MEMORANDUM M-622

Guide to the datareduction of imperfection surveys on circular shells

Ch. Cartalas
H.J.C. van der Hoeven
A.W.H. Klompe

Delft University of Technology
Faculty of Aerospace Engineering
Delft, The Netherlands

February 1990
Table of Contents

Acknowledgement

List of Symbols

Abstract

1. An introduction to the Data Reduction Process
   1.1. The imperfection survey
   1.2. AMIVAS
   1.3. Overview of the Data Reduction Programs

2. Data Reduction Programs
   2.1. Introduction
       2.1.1. Dimensions
   2.2. Guide on the use of the Data Reduction Programs
       2.2.1. Preliminary work
       2.2.2. Creating the file [shellcode]var
       2.2.3. Formatting program xx00
       2.2.4. Correction program xx01
       2.2.5. Best-fit program xx02
       2.2.6. Harmonic analysis program xx03
       2.2.7. Analysis check program xx04
       2.2.8. Program mapprint
       2.2.9. Program hargraph
       2.2.10. Program axigraph
       2.2.11. Program graph3d

References

Figures

Appendix A
Appendix B
Appendix C
Appendix D
Acknowledgement

The work presented in this paper was carried out by Ch. Cartalas during the eight weeks he resided in Delft as an IAESTE-trainee and by H.J.C. van der Hoeven as part of his study at the Delft University of Technology. The authors wish to express their thanks to Mrs. Marijke Mooyman for the skillful typing of the manuscript.

List of Symbols

\[ A_{00}, A_{0 k}, A_{i0}, A_{ik}^l, B_{kl}, C_{i0}, C_{k0}, C_{kl}^l, D_{kl} \]

- coefficients of the half-wave cosine Fourier representation.
- coefficients of the half-wave sine Fourier representation.

- \( i, k \) number of half-waves in the axial direction.
- \( \ell \) number of full-waves in the circumferential direction
- \( L \) shell length
- \( L_{HA} \) shell length used for harmonic analysis
- \( M \) integer corresponding to the last axial station
- \( N_C \) number of data points in the circumferential direction
- \( N_R \) number of data points in the axial direction
- \( R \) shell radius
- \( t \) shell thickness
- \( \bar{w}, \bar{w} \) radial imperfection with respect to the perfect cylinder, positive outward
- \( x, y \) axial and circumferential coordinates in the middle surface of the shell respectively
- \( X_1, Y_1 \) eccentricities (see figure 4)
- \( \epsilon_1, \epsilon_2 \) small angles in radians-inclination of the best-fit reference axis (see figure 4)
- \( \epsilon_3 \) semi-vertex angle of a conical shell (see figure 4)
- \( \theta = \frac{Y}{R} \) non-dimensional circumferential coordinate
- \( \xi \) equivalent initial imperfection amplitude
Abstract

In this paper a description is given of the structure and use of the computer programs related to the data reduction process of the initial imperfection surveys on circular shells.

The purpose of these programs are:

1. Putting the measured imperfection data in a standard format (program xx00).
2. Correcting the measured imperfections for obvious errors made during the measurement stage (program xx01).
3. Performing a best-fit on the, in the former step, corrected data (using the least squares method) to find the imperfections with respect to the 'perfect' (best fitted) shell (program xx02).
4. Performing harmonic analysis on the best-fitted data to find the two-dimensional fourier coefficients which describe the measured imperfections (program xx03).

With a plot program (graph3d) a pseudo 3-dimensional plot can be made at allmost any stage of the data reduction process to verify the result of each reduction step.

This paper is divided into four major parts:

1. An introduction to the data reduction process.
2. A guide on the use and structure of the programs.
3. An example of all outputs that can be obtained from these programs. For this purpose, the imperfections of the ARIANE interstage I/II, shell 'a12114' has been processed.
4. All program listings, as well as a listing of all variables needed for the proper working of the program.

The final results of this data reduction process are presented in the form recommended by the Initial Imperfection Data Bank publications [1] and [2].
1. An introduction to the Data Reduction Process

This chapter gives a short description of the imperfection survey in order to understand the input for the programs. After that a summary of the available programs is given.

1.1. The imperfection survey

Whereas the degrading effect of the unavoidable initial imperfections on the load carrying capability of thin-walled structures is now universally known, still in general, very little information is available about the shape and size of initial imperfections produced by the different fabrication processes. If one wants to improve the buckling load predictions and make full use of the non-linear analysis capabilities now available with several of the advanced structure analysis codes[3], then a detailed knowledge of the actual imperfections that are present in the real structures is a must.

For measuring the initial imperfection of real structures 3 instruments were designed and manufactured at the Aerospace Faculty and the Central machine shop of the Delft University of Technology.

1. AMIVAS, which is an acronym for 'Automatisch Meetsysteem voor het bepalen van de Initiële imperfecties Van Schaalkonstructies' (Initial imperfection survey instrument for thin-walled shells). This apparatus is capable of handling circular shells with a diameter of more than 1 meter and a height of up to 4 meter.

2. STONIVOXS, an acronym for 'Statistisch ONderzoek naar de invloed van Initiële VORMonzuiverheden op de Kniklast van Schaalkonstructies' (Statistical research into the influence of initial deformation on the buckling load of shells) [4]. Stonivoks can handle cylinders of a diameter of 66 mm and a height of 100 mm.

3. UNIVIMP, an acronym for 'UNIviersel instrument Voor het bepalen van initiële IMPerfecties van schaalkonstructies' (Universal instrument for the survey of initial imperfections of thin-walled shells) [5]. Univimp is designed for cylinders with a diameter of 240, 360 and 480 mm and a height of respectively 240, 360 and 480 mm.
Because the data reduction programs were originally developed for the processing of the initial imperfections measured on interstages of the ARIANE launch vehicle with the aid of AMIVAS in the next section a short description of AMIVAS will be given.

1.2. AMIVAS

Figure 1 shows the experimental set-up. The shells were positioned on a turntable and the reference beam placed parallel to it on an adjustable tripod. As can be seen from this figure and figure 2 use has been made of 7 linear voltage displacement transducers (L.V.D.T.) to carry out measurements.

Two fixed pick-ups (A) are bearing against the outer rim of the two end rings, and they are used for measuring the shape of the top and bottom ring of the shell. An accurate machined reference ring is placed on a frame attached to the top part of the turntable. With four transducers (C) bearing against the inner diameter of this ring (see figure 2) it is possible to trace the translation of the shell (rigid body motion). For more detail see reference [6].

The imperfection transducer (B) is installed on a carriage which can move along the reference beam by an electric drive. The transducer bears against the surface of the shell and is used to record the shape of the shell generator. Upon completion of an axial scan the shell is rotated to a new angular position, followed by another axial scan. The process is continued until the whole surface has been surveyed and recorded. The axial scan start just below the upper end and terminates just above the lower end of the shell.

The exact shape of the reference beam has been measured optically (see figure 3) in order to correct the measured initial imperfection scans during each imperfection survey step. The axial position of the carriage is recorded by an electro-optical device which scans a strip with equally spaced cut-outs fixed to the reference beam. Each time a cut-out is detected an electric pulse is generated. This pulse is used as a signal for the computer to digitize and store the signal from the imperfection transducer (B) at that moment. This system makes it possible to recor the imperfection data every 10 mm in axial direction. The data-aquisition system consist of an HP9825T programmable calculator, an HP6940B multiprogrammer (that acts as a scanner) and a digital XY-plotter. The analog to digital converter is a 12 bit system giving a +/- 5mV accuracy for an input of 10 Volt. The digitized signals are
recorded directly in the standard calculator format (8 byte words) and are stored on the cassette tape after each scan (that is after each axial measurement). Also each scan is recorded on the XY-plotter for a quick look examination.

At the University the data is transferred to a HP9000 computer for the data reduction process described in this paper.

1.3. Overview of the Data Reduction Programs

See the program flow chart in figure 5.

xx00
This program creates a file with the measured data in such a way that the data reduction programs can easily read it. The program fills possibly missing scans with the value 99.999.

xx01
This program makes the following corrections on the formatted data (result from xx00):
- Translation of the shell
  This correction is done on basis of the reference measurements (see figure 1, transducers C).

- Beam shape correction
  Normally the imperfection data has already been corrected for the beam shape during the measurement step. For some shells no beam correction (or a wrong one) has done, and if necessary this can be done (or corrected) by xx01.

- Extra reinforcement (if present)
  Extra reinforcement at the outside of the shell is handled by subtracting the thickness of the reinforcement from the measured data in the affected area(s).

- Missing scans
  The imperfection values of the missing scans are determined by a linear interpolation process between the scans before and after the missing scan.
- Inaccurate measurements around the hatches, rivets, etc.  
  These correction are done via interpolation and with a filter technique based on 
  fourier coefficients.

**xx02**

In this program the least-squares method is used to find a perfect circular shell 
which best fits the imperfection data.

The output of this program is a file which contains the imperfections of the 
measured shell with respect to this perfect shell.

**xx03**

In this program by a double harmonic analysis the Fourier coefficients of the 
imperfection surfaces with respect to the best fit shell (output of xx02) can be 
obtained.

The half-wave cosine representation in the axial direction involves the 
determination of two sets of harmonic components \( A_{kl} \) \((A_{i0})\) and \( B_{kl} \) where:

\[
\tilde{w}(x, \theta) = t \sum_{i=0}^{N} A_{10} \cos \frac{inx}{L} + t \sum_{k=0, l=1}^{N} \cos \frac{knx}{L} (A_{kl} \cos \ell \theta + B_{kl} \sin \ell \theta)
\]

and \( \theta = y/R \). Similarly, the half-wave sine representation in the axial direction 
involves the determination of two sets of harmonic components \( C_{kl} \) \((C_{i0})\) and \( D_{kl} \) where:

\[
\tilde{w}(x, \theta) = t \sum_{i=0}^{N} C_{i0} \sin \frac{inx}{L} + t \sum_{k=1, l=1}^{N} \cos \frac{knx}{L} (C_{kl} \cos \ell \theta + D_{kl} \sin \ell \theta)
\]

In xx03 this four sets of harmonic components are calculated. The user can specify 
the number of fourier coefficients to be calculated and whether the cosine, sine or 
both representations have to be calculated.

**NOTE:**

The number of fourier coefficients calculated may not exceed the number of data 
points/2 in each direction.

The number of data points must be odd. If this is not so, this number is adjusted 
automatically in program xx03 before calculation takes place (see section 2.2.6).

The result of this process is a file for the sine components or a file for the 
cosine components or both (depends on the users choice).
xx04
Program xx04 recalculates the imperfections of a shell from a given set of fourier coefficients. Either the half-wave sine or the half-wave cosine coefficients can be used as the input.

The resulting output file can be used to compare with the result from xx02 to check if in the harmonic analysis program xx03 enough fourier coefficients have been calculated to get a good representation of the original imperfections.

mapprint
This program makes a neat print out of the calculated Fourier coefficients.

hargraph
Program hargraph makes the axial and circumferential cross plots of $\sqrt{A_{k\ell}^2 + B_{k\ell}^2}$ and/or $\sqrt{C_{k\ell}^2 + D_{k\ell}^2}$ as either a function of the axial half-wave number $k$ and/or the circumferential full-wave number $\ell$.

axigraph
This program makes a plot of the axisymmetric components $A_{10}$ and $C_{10}$ for a maximum of 4 different shells per plot.

graph3d
This program can make pseudo 3-dimensional plots of the processed imperfection data of the programs xx00, xx01, xx02 and xx04 to check the correctness of the data reduction process.

loadvar
With this program one can create a file with the variables needed for all programs mentioned above.
2. Data Reduction Programs

2.1. Introduction

The data reduction programs were originally developed for the processing of imperfection data of ARIANE shells.

Whenever the reader encounters [shellcode] in this paper he should replace this by the identification code of the shell the data reduction process has to work on. All files concerning certain shell data are named [shellcode], followed by one or a few letters (see appendix A for the shellcode/filename convention).

Wherever the user finds input is requested from a list (list1 to list17), he can find those lists in the file [shellcode]var. For the input lists the NAMELIST-directed I/O method is used.

Before starting the data reduction process for a new shell one has to copy first an empty [shellcode]var file from the file /users/data/list or one can use the program loadvar to create such a [shellcode]var file.

All programs and files can be reached via the HP9000 serie 300 computer at the Faculty of Aerospace Engineering of the Delft University of Technology, known as the dutlbc2 (or the alias vakgr_c) computer. Wherever in this paper is refered to the computer, this computer is meant.

All source files, containing the FORTRAN 77 programs, can be found in the directory: /users/datared.

The object files, containing the compiled (sub)programs, are in the directory: /users/datared/odatared.

All data files (including [shellcode]var) can be found in the directory: /users/data. Which is on the hard disk of the other computer known as dutlbc1 (or the alias halsys). The computers vakgr_c and halsys are connected via a LAN. The data directory of the halsys is mounted automatically to the directory structure of the vakgr_c computer.

Also in this /users/data directory are the files variables, variablealphab and list, containing respectively a list of all needed variables in list order with comment, a list of all variables in alphabetic order with comment and an empty [shellcode]var file.
In the data reduction programs a scan is a set of measured imperfection data, measured in succession on a 'straight' line of the shell, with intervals of equal length between the data points.

A scan can either be oriented in axial direction (from top to bottom of the shell), or in circumferential direction (once around the shell).

By the length of the shell is meant the distance between the highest and lowest point on the shell in axial direction. In case of an ARIANE shell, this is equal to 10*(number of points per scan - 1), because the scans are oriented in axial direction and the distance between two data points on a scan is 10 mm.

2.1.1. Dimensions

A maximum of 400 scans, each containing a maximum of 400 data points, can be processed.

If more scans are present or if each scan contains more data points, the dimensions of the affected arrays in all programs must be adjusted to this situation.

Because the file protection of each file is set to 'read only' this can only be done by the system administrator.

If, however, the number of scans multiplied by the number of data points per scan (in the default situation 400*400=160000) exceeds 200000 (approximately), the computer vakgr_c will not have enough memory to process the shell.

The highest Fourier coefficient that can be calculated is 100 axial half-waves and/or 100 circumferential full-waves. If this is not sufficient, the user should ask the system administrator to change the dimension statement in the affected programs.

NOTE:

The number of fourier coefficients calculated may not exceed the number of data point/2 in each direction (see also note on page 7).
2.2. Guide on the use of the data reduction programs

2.2.1 Preliminary work

In case the data to be processed is derived from measurements on ARIANE shells or shells measured in a similar fashion, the first step in the data reduction procedure is reading the needed imperfection data from tape, making use of the HP9825T programmable calculator.

If the data has another origin, another data transfer method may be used. Consecutively this data is sent to the computer.

It is customary to write this data to a file which has the same code name as the shell under consideration: [shellcode]. In case the data comes from imperfection surveys on ARIANE shells this file consists of a heading which contains general information about the shell, followed by some reference data and the shape of the reference beam, which in turn is followed by the actual imperfection data.

Before processing of the data can commence, the data file has to be brought into standard shape.

In file [shellcode], only the imperfection data should be present in the form of consecutively numbered scans.

Each scan begins with a scan number, followed by a certain number (for ARIANE shells normally 6) of reference points and a fixed number of data points.

At the end of the imperfection data, a maximum of 10 control scans may be present in the data file. In case of ARIANE data, these control scans are measured at the end of the measurement session, to check the correct alignment of the measuring device. A maximum of 20 scans may be missing from the data file. These scans will then lateron be approximated by linear interpolation.

All other data present on the original data tape (the heading, etc.) should be written to the file [shellcode].

SPECIAL NOTE: Normally all the output files of a shell measured in the axial direction will be represented in axial way, for a shell measured in circumferential direction the output is always a circumferential shell presentation.

If the user wants to process a shell which contains hatches and/or extra reinforcements, this shell must be measured with scans oriented in axial direction.
If, however, the scans are oriented in circumferential direction, the user may write a small program which puts the data in 'axial representation' first, and from thereon treats the shell as if it is 'axially oriented'.

2.2.2. Creating the file [shellcode]var

This file [shellcode]var is used for the NAMELIST directed I/O. This file contains 17 lists of variables for the shell 'shellcode' and each data reduction program calls some of them. In order to create this file one can make a copy of the file /users/data/list to /users/data/[shellcode]var and edit this file (with the vi editor).

Another way to create and modify a file [shellcode]var is to use the program 'loadvar' in the directory /users/data/red. To start the program, type:

```
/users/data/red/loadvar [Return]
```

or

```
loadvar [Return]
```

The program displays the following menu:

- 1 list of parameters
- 2 modify one parameter
- 3 modify several parameters
- 4 create a new file [shellcode]var
- 5 quit

1. This option displays a complete list of all parameters in two columns to get a general view of this [shellcode]var file.

2. This option updates one variable only.
3. Option 3 allows the user to update several variables.
   This option stops by typing q.

4. This option creates a new file [shellcode]var, if this file already exists this
   option cannot be run.
   In all other cases a new [shellcode]var file is created. Option 4 copies the file
   /users/data/list and one can type in sequentially the values of the variables
   that are known at that moment. This option can be ended by typing q.

5. Option 5 terminates the loadvar program.

For the meaning of the used variable names the user is referred to the list in
appendix B.

2.2.3. Formatting program xx00

This program formats the measured imperfection data in such a way that the other
data reduction programs easily can read it.

The input of this program consists of the file [shellcode] and the file
[shellcode]var from which the list1 to list4 are read.

Before starting this program one has to fill the list 1 to 4 in the file
[shellcode]var for this shell. The needed information can be found in the report
which is made during the measurement session (measurement 'logbook').
This program can handle axially and circumferentially measured imperfection data.

Program xx00 creates two output files: one with the formatted imperfection data
[shellcode]b and one file with information about the formatting process, file
[shellcode]b_out.

The program is started by typing:
The first thing the program does is prompting the user for a 'shellcode'. After this code has been given, the program reads the imperfection data from file [shellcode], fills the missing scans with the value 99.999 and subtracts the average of all imperfection data from the imperfection data points. Consecutively the new processed data is written to the file [shellcode]b in the standard format for the data reduction programs (FORTRAN format 8f9.5). After program xx00 has terminated, the user is recommended to check file [shellcode]b_out to see if everything went right.

With program 'graph3d' a pseudo 3-d plot can be made of this formatted imperfection data (see section 2.2.11).

2.2.4. Correction program xx01

In this program errors made during the measurement session are corrected. Corrections are made for:

1. translation of the shell during the measurement session (as measured by the pickups C of the reference ring - see figures 1 and 2).

2. reference beam influences.

3. extra reinforcements (if present).

4. missing scans (if present).

5. inaccurate or faulty measurements around hatches, faulty measurements at rivets, etc.
The input for this program consists of the file [shellcode]b and file [shellcode]var from which list1, list2, list3, list5, list6 and list7 are read. Lists 1, 2 and 3 have already been filled for program xx00, so only lists 5, 6 and 7 have to be filled.

This program creates two output files, one with the corrected imperfection data [shellcode]c and one file with information about the correction process, file [shellcode]c_out.

The program is started by typing:

    /users/datedred/xx01  [Return]

or

    xx01  [Return]

The way the correction will be done by xx01 is described below.

**translation of the shell**

During the measurement session, the relative position of the shell with respect to the measurement device must constantly be monitored.

In case of ARIANE data, this results in 6 reference points for each scan measured. With the information contained in four of these reference points the program can make a correction for this relative translation of the shell during the measurement session.

From the other two reference points the shape of the two end rings can be obtained (if necessary).

**reference beam shape influence**

For ARIANE data, correction for the influence of the non perfect shape of the reference beam on the measured data usually is done during the actual measurement, so this correction step can be by-passed.
Whenever this correction has not been done yet, the user can use the beam shape correction option of this program. If this correction is necessary, the user has to create a file called [shellcode]beam, which contains:

1. A standard number of reference points (NREFPOINT) with a value of 0.0.

2. Data which describes the shape of the reference beam, from top to bottom, with intervals equal to the intervals used during the measurement session and for the corresponding part of the beam.

If the imperfection data has been derived in a different manner, the reference beam correction procedure may have to be changed.

**extra reinforcement**

This extra reinforcement is handled by subtracting the thickness of the reinforcement from the measured imperfection data in the affected area(s). The user therefore has to specify how many scans contain areas with extra reinforcement (NSCANPLATING in list5), the matching scan numbers (ARRAYPLATING in list6), the axial data point numbers of the beginning and the end of this reinforcement (NFIRSTPLATING and NLASTPLATING in list6) and the thickness (PLATETHICKNESS in list6).

If no extra reinforcement is present (NSCANPLATING = 0 in list5) list6 will not be read.

**missing scans**

In program xx00 missing scans have already been specified and filled with the value 99.999.

In xx01 these missing scans are filled in by linear interpolation between the scans before and after the missing scan.

**inaccurate measurements around hatches, rivets, etc.**

To make the influence of the hatches on the corrections as small as possible, the hatch areas are first 'filled in' by linear interpolation.
If the scans are oriented in circumferential direction, the data is 'switched', so that scans are always oriented in axial direction during correction. This is done because the imperfections in a shell (always) vary much more in circumferential direction than they do in axial direction, so that approximation of the data in axial direction with a few fourier coefficients is much more accurate.

Correction is performed on each scan separately and consists of the following few steps:

1. The scan is copied to an 'approximation scan'.

2. All data in the 'approximation scan' higher than MAX (list5) or lower than -MAX is replaced by the nearest value in axial direction that is smaller than MAX and greater than -MAX.

3. A one dimensional fourier analysis is done of the 'approximation scan', calculating the first NCOEF(list5) fourier coefficients.

4. With the lower order NCOEF coefficients, a 'recalculated scan' is composed.

5. The original scan is now compared to the 'recalculated scan'. If a data point in the scan deviates more than PEAKMAX (list5, recommended value: 0.6 to 0.8 mm) from the 'recalculated scan', this data point is corrected. Correction consists in removing the incorrect part from the scan and fitting in the corresponding part of the 'recalculated scan'.

This way even large parts of a scan can be corrected fairly accurately. If the errors occur at the beginning or end of a scan, correction is performed by replacing the incorrect beginning or end of the scan by the corresponding part of the 'recalculated scan' in such a way that, the substituted part 'fits' to the correct part of the scan.

The corrected data is put back in its original orientation (axial or circumferential).

The hatches, which contained only incorrect data, will now have been 'corrected'. The data points in a hatch area are therefore set to 99.999, otherwise the hatches will not be visible in subsequent 3-d plots.
The corrected data is now written to the file [shellcode]c, while file [shellcode]c_out contains information on the corrections performed. If all went right, file [shellcode]c_out contains the message 'AMIS CALLED FOR SCAN [scan number]' for each missing scan, followed by a list of all error-corrections performed on the separate scans.

For each correction the following information is given: the scan number, the data point number, the value before correction, and the value after the correction has been completed.

If the list contains (too many) points in which a very small correction has been made, either PEAKMAX or MAX have been chosen too small. In that case check and adjust the values of PEAKMAX and/or Max and run xx01 again.

If from file [shellcode]c_out the top and bottom measurements on the shell appear to be mostly incorrect, the user may decide to omit the measurements at these places from the file [shellcode], adjust NPOINT and/or NSCAN in list1, and start again the program xx01. Now a smaller part of the shell will be processed, omitting the top and bottom of the shell.

It is not recommended to take NCOEF much higher than 4, because this has almost no effect on the quality of the approximation obtained, while for a very good approximation NCOEF should be at least 30, and then the approximation becomes so accurate, that unwanted peaks will not be corrected in the process described above. Usually MAX=3 is a usable value, however, care must be taken that in file [shellcode]b no significant data exceeds MAX.

A very convenient way to check if (almost) all unwanted peaks have been corrected, is to compare the 3-d plot from file [shellcode]b with the 3-d plot from file [shellcode]c. The process of making the 3-d plots is described in section 2.2.11 of this paper.

2.2.5 Best-fit program xx02

In order to be able to compare the results obtained from one shell with the results of another one, it is necessary to perform a best-fit procedure on the shell imperfection data. In program xx02 the least-squares method is used to find a perfect cylinder or cone which best fits the given imperfection data.
Before xx02 can be run list8 in file [shellcode]var has to be filled in. If INITIALGUESS=1 in list8, list9 also has to be filled in.

Program xx02 uses file [shellcode]c as input, and makes the file [shellcode]f as output. The file [shellcode]f_out contains extra information on the best-fit process. In that file one can find the solution vector E of the best-fit process (see also figure 4):

- $E(1) = X_1$
- $E(2) = Y_1$
- $E(3) = \cos(\epsilon_1)$
- $E(4) = \cos(\epsilon_2)$
- $E(5) = \tan(\epsilon_3)$
- $E(6) = R$ at $Z=0$

Program xx02 is started by typing:

/users/dated/xx02 [Return]

or

xx02 [Return]

With program graph3d (section 2.2.11) a pseudo 3-d plot can be made of the output data, [shellcode]f, of this program.
2.2.6. Harmonic analysis program xx03

In program xx03 the actual two dimensional harmonic analysis is performed. This program uses file [shellcode]f and list1 and list12 from file [shellcode]var as the input. In list12, the user should specify the fourier coefficients to be calculated and whether a half-wave sine, half-wave cosine or both representations have to be calculated.

Program xx03 is started by typing:

/users/datared/xx03  [Return]

or

xx03  [Return]

Output is written to the files [shellcode]hs (the sine coefficients), and [shellcode]hc (the cosine coefficients). Extra output about the analysis process can be found in the file [shellcode]f_out.

Before harmonic analysis can begin, the data in the hatch areas has to be interpolated, because that way, the influence of the hatches on the harmonic analysis is minimal. For the calculation of the necessary integrals use is made of the trapezoid rule.

For better accuracy the number of point in the axial- and circumferential direction must be odd. If this is not so, the numbers are adjusted automatically to odd by the following steps:

- In the circumferential direction program xx02 adds the first scan at the end of the scans measured, and the program runs the analysis with a circumferential number of scans = NSCAN+1.

- In the axial direction the last points of each scan are ignored and the program runs the analysis process with a number of axial point = NPOIN -1.

In the half-wave cosine axial, short 'axial cosine', representation the fourier coefficients are given by:
\[
A_{00} = \frac{1}{2\pi L} \int_0^L \int_0^{2\pi} W(x, \theta) \, dx \, d\theta \\
A_{k0} = \frac{1}{\pi L} \int_0^L \int_0^{2\pi} W(x, \theta) \cos \frac{k\pi x}{L} \, dx \, d\theta \quad k > 0 \\
A_{0l} = \frac{1}{\pi L} \int_0^L \int_0^{2\pi} W(x, \theta) \cos \ell \theta \, dx \, d\theta \quad \ell > 0 \\
A_{kl} = \frac{2}{\pi L} \int_0^L \int_0^{2\pi} W(x, \theta) \cos \frac{k\pi x}{L} \cos \ell \theta \, dx \, d\theta \quad k > 0, \ell > 0 \\
B_{0l} = \frac{1}{\pi L} \int_0^L \int_0^{2\pi} W(x, \theta) \sin \ell \theta \, dx \, d\theta \quad \ell > 0 \\
B_{kl} = \frac{2}{\pi L} \int_0^L \int_0^{2\pi} W(x, \theta) \cos \frac{k\pi x}{L} \sin \ell \theta \, dx \, d\theta \quad k > 0, \ell > 0
\]

The imperfections are now represented by:

\[
W(x, \theta) = t \sum_{k=0}^{\infty} \sum_{\ell=0}^{\infty} \cos \frac{k\pi x}{L} \left[ A_{kl} \cos \ell \theta + B_{kl} \sin \ell \theta \right]
\]

In the half-wave sine axial, short 'axial sine', representation the fourier coefficients are given by:

\[
C_{k0} = \frac{1}{\pi L} \int_0^L \int_0^{2\pi} W(x, \theta) \sin \frac{k\pi x}{L} \, dx \, d\theta \quad k > 0 \\
C_{kl} = \frac{2}{\pi L} \int_0^L \int_0^{2\pi} W(x, \theta) \sin \frac{k\pi x}{L} \cos \ell \theta \, dx \, d\theta \quad k > 0, \ell > 0 \\
D_{kl} = \frac{2}{\pi L} \int_0^L \int_0^{2\pi} W(x, \theta) \sin \frac{k\pi x}{L} \sin \ell \theta \, dx \, d\theta \quad k > 0, \ell > 0
\]
The imperfections are now given by:

\[ W(x, \theta) = \sum_{k=0}^{\infty} \sum_{\ell=0}^{\infty} \frac{a_{k\ell}}{L} \left( C_{k\ell} \cos \ell \theta + D_{k\ell} \sin \ell \theta \right) \]

These representations are like a 'finger print' displaying the imperfections of the structure and the manufacturing method of the shell.

2.2.7. Analysis check program xx04

Program xx04 recalculates the imperfection data from a given set of fourier coefficients.

Either a [shellcode]hs or a [shellcode]hc file can be used as the input for this program.

This program is started by typing:

```
/users/datared/xx04   [Return]
```

or

```
xx04                 [Return]
```

The recalculated imperfection data can be found in the file [shellcode]k.

When comparing a 3-d plot of this file with a similar plot of file [shellcode]f, the user can readily determine if enough fourier coefficients have been calculated in xx03 to get a good approximation of the original imperfections.

Furthermore, the user can also see whether the 'axial cosine' or the 'axial sine' representation gives the best result.

If the user so chooses, he can rename file [shellcode]k to file [shellcode]f, rerun program xx03, and compare the new files [shellcode]hs and [shellcode]hc with the old ones.

This is only usefull when doubt exists in the proper working of program xx03. If the old and new files are almost identical, the program works correctly.
In principal, in the theoretical case that an infinite number of two-dimensional fourier coefficients have been calculated, the files [shellcode]f and [shellcode]k should be identical. Watch for 'high frequency throw-back'.

In the practical case where K2=30 and L2=80 (recommended for ARIANE shells), no easy comparison is possible between the files [shellcode]f and [shellcode]k, except for the 3-d plots.

Remember, to avoid what is called 'high frequency throw-back' in signal theory, the number of fourier coefficients calculated may not exceed the number of data points/2 in each direction (see also the corresponding notes on page 7 and page 10).

2.2.8. Program mapprint

Program mapprint was written to make a neat printout of the calculated fourier coefficients.
The input is [shellcode]var (list16) and the files [shellcode]hc and/or [shellcode]hs, depending of the value of CHOICE in list16.

This program is started by typing:

```
/users/datared/mapprint [Return]
```

or

```
mapprint [Return]
```

The output in the form of tables is printed on the printer halsys:/dev/lp.
The user only has to specify which coefficients should be printed and in which format the data has to appear on paper (3, 4 or 5 decimals behind the decimal point).

2.2.9. Program hargraph

This program was written to make axial and circumferential cross plots of $\sqrt{(A_{k\ell}^2 + B_{k\ell}^2)}$ and/or $\sqrt{(C_{k\ell}^2 + D_{k\ell}^2)}$ as either a function of the axial half-wave number $k$ and/or the circumferential full-wave number $\ell$. 
The input is in list13, list14 and list15 of file [shellcode]var and either file [shellcode]hc and/or [shellcode]hs.

Program hargraph is started by typing:

```
/users/datared/hargraph [Return]
```

or

```
hargraph [Return]
```

Via a menu the user can choose an output of maximum 4 plots on the console and/or on the halsys plotter.

2.2.10. Program axigraph

This program makes a cross plot of the axisymmetric sine and/or cosine components $A_{10}$ and $C_{10}$ of a maximum of 4 shells per plot.

The input of this program is either file [shellcode]hc and/or file [shellcode]hs.

The program is started by typing:

```
/users/datared/axigraph [Return]
```

or

```
axigraph [Return]
```

The program will prompt the user to enter:

- the name of the output device.

- the number of shells to be plot in axi-symmetric cross plots.

- the shell identification code.

- the users choice: a plot of $A_{10}$ or $C_{10}$ or both.
Fig. 1 Experimental test set-up
Fig. 2 Reference ring and transducer couples
Fig. 3 Shape of the reference beam for different vertical positions
**SOLUTION VECTOR:**

\[ E(1) = X_i \]
\[ E(2) = Y_i \]
\[ E(3) = \cos(E_1) \]
\[ E(4) = \cos(E_2) \]
\[ E(5) = \tan(E_3) \]
\[ E(6) = R \text{ at } Z = 0 \]

**X, Y, Z** Reference axis of traversing pick-up

**X', Y', Z'** Reference axis of best-fit cylinder

**d_i** Normal distance from measured point to best-fit cylinder

Fig. 4 Best-fit cone and cylinder reference axis
Fig. 5 Program flow chart
APPENDIX A

File name convention

The imperfection data, in different stages of the data reduction process, is stored in the directory /users/data.

Each file has a name which includes a 6 character shell identification code, the shellcode, and one or more characters which indicate what stage the reduction process has passed.

For an ARIANE shell with the shellcode a12114 the file names are as follows:

a12l14   rough imperfection data
a12l14a  heading of the rough data
a12l14b  formatted data (output xx00)
a12l14c  corrected data (output xx01)
a12l14f  data with respect to the best-fit (output xx02)
a12l14hc components of the half-wave cosine representation (output xx03)
a12l14hs components of the half-wave sine representation (output xx03)
a12l14k  recalculated imperfections (output xx04)
a12l14var contains the variables of the shell
APPENDIX B

Listing of all variables
used in the file
[shellcode]var
in
- order listed
- alphabetical order
VARIABLES NEEDED FOR DATEREDUCTION PROGRAMS IN ORDER LISTED.

list1: xx00  xx01  xx02  xx03  xx04  plot3d
SHELLCODE - Identification code for the shell used.
MEASUREDATE - date of imperfection measurement.
NSCAN - total number of scans,
           including number of missing scans,
           excluding number of control scans.
NPOINT - number of data points per scan.
CONICITY - conicity of shell, deviation from cylinder,
           measured in degrees, positive if R(top) < R(bottom).
THICKNESS - wall thickness of the shell (in mm).
RADIUS - radius at the bottom of the shell (in mm).
LENGTH - length used by measurement(mostly equal LHA) (in mm).
SCANTYPE - =0 => scans are oriented in axial direction.
           =1 => scans are oriented in circumferential direction.
           If SCANTYPE=1, hatches and extra plating cannot be
           properly 'placed'.
           In that case write a small program to put the available
           data in axial representation first.

list2: xx00  xx01  xx04
FIRSTSCAN - first scan number.
SCANINCR - increment in scan numbering.
MMISSCAN - number of missing scans (maximum 20).
NCONTSCAN - number of control scans (maximum 10).
NNREFPOINT - number of reference data points.
REFOCODE - handling code reference data.
           =0 => 6 reference points - standard handling.
           =1 => 6 reference points - user defined handling.

list3: xx00  xx01
MISSINGSCAN - array containing the scan numbers of the
               missing scans (maximum 20).

list4: xx00  xx04
CONTROLSCAN - array containing the scan numbers of the
               control scans (maximum 10).

list5: xx01  xx04
EXTRAOUT1 - =0 => no extra output from correction procedure.
           =1 => extra output.
PEAKMAX - Each axial scan is approximated by calculating its
           first NOE  one-dimensional fourier coefficients.
           Watch for 'high frequency throw-back'!
           All data on the actual scan that deviates more than
           PEAKMAX from the approximation is then corrected.
           Recommended for Ariane-shells: PEAKMAX=0.6-0.8 mm.
MAX - all correct data in the b-file must be between
       MAX and -(MAX).
       If MAX is chosen much too large, correction will
       be inaccurate, MAX must not be chosen too small !!
       (All data above MAX or below -(MAX) is made equal
       to the nearest data (in axial direction) not greater
       than MAX, before fourier analysis is performed!)
       For Ariane-shells: MAX = 3.0-4.0 mm.
NCOEF - Number of Fourier coefficients used to approximate each scan (usually 4 is sufficient). This approximation is used for peak correction. If NCOEF is chosen too high, the peaks which should be corrected have too much effect on the 'approximation-scan', so that correction will not be sufficient.

NSCANPLATING - number of scans with extra plating.
NSCANHATCH - number of scans with hatches.
BEAMCORRECT - =0 -> no beam correction needed.
=1 -> beam correction has to be done.

list6: xx01
NFIRSTPLATING - axial data point, beginning of extra plating.
NLASTPLATING - axial data point, end of extra plating.
PLATETHICKNESS - thickness of extra plating.
ARRAYPLATING - array containing the numbers of the scans which need correction for extra plating.

list7: xx01 xx04
NFIRSTHATCH - axial data point, beginning of hatch.
NLASTHATCH - axial data point, end of hatch.
ARRAYHATCH - array containing the numbers of the scans with hatches, including missing scans.

list8: xx02
EXTRAOUTZ - =0 -> no extra output from least-squares fit procedure.
=1 -> extra output.
INITIALGUESS - =0 -> no initial guesses (see list9).
=1 -> initial guesses.

list9: xx02
ARRAYINITIAL(1) - initial guess for eccentricity in X-direction.
ARRAYINITIAL(2) - initial guess for eccentricity in Y-direction.
ARRAYINITIAL(3) - initial guess for cos(E1).
ARRAYINITIAL(4) - initial guess for cos(E2).
ARRAYINITIAL(5) - initial guess for tan(E3).
ARRAYINITIAL(6) - initial guess for radius of shell at 2=0.

list10: plot3d
PLOTINCR - plot interval in axial data points.
PLOTCRIT - plot criterion:
=0 -> lift pen if abs(W) > 3*T
else -> lift pen if abs(W) > PLOTCRIT*T
SCALESET - =0 -> default scale setting.
=1 -> user defined scale setting.

list11: plot3d
YMIN - user defined Y minimum (scale setting).
YMAX - user defined Y maximum (scale setting).

list12: xx03 xx04
K1 - lowest harmonic component to be calculated in axial direction (usually 0).
K2  - highest harmonic component to be calculated in axial direction (usually 30 for ARIANE shells). Watch for 'high frequency throw-back (see remarks on pages 9, 12 and 22).

L1  - lowest harmonic component to be calculated in circumferential direction (usually 0).

L2  - highest harmonic component to be calculated in circumferential direction (usually 40 for ARIANE shells). Watch for 'high frequency throw-back (see remarks on pages 9, 12 and 22).

COEF  - =0 => both sine and cosine coefficients are calculated.
       =1 => only cosine coefficients are calculated.
       =2 => only sine coefficients are calculated.

list13:  hargraph
KMAX  - number of points to be plotted in circumferential crossplots.

LMAX  - number of points to be plotted in axial crossplots.

XMAXH  - length of X-axis (scale setting) user defined.

YMAXH  - length of Y-axis (scale setting) user defined.

NLINEPLOT  - number of lines per plot (maximum 4).

PLOTCOEF  - =0 => sine and cosine plots are made.
            =1 => only cosine plots are made.
            =2 => only sine plots are made.

list14:  hargraph
ARRAYK  - array containing numbers of selected axial half-waves to be plotted

list15:  hargraph
ARRAYL  - array containing numbers of selected circumferential waves to be plotted.

list16:  mapprint
FORMAT  - number of decimals in printout of fourrier coefficients (FORMAT = 3,4 or 5)

K1  - number of lowest axial coefficient to be printed.

Kh  - number of highest axial coefficient to be printed.

L1  - number of lowest circumf. coefficient to be printed.

Lh  - number of highest circumf. coefficient to be printed.

CHOICE  - =0 => A, B, C, and D matrices will be printed.
         =1 => A and B matrices will be printed.
         =2 => C and D matrices will be printed.
VARIABLES NEEDED FOR DATA REDUCTION PROGRAMS IN ALPHABETICAL ORDER.

ARRAYHATCH - array containing the numbers of the scans with hatches, including missing scans.
ARRAYINITIAL(1) - initial guess for eccentricity in X-direction.
ARRAYINITIAL(Z) - initial guess for eccentricity in Y-direction.
ARRAYINITIAL(3) - initial guess for cos(E1).
ARRAYINITIAL(4) - initial guess for cos(E2).
ARRAYINITIAL(5) - initial guess for tan(E3).
ARRAYINITIAL(6) - initial guess for radius of shell at Z=0.
ARRAYK - array containing numbers of axial half-waves selected for plotting.
ARRAYL - array containing numbers of circumferential waves selected for plotting.
ARRAYPLATING - array containing the numbers of the scans which need correction for extra plating.
BEAMCORRECT - @ > no beam correction needed.
@1 > beam correction has to be done.
CHOICE - 0 > A, B, C, and D matrices will be printed.
@1 > A and B matrices will be printed.
@2 > C and D matrices will be printed.
COEF - 0 > both sine and cosine coefficients are calculated.
@1 > only cosine coefficients are calculated.
@2 > only sine coefficients are calculated.
CONICITY - conicity of shell, deviation from cylinder, measured in degrees, positive if R(top) < R(bottom).
CONTROLSAN - array containing the scan numbers of the control scans (maximum 10).
EXTRAOUT1 - @ > no extra output from correction procedure.
@1 > extra output.
EXTRAOUT2 - @ > no extra output from least-squares fit procedure.
@1 > extra output.
FIRSTSCAN - first scan number.
FORMAT - number of decimals in printout of fourier coefficients (FORMAT = 3, 4 or 5)
INALGUESS - @ > no initial guesses (see list9).
@1 > initial guesses.
K1 - lowest harmonic component to be calculated in axial direction (usually 0).
K1 (list16) - number of lowest axial coefficient to be printed.
K2 - highest harmonic component to be calculated in axial direction (usually 30 for ARIANE shells).
Watch for 'high frequency throw-back' (see remarks on pages 9, 12 and 22).
Kh (list16) - number of highest axial coefficient to be printed.
KMAX - number of points to be plotted in circumferential crossplots.
L1 - lowest harmonic component to be calculated in circumferential direction (usually 0).
L1 (list16) - number of lowest circumf. coefficient to be printed.
L2 - highest harmonic component to be calculated in circumferential direction (usually 40 for ARIANE shells).
Watch for 'high frequency throw-back' (see remarks on pages 9, 12 and 22).
Lh (list16) - number of highest circumf. coefficient to be printed.
LENGTH - length used by measurement (mostly equal LHA) (in mm).
LMAX - number of points to be plotted in axial crossplots.
MAX - all correct data in the b-file must be between
      MAX and -(MAX).
      If MAX is chosen much too large, correction will
      be inaccurate, MAX must not be chosen too small !
      (All data above MAX or below -(MAX) is made equal
      to the nearest data in axial direction)
For Ariane-shells: MAX = 3.0-4.0 mm.
MEASUREDATE - date of imperfection measurement.
MISSINGSCAN - array containing the scan numbers of the
                missing scans (maximum 20).
NCOEF - Number of fourier coefficients used to approximate
        each scan (usually 4 is sufficient).
        This approximation is used for peak correction.
        If NCOEF is chosen too high, the peaks which should
        be corrected have too much effect on the
        'approximation-scan', so that correction will not
        be sufficient.
NCONTSCAN - number of control scans (maximum 10).
NFIRSTHATCH - axial data point, begin of hatch.
NFIRSTPLATING - axial data point, begin of extra plating.
NLASTHATCH - axial data point, end of hatch.
NLASTPLATING - axial data point, end of extra plating.
NLINEPLOT - number of lines per plot (maximum 4).
NMISSCAN - number of missing scans (maximum 20).
NPOINT - number of data points per scan.
NREFPOINT - number of reference data points.
NSCAN - total number of scans,
        including number of missing scans,
        excluding number of control scans.
NSCANCHATCH - number of scans with hatches.
NSCANPLATING - number of scans with extra plating.
PEAKMAX - Each axial scan is approximated by calculating it's
          first NCOEF one-dimensional fourier coefficients.
          All data on the actual scan that deviates more than
          PEAKMAX from the approximation is then corrected.
          Recommended for Ariane-shells: PEAKMAX=0.6-0.8 mm.
PLATETHICKNESS - thickness of extra plating.
PLOTCOEF - =0 => sine and cosine plots are made.
            =1 => only cosine plots are made.
            =2 => only sine plots are made.
PLOTCRIT - plot criterion:
            =0 => lift pen if abs(W) > 3*T
            else => lift pen if abs(W) > PLOTCRIT*T
PLOTRINC - plot interval in axial datapoints.
RADIUS - radius at the bottom of the shell (in mm).
REFCODE - handling code reference data.
            =0 => 6 reference points - standard handling.
            =1 => 6 reference points - user defined handling.
SCALESET - =0 => default scale setting.
            =1 => user defined scale setting.
SCANINCR - increment in scan numbering.
SCANTYPE - =0 => scans are oriented in axial direction.
            =1 => scans are oriented in circumferential direction.
            If SCANTYPE=1, hatches and extra plating cannot be
            properly 'placed'.
In that case write a small program to put the available data in axial representation first.

- Identification code for the shell used.
- Wall thickness of the shell (in mm).
- X maximum (scale setting 3-d plots) defined by user.
- Y minimum (scale setting 3-d plots) defined by user.
- Length of X-axis (in axial/circumferential crossplots).
- Length of Y-axis (in axial/circumferential crossplots).
APPENDIX C

Example of output derived from the data reduction programs for an ARIANE shell
(shellcode = a12f14)
$list1
SHELLCODE = 'al2114 fields'
MEASUREDATE = '06-01-1983'
NSCAN = 80
NPOINT = 303
CONICITY = 7.0
THICKNESS = 1.8
RADIUS = 1900.0
LENGTH = 3020
SCANTYPE = 0
$end
$list2
FIRSTSCAN = 201
SCANINCR = 1
NMISSCAN = 4
NCONTSCAN = 0
NREFPOINT = 6
REFCODE = 0
$end
$list3
MISSINGSCAN = 222, 254, 255, 268
$end
$list4
CONTROLSCAN =
$end
$list5
EXTRADOUT1 = 1
PEAKMAX = 0.9
MAX = 3
NCOEF = 4
NSCANPLATING = 0
NSCANCHATCH = 9
BEAMCORRECT = 0
$end
$list6
NFIRSTPLATING =
NLASTPLATING =
PLATETHICKNESS =
ARRAYPLATING =
$end
$list7
NFIRSTHATCH = 123
NLASTHATCH = 171
ARRAYHATCH = 219, 220, 221, 243, 244, 245, 267, 268, 269
$end
$list8
EXTRADOUT2 = 1
INITIALGUESS = 0
$end
$list9
ARRAYINITIAL(1) =
ARRAYINITIAL(2) =
ARRAYINITIAL(3) =
ARRAYINITIAL(4) =
ARRAYINITIAL(5) =
ARRAYINITIAL(6) =
$end
Fig. C1 Measured initial shape after the formatting step

Fig. C2 Measured initial shape after the correction step
Fig. C3 Measured initial shape after the best-fit step

Fig. C4 Recalculated initial shape
INPUT DATA
NPOINT = 303
NSCAN = 00
EXTRAOUTZ = 1

INITIAL GUESS RADIUS = 1900.000
INITIAL GUESS CONE-ANGLE DELTA = 7.000
LENGTH OF SCAN = 3020.000

EPS= 1.0000000013E-09

0 0
X(I)= -3.241904378E+00 1
X(I)= 4.434004520E-01 1
X(I)= 1.958111897E-03 1
X(I)= 1.049002822E-03 1
X(I)= -2.629900872E-03 2
X(I)= -9.485720396E+00 3

3 3
X(I)= 7.279856072E-04 0
X(I)= -1.424799745E-07 1
X(I)= -3.831230485E-07 2
X(I)= 2.093067903E-10 3
X(I)= -2.512310082E-09 4
X(I)= 4.758855425E-06 5

5 2
X(I)= -4.768304062E-07 1
X(I)= -2.566061603E-10 2
X(I)= 2.509244268E-10 3
X(I)= -1.998974871E-14 4
X(I)= 5.476582035E-10 5
X(I)= 8.216438770E-07 5

3 5
X(I)= 7.942986713E-07 0
X(I)= 5.935482868E-10 1
X(I)= -4.180408961E-10 2
X(I)= -5.924065847E-14 3
X(I)= 5.475403117E-10 4
X(I)= 8.204585811E-07 4

4 4
X(I)= -4.764249084E-07 1
X(I)= -4.889089480E-10 2
X(I)= 2.508476056E-10 3
X(I)= 5.897892106E-14 4
X(I)= -1.642797703E-09 5
X(I)= -2.463162616E-06 6

RESULTS AFTER ITERATION NO.: 1
MATRICES A AND B

\[
\begin{align*}
\text{SUM OF SQUARES IN PREVIOUS STEP WAS: } SS &= 0.1854E+05 \\
\text{NUMBER OF ITERATIONS IN EQSOV: } IT &= 5
\end{align*}
\]

\[
\begin{align*}
\text{THE SOLUTION VECTOR IS:} & & \text{THE IMPROVED SOLUTION VECTOR IS:} \\
C(1) &= -0.3241 & E(1) &= -0.3241 \\
C(2) &= 0.0443 & E(2) &= 0.0443 \\
C(3) &= 0.0003 & E(3) &= 0.0002 \\
C(4) &= 0.001 & E(4) &= 0.0001 \\
C(5) &= 0.0003 & E(5) &= 0.1225 \\
C(6) &= -0.3486 & E(6) &= 1.8399.0514
\end{align*}
\]

\[
\begin{align*}
\text{EPS} &= 0.1000000013E-09 \\
\begin{array}{ll}
\times(1) &= -0.6913772813E-04 & 1 \\
\times(2) &= -0.1942603012E-04 & 1 \\
\times(3) &= -0.138912761E-07 & 2 \\
\times(4) &= -0.4605457349E-07 & 3 \\
\times(5) &= -0.1090538659E-08 & 3 \\
\times(6) &= -0.8345336028E-05 & 3
\end{array}
\end{align*}
\]

\[
\begin{align*}
\text{RESULTS AFTER ITERATION NO.:} & & 2 \\
\text{MATRICES A AND B}
\end{align*}
\]

\[
\begin{align*}
\text{SUM OF SQUARES IN PREVIOUS STEP WAS: } SS &= 0.3049E+04 \\
\text{NUMBER OF ITERATIONS IN EQSOV: } IT &= 2
\end{align*}
\]
THE SOLUTION VECTOR IS:

\begin{align*}
C(1) &= -0.0001 \\
C(2) &= 0.0000 \\
C(3) &= 0.0000 \\
C(4) &= 0.0000 \\
C(5) &= 0.0000 \\
C(6) &= 0.0000 \\
\end{align*}

THE IMPROVED SOLUTION VECTOR IS:

\begin{align*}
E(1) &= -0.3242 \\
E(2) &= 0.0444 \\
E(3) &= 0.0002 \\
E(4) &= 0.0001 \\
E(5) &= 0.1225 \\
E(6) &= 1899.0514 \\
\end{align*}

SOLUTION VECTOR:

\begin{align*}
E(1) &= -0.3242 \\
E(2) &= 0.0444 \\
E(3) &= 0.0002 \\
E(4) &= 0.0001 \\
E(5) &= 0.1225 \\
E(6) &= 1899.0514 \\
\end{align*}
| \( L \) | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| \( K \) | 0.005 | 0.023 | 0.030 | 0.048 | 0.005 | 0.006 | 0.007 | 0.008 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 |
| \( K \) | 0.005 | 0.007 | 0.006 | 0.005 | 0.006 | 0.007 | 0.008 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.007 | 0.006 | 0.005 | 0.006 | 0.007 | 0.008 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.007 | 0.006 | 0.005 | 0.006 | 0.007 | 0.008 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.007 | 0.006 | 0.005 | 0.006 | 0.007 | 0.008 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.007 | 0.006 | 0.005 | 0.006 | 0.007 | 0.008 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.007 | 0.006 | 0.005 | 0.006 | 0.007 | 0.008 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.007 | 0.006 | 0.005 | 0.006 | 0.007 | 0.008 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.007 | 0.006 | 0.005 | 0.006 | 0.007 | 0.008 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.007 | 0.006 | 0.005 | 0.006 | 0.007 | 0.008 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.007 | 0.006 | 0.005 | 0.006 | 0.007 | 0.008 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.007 | 0.006 | 0.005 | 0.006 | 0.007 | 0.008 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |

| \( L \) | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| \( K \) | 0.005 | 0.003 | 0.001 | 0.000 | 0.005 | 0.007 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.003 | 0.001 | 0.000 | 0.005 | 0.007 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.003 | 0.001 | 0.000 | 0.005 | 0.007 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.003 | 0.001 | 0.000 | 0.005 | 0.007 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.003 | 0.001 | 0.000 | 0.005 | 0.007 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.003 | 0.001 | 0.000 | 0.005 | 0.007 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |
| \( K \) | 0.005 | 0.003 | 0.001 | 0.000 | 0.005 | 0.007 | 0.009 | 0.010 | 0.011 | 0.012 | 0.013 | 0.014 | 0.015 | 0.016 | 0.017 | 0.018 | 0.019 |

Table 1 Fourier coefficients of the half-wave cosine representation

\[
A_{kf} \text{ components } - \cos \frac{kf_{x}}{L} \cos \frac{ly}{R}
\]

\[
B_{kf} \text{ components } - \cos \frac{kf_{x}}{L} \sin \frac{ly}{R}
\]
### Table 2  Fourier coefficients of the half-wave sine representation

<table>
<thead>
<tr>
<th>C&lt;sub&gt;N&lt;/sub&gt; components</th>
<th>( \sin \frac{k \pi x}{L} \cos \frac{\pi y}{R} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( L )</td>
<td>0</td>
</tr>
<tr>
<td>( N )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>D&lt;sub&gt;N&lt;/sub&gt; components</th>
<th>( \sin \frac{k \pi x}{L} \sin \frac{\pi y}{R} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( L )</td>
<td>0</td>
</tr>
<tr>
<td>( N )</td>
<td></td>
</tr>
</tbody>
</table>
Fig. C5 Axial variation of the half-wave cosine Fourier representation

Fig. C6 Circumferential variation of the half-wave cosine Fourier representation
Fig. C7 Axial variation of the half-wave sine Fourier representation

Fig. C8 Circumferential variation of the half-wave sine Fourier representation
Fig. C9 Axial variation of the half-wave cosine axisymmetric Fourier representation

Fig. C10 Axial variation of the half-wave sine axisymmetric Fourier representation
APPENDIX D

Listing of the
Data Reduction programs
# Program xx00
#
# Program made by A.W.H.Klompe, May 1989
#
# Program designed to create a formatted dataset
#
# Rough data is given in a file named [shellcode].
# All needed input variables are read from the file
# /users/data/shellcode/ VAR, list1, list2, list3 and list4.
#
# Input variables in file /users/data/[shellcode] VAR
# list1:
# SHELLCODE - Identification code for used shell.
# MEASUREDATE - Date of imperfection measurement.
# NSCAN - Total number of scans,
# including number of missing scans,
# excluding number of control scans.
# NPOINT - Number of data points per scan.
# CONICITY - Cone angle of shell, measured in degrees,
# positive if R(top) < R(bottom).
# THICKNESS - Wall thickness of the shell (in mm).
# RADIUS - Radius of the shell (in mm).
# LENGTH - Length of the shell (in mm).
# SCANTYPE - =0 => Scans are oriented in axial direction.
# =1 => Scans are oriented in circum. direction.
# If SCANTYPE=1, hatches and extra plating can not be
# properly 'placed'.
# In that case write a small program to put the
# available data in axial representation first.
#
# list2:
# FIRSTSCAN - First scan number.
# SCANINCR - Increment in scan numbering.
# NMISSSCAN - Number of missing scans (maximum 20).
# NCONTSNAN - Number of control scans (maximum 10).
# NREFPOINT - Number of reference data points.
# REFCODE - Handling code reference data.
# =0 => 2 or 6 reference points - standard handling
# =1 => 6 reference points - user defined handling
# OFFSET - Offset value for the imperfections.
#
# list3:
# MISSINGSCAN - Array containing the scan numbers of the
# missing scans (maximum 20).
#
# list4:
# CONTROLSCAN - Array containing the scan numbers of the
# control scans (maximum 10).
#
# ARRAY:
# REF - Contains reference data (maximum 10).
# NSCAN - Contains scan (run) numbers.
# SCAN - Contains data without reference data.
#
fc -o /tmp/xx00 /users/datatred/odatatred/xx00format.o
/tmp/xx00

rm /tmp/xx00

# End of program
program xx00format
  Program made by A.W.H. Klop, may 1989.
  This program creates the formatted dataset on file
  [shellcode].
  Unformatted data input is in file [shellcode].
  Program is called by xx00.

******************************************************************************
  Dimension:
  ONESCANT,POINT+1),SCAN(POINT,NSCAN)
  REF(POINT,NSCAN),SCANR(POINT)
******************************************************************************
  implicit none
  real*4 ONESCANT(410),SCAN(400,400),REF(10,400)
  real*4 CONIENCY,THICKNESS,RADIUS,LENGTH,AVERAGE
  integer*2 MISSSCANS(20),CONTROLSCAN(10),SCANR(400)
  integer*2 REFCODE,IOS,1,K,NSCAN,POINT,FIRSTSCAN
  integer*2 SCANNCR,NMISSCANS,NCONTSCAN,REPPOINT,SCANTYPE
  character*1 REP
  character*14 NFIL
  character*25 SHELLCODE,MISSEDDATE
  character*31 FILE,FILEVAR,FILEB_OUT,FILEB
******************************************************************************
  Reading the name of the shell to be processed.
  Opening of the files [shellcode],[shellcode]var,[shellcode]b and
  [shellcode]b.out.
******************************************************************************
  write(*,101)
  10 write(*,102)
    I=1
    read(*,103) NFIL
    do while(.char(NFIL(I:I)).ge.33)
      I=I+1
    end do
    FILE = '/users/data'/NFIL(1:1-1)
    FILEVAR = '/users/data'/NFIL(1:1-1)/'var'
    FILEB_OUT = '/users/data'/NFIL(1:1-1)/'b_out'
    FILEB = '/users/data'/NFIL(1:1-1)/'b'
    write(*,106) FILE
    write(*,107) FILEVAR
    write(*,108) FILEB_OUT
    write(*,109) FILEB
    write(*,104)
    read(*,105) REP
    if (REPLY.eq.'n'.or.REPLY.eq.'N') goto 10
    if (REPLY.ne.'y'.and.REPLY.ne.'Y') stop
    write(*,90)
    write(*,91) "EXECUTING xx00"
    write(*,91) "EXECUTING xx00"
open (10,status='old',file=FILE ,iostat=I0S)
if (I0S.ne.0) then
   print 111,FILE
   goto 10
end if
open(15,status='old',file=FILEVAR )
open(16,status='unknown',file=FILEB_OUT)
open(20,status='unknown',file=FILEB )

50 format(10(/))
91 format(15x,A40)
101 format(50(/))
102 format(/*Enter the name of the shell to be processed :',"$"))
103 format(A14)
104 format(/'Are the right files selected? y(es),n(o) or s(top) :',"$")
105 format(A1)
106 format(/'Unformatted data is read in from file : ',a27)
107 format(/'All needed variables are read from file : ',a27)
108 format(/'Intermediate results and other extra output : ',a27)
109 format(/'Formatted data is written to file : ',a27)
111 format(/'Error: file '','a27,' not found')

*****************************************************************************
* input variables
*****************************************************************************
     namelist /list1/ SHELLCODE,MEASUREDATE,NSCAN,NPOINT,CONICITY,THIC
     KNESS,RADIUS,LENGTH,SCANTYPE
     namelist /list2/ FIRSTSCAN,SCANINCR,NMISSCAN,NCONTSCAN,NREFPOINT,
     BREFCODE
     namelist /list3/ MISSINGSNAN
     namelist /list4/ CONTROLSCAN
     read(15,list1)
     write(16,list1)
     read(15,list2)
     write(16,list2)
     if (NMISSCAN .ne. 0) then
       read(15,list3)
     endif
     if (NCONTSCAN .ne. 0) then
       read(15,list4)
     endif
     close (15)
     write(16,220) FILE

*****************************************************************************
* reading input data
*****************************************************************************
do 60 J=1,NSCAN+NCONTSCAN
   do I=1,NMISSCAN
      if (((FIRSTSCAN+(J-1)*SCANINCR).eq.MISSINGSNAN(I)) then
         SCAN(J)=MISSINGSNAN(I)
      do K=1,NREFPOINT
         REF(K,J)=0.0
      end do
   do K=1,NPOINT
      SCAN(K,J)=.9999
end do
write(16,200) SCANNR(J)
goto 60
endif
end do
read(10,100) (ONESCAN(I),I=1,NPOINT+NREFPOINT+1)
SCANNR(J)=ifix(ONESCAN(I))
if (J.eq.NSCAN .and. SCANNR(J).ne.SCANNR(J-1)+SCANINCR) then
  if (J.gt.1) then
    write(16,210) SCANNR(J)
    write(*,210) SCANNR(J)
    stop
  endif
endif
if (REFCODE.eq.0) then
  if (NREFPOINT.gt.0) then
    do 30 I=1,NREFPOINT
      REF(I,J)=ONESCAN(I+1)
    continue
  else
    stop
  endif
else if (REFCODE.eq.1) then
  * user defined reference data handling
  REF(1,J)=-ONESCAN(2)
  REF(2,J)=0.000
  REF(3,J)=ONESCAN(2)
  REF(4,J)=0.000
  REF(5,J)=ONESCAN(6)
  REF(6,J)=ONESCAN(7)
else
  write(16,230) REFCODE
  stop
endif
*******************************************************************************
* 'stop' can be replaced by a userdefined refmode handler.
*******************************************************************************
end if
K=NREFPOINT+2
do I=1,NPOINT
  SCAN(I,J)=ONESCAN(K)
  K=K+1.
end do
60 continue
*******************************************************************************
* Estimation of average of imperfection data.
** Subtracting this average from data.
*******************************************************************************
AVERAGE=0
do J=1,NSCAN
  do I=1,NPOINT
    AVERAGE=AVERAGE+SCAN(I,J)
  end do
end do
AVERAGE=(AVERAGE-NMISSCAN*NPOINT*99.999)/((NSCAN-NMISSCAN)*NPOINT)
write(16,120) AVERAGE
do J=1,NSCAN
   do I=1,NPOINT
      SCAN(I,J)=SCAN(I,J)-AVERAGE
   end do
end do

******************************************************
* output to unit 20
******************************************************

   do J=1,NSCAN+NCONTSCAN
      write(20,110) SCANNR(J)
      write(20,100) (REF(I,J),I=1,1,REFPOINT)
      write(20,100) (SCAN(I,J),I=1,NPOINT)
   end do
   close (16)
close (20)
stop

100 format (8f9.5)
110 format (i5)
120 format (/,'Estimated average ',f8.5,' is subtracted from data')
200 format (5x,'scan nr. :',i5,' not found'
210 format (5x,'error: wrong scan nr.',i5,5x,'unformatted datafile probably damaged.')
220 format (//x,'input data :''a27')
230 format (5x,'error: unknown refmode nr.',i5/)
end
This program makes corrections for:
- translation of the shell on basis of the reference measurements
- imperfections of the beam
- missing scans
- hatches
- rivets
- extra plating

Missing scans:
The missing scans are added by linear interpolation.
The successor and predecessor of the missing scans are used for this interpolation.

Hatches:
The data for the hatch areas is set to 99.999.
In a 3-d plot a gap will appear at the location of a hatch, because 99.999 is too high a value to be plotted.

Rivets:
The rivets are corrected by linear interpolation in axial direction.
The interpolation takes place from a point before the rivet to a point after the rivet, except for the rivets at the beginning or end of an 'axial scan'.

Extra plating:
This correction is done by subtracting the plate thickness from the data.

Program history and the extra output can be found in the file /users/data/[shellcode]c_out.

This program uses the procedures:
XX01MCORRE , XX01CORRECT

INPUT VARIABLES in file /users/data/[shellcode]var:

SHELLCODE - Identification code for used shell.
MEASUREDATE - date of imperfection measurement.
NSCAN - total number of scans,
including number of missing scans,
excluding number of control scans.
NPOINT - number of data points per scan.
CONICITY - conicity of shell, deviation from cylinder,
measured in degrees, positive if R(top) < R(bottom).
THICKNESS - wall thickness of the shell (in mm).
RADIUS - radius of the shell (in mm).
LENGTH - length of the shell (in mm).
SCANTYPE = 0 => scans are oriented in axial direction.
1 => scans are oriented in circumf. direction.
If SCANTYPE=1, hatches and extra plating can not be
properly 'placed'.
In that case write a small program to put the
available data in axial representation first.

# list2:
  # FIRSTSCAN - first scan number.
  # SCANINC - increment in scan numbering.
  # NMISSCAN - number of missing scans (maximum 20).
  # NCONTS - number of control scans (maximum 10).
  # NREFPO - number of reference data points.
  # REFN - handling code reference data.
  # =0 -> 6 reference points - standard handling.
  # =1 -> 6 reference points - user defined handling.
  # =2 -> 2 reference points - standard handling (not
implemented yet).

# list3:
  # MISC - array containing the scan numbers of the
  # missing scans (maximum 20).

# list5
  # EXTRAOUT = no extra output from correction procedure.
  # =1 -> extra output.
  # PEAKMAX - Each axial scan is approximated by calculating its
  # first NCOEF one-dimensional fourier coefficients.
  # All data on the actual scan that deviates more than
  # PEAKMAX from the approximation is then corrected.
  # Recommended for Ariane-shells: PEAKMAX=0.6-0.8 mm.
  # MAX - all correct data in the b-file must be between
  # MAX and -(MAX).
  # If MAX is chosen much too large, correction will be
  # inaccurate, MAX must not be chosen too small !!
  # (All data above MAX or below -(MAX) is made equal to
  # the nearest data (in axial direction) not greater
  # than MAX, before fourier analysis is performed!)
  # For Ariane-shells: MAX = 3.0-4.0 mm.
  # NCOEF - Number of fourier coefficients used to approximate
  # each scan (usually 4 is sufficient).
  # This approximation is used for peakcorrection.
  # If NCOEF is chosen too high, the peaks which should
  # be corrected have too much effect on the
  # 'approximation-scan', so that correction will not
  # be sufficient.
  # NSCANPLATING - number of scans with extra plating.
  # NSCANHATCH - number of scans with hatches.
  # BEAMCORRECT - =0 -> no beam correction needed.
  # =1 -> beam correction has to be done.

# list6
  # FIRSTPLATING - axial datapoint, begin of extra plating.
  # NLASTPLATING - axial datapoint, end of extra plating.
  # PLATEH - thickness of extra plating.
  # ARRAYPLATING - array containing the numbers of the scans which need
  # correction for extra plating.

# list7
Mar 27 07:50 1990  xx01 Page 3

# NFIRSTHATCH - axial datapoint, begin of hatch.
# NLASTHATCH  - axial datapoint, end of hatch.
# ARRAYHATCH  - array containing the numbers of the scans with
#                hatches, including missing scans.

fc -o /tmp/x1 /users/datared/odatared/xx01mcorre.o /users/datared/odatared/x
/tmp/x1

rm /tmp/x1

# End of program
program xx0lmcorre

**********************************************************************
* Program modifications done by R.W.H. Klompe, May 1989
* Further modifications done by H.J.C. van der Hoeven, September 1989.
*
* Correction program for imperfection data
*
* Program is called by xx0l

**********************************************************************

implicit none
real*4 PLT(2,400),REF(10,400),W(400,400),B(400,400)
real*4 BEAM(400),ARR(10),X(400,400)
real*4 THICKNESS,RADIUS,LENGTH,CONICITY,PEAKMAX,MAX
real*4 PLATETHICKNESS,PLTX,PLTY,CC,SLICE,DIFFERENCE
integer*2 MISSINGSCAN(25),SCANARRAY(400),ARRAYPLATING(50)
integer*2 ARRAYHATCH(50)
integer*2 I,J,IOS,NPOINT,EXTRAOUT,SCANTYPE,NLASTHATCH,N
integer*2 NSCAN,NSCANPLATING,NSCANHATCH,NCOEF,BEGIN,END,ERROR
integer*2 NFIRSTPLATING,NLASTPLATING,FIRSTSCAN,SCANIINC,NMISSCAN
integer*2 NCONTSCAN,NREFPOINT,REFCODE,BEAMCORRECT,NFIRSTHATCH
character*100 REPLY
character*14 FILE
character*25 SHELLCODE,MEASUREDATE
character*34 FILEB,FILEVAR,FILEC,FILEC_OUT,FILEBEAM
common /SCAN/ FIRSTSCAN,SCANINC,NMISSCAN,SCANARRAY,MISSINGSCAN
common /HATCH/ NSCANHATCH,NFIRSTHATCH,NLASTHATCH
common /DATA/ W,NPOINT,NSCAN

**********************************************************************
* Reading the name of the shell to be processed.
* Opening of the files [shellcode]b,[shellcode]var,
* [shellcode]out and, if needed, [shellcode]beam.
**********************************************************************

write(*,101)
10 write(*,102)
I=1
read(*,103) NFILE
do while(ichar(NFILE(I:I)).gt.33)
   I=I+1
end do
FILEB='/users/data/\'//NFILE(1:I-1)//'b'
FILEVAR='/users/data/\'//NFILE(1:I-1)//'var'
FILEC_OUT='/users/data/\'//NFILE(1:I-1)//'c_out'
FILEC='/users/data/\'//NFILE(1:I-1)//'c'
FILEBEAM='/users/data/\'//NFILE(1:I-1)//'beam'
write(*,106) FILEB
write(*,107) FILEVAR
write(*,108) FILEBEAM
write(*,109) FILEC_OUT
write(*,112) FILEC
write(*,104)
read(*,105) REPLY
if (REPLY.eq.'n'.or.REPLY.eq.'N') goto 10
if (REPLY.ne.'y'.and.REPLY.ne.'Y') stop
write(*,90)
write(*,91) "***************************************************************************"
write(*,91) "* EXECUTING *"
write(*,91) "* xx01 *
write(*,90) "***************************************************************************"
write(*,90)
on 10 open (10,status='old',file=FILEB,iostat=IOS)
if (IOS.ne.0) then
   print 111,FILEB
   goto 10
endif
open(15,status='old',file=FILEVAR)
open (16,status='unknown',file=FILEC_OUT)
on 20 if (BEAMCORRECT.eq.1) then
   open(17,status='old',file=FILEBEAM)
endif

90 format(10(/))
91 format(15x,A40)
101 format(50(/))
102 format(/,x,'Enter the name of the shell to be processed :',$,)
103 format(A14)
104 format(/,x,'Is the above data correct? y(es),n(no) or s(stop) :',$,)
105 format(A1l)
106 format(/,x,'Formatted data is read in from file : ',a27)
107 format(x,'All needed variables are read from file : ',a27)
108 format(x,'If needed, beamshape is read from file : ',a27)
109 format(x,'Intermediary results and other extra output : ',a27)
112 format(x,'Corrected data is written to file : ',a27)
111 format(x,'Error: file ',',a27,'not found')

***************************************************************************
* INPUT VARIABLES
***************************************************************************

name list /list1/ SHELLCODE,MEASUREDATE,NSCAN,NPOINT,CONICITY,THIC
KNESS,RADIUS,LENGTH,SCANTYPE
name list /list2/ FIRSTSCAN,SCANINC,NNISSCAN,NCONTSCAN,REFPOINT,
TREFFCODE
name list /list3/ MISSINGSCAN
name list /list5/ EXTRAO,PEAKMAX,MAX,NCOEF,NSCANNPLATING,
1NSCANNHATCH,BEAMCORRECT
name list /list6/ NFIRSTPLATING,NLASTPLATING,PLATETHICKNESS,
1ARRAYPLATING
name list /list7/ NFIRSTHATCH,NLASTHATCH,ARRAYHATCH
read(15,list1)
read(15,list2)
if (NNISSCAN.ne.0) read(15,list3)
read(15,list5)
if (NSCANNPLATING.ne.0) read(15,list6)
if (NSCANNHATCH.ne.0) then
   read(15,list7)
else
   NFIRSTHATCH=0
   NLASTHATCH=-1
endif
close (15)

******************************************************************************
* READ DATA SCAN
******************************************************************************
do J=1,NSCAN
   read(10,25) SCANARRAY(J)
   read(10,30) (REF(I,J),I=1,NREFPOINT)
   read(10,30) (W(I,J),I=1,NPOINT)
end do
close (10)

******************************************************************************
* CORRECTION FOR TRANSLATION OF THE SHELL
* calculate the offset in translation PLTX and PLTY
* PLT(1,J) translation in x-direction
* PLT(2,J) translation in y-direction
* CC is the cosine of the cone angle
******************************************************************************
if (NREFPOINT.eq.6) then
   PLTX= (REF(3,1)-REF(1,1))/2.0
   PLTY= (REF(2,1)-REF(4,1))/2.0
   do J=1,NSCAN
      PLT(1,J)= (REF(3,J)-REF(1,J))/2.0-PLTX
      PLT(2,J)= (REF(2,J)-REF(4,J))/2.0-PLTY
   end do
   CC= cos(4.0*CONICITY*atan(1.0)/180.0)
   do J=1,NSCAN
      do I=1,NPOINT
         W(I,J)= W(I,J)-PLT(1,J)*CC
      end do
   end do
endif

******************************************************************************
* READ BEAM CORRECTION
* CORRECT DATA FOR BEAM-SHAPE
******************************************************************************
if (BEAMCORRECT.eq.1) then
   read(17,30) (ARR(J),J=1,6),(BEAM(I),I=NPOINT,1,-1)
close(17)
write(16,145) (BEAM(I),I=1,NPOINT)
do J=1,NSCAN
   do I=1,NPOINT
      W(I,J)= W(I,J)+BEAM(I)
   end do
end do
endif

******************************************************************************
* CORRECTION EXTRA PLATING
******************************************************************************
do N=1,NSCANPLATING
   J=(ARRAYPLATING(N)-FIRSTSCAN)/SCANINCR+1
   do I=NFIRSTPLATING,NLASTPLATING

W(I,J)=W(I,J)-PLATETHICKNESS  
end do  
end do

*****************************************************************************
* CORRECTION MISSING SCANS
*****************************************************************************
write(16,130) SHELLCODE,MEASUREDATE  
if (NMISSCAN.ne.0) then  
call amis(EXTRAOUT1,NREFPOINT,REF)  
else  
write(16,4)  
endif

*****************************************************************************
* OUTPUT TO UNIT 16
*****************************************************************************
write(16,135) EXTRAOUT1,NSCAN,NPOINT,CONICITY  
write(16,140) NFIRSTHATCH,NLASTHATCH  
write (16,3) PEAKMAX  
write (16,5) NCOEF

if (NSCANPLATING.ne.0) then  
write(16,100)  
write(16,110) (ARRAYPLATING(J),J=1,NSCANPLATING)  
write(16,150) NFIRSTPLATING,NLASTPLATING,PLATETHICKNESS  
endif

if (NSCANHATCH.ne.0) then  
write(16,120)  
write(16,110) (ARRAYHATCH(J),J=1,NSCANHATCH)  
endif

if (NMISSCAN.ne.0) then  
write(16,155)  
write(16,110) (MISSINGSCAN(J),J=1,NMISSCAN)  
endif

if (EXTRAOUT1.eq.1) then  
do J=1,NSCAN  
do I=1,NPOINT  
   X(I,J)=W(I,J)  
end do  
end do
endif

*****************************************************************************
* INTERPOLATING OVER ALL THE HATCHES.
*****************************************************************************
if(NSCANHATCH .ne. 0) then  
do N=1,NSCANHATCH  
   J=(ARRAYHATCH(N)-FIRSTSCAN)/SCANINC+1  
   I=NFIRSTHATCH+1  
   SLOPE=abs(W(I,J)-W(I-1,J))  
do while (SLOPE.ge.(0.1).or.abs(W(I,J)).ge.(MAX))  
   I=I-1  
   SLOPE=abs(W(I,J)-W(I-1,J))  
   end do
BEGIN=I
I=NLASTHATCH+1
SLOPE=abs(W(I,J)-W(I+1,J))
do while (SLOPE.ge.(0.1).or.abs(W(I,J)).ge.(MAX))
   I=I+1
   SLOPE=abs(W(I,J)-W(I+1,J))
end do
END=I
do I=NFIRSTHATCH,NLASTHATCH
   W(I,J)=W(BEGIN,J)+(W(END,J)-W(BEGIN,J))*(I-BEGIN)/(END-BEGIN)
end do
end if

******************************************************************************
* PUTTING MATRIX W() IN 'AXIAL' REPRESENTATION                          *
* During correction the scans are always oriented                        *
* in axial direction.                                                   *
******************************************************************************
if (SCANTYPE.eq.1) then
   do I=1,NSCAN
      do J=1,NPOINT
         B(I,J)=W(J,I)
      end do
   end do
   N=NPOINT
   NPOINT=NSCAN
   NSCAN=N
   do I=1,NPOINT
      do J=1,NSCAN
         WI,J=BI,J
      end do
   end do
endif

******************************************************************************
* START CORRECTION                                     *
******************************************************************************
call correction(PEAKMAX,MAX,NCOEF)

******************************************************************************
* PUTTING MATRIX W() BACK IN ORIGINAL REPRESENTATION   *
******************************************************************************
if (SCANTYPE.eq.1) then
   do I=1,NPOINT
      do J=1,NSCAN
         BI,J=W(I,J)
      end do
   end do
   N=NPOINT
   NPOINT=NSCAN
   NSCAN=N
   do I=1,NPOINT
      do J=1,NSCAN
         WI,J=BI,J
      end do
   end do
end do
endif

******************************************************************************
* SETTING VALUES OF W() IN ALL HATCHES TO 99.999
* Due to the corrections performed in subroutine correction, the hatches will 'disappear', and
* therefore have to be 'regenerated'.
******************************************************************************
if(NSCANHATCH .ne. 0) then
   do N=1,NSCANHATCH
      J=(ARRAYHATCH(N)-FIRSTSCAN)/SCANINC+1
      do I=1,NFIRSTHATCH,NLASTHATCH
         W(I,J)=99.999
      end do
   end do
end if

******************************************************************************
* WRITE EXTRA OUTPUT
******************************************************************************
if (EXTRAOUT1.eq.1) then
   write (16,250)
   write(16,200)
   ERROR=0
   do J=1,NSCAN
      do I=1,NPOINTER
         DIFFERENCE=W(I,J)-X(I,J)
         if (DIFFERENCE .ne. 0 .and. W(I,J).lt.99) then
            ERROR=ERROR+1
            write(16,300) J,I,X(I,J),W(I,J)
         endif
      end do
   end do
   write(16,310) ERROR
end if

******************************************************************************
* WRITE DATA SCAN
******************************************************************************
do J=1,NSCAN
   write(20,25) SCANARRAY(J)
   write(20,30) (W(I,J),I=1,NPOINTER)
end do

close (16)
close (20)
stop

1 format (/,'Average of imperfection data is ',F9.5)
2 format (6X,'Standard deviation is ',F9.5)
3 format (6X,'PEAKMAX is ',F9.5)
4 format ('No missing scans present'/)
5 format (6X,'NCOEF is ',I3,)
25 format (I5)
30 format (BF9.5)
100  format (/'ARRAY ARRAYPLATING'/)
110  format (10I5)
120  format (/'ARRAY ARRAYHATCH'/)
130  format (/5X,1A25/5X,1A25)
135  format (/5X,'INPUT DATA :',6X,'EXTRAOUT1',T40,=' ',I6/
16X,'NUMBER OF SCANS',T40,=' ',I6/
26X,'NUMBER OF DATA POINTS PER SCAN',T40,=' ',I6/
36X,'CONE ANGLE',T40,=' ',F9.5)
140  format (/6X,'AXIAL DATAPoint. BEGIN O'F HATCH',T40,=' ',I6/
16X,'AXIAL DATAPoint. END OF HATCH',T40,=' ',I6/
145  format (/5X,'BEAM CORRECTION (FROM TOP TO BOTTOM) :'/
1(6X,0F9.5))
150  format (/6X,'BEGIN OF PLATE ',T40,=' ',I6/
16X,'END OF PLATE ',T40,=' ',I6/
26X,'THICKNESS OF PLATE',T40,=' ',F9.5/)
155  format (/,'ARRAY MISSINGSCAN'/)
200  format(/,'SCAN POINT VALUE BEFORE AND AFTER CORRECTION',/)
250  format(/,' CORRECTIONS PERFORMED '/)
300  format(I4,5X,I4,4X,F9.5,7X,F9.5)
310  format(/,'Total number of corrected errors: ',I5)
      end
Program part xx01correct.f

Subroutines used for correction of imperfection data
(so called ARIANE data).

The routines are called by 'xx01mcorre'.

Subroutine amis(EXTRAOUT1,NREFPOINT,REF)


Implicit none

Real*4 W(400,400),REF(10,400),XX
Integer*2 SCANARRAY(400),ARRAYHATCH(50),MISSINGSCAN(25)
Integer*2 NPOINT,NREFPOINT,EXTRAOUT1,NFIRSTHATCH,NLASTHATCH
Integer*2 NSCAN,FIRSTSCAN,SCANINCR,NMISSSCAN,SCANHATCH,NL,NH
Integer*2 I,J,K,N,MIS,SCANLOW,SCANHIGH,LFLAG
Common /SCAN/ FIRSTSCAN,SCANINCR,NMISSSCAN,SCANARRAY,MISSINGSCAN
Common /HATCH/ SCANHATCH,NFIRSTHATCH,NLASTHATCH
Common /DATA/ W,NPOINT,NSCAN

Searching predecessors and
successors of the missing scans.

LASTSCAN=FIRSTSCAN+(NSCAN-1)*SCANINCR
Do 50,k=1,NMISSSCAN
MIS=MISSINGSCAN(K)
If (MIS.eq.FIRSTSCAN) then
SCANLOW=LASTSCAN
SCANHIGH=FIRSTSCAN+SCANINCR
Else if (MIS.eq.LASTSCAN) then
SCANLOW=LASTSCAN-SCANINCR
SCANHIGH=FIRSTSCAN
Else
SCANLOW=MIS-SCANINCR
SCANHIGH=MIS+SCANINCR
Endif
I=0
Do While (I.eq.0)
Do J=K+1,NMISSSCAN
If (SCANLOW.eq.MISSINGSCAN(J)) then
I=1
Endif
End Do
If (I.eq.1) then
SCANLOW=SCANLOW-SCANINCR
I=0
If (SCANLOW.lt.FIRSTSCAN) then
SCANLOW=LASTSCAN
Endif
Else
I=1
endif
end do
I=0
do while (I.eq.0)
do J=K+1,NMISSCAN
   if (SCANHIGH.eq.MISSINGSCAN(J)) then
      I=1
   endif
end do
if (I.eq.1) then
   SCANHIGH=SCANHIGH+SCANINCR
   I=0
   if (SCANHIGH.gt.LASTSCAN) then
      SCANHIGH=FIRSTSCAN
   endif
else
   I=1
endif
end do
if (SCANLOW.gt.SCANHIGH) then
   N=(SCANHIGH+NSCAN+SCANINCR-SCANLOW)/SCANINCR
else
   N=(SCANHIGH-SCANLOW)/SCANINCR
endif
LFLAG=0
do J=1,NSCANHATCH
   if (SCANLOW.eq.ARRAYHATCH(J)) then
      LFLAG=LFLAG+1
   endif
   if (SCANHIGH.eq.ARRAYHATCH(J)) then
      LFLAG=LFLAG+2
   endif
end do

*****************************************************************************
* interpolation of datapoints for missing scans  
*****************************************************************************
J=(MIS-FIRSTSCAN)/SCANINCR+1
NL=(SCANLOW-FIRSTSCAN)/SCANINCR+1
NH=(SCANHIGH-FIRSTSCAN)/SCANINCR+1
SCANARRAY(J)=MIS
do I=1,NPOINT
   W(I,J)=W(I,NL)+(W(I,NH)-W(I,NL))/N
end do

*****************************************************************************
* If LFLAG=1, scan NL crosses a hatch.
* scan MIS is now interpolated from five points before to five points 
* after the hatch, so that scan MIS 'follows' scan NH.
* If LFLAG=2, scan NH crosses a hatch.
* scan MIS is now interpolated from five points before to five points 
* after the hatch, so that scan MIS 'follows' scan NL.
* If LFLAG=3, both scan NL and scan NH cross the same hatch.
* No extra step is necessary now, because linear interpolation 
* between SCANLOW and SCANHIGH automatically extends the hatch over
* the missing scan(s).

if (LFLAG.eq.1) then
  do I=NFIRSTHATCH-5,NLASTHATCH+5
    XX=0.5-0.5*(2*I-NLASTHATCH-NFIRSTHATCH)/(NLASTHATCH-10)
    W(I,J)=W(NFIRSTHATCH-5,J)+XX*(W(NLASTHATCH+5,J)+W(I,NH)-
    W(NFIRSTHATCH-5,J)-W(NLASTHATCH+5,NH))
  end do
endif

if (LFLAG.eq.2) then
  do I=NFIRSTHATCH-5,NLASTHATCH+5
    XX=0.5-0.5*(2*I-NLASTHATCH-NFIRSTHATCH)/(NLASTHATCH-10)
    W(I,J)=W(NFIRSTHATCH-5,J)+XX*(W(NLASTHATCH+5,J)+W(I,NL)-
    W(NFIRSTHATCH-5,J)-W(NLASTHATCH+5,NL))
  end do
endif

* output to unit 16 after interpolation of the missing scan.

write(16,150) SCANARRAY(J)
write(16,*)
60 continue
return

150 format(/5X,'AMIS IS CALLED FOR SCAN:',I5)
end

* End of routine amis

*-------------------------------------------------

subroutine correction(PEAKMAX, MAX, NCOEF)

* Subroutine written by H.J.C. van der Hoeven, September 1989.

* This subroutine corrects rivets, inaccuracies around hatches etc.
* In this subroutine, the scans are always oriented in axial direction.
* This is accomplished in the main program xx01mcorre.f by swapping W(I,J) with W(J,I) if the scans were oriented in circumferential direction originally.
* This is done because the imperfections in most shells vary much more in circumferential direction than in axial direction, so correction in axial direction is likely to be more accurate.

implicit none
real*4 W(400,400),WW(0:401),AI(0:50),BI(0:50)
real*4 PEAKMAX,MAX,PI,SUM1,SUM2,ARG,XX
integer*2 I,I1,I2,K,N,NPOINT,NSCAN,NCOEF,BEGIN,END
common /DATA/ W,NPOINT,NSCAN
PI=4.0*atan(1.0)
do J=1,NSCAN
  do I=1,NPOINT
WW(I)=W(I,J)
end do
WW(0)=WW(1)
WW(-1)=WW(1)
do I=1,(NPOINT/2)+1
  if (abs(WW(I)).gt.MAX) then
    I1=I+2
    K=0
    do while ((abs(WW(I1)).gt.MAX.or.abs(WW(I1-1)).gt.MAX).
      .and.K.eq.0)
      I1=I1+1
      if (I1.eq.NPOINT-10) then
        K=1
      endif
    end do
    WW(I)=WW(I1)
  endif
end do
do I=NPOINT,(NPOINT/2)-1,-1
  if (abs(WW(I)).gt.MAX) then
    I1=I-2
    K=0
    do while ((abs(WW(I1)).gt.MAX.or.abs(WW(I1+1)).gt.MAX).
      .and.K.eq.0)
      I1=I1-1
      if (I1.eq.10) then
        K=1
      endif
    end do
    WW(I)=WW(I1)
  endif
end do
do N=0,NCOEF
  SUM1=0.5*WW(1)+0.5*WW(NPOINT)*cos(N*PI)
  SUM2=0
  do I=2,NPOINT-1
    ARG=N*PI*(I-1)/(NPOINT-1)
    SUM1=SUM1+WW(I)*cos(ARG)
    SUM2=SUM2+WW(I)*sin(ARG)
  end do
  A(N)=SUM1/(NPOINT-1)
  B(N)=SUM2/(NPOINT-1)
end do
do I=1,NPOINT
  WW(I)=A(0)/Z.0
  do N=1,NCOEF
    ARG=N*PI*(I-1)/(NPOINT-1)
    WW(I)=WW(I)+A(N)*cos(ARG)+B(N)*sin(ARG)
  end do
end do
end do
N=1
K=0
do while (K.eq.0)
  do while (abs(W(N,J)-WW(N)).lt.PEAKMAX.and.K.eq.0)
    if (N.eq.NPOINT) then
      K=1
    endif
  end do
else
    N=N+1
endif
end do
if (K.eq.1) then
    BEGIN=NPOINT+1
else
    BEGIN=N
endif
do while (abs(W(N,J)-WW(N)).ge.PEAKMAX.and.K.eq.0)
    if (N.eq.NPOINT) then
        K=1
    else
        N=N+1
    endif
end do
if (K.eq.1) then
    END=NPOINT+1
else
    END=N
endif
if (BEGIN.eq.1.and.END.lt.NPOINT+1) then
    do I=1,END-1
        W(I,J)=W(END,J)+WW(I)-WW(END)
    end do
else if (BEGIN.gt.1.and.END.lt.NPOINT+1) then
    do I=BEGIN,END-1
        XX=0.5-0.5*(Z*BEGIN+1-END)/(BEGIN-1-END)
        W(I,J)=W(BEGIN-1,J)+WW(I)-WW(BEGIN-1)
        +XX*(W(END,J)-W(BEGIN-1,J)+WW(BEGIN-1)-WW(END))
    end do
else if (BEGIN.gt.1.and.BEGIN.le.NPOINT.and.K.eq.1) then
    do I-BEGIN,NPOINT
        W(I,J)=W(BEGIN-1,J)+WW(I)-WW(BEGIN-1)
    end do
endif
end do
end
return
Mar 27 08:02 1990 xx02 Page 1

# Program xx02
#
# program made by A.W.H. Klompe, May 1989.
#
# Program to calculate the imperfections w.r.t. the bestfit
cylinder (or cone).
#
# DATA INPUT on unit 10.
# DATA OUTPUT on unit 20.
# INPUT VARIABLES on unit 15.
#
# INPUT VARIABLES in file /users/data/[shellcode]var:
#
# list1
# SHELLCODE - identification code for used shell.
# MEASUREDATE - date of imperfection measurement.
# NSCAN - total number of scans,
# including number of missing scans,
# excluding number of control scans.
# NPOINT - number of data points per scan.
# CONICITY - cone angle of shell, measured in degrees,
# positive if R(top) < R(bottom).
# THICKNESS - wallthickness of the shell (in mm).
# RADIUS - radius at the bottom of the shell (in mm).
# LENGTH - length of the shell (in mm).
#
# list8
# EXTRAOUT2 - =1 -> extra output from least-squares fit procedure
# =0 -> no extra output.
# INITIALGUESS - =0 -> no initial guesses (see list9).
# =1 -> initial guesses.
# list9
# ARRAYINITIAL(1) - initial guess for eccentricity in X-direction.
# ARRAYINITIAL(2) - initial guess for eccentricity in Y-direction.
# ARRAYINITIAL(3) - initial guess for cos(alpha).
# ARRAYINITIAL(4) - initial guess for cos(beta).
# ARRAYINITIAL(5) - initial guess for cos(gamma).
# ARRAYINITIAL(6) - initial guess for radius of shell at Z=0.
#
# This program uses the procedures:
# MDATA, LSQ.
#
fc -o /tmp/xx02 /users/datared/odatared/xx02mdata.o /users/datared/odatared/
/tmp/xx02
rm /tmp/xx02
#
# End of program
program xx02mdatof
*******************************************************************************
* Program modifications done by A.W.H. KIompe, may 1989
* Program is called by xx02
*******************************************************************************
*******************************************************************************
* DIMENSION W(NSCAN,NPOINT),SCAN(NSCAN,NP),SCANNUMBER(NSCAN)
* DIMENSION X(NSCAN,NPOINT),Y(NSCAN,NP),Z(NSCAN,NP),WI(NSCAN,NP)
*******************************************************************************
implicit none
real*4 W(400,400),SCAN(400,400),SCANNUMBER(400)
real*4 X(1500000),Y(1500000),Z(1500000),WI(1500000)
real*4 LENGTH,CONICITY,THICKNESS,RADIUS,P1,CC,SC,STEP,AXST,TETA
real*4 DC,DS,RR
real*4 ARRAYINITIAL(6)
integer*4 I,IOS,NSCAN,NPOINT,EXTRAOUT,INITIALGUESS,J,ICOR,NN
integer*2 INS,INP,SCANTYPE
character*25 SHELLCODE,MEASUREDATE
character*34 FILEC,FILEVAR,FILEF_OUT,FILEF
character*14 REPLY
character*14 NFSFILE
common /COORD/ X,Y,Z,WI
common /ITEXT/ SHELLCODE,MEASUREDATE
*******************************************************************************
* Reading the name of the shell to be processed.
* Opening of the files [shellcode]c,[shellcode]var,[shellcode]f and
* [shellcode]f_out.
*******************************************************************************
write(*,101)
10 write(*,102)
I=1
read(*,103) NFSFILE
do while (ichar(NFSFILE(1:1)).gt.33)
  I=I+1
end do
FILEC='/users/data//'//NFSFILE(1:I-1)//'c'
FILEVAR='/users/data//'//NFSFILE(1:I-1)//'var'
FILEF_OUT='/users/data//'//NFSFILE(1:I-1)//'f_out'
FILEF='/users/data//'//NFSFILE(1:I-1)//'f'
write(*,106) FILEC
write(*,107) FILEVAR
write(*,108) FILEF_OUT
write(*,109) FILEF
write(*,104) read(*,105) REPLY
if (REPLY.eq.'n'.or.REPLY.eq.'N') goto 10
if (REPLY.ne.'y'.and.REPLY.ne.'Y') stop
write(*,90)
write(*,91) "EXECUTE" .xx02"
write(*,91) "*****************************************************************************
write(*,90)
open (10,status='old',file=FILEC,iostat=IOS)
if (IOS.ne.0) then
   print 111,FILEC
   goto 10
endif
open(15,status='old',file=FILEVAR)
open(16,status='unknown',file=FILEF_OUT)
open(20,status='unknown',file=FILEF)

90 format(10(/))
91 format(15x,A40)
101 format(50(/))
102 format(/,x,'Enter the name of the shell to be processed : ',$)
103 format(A14)
104 format(/,x,'Is the above data correct? y(es),n(no) or s(top) : ',$)
105 format(A1)
106 format(/,x,'Corrected data is read in from file : ',a27)
107 format(x,'All needed variables are read from file : ',a27)
108 format(x,'Intermediary results and other extra output : ',a27)
109 format(x,'Best-fitted data is written to file : ',a27)
111 format(x,'Error: file ',a27,' not found')

*****************************************************************************
* input variables
*****************************************************************************
namelist /list1/ SHELLCODE,MEASUREDATE,NSCAN,NPOINT,CONICITY,
   THICKNESS,RADIUS,LENGTH,SCANTYPE
namelist /list8/ EXTRAOUTZ,INITIALGUESS
namelist /list9/ ARRAYINITIAL
read(15,list1)
read(15,list8)
write(16,130) SHELLCODE,MEASUREDATE
if (INITIALGUESS.eq.1) read(15,list9)
write (16,100) NPOINT,NSCAN,EXTRAOUTZ,RADIUS,CONICITY,LENGTH
if (INITIALGUESS.eq.1) write (16,110) (I,ARRAYINITIAL(I),I=1,6)

*****************************************************************************
* READ DATA SCANS FROM UNIT 10  *
*****************************************************************************
do 30 I=1,NSCAN
   read (10,3) SCANNUMBER(I)
   read (10,2) (SCAN(I,J),J=1,NPOINT)
30 continue
C
C*****************************************************************************
C*  CALCULATE COORDINATES X,Y and Z FOR LSQ  *
C*****************************************************************************
PI=4.0*atan(1.0)
CONICITY=CONICITY*PI/180.0
CC=cos(CONICITY)
SC=sin(CONICITY)
STEP=2.0*PI/float(NSCAN)
AXST=LENGTH/float(NPOINT-1)
ICOR=201
if (SCANNUMBER(1).lt.200) then
    ICOR=1
endif
NN=0
do 50 I=1,NSCAN
   TETA=FLOAT(SCANNUMBER(I)-ICOR)*STEP
   DC=cos(TETA)
   DS=sin(TETA)
   do 40 J=1,NPOINT
      if (SCAN(I,J).gt.98.0) goto 40
      NN=NN+1
      Z(NN)=FLOAT(J-1)*CC*AXST+SCAN(I,J)*SC
      RR=RADIUS-tan(CONICITY)*Z(NN)+SCAN(I,J)/CC
      X(NN)=RR*DC
      Y(NN)=RR*DS
   40 continue
50 continue

*****************************************************************************
*    FIND BEST-FIT SHELL            *
*****************************************************************************
call LSQ(NN,RADIUS,CONICITY,ARRAYINITIAL,EXTRADOUTZ,INITIALGUESS)

*****************************************************************************
*    CALCULATE NEW IMPERFECTIONS    *
*    W.R.T. BEST FIT SHELL          *
*    POSITIVE OUTWARD               *
*****************************************************************************
   ins=1
   inp=1
do 60 I=1,NN
   if (SCAN(INS,INP).lt.98.0) goto 52
   W(INS,INP)=99.999
   INP=INP+1
   goto 51
52   W(INS,INP)=WI(I)*CC
   INP=INP+1
   if (INP.eq.NPOINT) goto 60
   INS=INS+1
   INP=1
60 continue

*****************************************************************************
*    WRITE IMPERFECTIONS TO OUTPUT UNIT 20     *
*****************************************************************************
do 70 I=1,NSCAN
   write(20,3) SCANNUMBER(I)
   write(20,2) (W(I,J),J=1,NPOINT)
70 continue
stop
2 format (8F9.5)
3 format (15)
100 format (/5X,'INPUT DATA',/6X,'NPOINT =',IS,T30,/6X,'NSCAN =',IS/
    16X,'EXTRADOUTZ =',IS//
    26X,'INITIAL GUESS RADIUS =',F12.3/
    36X,'INITIAL GUESS CONE-ANGLE DELTA = ',F9.3/)
46X,'LENGTH OF SCAN' =',f12.3//')
110 format(/5x,'Initial guess :'/,5x,'Arrayinitial(II)= ',f12.5))
130 format(/5x,1A25/5x,1A25)
end
Program part xx02lsq.f

SUBROUTINE SEARCHES BEST-FIT SHELL USING A LEAST SQUARE METHOD.
RESIDUE IS THE DIFFERENCE IN RADIUS OF CIRCLE AT Z = 0.0
SUBROUTINE CALCULATES:

1 TRANSLATION OF ORIGIN:
   E(1) - X DIRECTION
   E(2) - Y DIRECTION

2 INCLINATION OF CENTERLINE SHELL
   E(3) - C(Z',X)
   E(4) - C(Z',Y)

3 CONE ANGLE
   E(5) - TAN(Z',S)

4 BEST FIT RADIUS
   E(6)

E()=ARRAYINITIAL() !!!

This program is called by xx02 .

******************************************************************************
subroutine lsq (NP0INT,RADIUS,CONICITY,E,EXTRAROUTZ,INITIALGUESS)
implicit none
real*4 A(6,6),B(6),C(6),E(6),P(6),X(160000),Y(160000),Z(160000)
real*4 EPS,RADIUS,CONICITY,SS,SI,S2,S3,S4,S5,SI1,SI2,SI3,SI4,SI5,SI6
integer*2 N,MAX,INEW,INITIALGUESS,NPOINT,EXTRAROUTZ,I,J,K,IT,ITEL
common /COORD/ X,Y,Z
common /EPS/ EPS
N=6
MAX=10
EPS=1.0E-7
INEW=0

******************************************************************************
* SET INITIAL GUESS *
******************************************************************************
if (INITIALGUESS.eq.1) goto 1
E(1)=0.0
E(2)=0.0
E(3)=0.0
E(4)=0.0
E(5)=tan(CONICITY)
E(6)=RADIUS
1 ITEL=0
2 ITEL=ITEL+1
SS=0.0
SI=sqrt(1.0-E(3)*E(3)-E(4)*E(4))
S2=1.0+E(5)*E(5)
S3=1.0-E(3)*E(3)*S2
Oct 18 14:41 1989 xx021sq.f Page 2

S4=1.0-E(4)*E(4)*S2
S5=E(3)*E(4)*S2
B1=2.0*E(5)*E(3)
B2=2.0*E(5)*E(4)
do 4 I=1,N
   do 3 J=1,N
      A(I,J)=0.0
   3 continue
   B(I)=0.0
4 continue

do 20 K=1,NPOINT
   B15=-2.0*(X(K)*E(3)+Y(K)*E(4)+Z(K)*S1-E(1)*E(3)-E(2)*E(4))
   B1=B15*E(5)
   C1=-((X(K)*X(K)*S3+Y(K)*Y(K)+Z(K)*S4+Z(K)*E((3)*E(3)+E(4)*E(4))
   S2-E(5)*E(5)+2.0*X(K)*Y(K)+S5+2.0*X(K)*Z(K)*(-E(3)*S1+S2)
   +2.0*Y(K)*E(K)*E(5)-S1+S2)
   3-2.0*X(K)*E(K)+S5-2.0*Y(K)*E(K)*E(5)-S4+S5+2.0*E(K)*S1+S2
   4-2.0*X(K)*E(K)+E(3)*E(3)+S2-2.0*X(K)*E(K)*E(5)-E(4)*S1+S2
   S6=E(2)*E(5)+S4+2.0*E(1)*E(2)*E(5)
   B13=-2.0*E(5)*X(K)*Z(K)*E(3)/S1-E(1))
   B14=-2.0*E(5)*Y(K)*Z(K)*E(4)/S1-E(2)
   C12=2.0*X(K)*S5+Z(K)*E(4)*S1+S2-2.0*Y(K)*E(5)*E(1)*E(5)
   C13=2.0*X(K)*X(K)*E(3)*S2-Z(K)*Z(K)*E(3)*S2-Z(K)*Y(K)*E(4)*S2
   1+X(K)*Z(K)*S1-E(3)*E(3)/S1+S2-Z(K)*Y(K)*E(3)*E(3)+E(4)/S2
   2-2.0*X(K)*E(K)*E(1)*E(3)+S2-Z(K)*E(2)*E(4)+S2-Z(K)*E(K)*E(1)*E(4)/S2
   3+E(1)*E(3)*E(3)+S2-Z(K)*E(1)*E(3)+S1-E(3)*E(3)/S1+S2
   4+Z(K)*E(K)*E(2)*E(3)+E(4)/S2+S1-E(1)*E(2)*E(4)+E(4)/S2
   C14=2.0*Y(K)*Y(K)*E(4)+S2-Z(K)*Z(K)*E(4)*S2+X(K)*Y(K)*E(3)*S2
   1+X(K)*Z(K)*S1-E(3)*E(3)+S2+Y(K)*Z(K)*S1-E(4)/S1+S2
   2-Z(K)*E(K)*E(3)+E(3)+S2-Z(K)*E(K)*E(2)*E(4)+S2-Z(K)*E(K)*E(1)*E(4)/S2
   3+Z(K)*E(K)*E(3)+S2+S1-Z(K)*E(K)*E(2)*S1-E(4)/S1+S2
   4+Z(K)*E(K)*E(2)*E(4)+S2+S1-E(2)*E(3)+S2
   C15=2.0*X(K)*X(K)*E(3)*E(3)+S5+Y(K)*Y(K)*E(4)*E(4)+E(5)
   1+X(K)*S1+S1=E(5)2.0*X(K)*Y(K)*E(3)*E(4)/S1+E(5)
   2+2.0*X(K)*Z(K)*E(3)+E(5)+S1+S2-2.0*X(K)*Y(K)*E(4)*E(5)+S1
   3-2.0*X(K)*E(K)*E(3)*E(3)+E(4)+E(5)-2.0*X(K)*E(K)*E(1)*E(3)+E(3)+E(5)
   4-2.0*Y(K)*E(K)*E(3)+E(4)+E(5)-2.0*Y(K)*E(2)*E(4)+E(4)+E(5)
   5*(E(1)*E(3)+S2*E(5)-2.0*Z(K)*E(K)*E(3)+E(5)+S1)
   6-2.0*Z(K)*E(K)*E(4)+E(5)+S1+(E(2)*E(4)+E(4))*2+E(5)
   7+2.0*E(1)*E(2)*E(3)+E(4)+E(5))
   SQ=sqrt(B1*B1-4.0*CI)
   W(K)=0.5*(-B1+SQ)-E(6)

C

P(1)=0.5*(-B11+(B1*B1-2.0*CI1)/SQ)
P(2)=0.5*(-B12+(B1*B12-2.0*CI1)/SQ)
P(3)=0.5*(-B13+(B1*B13-2.0*CI1)/SQ)
P(4)=0.5*(-B14+(B1*B14-2.0*CI1)/SQ)
P(5)=0.5*(-B15+(B1*B15-2.0*CI1)/SQ)
P(6)=-1.0

do 10 I=1,N
   do 5 J=1,N
      A(I,J)=A(I,J)+P(I)*P(J)
   5 continue

-80-
B(I)=B(I)-W(K)*P(I)
10 continue
SS=SS+W(K)*W(K)
20 continue

call EQSOV (N,A,B,MAX,C,IT,INEW)
if (IT.eq.MAX) goto 40
if (EXTRAROUTZ.eq.1) write(16,100) ITEL
if (EXTRAROUTZ.eq.1) write(16,105) ((A(I,J),J=1,N),B(I),I=1,N)
if (EXTRAROUTZ.eq.1) write(16,110) SS
do 25 I=1,N
   if (abs(C(I)).lt.1.0E-07) C(I)=0.0
   E(I)=E(I)+C(I)
25 continue
if (EXTRAROUTZ.eq.1) write(16,106) IT
if (EXTRAROUTZ.eq.1) write(16,110) (I,E(I),I=1,N)
if (ITEL.lt.2) goto 2
30 write(16,130) (I,E(I),I=1,N)
   return
40 write(16,140)
100 format (/5X,'RESULTS AFTER ITERATION NO. : ',IS)
105 format (/5X,'MATRICES A AND B'/(*8E9.3))
106 format (/5X,'NUMBER OF ITERATIONS IN EQSOV : IT = ',IS//)
110 format (/5X,'THE SOLUTION VECTOR IS : ','T35,
1   'THE IMPROVED SOLUTION VECTOR IS : ''T35,
2  ('6X,'C'I,I') = ','F12.4',T35,'E('I,I') = ','F12.4)
120 format (/5X,'SUM OF SQUARES IN PREVIOUS STEP WAS : SS = ','F12.4)
130 format (/5X,'SOLUTION VECTOR : '/(*6X,'E('I,I') = ','F12.4)
140 format (/5X,'NO CONVERGED SOLUTION FROM EQSOV')
end

subroutine EQSOV (N,BDMTX,V,ITER,F,IT,INEW)
*******************************************************************************
* M = ORDER OF MATRIX
* BDMTX = TWO- DIMENSIONAL ARRAY OF COEFFICIENTS
* V = RIGHT-HAND VECTOR
* ITER = MAXIMUM NUMBER OF ITERATIONS DESIRED
* EPS = TOLERANCE FOR CONVERGENCE (.GE. 1.E-7)
* F = RESULTING VECTOR
* IT = OUTPUT FROM ROUTINE SPECIFYING NUMBER OF ITERATIONS ACTUALLY
* INEW (FIRST CALL) SET INEW .NE. 1
* (LATER CALLS) IF THE MATRIX IS UNCHANGED AND ONLY THE
* COLUMN VECTOR B IS CHANGED, THEN SET INEW = 1
*******************************************************************************
implicit none
integer*2 N,M,I,J,INEW,IT,SG1,N1,IM1,JMX,K,II,K2,IT2,ITER,IFLAG,II
integer*2 IDX(50)
real*4 BDMTX(M,M),V(M),F(M),X(50),A(50,50),XT(50)
real*4 RADIUS,EPS,AMX,ABSA,T,XI,CX,SUM,XMAX
common /EPS/ EPS
write(16,350) EPS
N=M
IT=0
do 9 I=1,N
    X(I)=V(I)
    F(I)=0.0
9    continue
N1=N-1
if (INEW.eq.1) goto 101
do 10 I=1,N
    do 10 J=1,N
        A(I,J)=BDMTX(I,J)
10    continue
do 12 I=1,N
    IDX(I)=1
12    continue
SG1=0
do 60 I=2,N
C    PARTIAL PIVOTING, CHECK FOR MAX ELEMENT IN (I-1)ST COLUMN.
AMX=abs(A(I-1,I-1))
IM1=I-1
JMX=IM1
do 16 J=I,N
    ABSA=abs(A(J,I-1))
    IF(AMX.ge.ABSA) goto 16
    AMX=ABSA
    JMX=J
16    continue
if (JMX.eq.IM1) goto 20
C    MOVE THE ROW WITH MAX A(J,I-1) TO (I-1)ST ROW.
do 18 K=1,N
    T=A(I-1,K)
    A(I-1,K)=A(JMX,K)
    A(JMX,K)=T
18    continue
II=IDX(I-1)
IDX(I-1)=IDX(JMX)
IDX(JMX)=II
XI=X(I-1)
X(I-1)=X(JMX)
X(JMX)=XI
SG1=1.0
20    continue
if (A(I-1,I-1).eq.0.) goto 200
do 55 J=I,N
    CX=A(J,I-1)/A(I-1,I-1)
    K2=I
    do 50 K=1,N
        A(J,K2)=A(J,K2)-CX*A(I-1,K2)
        K2=K2+1
50    continue
A(J,I-1)=CX
55    continue
60    continue
C    FORWARD PASS - OPERATE ON RIGHT HAND SIDE AS
C    ON MATRIX
62    continue
do 70 I=2,N
do 65 J=I,N
   X(J)=X(J)-X(I-1)*A(J,I-1)
65 continue
70 continue
C
C BACKWARD PASS - SOLVE FOR AX = B
C
X(N)=X(N)/A(N,N)
do 80 I=N,1,-1
   SUM=0.0
   I2=N-I+1
do 75 J=I2,N
      SUM=SUM+A(I2-1,J)*X(J)
75 continue
   X(I2-1)=(X(I2-1)-SUM)/A(I2-1,I2-1)
80 continue
C
XMAX=0.0
X
4 continue
F(I)=F(I)+X(I)
ifeq abs(F(I)).gt.XMAX then
   XMAX=abs(F(I))
eendif
90 continue
if (IT.eq.ITER) return
write(16,300) IT,IFLAG
IT=IT+1
IFLAG=0
do 95 I=1,N
   if (X(I).lt.EPS) then
      IFLAG=IFLAG+1
   endif
write(16,400) X(I),IFLAG
95 continue
if (IFLAG.eq.N) return
C
MATRIX MULTIPLICATION
40 do 170 I=1,N
   RADIUS=0.0*
170 continue
X(I)=V(I)-RADIUS
160 continue
181 if (SG1.eq.0) goto GZ
C IF SG1 .NE. 0, PERMUTE X BEFORE PERFORMING FORWARD PASS.
do 182 I=1,N
   XT(I)=X(I)
182 continue
do 184 I=1,N
   K=IDX(I)
   X(I)=XT(K)
184 continue
GZ
200 I=I-1
write(16,510) I1
300 format(215)
350 format(/,'EPS=',E20.10)
400 format('X(I)=' E20.10,5x,15)
500 FORMAT ('R = ', E15.5)
510 FORMAT ('ERROR RETURN FROM SEQSOV ', I10,
     1 'DIAGONAL TERM REDUCED TO ZERO ', /)
     RETURN
     END
Mar 27 00:05 1990 xx03 Page 1

# Program xx03

fc -o /tmp/xx03 /users/datared/odatared/xx03arimain.o
/tmp/xx03
rm /tmp/xx03

# End of program
# program made by H.J.C. van der Hooven, May 1989.
#
# program designed to perform harmonic analysis on the
# measured imperfections of a cylinder.
#
# The measured data must have been corrected by program xx01
# and those data must have been recalculated as imperfections
# with respect to the best-fit cone by program xx03.
#
# DATA INPUT on unit 10.
# DATA OUTPUT on units 16, 20 and 21.
# INPUT VARIABLES on unit 15.
#
# Dimensions of used arrays is:
# W(NA,NC),A(K1:K2,L1:L2), B(K1:K2,L1:L2), F1(NA,L1:L2), FZ(NA,L1:L2)
# The maximum allowed for NA and NC is 400.
# The minimum allowed for K1 and L1 is 0.
# The maximum allowed for K2 and L2 is 100.
# If given maximum dimensions are not adequate ask system manager
# to increase dimensions in program arimain.f (don’t forget the sub-
# routines!)
# NA = number of axial datapoints. (=NSC if MRI=1, =NP if MRI=0)
# NC = number of circmf. datapoints. (=NP if MRI=1, =NSC if MRI=0)
#
# In this program the measured data is presented in W(I,J),
# where I = axial point number and J = circumferential point number.
#
# INPUT VARIABLES in file /users/data/[shellcode]var :
# list1:
# SHELLCODE - Identification code for used shell.
# MEASUREDATE- date of imperfection measurement.
# NSCAN - total number of scans,
# including number of missing scans,
# excluding number of control scans.
# NPOINT - number of data points per scan.
# CONICITY - cone angle of shell, measured in degrees.
# THICKNESS - wall thickness of the shell (in mm).
# RADIUS - radius of the shell (in mm).
# LENGTH - length of the shell (in mm).
# SCANTYPE =0 => scans are oriented in axial direction.
# =1 => scans are oriented in circumf. direction.
# In that case write a small program to put the
# available data in axial representation first.

list12:
# K1 - lowest harmonic component to be calculated in axial
direction (usually 0).

- highest harmonic component to be calculated in axial
direction (usually 30 for ARIANE shells).

- lowest harmonic component to be calculated in
circumferential direction (usually 0).

- highest harmonic component to be calculated in circum-
ferrential direction (usually 40 for ARIANE shells).

-0 => both sine and cosine coefficients are calculated
-1 => only cosine coefficients are calculated.
-2 => only sine coefficients are calculated.
program xx03arimainf
*********************************************************************
* Program written by H.J.C. van der Hoeven, June through August 1989.
* Program is called by xx03.
*********************************************************************
implicit none
real*4 W(400,400),A(0:100,0:100),B(0:100,0:100)
real*4 F1(400,0:100),FZ(400,0:100)
real*4 CONICITY,THICKNESS,RADIUS,LENGTH
integer*2 NSCAN,NPOINT,K1,KZ,L1,L2,IOS,BEGIN,END
integer*2 COEF,NA,NC,1,1,J,IOUT,SCANNUMBER,SCANTYPE
character*34 FILE,FILEVAR,FILEHS,FILEHC,FILEOUT
character*25 SHELLCODE,MEASUREDATE
character*16 NFILE
character*1 NFILE
common /matrix/ W,F1,FZ
*********************************************************************
* Reading the name of the shell to be processed.
* Opening of the files [shellcode]f, [shellcode]var, [shellcode]hs, 
* [shellcode]hc and [shellcode]out.
*********************************************************************
write(*,101)
write(*,102)
'I'=
read(*,103) NFIL
do while (ichar(NFILE(I:I)).gt.33)
'I'=I+1
end do
FILE= '/users/data//NFILE(I:I-1)//f'
FILEVAR= '/users/data//NFILE(I:I-1)//var'
FILEHS= '/users/data//NFILE(I:I-1)//hs'
FILEHC= '/users/data//NFILE(I:I-1)//hc'
FILEOUT= '/users/data//NFILE(I:I-1)//h_out'
write(*,106) FILE
write(*,107) FILEVAR
write(*,108) FILEHS
write(*,109) FILEHC
write(*,110) FILEOUT
write(*,104)
read(*,105) REPLY

open (10,status='old',file=FILE,iostat=IOS)
if (IOS.ne.0) then
  print 111,FILE
  REPLY='n'
end if
if (REPLY.eq.'n'.or.REPLY.eq.'N') goto 10
if (REPLY.ne.'y'.and.REPLY.ne.'Y') stop
write(*,90)
write(*,91) "********************************************************************"
write(*,91) "* EXECUTING xx03 *"
write(*,91) "********************************************************************"
write(*,90)
Mar 27 08:07 1990 xx03arimain.f Page 2

90  FORMAT(10(/))
91  FORMAT(15x,A40)
101  FORMAT(50(/))
102  FORMAT(/,X,'Enter the name of the shell to be processed : ',$,S)
103  FORMAT(A14)
104  FORMAT(/,X,'Is the above data correct? y(es),n(no) or s(top) : ',$,S)
105  FORMAT(A1)
106  FORMAT(/,3x,'Data is read in from file : ',14x,a34)
107  FORMAT(3x,'All needed variables are read from file : ',a34)
108  FORMAT(3x,'Sine coefficients are written to file : ',a34)
109  FORMAT(3x,'Cosine coefficients are written to file : ',a34)
110  FORMAT(3x,'Extra output is written to file : ',a34)
111  FORMAT(/,10x,'Error: file ',a34,' not found')

***************************************************************************

Reading in the necessary variables.
***************************************************************************

namelist /list1/ SHELLCODE,MEASUREDATE,NSCAN,NPOINT,CONICITY,
      THICKNESS,RADIUS,LENGTH,SCANTYPE
namelist /list12/ K1,K2,L1,L2,COEF
open (15,status='old',file=FILEVAR)
read(15,list1)
read(15,list12)
close(15)

open(16,status='unknown',file=FILEOUT)
if (L1.ge.0. and.K1.ge.0. and.L1.LT.L2.and.K1.LT.K2) then
  write(16,200)SHELLCODE
  write(16,175)MEASUREDATE
  write(16,225)
  write(16,250)LENGTH,abs(THICKNESS)
  write(16,325)K1,K2,L1,L2
endif

***************************************************************************

Reading in the imperfections.
***************************************************************************

if (SCANTYPE.eq.0) then
  NC=NSCAN
  NA=NPOINT
  do J=1,NC
    read(10,150) SCANNUMBER
    read(10,100) (W(I,J),I=1,NA)
  end do
else if (SCANTYPE.eq.1) then
  NA=NSCAN
  NC=NPOINT
  do J=1,NA
    read(10,150) SCANNUMBER
    read(10,100) (W(I,J),J=1,NC)
  end do
endif

close(10)
if(NC/2 .ne. int(NC/2)) then
  NC=NC+1
  do I=1,NA
    W(I,NC)=W(I,1)
enddo
end do
endif
if(NA/2 .ne. int(NA/2)) NA=NA-1
* NA : number of axial point (odd)
* NC : number of circ. point (odd)

******************************************************************************
* Interpolating over all hatches, so that the hatches have no
* adverse effect on the fourier analysis.
* (only a value of W(I,J) in a hatch can be greater than 99.)
******************************************************************************
do J=1,NC
   I=0
   do while (W(I,J).lt.99.and.I.lt.NA)
      I=I+1
   end do
   BEGIN=I-1
   write(16,*) BEGIN
   do while (W(I,J).gt.99.and.I.lt.NA)
      I=I+1
   end do
   END=I
   write(16,*) END
   do I=BEGIN+1,END-1
      W(I,J)=W(BEGIN,J)+(W(END,J)-W(BEGIN,J))*(I-BEGIN)/(END-BEGIN)
   end do
end do

******************************************************************************
* Calculation of the circumferential integrals (trapezoid rule).
******************************************************************************
call integral(NA,NC,L1,L2,IOUT)

******************************************************************************
* Calculation of the fourier sine coefficients.
******************************************************************************
open (20,status='unknown',file=FILEHS)
if (COEF.eq.2.or.COEF.eq.0) then
   write(20,125)SHELLCODE
   write(20,125)MEASUREMENT
   write(20,160)K1,K2,L1,L2
   call hars(A,B,NA,K1,K2,L1,L2,THICKNESS,IOUT)
endif
close(20)

******************************************************************************
* Calculation of the fourier cosine coefficients.
******************************************************************************
open (21,status='unknown',file=FILEHC)
if (COEF.eq.1.or.COEF.eq.0) then
   write(21,125)SHELLCODE
   write(21,125)MEASUREMENT
   write(21,160)K1,K2,L1,L2
   call harc(A,B,NA,K1,K2,L1,L2,THICKNESS,IOUT)
endif
close(21)
else
    write(16,300)
    write(16,275)
    write(16,285)
    write(16,300)
endif
close(16)
stop
100 format(8F9.5)
125 format(A25)
150 format(I5)
160 format(4I5)
175 format ('Date of measurement: ',A25)
200 format ('Harmonic analysis of shell ',A25)
225 format ('Positive outward.')
250 format ('Length of shell = ',F10.2,'Wall thickness = ',F6.2)
275 format ('ERROR: illogical boundaries L1,L2,K1,K2.')
285 format ('No output generated. Please try again.')
300 format ('*******************************')
325 format ('K1= ',IS,'/K2= ',IS,'/L1= ',IS,'/L2= ',IS)
end

subroutine integral(NA,NC,L1,L2,IOUT)
******************************************************************************
* Subroutine calculates for ((I=1,NA) L=L1,L2) the integrals:
* F1(I,L)= integral (W(I,J)*COS(L*THETA)) dTHETA
* F2(I,L)= integral (W(I,J)*SIN(L*THETA)) dTHETA
* for THETA= 0 to 2*PI, using the trapezoid rule.
******************************************************************************

implicit none
integer*2 I,J,L,L1,L2,NA,NC
real*4 W(400,400),F1(400,0:100),F2(400,0:100)
real*4 Y1,Y2,ARG,SUM1,SUM2,PI
common /matrix/ W,F1,F2
PI=4.0*atan(1.0)

do 10 I=1,NA
   do 20 L=L1,L2
      SUM1=W(I,1)
      SUM2=0.0
      do 30 J=2,NC
         ARG=2.0*PI*(J-1)/NC
         Y1=W(I,J)*cos(ARG)
         Y2=W(I,J)*sin(ARG)
         SUM1=SUM1+Y1
         SUM2=SUM2+Y2
      30   continue
      F1(I,L)=2.0*PI*SUM1/NC
      F2(I,L)=2.0*PI*SUM2/NC
  20  continue
  10  continue
return
end
subroutine hars(C,D,NA,K1,K2,L1,L2,THICKNESS,IOUT)

*C(C,K,L) is divided by 2 if L=0.

implicit none
integer*2 K1,KK1,K2,L1,L2,L,L,K,NA
real*4 C(K1,K2,L1:L2),D(K1,K2,L1:L2)
real*4 W(400,400),F1(400,0:100),F2(400,0:100)
real*4 Y1,Y2,ARG,PI,SUM1,SUM2,THICKNESS
common /matrix/ W,F1,F2
PI=4.0*atan(1.0)

do 10 L=L1,L2
   do 20 K=K1,K2
      SUM1=0.0
      SUM2=0.0
      do 30 I=2,NA-1
         ARG=K*PI*(I-1)/(NA-1)
         Y1=F1(I,0)*sin(ARG)
         Y2=F2(I,0)*sin(ARG)
         SUM1=SUM1+Y1
         SUM2=SUM2+Y2
      30 continue
      C(K,L)=2.0*SUM1/(PI*(NA-1)*(abs(THICKNESS)))
      D(K,L)=2.0*SUM2/(PI*(NA-1)*(abs(THICKNESS)))
      if (L.eq.0) then
         C(K,L)=C(K,L)/2.0
      endif
      if (K1.eq.0) then
         KK1=1
      else
         KK1=K1
      endif
      if (L1.eq.0) then
         LLI=L1
      else
         LLI=LL1
      endif
      write(16,400)
      write(16,200)
      write(16,100)((C(K,L),L=L1,L2),K=KK1,K2)
      write(20,100)((C(K,L),L=L1,L2),K=KK1,K2)
      write(16,300)
      write(16,100)((D(K,L),L=LL1,L2),K=KK1,K2)
      write(20,100)((D(K,L),L=LL1,L2),K=KK1,K2)
   20 continue
10 continue
return
100 format(8F9.5)
200 format(/,'C(K,L) components - SIN(K*PI*X/LENGTH)*COS(L*THETA)')
300 format(/,'D(K,L) components - SIN(K*PI*X/LENGTH)*SIN(L*THETA)')
400 format(/,'Half wave sine axial representation')
end

subroutine harc(A,B,NA,K1,K2,L1,L2,THICKNESS,IOUT)

* Subroutine computes the harmonic components of the deflection
* surface using 1/2-wave cosine axial representation:
* A(K,L) = Z/(PI*T*AL)*integral (F1(I,L)*COS(K*PI*X/AL)) dX
* B(K,L) = Z/(PI*T*AL)*integral (F2(I,L)*COS(K*PI*X/AL)) dX
* using trapezoid rule, for X=0 to LENGTH.
* Exception:
* A(K,L) and B(K,L) are divided by 2 if K=0.
* A(K,L) is divided by 2 if L=0.

implicit none
integer*2 K1,K2,L1,L2,K,LI,NA
real*4 A(K1:K2,L1:L2),B(K1:K2,L1:L2)
real*4 W(400,400),F1(400,0:100),F2(400,0:100)
real*4 Y1,Y2,ARG,PI,SUM1,SUM2,THICKNESS
common /matrix/ W,F1,F2
PI=4.0*atan(1.0)

do 10 L=L1,L2
   do 20 K=K1,K2
      SUM1=0.5*(F1(1,L)+F1(NA,L)*cos(K*PI))
      SUM2=0.5*(F2(1,L)+F2(NA,L)*cos(K*PI))
      do 30 I=2,NA-1
         ARG=K*PI*(I-1)/(NA-1)
         Y1=F1(I,L)*cos(ARG)
         Y2=F2(I,L)*cos(ARG)
         SUM1=SUM1+Y1
         SUM2=SUM2+Y2
      30 continue
      A(K,L)=2.0*SUM1/(PI*(NA-1)*(abs(THICKNESS)))
      B(K,L)=2.0*SUM2/(PI*(NA-1)*(abs(THICKNESS)))
      if (K.eq.0) then
         A(K,L)=A(K,L)/2.0
         B(K,L)=B(K,L)/2.0
      endif
      if (L.eq.0) then
         A(K,L)=A(K,L)/2.0
      endif
   20 continue
10 continue

if (L1.eq.0) then
   LLL=1
else
   LLL=L1
endif

write(16,410)
write(16,307)
write(16,100)((A(K,L),L=L1,L2),K=K1,K2)
write(21,100)((A(K,L),L=L1,L2),K=K1,K2)

write(16,309)
write(16,100)((B(K,L),L=LL1,L2),K=K1,K2)
write(21,100)((B(K,L),L=LL1,L2),K=K1,K2)
return

307 format(/,'A(K,L) components - COS(K*PI*X/LENGTH)*COS(L*THETA)'/)
309 format(/,'B(K,L) components - COS(K*PI*X/LENGTH)*SIN(L*THETA)'/)
410 format(/,'Half wave cosine axial representation' )
100 format(8F9.5)
end
Program xx04

Program written by H.J.C. van der Hoeven, June through August 1989.

This is done by recalculating the imperfections from the calculated Fourier coefficients.

The proper working of HARANA and CHECK can be verified by the following procedure:
2. Take either file [shellcode]hs or [shellcode]hc as input for program CHECK and generate a file [shellcode]k.
3. Run HARANA again.

The files [shellcode]hs and [shellcode]hc will now be generated.
If both HARANA and CHECK operate properly, the files [shellcode]hc and [shellcode]hcc should be almost identical, as should the files [shellcode]hs and [shellcode]hss.
If 3-d plots are made (with xx04) of files [shellcode]f and [shellcode]ff those plots should be very much alike.

# INPUT VARIABLES in file /users/data/[shellcode]f,var:
# list1:
#   SHELLCODE - Identification code for used shell.
#   MEASUREDATE- date of imperfection measurement.
#   NSCAN - total number of scans,
#         - including number of missing scans,
#         - excluding number of control scans.
#   NPOINT - number of data points per scan.
#   CONICITY - cone angle of shell, measured in degrees.
#         - positive if R(top) < R(bottom).
#   THICKNESS - wall thickness of the shell (in mm).
#   RADIUS - radius of the shell (in mm).
#   LENGTH - length of the shell (in mm).
#   SCANTYPE - =0 => scans are oriented in axial direction.
#         - =1 => scans are oriented in circumf. direction.
#         - If SCANTYPE=1, hatches and extra plating can not be properly placed.
#         - In that case write a small program to put the available data in axial representation first.
# list12:
#   K1 - lowest harmonic component to be calculated in axial direction (usually 0).
#   K2 - highest harmonic component to be calculated in axial direction (usually 30 for ARIANE shells).
#   L1 - lowest harmonic component to be calculated in circumferential direction (usually 0).
# L2 - highest harmonic component to be calculated in circumferential direction (usually 40 for ARIANE shells).
# COEF - =0 => both sine and cosine coefficients are calculated
#        =1 => only cosine coefficients are calculated.
#        =2 => only sine coefficients are calculated.

fc -o /tmp/xx04 /users/datared/odatared/xx04check.o
/tmp/xx04
rm /tmp/xx04
program xx04check
******************************************************************************
* This program is made by H.J.C. van der Hoeven - august 1989         *
* Program is called by xx04.                                          *
******************************************************************************
implicit none
real*4 W(400,400),A(0:100,0:100),B(0:100,0:100)
real*4 CONICITY,THICKNESS,RADIUS,LENGTH,SCANTYPE
real*4 PEAKMAX,MAX
integer*2 NFIRSTHATCH,NLASTHATCH,ARRAYHATCH(50)
integer*2 NSCAN,NPOINT,NA,NC,K1,K2,L1,L2,COEF,OPTION,I
integer*2 FIRSTSCAN,SCANINCR,NMISSSCAN,NCONTSCAN,NREFPOINT
integer*2 RFCODE,EXTRAOUT1,NCOEF,NSCANPLATING,NSCHANHATCH
integer*2 BEAMCORRECT
character*25 SHELLCODE,MEASUREDATE
character*34 FILE,FILEVAR,FILEOUT
character*16 NFILE
character=1 REPLY
common /HATCH/,FIRSTSCAN,SCANINCR,NCHANHATCH,ARRAYHATCH,
+NFIRSTHATCH,NLASTHATCH

C******************************************************************************
C Reading the name of the shell to be processed.             
C Opening of the files [shellcode] and [shellcode]var.     
C******************************************************************************
write(*,81)
write(*,82)
write(*,83)
write(*,84)
write(*,85)
write(*,86)
write(*,87)
write(*,88)
write(*,89)
write(*,90)
read(*,103) NFILF
I = 1
do while (ichar(NFILE(1:1)).gt.33)
  I = I+1
end do
FILE = '/users/data//'NFILE(1:1-1)
FILEVAR = '/users/data//'NFILE(1:1-3)//'var'
FILEOUT = '/users/data//'NFILE(1:1-3)//'k'
write(*,106) FILE
write(*,107) FILEVAR
write(*,108) FILEOUT
write(*,104)
read(*,105) REPLY
if (REPLY.eq.'n'.or.REPLY.eq.'N') goto 10
if (REPLY.eq.'s'.or.REPLY.eq.'S') goto 103
write(*,95)
write(*,96) 'executing xx04'
write(*,96) **
write(*,96) **
write(*,96)'* For K2=30 and L2=40 this will *'
write(*,96)'* take about 60 minutes. *'
write(*,96)'******************************************************************************'
write(*,95)
81 format(8/*))
82 format('This program recalculates imperfections from a file',
+ ' with known')
83 format('Fourier coefficients.')
84 format('A file named [shellcodek] is generated.')
85 format('A 3-d plot of this file, made with program graph3d',
+ ' should be compared with')
86 format('A 3-d plot of the file named [shellcodef].')
87 format('If sufficient Fourier coefficients have been calculated',
+ ' with program xx03.')
88 format('the difference between the two plots should be minor.')
89 format('Enter the name of a file containing cosine or sine',
+ ' coefficients.')
90 format('The FILE NAME should end with hs or hc --> ',$)
91 format(10/*))
92 format(15x,A40)
93 format(A14)
94 format(3x,'Is the above data correct? y(es),n(no) or s(top) : ',$)
95 format(A1)
96 format(3x,'Data is read in from file : ','14x,a34)
97 format(3x,'All needed variables are read from file : ','a34)
98 format(3x,'Imperfections are written to file : ','a34)

name!!list1/ SHELLCODE,MEASUREDATE,NSCAN,NPOINT,CONICITY,
+THICKNESS,RADIUS,LENGTH,SCANTYPE
name!!list2/ FIRSTSCAN,SCANINCRR,MISSCAN,NCONTSCAN,
+NRFPPOINT,REFCODE
name!!list5/ EXTRAOUTI,PEAKMAX,MAX,NCOEF,SCANPLATING,
+SCANCHANGET,BEAMCORRECT
name!!list7/ NFIRSTTHATCH,NLASTTHATCH,ARRAYHATCH
name!!list12/ K1,K2,L1,L2,COEF

if (NFILE(I-2:I-1).eq.'hs') then
    OPTION=2
else if (NFILE(I-2:I-1).eq.'hc') then
    OPTION=1
else
    stop
endif
open(15,status='old',file=FILEVAR)
open(21,status='old',file=FILE)
open(23,status='unknown',file=FILEOUT)
read(15,list1)
read(15,list2)
read(15,list5)
read(15,list7)
read(15,list12)
close(15)
if (SCANTYPE.eq.1) then
    NC=NSCAN
Mar 27 08:15 1990  xx04check.f Page 3

NA=NPOINT
else if (SCANTYPE.eq.0) then
   NA=NPOINT
   NC=NSCAN
endif

call haranachcek(W,A,B,K1,K2,L1,L2,NA,NC,THICKNESS,OPTION,ISCANTYPE)
103 end

******************************************************************************
 subroutine haranachcek(W,A,B,K1,K2,L1,L2,NA,NC,THICKNESS,OPTION,SCANTYPE)
 implicit none
 integer*2 L1,L2,K1,K2,NA,NC,KK1,LL1,OPTION,I,J,K,L,SCANTYPE
 integer*2 FIRSTSCAN,SCANINC,NSCANHATCH,ARRAYHATCH(S0)
 integer*2 NFIRSTHATCH,NLASTHATCH
 integer*2 SCANNUMBER,N
 real*4 A(K1,K2,L1:L2),B(K1,K2,L1:L2),W(NA,NC)
 real*4 THICKNESS,PI,WW,ARG1,ARG2
 character*25 SHELLCODE,MEASUREDATE
 common /HATCH/ FIRSTSCAN,SCANINC,NSCANHATCH,ARRAYHATCH,N
FIRSTHATCH,NLASTHATCH

PI=4.0*atan(1.0)
 read(21,125)SHELLCODE
 read(21,125)MEASUREDATE
 read(21,175)K1,K2,L1,L2
 if (K1.eq.0) then
    KK1=1
 else
    KK1=K1
 endif
 if (L1.eq.0) then
    LL1=1
 else
    LL1=L1
 endif
 if (OPTION.eq.1) then
    read(21,100)((A(K,L),L=LL1,L2),K=K1,K2)
    read(21,100)((B(K,L),L=LL1,L2),K=K1,K2)
 endif
 if (OPTION.eq.2) then
    read(21,100)((A(K,L),L=LL1,L2),K=KK1,K2)
    read(21,100)((B(K,L),L=LL1,L2),K=KK1,K2)
 endif
 close(21)
do 10,I=1,NA
 ARG1=PI*(I-1)/(NA-1)
do 20,J=1,NC
  ARG2=2*PI*(J-1)/NC
  WI,J=0.0
 do 30,K=K1,K2
    do 40,L=1,L2
       WW=A(K,L)*cos(ARG2*L)+B(K,L)*sin(ARG2*L)
       if (OPTION.eq.1) then

W(I,J)=W(I,J)+THICKNESS*cos(ARG1*K)*WW
endif
if (OPTION.eq.2) then
  W(I,J)=W(I,J)+THICKNESS*sin(ARG1*K)*WW
endif
40  continue
30  continue
20  continue
10  continue
if (NSCANHATCH.ne. 0) then
  do N=1,NSCANHATCH
    J=(ARRAYHATCH(N)-FIRSTSCAN)/SCANINCR+1
    do I=FIRSTHATCH,NLASTHATCH
      W(I,J)=99.999
    end do
  end do
end do
end if
if (SCANTYPE.eq.1) then
  do I=1,NA
    SCANNUMBER=FIRSTSCAN+SCANINCR*(I-1)
    write(23,150) SCANNUMBER
    write(23,100) (W(I,J),J=1,NC)
  end do
else if (SCANTYPE.eq.0) then
  do J=1,NC
    SCANNUMBER=FIRSTSCAN+SCANINCR*(J-1)
    write(23,150) SCANNUMBER
    write(23,100) (W(I,J),I=1,NA)
  end do
endif
C
  return
100  format(8F9.5)
125  format(A25)
150  format(I5)
175  format(4I5)
end
# Program mapprint

# warning !!! : set the printer at the top of page.

fc -o /tmp/print /users/dateded/odedared/mapprint.o
/tmp/print

rcp /tmp/result halsys:/dev/1p
rm /tmp/print
rm /tmp/result

# end of program

# ***********************************************
# program made by Christian Cattalas, July 1989.
# minor modifications performed by H.J.C. van der Hoeven, September '89.
# This program writes the two files containing the Fourier coefficients
# in a format selected by the user in a file called RESULT in order to
# be sent to a printer.
# If the circumferential wave number L is too high to be contained in
# only one array, the user could choose to print another one.
# Just the code name of the SHELL is requested to run this program.
# For clearer representation any data smaller than 0.0001 for first
# format, 0.0001 for the second and 0.00001 for the third is replaced
# by the number 0.0.
# This process calls the Fortran program 'mapprint.f'.
# The 'mapprint.f' program uses the subroutines AFF and COLUMN.
# Data
# ----
# This program uses the same variables as program harana.

# INPUT:
#   - NFILE : character*14  filename of the Fourier coef.
#   - FORMAT : integer*2    number of decimals, 3,4 or 5.
#   - K1 ,K2 : integer*2     lowest axial coeff. to be printed.
#   - L1 ,L2 : integer*2     highest circumf. coeff. to be printed.
#   - CHOICE : integer*2    =0 => both sine and cosine representations printed.
#                        =1 => only cosine representation.
#                        =2 => only sine representation.
# These input variables are stored in file /users/data/shellcode/var,
# unit 15 in the namelist List17.

# The input filenames [shellcode]hc and [shellcode]hs must be present
# in the directory /users/data/.
# The data input units are 39 & 40.

# OUTPUT:
# none
The new file created is: '/users/tmp/result', unit 41.
```plaintext
program mapprintf

*****************************************************************************
* Program made by Christian Cartalas - july 1989.                        *
* This program prints the files containing the fourier                   *
* coefficients on the halsys printer.                                    *
* It is called by mapprint, and uses AFF and COLUMN.                     *
*****************************************************************************

implicit none
character*24 DATE, NAME
character*28 FILE1, FILE2
character*29 FILEVAR
character*14 NFILE
character*1 E, REPLY
character*4 Hi
character*5 Lo
real*4 AR(0:100, 0:100)
real*4 SL1, V, W
integer*2 K1, K2, L1, L2, I, J, L, KK1, KK2, LL1, LL2, CHOICE, COUNTER
integer*2 FORMAT, ANS, MAXIMUM, IOS1, IOS2, LIM_P, PR, NL
integer*2 NL1, NK1, MATRIX
    IMATRIX=0 -> A & B matrix
    IMATRIX=1 -> C & D matrix
integer*2 NB, MARQ
    IMARQ=0 -> A & C matrix
    IMARQ=1 -> B & D matrix
integer*2 K1, Kh, L1, Lh
name list /list16/ FORMAT, K1, Kh, L1, Lh, CHOICE

*****************************************************************************
initialisation  **************************************************************************

PR=0
I=1
E=CHAR(27)
Hi='(s5H'
Lo='(s10H'

*****************************************************************************
opening of the input files  **************************************************************************

write(*,25)
write(*,20)
read(*,90) NFILE
write(*,9) NFILE
write(*,10)
read(*,12) REPLY
if (REPLY.ne.'y' .and. REPLY.ne.'Y') goto 1
do while(NFILE(I:I).ne.' ')
    I=I+1
end do
I=I-1
FILE1='/users/data/'//NFILE(1:I)//'hc'
FILE2='/users/data/'//NFILE(1:I)//'hs'
write(*,5)
write(*,6) "*****************************************************************************"
```
write(*,6) "* EXECUTING *"  
write(*,6) "* MAPPRINT *"  
write(*,6) "*******************************"  
write(*,5)  
open (39,status='old',file=FILE1,iostat=IOS1)  
if (IOS1.ne.0) then  
print 2,FILE1  
endif  

open (40,status='old',file=FILE2,iostat=IOS2)  
if (IOS2.ne.0) then  
print 2,FILE2  
endif  
if (IOS1.ne.0 .and. IOS2.ne.0) then  
goto 1  
endif  

open (41,status='unknown',file='/tmp/result')  
write(41,3) char(27),'(s3b3tZQ'

******************************* reading of input data *******************************

FILEVAR = '/users/data//'NFILE(l:i)//'var'  
open(unit=15,status='old',file= FILEVAR)  
read(15,list16)  
close(15)  

K1=K1  
K2=Kh  
L1=L1  
L2=Lh  
if ((CHOICE.eq.0.or.CHOICE.eq.1).and.IOS1.eq.0) then  
read(39,200) NAME,DATE,KK1,KK2,LL1,LL2  
endif  
if ((CHOICE.eq.0.or.CHOICE.eq.2).and.IOS2.eq.0) then  
read(40,200) NAME,DATE,KK1,KK2,LL1,LL2  
endif  
if (CHOICE.eq.2) then  
MATRIX=1  
else  
MATRIX=0  
endif  
if (K1.lt.KK1.or.L1.lt.LL1.or.K2.gt.KK2.or.L2.gt.LL2) then  
print *,' ERROR: wrong K1, Kh, L1, Lh in list16'  
stop  
endif  
write(41,201) E,Hi,NAME,E,Lo,E,Hi,DATE,E,Lo,K1,K2,L1,L2  
write(41,70) char(12)  
SL1=L1  

******************************* determination of the best format *******************************

if (FORMAT.eq.3) then  
LIM_P=18
else if (FORMAT.eq.4) then  
    LIM_P=16  
else  
    LIM_P=14  
endif

NB=L1+LIM_P  
if ((L2-L1).le.LIM_P) then  
    NB=L2  
    ANS=1  
    LIM_P=L2-L1  
else if ((L2-L1).gt.2*LIM_P) then  
    MAXIMUM=2*LIM_P+1+L1  
    ANS=2  
else  
    MAXIMUM=L2  
    ANS=2  
endif

************************************************************************** determination of the best sheet *******************************

if (ANS.eq.1) then  
    if (2*(K2-K1)+7).ge.B2) then  
        PR=1  
    endif  
else if (ANS.eq.2) then  
    if (2*(K2-K1)+8).ge.B2) then  
        PR=2  
    else if ((4*(K2-K1)+2).ge.B2) then  
        PR=1  
    endif  
endif

************************************************************************** A & B & C & D matrix *******************************

NK1=K1  
if (CHOICE.eq.0) then  
    COUNTER=2  
else if (CHOICE.eq.1 .or. CHOICE.eq.2) then  
    COUNTER=1  
endif  
do V=1,COUNTER  
    MARQ=0  
    NL=1  
    NL1=L1  
do W=1,2  
    if (PR.eq.2) then  
        do L=0,nint((64-(K2-K1)*6))*.5)  
            write(41,B80)  
        end do  
    endif  
    if (MARQ.eq.0 .and. MATRIX.eq.0 .and. IOS1.eq.0) then  
        read(39,250) ((AR(I,J),J=LL1,LL2),I=KK1,KK2)  
        write(41,230)  
        if (NL1 .eq. 0) then  
            NL1=L1  
        endif  
    endif  
end do
endif
else if (MARQ.eq.1.and.MATRIX.eq.0.and.IOS1.eq.0) then
  read(39,250) ((AR(I,J),J=max(1,LL1),LL2),I=KK1 KK2)
  write(41,240)
  if (NL1.eq.0) then
    NL1=L1
  endif
else if (MARQ.eq.0.and.MATRIX.eq.1.and.IOS2.eq.0) then
  read(40,250) ((AR(I,J),J=LL2,LL2),I=max(1,KK1),KK2)
  write(41,270)
  if (NL1.eq.0) then
    NL1=L1
  endif
else if (MARQ.eq.1.and.MATRIX.eq.1.and.IOS2.eq.0) then
  read(40,250) ((AR(I,J),J=max(1,LL1),LL2),I=max(1,KK1),KK2)
  write(41,280)
  if (NL1.eq.0) then
    NL1=L1
  endif
endif
call COLUMN(FORMAT,K2,NB,L1)
call AFF(AR,NK1,K2,NL1,L1,NB,FORMAT)
NL=NL+K2-K1+6
if (ANS.eq.2) then
  if (PR.eq.2) then
    write(41,70) char(12)
    do L=0,nint((S6-(K2-K1+3))*0.5)
      write(41,80)
    end do
    if (MARQ.eq.0 .and. MATRIX.eq.0) then
      write(41,230)
    else if (MARQ.eq.1 .and. MATRIX.eq.0) then
      write(41,240)
    else if (MARQ.eq.0 .and. MATRIX.eq.1) then
      write(41,270)
    else if (MARQ.eq.1 .and. MATRIX.eq.1) then
      write(41,280)
    endif
  else if (PR.eq.0) then
    do L=0,((10+K1)-K2)
      write(41,80)
    end do
    if (MARQ.eq.0) then
      NL=16
    else
      NL=49
    endif
  else
    do L=0,(30-NL)
      write(41,80)
    end do
    NL=34
  endif
L1=LIM_P+SL1+1
NL1=L1
NB=MAXIMUM
call COLUMN(FORMAT,K2,NB,L1)
call AFF(AR,NK1,K2,NL1,L1,NB,FORMAT)
NL=NL+K2-K1+3
endif
NB=LIM_P+SL1
L1=SL1
if (PR.eq.1 .or. PR.eq.2) then
  write(41,70) char(12)
  NL=1
else if (MARQ.eq.0) then
  do L=0,(30-NL)
   write(41,80)
  end do
  NL=33
endif
MARQ=1
NL1=max(L1,1)
end do
if (PR.eq.0) then
  write(41,70) char(12)
endif
if (K1.eq.0) then
  NK1=K1+1
endif
MATRIX=1  ! C & D matrix
end do
write(41,3) char(27),'(s@b@t@Q' if (IOS1.eq.0) then
  close(39)
endif
if (IOS2.eq.0) then
  close(40)
endif
close(41)
stop

! *** warning: file '.A14,'not found ***')
2 format(x,/,')
3 format(x,A1,A)
5 format(10//)
6 format(15x,A40)
9 format(//,3x,'You typed in the shellname : ',A14)
10 format(//,3x,'Confirm this name with Yes or No, please ?:',$)
12 format(A1)
15 format(A8,I3,/,A5,I3)
20 format(//,3x,'Enter the shellname : ',$)
25 format(22//),T10,'WARNING !! : SET the HALSYS-printer at the top ',
1'of page', //////////)
65 format(//)
70 format(A)
80 format(x)
90 format(A14)
200 format(A24,/,A24,/,A15)
201 format(20//),38x,'name of the shell : ',A1,A,A12,A1,A,/,38x,
1'date of the measurement : ',A1,A,A10,A1,A,/,38x,'Parameters :'
2,2x,'K1 =',I3,/,S2x,'Kh =',I3,/,S2x,'L1 =',I3,/,S2x,'Lh =',I3)
230 format(//,38x,'A(K,L) components',/)
Feb 12 14:49 1990  maprint.f Page 6

240  format(//,38x,'B(K,L) components',/) 
250  format(8FS,5) 
270  format(//,38x,'C(K,L) components',/) 
280  format(//,38x,'D(K,L) components',/) 
end 

c ####################################################################

subroutine AFF (AR,NK1,K2,NL1,L1, NB,FORMAT) 
imPLICIT none 
real*4 AR(0:100,0:100), NUL 
real*4 MIN,MIN4,MIN5 
integer*2 NB, FORMAT, I, NK1, K2, NL1, L1, J 

MIN=.001 
MIN4=.0001 
MIN5=.00001 
NUL=.0 
do I=NK1,K2 
   if (NL1.eq.L1 .and. K2.lt.100) then 
      write(41,100) I 
   else if (NL1.ne.L1 .and. K2.lt.100) then 
      if (FORMAT.eq.3) then 
         write(41,200) I 
      else if (FORMAT.eq.4) then 
         write(41,201) I 
      else 
         write(41,202) I 
      endif 
   else if (NL1.eq.L1 .and. K2.ge.100) then 
      write(41,300) I 
   else if (NL1.ne.L1 .and. K2.ge.100) then 
      if (FORMAT.eq.3) then 
         write(41,400) I 
      else if (FORMAT.eq.5) then 
         write(41,401) I 
      else 
         write(41,402) I 
      endif 
   endif 
do I=NK1,NB 
   if (FORMAT.eq.4) then 
      if (abs(AR(I,J)).ge.MIN4) then 
         write(41,24) AR(I,J) 
      else 
         write(41,14) NUL 
      endif 
   else if (FORMAT.eq.5) then 
      if (abs(AR(I,J)).ge.MIN5) then 
         write(41,25) AR(I,J) 
      else 
         write(41,15) NUL 
      endif 
   else 
      if (abs(AR(I,J)).ge.MIN) then 
         write(41,20) AR(I,J) 
   endif 
end
else
   write(41,10) NUL
endif
endif
end do
write(41,30)
end do
return

10 format(TR1,f3.1,TR2,$)
14 format(TR1,f3.1,TR3,$)
15 format(TR1,f3.1,TR4,$)
20 format(f6.3,$)
24 format(f7.4,$)
25 format(f8.5,$)
30 format(/,$)
100 format('K=','I2,$)
200 format('K=','I2,’ ** ’,$)
201 format('K=','I2,’ *** ’,$)
202 format('K=','I2,’ **** ’,$)
300 format('K=','I3,$)
400 format('K=','I3,’ ** ’,$)
401 format('K=','I3,’ *** ’,$)
402 format('K=','I3,’ **** ’,$)
end

c #--------------------------------------------------------------------------------------

subroutine COLUMN (FORMAT,K2,NB,L1)
implicit none
integer*2 FORMAT
integer*2 K2,NB,L1,I

c ********************* initialisation *********************

if (FORMAT.eq.4 .and. K2.lt.100) then
   write(41,304) (I,I=L1,NB)
else if (FORMAT.eq.4 .and. K2.ge.100) then
   write(41,314) (I,I=L1,NB)
else if (FORMAT.eq.5 .and. K2.lt.100) then
   write(41,305) (I,I=L1,NB)
else if (FORMAT.eq.5 .and. K2.ge.100) then
   write(41,315) (I,I=L1,NB)
else if (FORMAT.eq.3 .and. K2.lt.100) then
   write(41,300) (I,I=L1,NB)
else
   write(41,310) (I,I=L1,NB)
endif
write(41,100)
return

100 format(x)
300 format( \\
   \ L= '19(I2,4x))
304 format( \\
   \ L= '17(I2,5x))
305 format( \\
   \ L= '15(I2,6x))
310 format( \\
   \ L= '19(I2,4x))
314 format(' \ L= ',17(I2,5x))
315 format(' \ L= ',15(I2,6x))
end
program hargraph

program to plot imperfection coefficients for shells.

program made by A.W.H. Klompe - januari 1990

After any modification compile with:
fcc -o hargraph hargraph.f -ldd98550 -lddhpgl -ldd3001 -ldvio -lsbl -lsb2
and run program hargraph

In this program the following variables are used:

list13
KMAX   - number of points to be plot in circumferential
crossplots.
LMAX   - number of points to be plotted in axial crossplots.
XMAXX  - length of X-axis.
YMAXX  - length of Y-axis.
NLINEPLOT   - number of lines per plot (maximum 4).
PLOTCOEF = 0 => sine and cosine plots are made
           = 1 => only cosine plots are made
           = 2 => only sine plots are made.

list14
ARRAYK  - array containing number of selected axial half-waves
to be plotted.

list15
ARRAYL  - array containing numbers of selected circumferential
full-waves to be plotted.

implicit none
include '/usr/include/starbase.f.l.h'
include '/usr/include/starbase.f2.h'
real*4 A(0:100,0:100),B(0:100,0:100),C(0:100,0:100),D(0:100,0:100)
real*4 KSI, YMIN, YMAX, YMAXX, AZ, BZ, CZ, D2, XP, YP, XI, Y1, X2, Y2, COORD(2)
integer*2 ARRAYK(4), ARRAYL(4), NLINEPLOT, PLOTCOEF
integer*2 DARRAYK(4), DARRAYL(4), DLINEPLOT, D PLOTCOEF
integer*2 XMIN, XMAX, XMAXX, XB, I, KMAX, LMAX, IOS, NOTCOS, NOTSIN
integer*2 K1, K2, L1, L2, KK1, LL1, K, L, OPTION, LOAD
integer*4 FILOES
character*1 NULL, REPLY
character*3 NUM
character*25 SHELLCODE, MEASUREDATE
character*27 NFILE, FILEHS, FILEHC, FILEVAR
parameter (NULL=char(0))
common /COM1/ XMIN, XMAX, YMIN, YMAX, SHELLCODE, FILOES
common /COM2/ ARRAYK, ARRAYL, NLINEPLOT,
              DARRAYK, DARRAYL, DLINEPLOT
name list /LIST13/ KMAX, LMAX, XMAXX, YMAXX, NLINEPLOT, PLOTCOEF
name list /LIST14/ ARRAYK
name list /LIST15/ ARRAYL

NOTCOS=0
NOTSIN=0
Mar 27 08:20 1990  hargraph.f Page 2

LOAD=0
1 OPTION=0
call menu(OPTION,FILDES)

**********************************************
• Reading the name of the shell to be processed.
**********************************************
do while(OPTION .lt. 4)
  if (LOAD .eq. 1) then
    write(*,90) 'Already prepared shell : ',NFFILE
    write(*,91) 'You want to use the same shell ? --> '
    read(*,105) REPLY
    if(REPLY .eq. 'y' .or. REPLY .eq. 'Y') then
      write(*,101)
call selection
      write(*,110)
      write(*,108) '****************************
      write(*,108) '* EXECUTING THE *
      write(*,108) '* PLOTS OF *
      write(*,108) '* HARMONIC COMPONENTS *
      write(*,110)
goto 20
    end if
  end if
write(*,101)
10 write(*,102)
I=1
read(*,103) NFFILE
do while (ichar(NFILE(I:I)).gt.33)
  I=I+1
end do
FILEHS='/users/data/)/NFFILE(I:I-1)'/hs'
FILEHC='/users/data/)/NFFILE(I:I-1)'/hc'
FILEVAR='/users/data/)/NFFILE(I:I-1)'/var'
write(*,106) FILEHS
write(*,107) FILEHC
write(*,108) FILEVAR
write(*,104)
read(*,105) REPLY
if (REPLY.eq.'n'.or.REPLY.eq.'N') goto 10
if (REPLY.ne.'y'.and.REPLY.ne.'Y') then
  write(*,200) 'Answer with y or n !'
goto 15
end if
write(*,110)
write(*,108) '*******************************
write(*,108) '* EXECUTING THE *
write(*,108) '* PLOTS OF *
write(*,108) '* HARMONIC COMPONENTS *
write(*,110)

**********************************************
• read in input variables
open(15, status='old', file=FILEVAR)
read(15, LIST13)
read(15, LIST14)
read(15, LIST15)
close(15)
XMAX=XMAXH
YMAX=YMAXH
ONLINEPLOT=NLINPLOT
PLOTCOEF=PLOTCOEF
do 1=1, NLINPLOT
   DARRAY(I)=ARRAY(K(I))
   DARRAYL(I)=ARRAYL(I)
10 continue

executing the plots of sine components

if (FILDES .eq. -1) goto 1
if (PLOTCOEF.eq.0.or.PLOTCOEF.eq.2) then
   open(11, status='old', file=FILEHS, iostat=IOS)
   if (IOS.ne.0) NOTSIN=1
   if (NOTSIN .eq. 1) goto 41

read sine components

if (LOAD .eq. 1) goto 25
read(11, 200) SHELLCODE
read(11, 200) MEASUREDATE
read(11, 210) K1, K2, L1, L2
if (K1.eq.0) then
   KK1=1
else
   KK1=K1
endif
if (L1.eq.0) then
   LL1=1
else
   LL1=L1
endif
read(11, 401) ((C(K,L), L=L1, L2), K=KK1, K2)
read(11, 401) ((D(K,L), L=LL1, L2), K=KK1, K2)
close(11)

plot of sine components

25 XMIN=-XMAX/5
YMIN=-2*YMAX/10
if (OPTION .eq. 1) then
   X1=real(XMIN)
   Y1=(-YMAX+3*abs(YMIN))
   Z1=real(2*XMAX+3*abs(XMIN))
   Y2=YMAX+abs(YMIN)
else
X1=real(-XMAX/5)
Y1=-2*YMAX/10
X2=real(XMAX+abs(XMIN))
Y2=YMAX+abs(YMIN)
end if
if(FILDES.eq.-1) goto 1
XB=L1
call SCALE(X1,Y1,X2,Y2, XB, OPTION)
do 30 I=1,NLINEPLOT
   call color(FILDES,I)
   if(OPTION.eq.3) then
      call marker(FILDES,I)
      COORD(1)=0.
      COORD(2)=abs(C(ARRAYK(I),0))
      call polymarker2d(FILDES,COORD,1,0)
   end if
   call move2d(FILDES,0.,abs(C(ARRAYK(I),0)))
do L=1,LMAX
   C2=C(ARRAYK(I),L)+C(ARRAYK(I),L)
   D2=D(ARRAYK(I),L)+D(ARRAYK(I),L)
   KSI=sqrt(C2+D2)
   call draw2d(FILDES,real(L), KSI)
   if(OPTION.eq.3) then
      COORD(1)=real(L)
      COORD(2)=KSI
      call polymarker2d(FILDES,COORD,1,0)
      call move2d(FILDES,real(L),KSI)
   end if
   end do
write(NUM,fmt='(I3)') ARRAYK(I)
XP=3*XMAX/4.
YP=YMAX-I*YMAX/10
if(OPTION.eq.3) then
   COORD(1)=XP
   COORD(2)=YP+YMAX/100
   call polymarker2d(FILDES,COORD,1,0)
   call text2d(FILDES,XP,YP,'-- k='//NUM//'NULL,'VDC_TEXT,0)
else
   call text2d(FILDES,XP,YP,'-- k='//NUM//'NULL,'VDC_TEXT,0)
end if
continue
30 call line_color(FILDES,1.,1.,1.)
call text_color(FILDES,1.,1.,1.)
call text2d(FILDES,real(XMAX/3),YMIN/2,'sine vs 1'//NULL,
   + VDC_TEXT,0)
if(OPTION.eq.2.or.OPTION.eq.3) then
   call text2d(FILDES,real(XMAX/3),YMAX+abs(YMIN/2),NFILE//NULL,
    + VDC_TEXT,0)
end if
if(OPTION.eq.2)then
   call text2d(FILDES,0.,Y1,'To continue PRESS Return'//NULL,
    + VDC_TEXT,0)
   FILDES=gclose(FILDES)
   read(*,105) REPLY
call dev_init(OPTION,FILDES)
end if
if(OPTION .eq. 3) then
   FILDES=gclose(FILDES)
   call dew_init(OPTION,FILDES)
   call clear_view_surface(FILDES)
   end if

X8=K1
if(OPTION .eq. 1) then
   XI=-real(XMAX+3*abs(XMIN))
   Y1=-(YMAX+3*abs(YMIN))
   XZ=real(XMAX+abs(XMIN))
   YZ=YMAX+abs(YMIN)
else
   XI=real(-XMAX/5)
   Y1=-2*YMAX/10
   XZ=real(XMAX+abs(XMIN))
   YZ=YMAX+abs(YMIN)
end if
   call SCALE(XI,Y1,XZ,YZ,X8,OPTION)
do 40 I=1,NLINEPLOT
   call color(FILDES,I)
   if(OPTION .eq. 3) then
      call marker(FILDES,I)
      COORD(1)=I.
      COORD(2)=abs(C(1,ARRAYL(I)))
      call polynmarker2d(FILDES,COORD,1,0)
   end if
   call move2d(FILDES,1.,abs(C(1,ARRAYL(I))))
do K=2,KMAX
   CZ=C(K,ARRAYL(I))*C(K,ARRAYL(I))
   DZ=D(K,ARRAYL(I))*D(K,ARRAYL(I))
   KSI=sqrt(CZ+DZ)
   call draw2d(FILDES,real(K),KSI)
   if(OPTION .eq. 3) then
      COORD(1)=real(K)
      COORD(2)=KSI
      call polynmarker2d(FILDES,COORD,1,0)
      call move2d(FILDES,real(K),KSI)
   end if
end do
write(NUM,fmt='(1I3)') ARRAYL(I)
XP=3*XMAX/4.
YP=YMAX-I*YMAX/10
if(OPTION .eq. 3) then
   COORD(1)=XP
   COORD(2)=YP+YMAX/100
   call polynmarker2d(FILDES,COORD,1,0)
   call text2d(FILDES,XP,YP,' 1="/NUMNullOrEmpty, VDC_TEXT,0)'
else
   call text2d(FILDES,XP,YP,'-- 1="/NUMNullOrEmpty, VDC_TEXT,0)
end if
40 continue
call line_color(FILDES,1.,1.,1.)
call text_color(FILDES,1.,1.,1.)
call text2d(FILDES,real(XMAX/3),YMIN/2,'sine vs k="/NULL, + VDC_TEXT,0)
if(OPTION .eq. 2 .or. OPTION .eq. 3) then
  call text2d(FILDES,real(XMAX/3),YMAX+abs(YMIN/2),NFILE//NULL,
  + VDC_TEXT,0)
end if
if(OPTION .eq.2)then
  call text2d(FILDES,0.,Y1,'To continue PRESS Return'/NULL, 
  + VDC_TEXT,0)
  FILDES=gclose(FILDES)
  read(*,105) REPLY
  call dev_init(OPTION,FILDES)
end if
if(OPTION .eq.3) then
  FILDES=gclose(FILDES)
  call dev_init(OPTION,FILDES)
end if
* end of plot sine components

******************************************************************************
* executing the plots of cosine components                                 *
******************************************************************************
41  if (PLOTCOEF.eq.0 .or. PLOTCOEF.eq.1) then
    open (10,status='old',file=FILEHC,iostat=105)
    if (IOS.ne.0) NOTCOS=1
    if (NOTCOS .eq. 1) goto 65
*--------------------------------------------------------------------------
* read cosine components                                                  *
*--------------------------------------------------------------------------
  if (LOAD .eq. 1) goto 45
  read(10,200) SHELLCODE
  read(10,200) MEASUREDATE
  read(10,210) K1,KZ,L1,L2
  if (L1.eq.0) then
    L1=1
  else
    L1=L1
  endif
  read(10,401) ((A(K,L),L=L1,L2),K=K1,K2)
  read(10,401) ((B(K,L),L=L1,L2),K=K1,K2)
  close(10)
*--------------------------------------------------------------------------
* plot of cosine components                                                *
*--------------------------------------------------------------------------
45  XMIN=-XMAX/5
    YMIN=-2*YMAX/10
    XB=L1
  if(OPTION .eq.1) then
    X1=real(XMIN)
    Y1=YMIN
    X2=real(2*YMAX+3*abs(XMIN))
    Y2=2*YMAX+3*abs(YMIN)
  else
    X1=real(-XMAX/5)
    Y1=-2*YMAX/10
    X2=real(XMAX+abs(XMIN))
    Y2=YMAX+abs(YMIN)
end if
  call SCALE(X1,Y1,X2,Y2,XB,OPTION)
call text2d(FILDES,real(XMAX/3)),YMIN/2,'cosine vs l'//NULL,
+ VDC_TEXT,0)
  if(OPTION .eq. 2 .or. OPTION .eq. 3) then
call text2d(FILDES,real(XMAX/3),YMAX+abs(YMIN/2),NFL+NULL,
+ VDC_TEXT,0)
end if
do 50 I=1,NLINEPLOT
  call color(FILDES,I)
  if(OPTION .eq. 3) then
    call marker(FILDES,I)
    COORD(1)=0.
    COORD(2)=abs(A(ARRAYK(I),0))
call polymarker2d(FILDES,COORD,1,0)
  end if
  call move2d(FILDES,0.,abs(A(ARRAYK(I),0)))
do L=1,LMAX
    A2=A(ARRAYK(I),L)*A(ARRAYK(I),L)
    B2=B(ARRAYK(I),L)*B(ARRAYK(I),L)
    KSI=SQRT(A2+B2)
call draw2d(FILDES,real(L),KSI)
  if(OPTION .eq. 3) then
    COORD(1)=real(L)
    COORD(2)=KSI
call polymarker2d(FILDES,COORD,1,0)
call move2d(FILDES,real(L),KSI)
  end if
end do
write(NUM,fmt='(1I3)') ARRAYK(I)
XP=3*XMAX/4.
YP=YMAX-1.*YMAX/10
if(OPTION .eq. 3) then
  COORD(1)=XP
  COORD(2)=YP+YMAX/100
  call polymarker2d(FILDES,COORD,1,0)
call text2d(FILDES,XP,YP,'** k='//NUM+NULL,VDC_TEXT,0)
else
  call text2d(FILDES,XP,YP,'-- k='//NUM+NULL,VDC_TEXT,0)
end if
50 continue
call line_color(FILDES,1.,1.,1.)
call text_color(FILDES,1.,1.,1.)
if(OPTION .eq.2)then
  call text2d(FILDES,0.,Y1,'To continue PRESS Return'//NULL,
+ VDC_TEXT,0)
  FILDES=gclose(FILDES)
read(*,105) REPLY
call dev_init(OPTION,FILDES)
end if
if(OPTION .eq.3) then
  FILDES=gclose(FILDES)
call dev_init(OPTION,FILDES)
end if
* XB=K1
if(OPTION .eq.1) then
    X1=real(XMAX+3*abs(XMIN))
    Y1=YMIN
    X2=real(XMAX+abs(XMIN))
    Y2=2*YMAX+3*abs(YMIN)
else
    X1=real(-XMAX/5)
    Y1=-2*YMAX/10
    X2=real(XMAX+abs(XMIN))
    Y2=YMAX+abs(YMIN)
end if

call SCALE(X1,Y1,X2,Y2,XB,OPTION)
call text2d(FILDES,real(XMAX/3),YMIN/2,'cosine vs k'/NULL,
+ VDC_TEXT,0)
if(OPTION .eq. 2 .or. OPTION .eq. 3) then
    call text2d(FILDES,real(XMAX/3),YMAX+abs(YMIN/2),NFL/E1000//NULL,
+ VDC_TEXT,0)
end if

do 80 I=1,NLINES
    call color(FILDES,I)
    if(OPTION .eq. 3) then
        call marker(FILDES,I)
        COORD(1)=0.
        COORD(2)=abs(A(0,ARRAYL(I)))
        call polymarker2d(FILDES,COORD,1,0)
    end if
    call move2d(FILDES,0.,abs(A(0,ARRAYL(I))))
    do K=1,XMAX
        A2=A(K,ARRAYL(I))*A(K,ARRAYL(I))
        B2=B(K,ARRAYL(I))*B(K,ARRAYL(I))
        KSI=sqrt(A2+B2)
        call draw2d(FILDES,real(K),KSI)
        if(OPTION .eq. 3) then
            COORD(1)=real(K)
            COORD(2)=KSI
            call polymarker2d(FILDES,COORD,1,0)
            call move2d(FILDES,real(K),KSI)
        end if
    end do
write(NUM,fmt='(19I3)') ARRAYL(I)
XP=3*XMAX/4.
YP=YMAX-I*YMAX/10
if(OPTION .eq. 3) then
    COORD(1)=XP
    COORD(2)=YP+YMAX/100
    call polymarker2d(FILDES,COORD,1,0)
call text2d(FILDES,XP,YP,'  1='//NUM//NULL,VDC_TEXT,0)
else
call text2d(FILDES,XP,YP,'-- 1='//NUM//NULL,VDC_TEXT,0)
end if
80 continue

write(NUM,fmt='(19I3)') ARRAYL(I)
call line_color(FILDES,1.1.1.)
call text_color(FILDES,1.1.1.)
if(OPTION .eq.2) then
    call text2d(FILDES,0.,Y1,'To continue PRESS Return'//NULL,
+ VDC_TEXT,0)
FILEDS=gclose(FILEDS)
read(*,105) REPLY
    call dev_init(OPTION,FILEDS)
end if
    if(OPTION .eq.3) then
        FILEDS=gclose(FILEDS)
        call dev_init(OPTION,FILEDS)
    end if
* end of plot cosine components

65 if (OPTION .eq.1) then
    X1=real(XMIN)
    Y1=YMIN
    X2=real(2*XMAX+3*abs(XMIN))
    Y2=2*YMAX+3*abs(YMIN)
call SCALE(X1,Y1,X2,Y2,XB,OPTION)
call text_color(FILEDS,0,1,0.)
call text2d(FILEDS,4*X2,.45*Y2,NFILE//NULL,VDC_TEXT,0)
call text2d(FILEDS,0,YMIN,'To continue PRESS Return '//NULL,
+ VDC_TEXT,0)
call text_color(FILEDS,1,1,1.)
    if (NOTSIN .eq.1) then
        call text2d(FILEDS,3*X2,.7*Y2,
+ 'SINE COMPONENTS NOT AVAILABLE'//NULL,VDC_TEXT,0)
    end if
    if (NOTCOS .eq.1) then
        call text2d(FILEDS,3*X2,.25*Y2,
+ 'COSINE COMPONENTS NOT AVAILABLE'//NULL,VDC_TEXT,0)
    end if
FILEDS=gclose(FILEDS)
read(*,105) REPLY
    call dev_init(OPTION,FILEDS)
end if
    call clear_view_surface(FILEDS)
FILEDS=gclose(FILEDS)
LOAD=1
write(*,101)
call menu(OPTION,FILEDS)
end do
write(*,101)
stop

**************************************************************************
90 format(x,/'1A33,1A30,/
91 format(x,1A40,0)
101 format(x,50/)')
102 format('/x,.'Enter the name of the shell to be processed :'.S)
103 format(A14)
104 format('/x,'Is the above data correct? y(es),n(o) or s(top) :'.S)
105 format(A1)
106 format('/x,'Sine coefficients are read in from file : ','a27)
107 format(x,'Cosine coefficients are read in from file : ','a27)
108 format(x,'All needed variables are read in from file : ','a27)
108  format(25x,A25)
110  format(10x/))
111  format(x,'Error: file 'a27', not found')
200  format(1A25)
210  format(4I5)
401  format(8F9.5)
407  format(X,1A1,1A3)
408  format(X,1A6,1I3,1R1)
409  format(X,1A2,2(F7.3,'",'),1A1)
410  format(X,1A3Z)
411  format(X,1A2,2F7.3,1A4)
412  format(X,1A2,1I5,'",1F7.3,1A4)
413  format(X,1A25,1R1)
414  format(X,1A3,1F7.3,1A4)
end

***********************************************************************
* subroutine: scaling the device
* This subroutine scales the graf area, draws the x- and
* y-axis and writes the labels.
* Set character size according to OPTION.
***********************************************************************

subroutine SCALE(X1,Y1,X2,Y2,XB,OPTION)
  implicit none
  include '/usr/include/starbase.fl.h'
  include '/usr/include/starbase.2z.h'
  real*4 YMIN,YMAX,YY,XT,YT,X1,Y1,X2,Y2
  integer*2 XX,XB,XMAX,XMIN,I,OPTION
  integer*4 FILDES
  character*1 NULL
  character*25 SHELLCODE
  character*5 NUM
  parameter (NULL=char(0))

  common /COM1/ XMIN,XMAX,YMIN,YMAX,SHELLCODE,FILDES

  XX=XMAX+abs(XMIN)
  YY=YMAX+abs(YMIN)
  XT=YY/50.
  YT=XX/150.
  call vdc_extent(FILDES,X1,Y1,0.,X2,Y2,0.)
  if (OPTION .eq. 3) then
    * Set character size for plotter
    call character_width(FILDES,(X2-X1)/75.)
    call character_height(FILDES,(Y2-Y1)/35)
  else
    * Set character size for console
    call character_width(FILDES,2*(XMAX-XMIN)/75.)
    call character_height(FILDES,2*(YMAX-YMIN)/35)
  end if
  call move2d(FILDES,real(XMAX),0.)
  call draw2d(FILDES,0.,0.)
  call draw2d(FILDES,0.,YMAX)
  call draw2d(FILDES,real(XMAX),YMAX)
call move2d(FILDES,0.,0.)
do I=0,10
  * Tic marks on y-axis
  call move2d(FILDES,0.,I*YMAX/10)
  if(I.eq.5) then
    call draw2d(FILDES,2*YT,I*YMAX/10)
  else
    call draw2d(FILDES,YT,I*YMAX/10)
  end if
end do
call text_alignment(FILDES,TARIGHT,TASHALF,0.,0.)
do I=0,10,5
  * Labels on y-axis
  write(NUM,fmt='(1f4.2)') I*YMAX/10
  call text2d(FILDES,-YT+2,I*YMAX/10,NUM(1:5)//NULL,VDC_TEXT,0)
end do
call text_alignment(FILDES,TACENTER,TABOTTOM,0.,0.)
do I=0,YMAX
  * Draw tic marks and labels on x-axis
  call move2d(FILDES,real(I),YMAX)
  if(I.eq.5*int(I/5)) then
    write(NUM,fmt='(1I2)') I+XB
    call draw2d(FILDES,real(I),YMAX-XT)
    call text2d(FILDES,real(I),YMAX+XT/2,NUM(1:2)//NULL,VDC_TEXT,0)
  else
    call draw2d(FILDES,real(I),YMAX-XT/2)
  end if
end do
call text_alignment(FILDES,TANORMAL,HORIZONTAL,TA_NORMAL_VERTICAL,0.,0.)
return
end

*****************************************************************************
* subroutine menu
* This subroutine displays the menu of hargraf and
  read the users choice in the variable OPTION.
*****************************************************************************
subroutine menu(OPTION,FILDES)
  integer*2 OPTION
  integer*4 FILDES
  if (FILDES .ne. -1) write(*,1)
  if (FILDES .eq. -1) write(*,2)
  write(*,10)
  read(*,20) OPTION
return

1  format(9(/))
2  format(3(/),'Your choice is not available !',5(/))
10  format(8x,'MENU HARGRAPH :',//,24x,
    +' 1 - four plots on console',//,24x,
    +' 2 - single plots on console',//,24x,
    +' 3 - plots on plotter halsys',//,24x,
    +' 4 - quit',8(/),'Your choice is --> ',$)
20  format(12)
end
subroutine dev_init
Subroutine to open the output device for graphics.

subroutine dev_init(OPTION,FILDES)
include'/usr/include/starbase.f1.h'
include'/usr/include/starbase.f2.h'
character NULL
integer*2 OPTION
integer*4 FILDES
parameter (NULL=char(0))

if (OPTION .lt. 3) then
   FILDES=gopen('/dev/crt'/NULL,OUTDEV,'hp3001'/NULL,INIT)
   if(FILDES .eq. -1) then
      FILDES=gopen('/dev/crt'/NULL,OUTDEV,'hp98550'/NULL,INIT)
      end if
   end if
if (OPTION .eq. 3) then
   FILDES=gopen('/dev/hp7550'/NULL,OUTDEV,'hpgl'/NULL,INIT)
end if
call clear_view_surface(FILDES)
call mapping_mode(FILDES,1)
return
end

subroutine color
This subroutine set the line- and text-colors.

subroutine color(FILDES,I)
include'/usr/include/starbase.f1.h'
include'/usr/include/starbase.f2.h'
integer*4 FILDES
integer*2 I
if(I.eq.1) call line_color(FILDES,1,2.,0.)
if(I.eq.2) call line_color(FILDES,0,1.,0.)
if(I.eq.3) call line_color(FILDES,0,7.,1.)
if(I.eq.4) call line_color(FILDES,1,1.,0.)
if(I.eq.1) call text_color(FILDES,1,2.,0.)
if(I.eq.2) call text_color(FILDES,0,1.,0.)
if(I.eq.3) call text_color(FILDES,0,7.,1.)
if(I.eq.4) call text_color(FILDES,1,1.,0.)
if(I.eq.1) call marker_color(FILDES,1,2.,0.)
if(I.eq.2) call marker_color(FILDES,0,1.,0.)
if(I.eq.3) call marker_color(FILDES,0,7.,1.)
if(I.eq.4) call marker_color(FILDES,1,1.,0.)

return
end

subroutine selection
In this subroutine the users can set new half-wave numbers k
and full-wave numbers \( l \).

```
subroutine selection
  integer*2 SEL,I
  integer*2 ARRAYK(4),ARRAYL(4),NLINEPLOT
  integer*2 DARRAYK(4),DARRAYL(4),ONLINEPLOT
  common /COM2/ ARRAYK,ARRAYL,NLINEPLOT,
        DARRAYK,DARRAYL,ONLINEPLOT

  SEL=0
  do while (SEL .lt. 4)
    write(*,106)
    write(*,100) 'Selected half-wave components \( k \) are :'
    do I=1,NLINEPLOT
      write(*,101) ARRAYK(I)
    end do
    write(*,100) 'Selected full-wave components \( l \) are :'
    do I=1,NLINEPLOT
      write(*,101) ARRAYL(I)
    end do
    write(*,100) 'Default components \( k \) are :'
    do I=1,NLINEPLOT
      write(*,101) DARRAYK(I)
    end do
    write(*,100) 'Default components \( l \) are :'
    do I=1,NLINEPLOT
      write(*,101) DARRAYL(I)
    end do
    write(*,102)
    read(*,103) SEL
    if (SEL .eq. 1 .or. SEL .eq. 2) then
      write(*,100) 'Number of lines to plot (max. 4) --> '
      read(*,103) NLINEPLOT
      write(*,107)
    end if
    if (SEL .eq. 1) then
      do I=1,NLINEPLOT
        write(*,104)
        read(*,103) ARRAYK(I)
      end do
    end if
    if (SEL .eq. 2) then
      do I=1,NLINEPLOT
        write(*,105)
        read(*,103) ARRAYL(I)
      end do
    end if
    if (SEL .eq. 3) then
      NLINEPLOT=ONLINEPLOT
      do I=1,NLINEPLOT
        ARRAYK(I)=DARRAYK(I)
        ARRAYL(I)=DARRAYL(I)
      end do
    end if
  end do
```
return

100  format(x,,,1A40,$)
101  format(x,,,112,$)
102  format(/,,4x,'You want to make changes in :',/,,10x,
+\'1 - selected half-wave components k',/,,10x,
+\'2 - selected full-wave components 1',/,,10x,
+\'3 - set k and 1 to the default values',/,,10x,
+\'4 - no (more) changes',/,,4x,
+\'Your choice is --> ',$)
103  format(/112)
104  format(x,,k --> ',$)
105  format(x,,l --> ',$)
106  format(/10(/))
107  format(/)

end

******************************************************************************
* subroutine marker type
******************************************************************************

subroutine marker(FILDES,I)
include'/usr/include/starbase.f1.h'
include'/usr/include/starbase.f2.h'
integer*4 FILDES
integer*2 I

if(I ,eq. 1) call marker_type(FILDES,I)
if(I ,eq. 2) call marker_type(FILDES,I+1)
if(I ,eq. 3) call marker_type(FILDES,I+2)
if(I ,eq. 4) call marker_type(FILDES,I+3)
call marker_size(FILDES,,3,0)

return
end

******************************************************************************
Feb 15 11:02 1980  axigraph.f  Page 1

program axigraph

******************************************************************************
* Program made by A.W.H. klompe - januari 1990.                          *
* Program to plot the axisymmetric fourier components                    *
* A(i,0) and C(i,0) of a maximum of 4 shells per plot.                    *
* After any modification compile with:                                    *
* fc -o axigraph axigraph.f -ldd98550 -lddhpgl -ldd3001 -ldvio -lsb1 -lsb2 *
* and run program axigraph                                                *
* This program prompts the user to enter:                                 *
* - the name of the output device                                         *
* - the number of shells to be plot in axi-symmetric cross plots          *
* - the shell identification codes.                                       *
* - the users choice: a plot of A(i,0) or C(i,0) or both components.      *
* - the length of the X-axis.                                             *
* - the length of the Y-axis.                                             *
******************************************************************************

implicit none
include'/usr/include/starbase.f1.h'
include'/usr/include/starbase.f2.h'
integer*2 NP,PL,OPTION,I,IOS
integer*2 K1,K2,L1,L2,KK1,K,L
integer*2 XMIN,XMAX,XB,SEL,MAX
integer*4 FILDES
real*4 YMIN,YMAX,X1,Y1,X2,Y2,XP,YP
real*4 A(0:100,0:100),COORD(2)
character*1 REPLY,NULL
character*25 SHELLCODE,MEASUREDATE
character*27 NFILE(4),FILEHC,FILEHS,HFILE
logical*2 AXI,COS,LOAD
parameter (NULL=char(0))
common /COMS/ XMIN,XMAX,YMIN,YMAX,SHELLCODE
common /COMP/ PL,NP,A,K2,COS

LOAD=.FALSE.
write(*,92)
call menu(FILDES,OPTION)
write(*,92)
do while(OPTION .lt. 3)
  if (LOAD) then
    write(*,*) 'Already prepared the shells : '
    do I=1,NP
      write(*,90) NFILE(I)
    end do
    write(*,91) 'You want to use the same shells ? -->'
    read(*,107) REPLY
    if (REPLY .eq. 'y' .or. REPLY .eq. 'Y') then
...
write(*,92)
goto 25end if
end if
10 write(*,100)
read(*,101) NP
if (NP.gt.4) then
  write(*,102)
goto 10
end if
PL=1
do while(PL .le. NP)
  write(*,104) PL
  read(*,105) NFILE(PL)
  write(*,106) NFILE(PL)
  read(*,107) REPLY
  if(REPLY.eq.'n' .or. REPLY.eq.'N') goto 20
  if(REPLY.ne.'y' .and. REPLY.ne.'Y') goto 20
  PL=PL+1
end do
write(*,108)
read(*,101) SEL
if(SEL.eq.0 .or. SEL.eq.1) COS=.TRUE.
PL=1
do while(PL .le. NP)
  I=1
  HFILE=NFILE(PL)
  do while(ichar(HFILE(I:I)) .gt. 33)
    I=I+1
  end do
  if(COS) then
    FILEHC='/users/data/'//HFILE(I:I-1),//'hc'
    open(11,status='old',file=FILEHC,err=21,iostat=IOS)
    continue
  end if
  if(IOS.ne.0) then
    print 113,FILEHC
  end if
else
  FILEHS='/users/data//'//HFILE(I:I-1)//'hs'
  open(11,status='old',file=FILEHS,err=22,iostat=IOS)
  continue
  if (IOS.ne.0) then
    print 113,FILEHS
  end if
end if
if(IOS.ne.0) then
  write(*,*) 'Selected files are :'
  do I=1,NP
    write(*,90) NFILE(I)
  end do
  goto 1
end if
PL=PL+1
end do
LOAD=.TRUE.
25 continue
write(*,110)
read(*,101) XMAX
write(*,111)
read(*,112) YMAX
if(YMAX .lt. .001) YMAX=1000000*YMAX
AXI=.TRUE.
XMIN=XMAX/5
YMIN=-2*YMAX/10.
write(*,93)
write(*,94) 'EXECUTING THE PLOTS OF THE *
write(*,94) 'AXISYMMETRIC COMPONENTS *
write(*,93)

do while(AXI)
   PL=1
   30 do while(PL .le. NP)
      I=1
      HFIL=NFIL(PL)
      do while(ichar(HFIL(I:I)) .gt. 33)
         I=I+1
      end do
      if(COS) then
         FILEHC='/users/data/''HFIL(1:I-1)/''hc'
         open(11,status='old',file=FILEHC)
      else
         FILEHS='/users/data/''HFIL(1:I-1)/''hs'
         open(11,status='old',file=FILEHS)
      end if
      read(11,200) SHELLCODE
      read(11,200) MEASUREDATE
      read(11,201) K1,K2,L1,L2
      if (COS) then
         KK1=1
      else
         if(K1.eq.0) then
            KK1=1
         else
            KK1=K1
         end if
      end if
      read(11,210) ((A(K,L),L=L1,L2),K=KK1,K2)
      close(11)
      if(PL .eq. 1) call dev_init(FILDES,OPTION)
      X1=real(XMIN)
      Y1=YMIN
      X2=real(XMAX+abs(XMIN))
      Y2=YMAX+abs(YMIN)
      XB=K1
      if(PL.eq.1) call scale(FILDES,X1,Y1,X2,Y2,XB,OPTION)
      call color(FILDES,PL)
      call marker(FILDES,PL)
      if(K2.gt.XMAX) then
MAX=XMAX
else
    MAX=KZ
end if

call move2d(FILDES,0.,abs(A(0,0)))
do I=1,MAX
    call draw2d(FILDES,real(I),abs(A(I,0)))
    if(OPTION.eq.2) then
        COORD(1)=real(I)
        COORD(2)=abs(A(I,0))
        call polymarker2d(FILDES,COORD,1,0)
        call move2d(FILDES,real(I),abs(A(I,0)))
    end if
end do

XP=3*XMAX/4.
YP=YMAX-NP+YMAX/10
if(OPTION.eq.2) then
    COORD(1)=XP
    COORD(2)=YP+YMAX/100
    call polymarker2d(FILDES,COORD,1,0)
    call text2d(FILDES,XP,YP,'"SHELLCODE(1:6)"//NULL,' +
        VDC_TEXT,0)
else
    call text2d(FILDES,XP,YP,'"SHELLCODE(1:6)"//NULL,' +
        VDC_TEXT,0)
end if

if(COS .and. PL.eq.1) then
    call text_color(FILDES,1.1.1.1.)
    call text2d(FILDES,real(XMAX/5),YMIN/2,
        'cosine axisym. components'"//NULL,VDC_TEXT,0)
else
    if(PL.eq.1) then
        call text_color(FILDES,1.1.1.1.)
        call text2d(FILDES,real(XMAX/5),YMIN/2,
            'sine axisym. components'"//NULL,VDC_TEXT,0)
    end if
end if

if(PL.eq.NP) then
    if(OPTION.eq.1) then
        call text_color(FILDES,1.1.1.1.)
        call text2d(FILDES,0.,Y1,'To continue PRESS Return'"//NULL,
            VDC_TEXT,0)
        FILDES=gclose(FILDES)
        read(*,107) REPLY
        call dev_init(FILDES,OPTION)
    else
        FILDES=gclose(FILDES)
        call dev_init(FILDES,OPTION)
    end if
end if

PL=PL+1
if(PL.gt.NP) then
    if(COS) then
        if(SEL.eq.0) then
```fortran
COS=.FALSE.
PL=1
else
   AXI=.FALSE.
end if
else
   AXI=.FALSE.
end if
end if
end do
* end while(PL)
end do
* end while(AXI)
call menu(FILDES,OPTION)
write(*,92)
end do
* end while(OPTION)

*-----------------------------
* Format statements
*-----------------------------
90 format(10x,1A14)
91 format(x,1A40,$)
92 format(x,50(//))
93 format(x,10(//))
94 format(25x,A40)
100 format(//,x,'Enter NUMBER OF SHELLS to be plot in',
      1' axisymmetric cross plots --> ','$)
101 format(I3)
102 format(//,x,'Number of shells to LARGE !')
104 format(//,x,'Enter SHELL IDENTIFICATION CODE for shell',
      1' number ',I1,' -> ','$)
105 format(A14)
106 format(//,x,'Is the SHELL IDENTIFICATION CODE : ',A10,' correct ?',
      1'( y or n ) -> ','$)
107 format(A1)
108 format(50(//),10x,'MENU COMPONENTS:/',
      1 '//,10x,0 - sine and cosine components',
      2//' ,10x,1 - cosine components only',
      3//' ,10x,2 - sine components only',
      4//' ,5x,'Your choice --> ','$)
110 format(5(//),10x,'Length of the X-axis (integer) --> ','$)
111 format(5(//),10x,'Length of the Y-axis',
      1 '+ ( 1000 > Length => .001 ) --> ','$)
112 format(1F12.6)
113 format(15(//),'ERROR : file for shell ',A18,' not found.',5(//))
200 format(A125)
201 format(A125)
210 format(8F9.5)
end

* subroutine scaling the plotter
*---------------------------------------------------------
subroutine scale(FILDES,X1,Y1,X2,Y2,XB,OFTION)
```
implicit none
include '/usr/include/starbase.f1.h'
include '/usr/include/starbase.f2.h'
real*4 YMIN,YMAX,YY,XT,YT,X1,Y1,X2,Y2
integer*2 XX,XMAX,XMIN,I,OPTION,XB
integer*4 FILDES
character*1 NULL
character*5 NUM
character*25 SHELLCODE
parameter (NULL=char(0))
common /COMS/ XMIN,XMAX,YMIN,YMAX,SHELLCODE

XX=XMAX+abs(XMIN)
YY=YMAX+abs(YMIN)
XT=YY/50.
YT=XX/150.
call vdc_extent(FILDES,X1,Y1,0..XZ,Y2,0.)
if(OPTION .eq. 2) then
  set character size for plotter
  call character_width(FILDES,(XZ-X1)/75.)
  call character_height(FILDES,(Y2-Y1)/35)
else
  set character size for console
  call character_width(FILDES,2*(XMAX-XMIN)/75.)
  call character_height(FILDES,2*(YMAX-YMIN)/35)
endif

call move2d(FILDES,real(XMAX),0.)
call draw2d(FILDES,0..0.)
call draw2d(FILDES,0..YMAX)
call draw2d(FILDES,real(XMAX),YMAX)
call move2d(FILDES,0..0.)
do I=1,10

  tic mark on y-axis

call move2d(FILDES,0..I*YMAX/10)
if (I .eq. 5) then
  call draw2d(FILDES,2*YT,I*YMAX/10)
else
  call draw2d(FILDES,YT,I*YMAX/10)
endif
endo

call text_alignment(FILDES,TA_RIGHT,TA_HALF,0..0.)
do I=0,10,5

  labels on y-axis
  write(NUM,fmt='(1f5.2)') real(I*YMAX/10)
call text2d(FILDES,-YT*2,I*YMAX/10,NUM(1:5)//NULL,VDC_TEXT,0)
endo

call text_alignment(FILDES,TA_CENTER,TA_BOTTOM,0..0.)
do I=0,XMAX

  draw tic marks and labels on x-axis
  call move2d(FILDES,real(I),YMAX)
if (I .eq. 5*int(I/5)) then
  write(NUM,fmt='(1I2)') I+XB
call draw2d(FILDES,real(I),YMAX-XT)
call text2d(FILDES,real(I),YMAX+XT/2,NUM(1:2)//NULL,VDC_TEXT,0)
else
call draw2d(FILDES,real(I),YMAX-XT/2)
end if
end do
call text_alignment(FILDES,TA_NORMAL_HORIZONTAL,  
+TA_NORMAL_Vertical,0.,0.)
call text_color(FILDES,1.,1.,1.)
return
end

*****************************************************************************
* subroutine color
* This routine set the line- and text- color
*****************************************************************************
subroutine color(FILDES,PL)
implicit none
include'/usr/include/starbase.f1.h'
include'/usr/include/starbase.f2.h'
integer*2 PL
integer*4 FILDES

if(PL.eq.1)call line_color(FILDES,1.,2.,0.)
if(PL.eq.2)call line_color(FILDES,0.,1.,0.)
if(PL.eq.3)call line_color(FILDES,0.,7.,1.)
if(PL.eq.4)call line_color(FILDES,1.,1.,0.)
if(PL.eq.1)call text_color(FILDES,1.,2.,0.)
if(PL.eq.2)call text_color(FILDES,0.,1.,0.)
if(PL.eq.3)call text_color(FILDES,0.,7.,1.)
if(PL.eq.4)call text_color(FILDES,1.,1.,0.)
if(PL.eq.1)call marker_color(FILDES,1.,2.,0.)
if(PL.eq.2)call marker_color(FILDES,0.,1.,0.)
if(PL.eq.3)call marker_color(FILDES,0.,7.,1.)
if(PL.eq.4)call marker_color(FILDES,1.,1.,0.)
return
end

*****************************************************************************
* subroutine dev_init
* Subroutine to open the output device for graphics
*****************************************************************************
subroutine dev_init(FILDES,OPTION)
implicit none
include'/usr/include/starbase.f1.h'
include'/usr/include/starbase.f2.h'
character*1 NULL
integer*2 OPTION
integer*4 FILDES

parameter (NULL=char(0))

if(OPTION .eq. 1) then
  FILDES=gopent('/dev/crt'/'NULL,OUTDEV,'hp98550'/'NULL,INIT)
if(FILDES .eq. -1) then
  FILDES=gopent('/dev/crt'/'NULL,OUTDEV,'hp3001'/'NULL,INIT)
end if
end if
if(OPTION .eq. 2) then
    FILDES=gopen('/dev/hp7550'//NULL,OUTDEV,'hpgl'//NULL,INIT)
end if
call clear_view_surface(FILDES)
call mapping_mode(FILDES,1)
return
end

******************************************************************************
* subroutine marker type
******************************************************************************
subroutine marker(FILDES,I)
implicit none
include'/usr/include/starbase.f1.h'
include'/usr/include/starbase.f2.h'
integer*2 I
integer*4 FILDES

if(I .eq. 1) call marker_type(FILDES,I)
if(I .eq. 2) call marker_type(FILDES,I+1)
if(I .eq. 3) call marker_type(FILDES,I+2)
if(I .eq. 4) call marker_type(FILDES,I+2)
call marker_size(FILDES,.3,0)
return
end

******************************************************************************
* subroutine menu axiplot
* This subroutine displays the menu of axiplot and reads
* the users choice in the variable OPTION.
******************************************************************************
subroutine menu(FILDES,OPTION)
implicit none
integer*2 OPTION
integer*4 FILDES
if(FILDES .ne. -1) write(*,1)
if(FILDES .eq. -1) write(*,2)
write(*,10)
read(*,20) OPTION
return
1  format(9(/))
2  format(3(/),'Your choice is not available ! ','5(/))
10  format(8x,'MENU AXIPL T :','','24x,'
          +'1 - Plot on console', ''24x,
          +'2 - Plot on plotter halsys', ''24x,
          +'3 - Quit',8(/),'Your choice is -- > ','$)
20  format(I2)
end
Program graph3d

The output of this program is a 3-dimensional plot of
the imperfection via starbase

Maximum of 400 axial stations and 400 circumferential
stations, if more stations are needed change the dimensions
of the arrays X and Y and compile this program again.

After any modification compile with:
fc -o graph3d graph3d.f -lidd98550 -liddhpgl -lidd300l -ldvio -lslbl -lslb2
and run program graph3d

INPUT VARIABLES in file [shellcode].var:
list1
SHELLCODE    - identification code for the used shell.
MEASUREDATE   - date of imperfection measurement.
NSCAN         - total numbers of scans,
               including the number of missing scans,
               excluding the number of control scans.
NPOINT        - number of data points per scan.
CONICITY      - cone angle of shell, measured in degrees,
               positive if R(top) < R(bottom).
THICKNESS     - wall thickness of the shell (in mm).
RADIUS        - radius of the shell (in mm).
LENGTH        - length of measured part of the shell (in mm).
SCANTYPE      - =0 => scans are oriented in axial direction,
               =1 => scans are oriented in circumf. direction.
               If SCANTYPE=1, hatches and extra plating cannot be
               properly 'placed'.
               In that case write a small program to put the
               available data in axial representation first.

list10
PLOTINCR      - plot interval in axial datapoints.
PLOTCRIT      - plot criterion:
               =0 => pen lift if abs(W) > 3*THICKNESS
               else => pen lift if abs(W) > PLOTCRIT*THICKNESS.
SCALESET      - =0 => default scale setting
               =1 => user defined scale setting.

list11
YMIN          - user defined Y minimum.
YMAX          - user defined Y maximum.

implicit none
include '/usr/include/starbase.f1.h'
include '/usr/include/starbase.f2.h'
real*4 X(400,400), Y(400,400)
real*4 CONICITY, DOX, DOY, DX, LENGTH, RADIUS, THICKNESS, XMIN, XMAX, YMIN
real*4 YMAX,XT,YT
integer*2 I,IOS,J,MARK_B,NA,NL,NPOINT,NSCAN,OUTCRIT,OPTION
integer*2 PLOTCRIT,PLOTINC,SCALESET,SCANNNUMBER,SCANTYPE,CALCDONE
integer*4 FILDES
character*1 REPLY,YNUM
character*16 NFILE
character*25 MEASUREDATE,SHELLCODE
character*34 FILE,FILEVAR
character NULL
parameter (NULL=char(0))

******************************************************************************
* Reading the name of the file to be processed.                                *
* Opening of the files (shellcode) or (shellcode)c or (shellcode)f          *
* and (shellcode)var.                                                       *
******************************************************************************

OPTION=0
CALCDONE=0

call menu(OPTION)
do while(OPTION .lt. 3)
  if (CALCDONE .eq. 1) then
    write(*,90) 'Already prepared is data file : ',NFILE
    write(*,91) 'You want to use the same data file? --> ',REPLY
    read(*,105) REPLY
    if (REPLY.eq.'y'.or.REPLY.eq.'Y') goto 20
  end if
  CALCDONE=0
  write(*,101)
  write(*,102)
  I=1
  read(*,103) NFILE
  do while(ichar(NFILE(I:I)).gt.33)
    I=I+1
  end do
  FILE='/users/data/''/NFILE(I:I-1)
  FILEVAR='/users/data/''/NFILE(I:I-2)//'var'
  write(*,106) FILE
  write(*,107) FILEVAR
  write(*,104)
  read(*,105) REPLY
  if (REPLY.eq.'n'.or.REPLY.eq.'N') goto 10
  if (REPLY.ne.'y'.and.REPLY.ne.'Y') stop
  open (10,status='old',file=FILE,iostat=IOS)
  if (IOS.ne.0) then
    print 111,FILE
  goto 10
  endif
  I=16
  do while(ichar(NFILE(I:I)).lt.47.or.ichar(NFILE(I:I)).gt.58)
    I=I-1
  end do
  if (NFILE(I+1:I+1).eq.'b') then
    MARK_B=1
  else
MARQ_B=0
endif

write(*,108)
write(*,109) '***********************************************'
write(*,109) '*     CALCULATING THE COORDINATES     '*
write(*,108) '*      FOR THE 3D- PLOT       '*
write(*,108) '***********************************************'
write(*,108)

90 format(x,//,1A33,1A30,/) 
91 format(x,1A40,$)
101 format(50(//))
102 format(x,'Enter the name of the FILE to be plotted: ',$)
103 format(A14)
104 format(x,'Is the above data correct? y(es),n(no) or s(top) : ',$,)
105 format(A1)
106 format(x,'Data is read in from file: ',a27)
107 format(x,'All needed variables are read from file: ',a27)
108 format(10(//))
109 format(15x,A40)
111 format(x,'Error: file ',a27,' not found')

***********************************************
read input variables
***********************************************
namelist /list1/ SHELLCODE,MEASUREDATE,SCAN,NPOINT,CONICITY,
THICKNESS,RADIUS,LENGTH,SCANTYPE
namelist /list10/ PLOTINCR,PLOTCRIT,SCALESET
namelist /list11/ YMIN,YMAX

20 continue
open(15,status='old',file=FILEVAR)
read(15,list1)
read(15,list10)
if (SCALESET.ne.0) read(15,list11)
close(15)

NA=2*int((NPOINT-1)*0.5)
NL=int((NA+1)/PLOTINCR)
if (SCALESET.eq.0) then
  YMIN=-2.*abs(THICKNESS)
  YMAX=12.*abs(THICKNESS)
endif
XMIN=-20.0
XMAX=480.0
ODX=2*360.0/(7*(NL-1))
ODY=(YMAX-abs(YMIN))/(NL-1)
if (CALCDONE .eq. 1) goto 30

***********************************************
read input data
***********************************************
I=1
do while (ichar(SHELLCODE(I:I)).le.33)
  I=I+1
end do
if (SCANTYPE.eq.0) then

*************** AXIAL DATA ***************

do J=1,NSCAN
  read(10,100) SCANNUMBER
  if (MARQ_B.eq.1) read(10,112)
  read(10,116) (W(I,J),I=1,NPOINT)
end do

*************** CIRCUMFERENTIAL DATA ***************

else if (SCANTYPE.eq.1) then
  do I=1,NSCAN
    read(10,100) SCANNUMBER
    if (MARQ_B.eq.1) read(10,112)
    read(10,116) (W(I,J),J=1,NPOINT)
  end do
  J=NPOINT
  NPOINT=NSCAN
  NSCAN=J
end if

close(10)

*************** GENERATING THE X-COORDINATES ***************

DX=360.0/NSCAN
do I=NA,1,-PLOTINCR
  X(I,1)=0.0
  do J=2,NSCAN+1
    X(I,J)=X(I,J-1)+DX
  end do
  if (CONICITY.ne.0.) then
    DX=DX-(DDX*1.5)/NSCAN
  endif
end do
if (CONICITY.ne.0.) DX=DX+(DDX*1.5)/NSCAN
CALCDONE=1

*************** EXECUTING THE PLOTS ***************

* Scaling
30 continue
if (PLOTCRIT.eq.0) then
  PLOTCRIT=3*abs(THICKNESS)
else
  PLOTCRIT=PLOTCRIT*abs(THICKNESS)
endif
call scinit(FILDES,OPTION)
call mapping_mode(FILDES,1)
call vdc_extent(FILDES,XMIN,YMIN,0.,XMAX,YMAX,0.)

* x and y axis
call line_color(FILDES,0.,1.,0.)
call text_color(FILDES,0.,1.,0.)
call character_width(FILDES,(XMAX-XMIN)/75)
call character_height(FILDES,(YMAX-YMIN)/35)
YT=(XMAX-XMIN)/75
XT=(YMAX-YMIN)/100

call move2d(FIODES,0.,0.)
call draw2d(FIODES,NL*DDX,NL*DDY)
call draw2d(FIODES,NL*DDX+NSCAN*DX,NL*DDY)
call draw2d(FIODES,360.,0.,0.)
call move2d(FIODES,0.,0.)
call draw2d(FIODES,380.,0.,0.)
call move2d(FIODES,360.,0.,0.)
call draw2d(FIODES,380.,-XT)
call text2d(FIODES,350.,-5*XT,'360'//NULL,VDC_TEXT,0)
call move2d(FIODES,0.,0.)
call draw2d(FIODES,0.,4*THICKNESS)

do I=0,3
  call move2d(FIODES,0.,I*THICKNESS)
call draw2d(FIODES,-YT/2,I*THICKNESS)
write(YNUM,fmt='(1I)') I
  call text2d(FIODES,-2*YT,I*THICKNESS,YNUM(1:1)//NULL,VDC_TEXT,0)
end do

call move2d(FIODES,-YT/2,4*THICKNESS-2*XT)
call draw2d(FIODES,0.,0.4*THICKNESS)
call draw2d(FIODES,YT/2,4*THICKNESS-2*XT)

call line_color(FIODES,1.,1.,1.,1.)
call text_color(FIODES,1.,1.,1.,1.)
call character_width(FIODES,(XMAX-XMIN)/50)
call character_height(FIODES,(YMAX-YMIN)/25)
call text2d(FIODES,0.,0.,YMAX-THICKNESS,NFILE//NULL,VDC_TEXT,0)
if (OPTION .eq. 1) then
  call text2d(FIODES,0.,-abs(THICKNESS)*1.25,
+ 'To continue PRESS Return'//NULL,VDC_TEXT,0)
end if

* actual plotting

do 00 I=NA,1,-PLOTPRINT
  if (1.1e.PLOTPRINT .or. I.eq.NA) then
    call move2d(FIODES,0.,0.)
    if(abs(W(I,1)).lt.PLOTCRIT) then
      call draw2d(FIODES,X(I,1),W(I,1)/abs(THICKNESS))
    end if
  end if
  OUTCRIT=0

  do J=1,NSCAN
    if (abs(W(I,J)).lt.PLOTCRIT) then
      if(J.eq.1 .or. OUTCRIT.eq.1) then
        call move2d(FIODES,X(I,J),W(I,J)/abs(THICKNESS))
        OUTCRIT=0
      else
        call draw2d(FIODES,X(I,J),W(I,J)/abs(THICKNESS))
      endif
    else
      OUTCRIT=1
      call move2d(FIODES,X(I,J),W(I,J)/abs(THICKNESS))
    endif
  enddo

}
Mar 27 08:24 1990  graph3d.f Page 6

end do

  call draw2d(FILDES,X(I,NSCAN+1),W(I,1)/abs(THICKNESS))
  if(I.eq.PLOTINCR.or.I.eq.NA) call draw2d(FILDES,X(I,NSCAN+1),0.0)
  XMIN=XMIN-DDX
  XMAX=XMAX-DDX
  YMIN=YMIN-DDY
  YMAX=YMAX-DDY
  call vdc_extent(FILDES,XMIN,YMIN,0.,XMAX,YMAX,0.)

80 continue

  FILDES=gclose(FILDES)
  if (OPTION .eq. 2) goto 85
  read(*,112) REPLY
  call scrint(FILDES,OPTION)

85 continue

  call menu(OPTION)
  end do

  write(*,480)

  stop

* 100 format(I5)
  112 format(A)
  116 format(BF9.5)
  400 format(X,1A2,3(IF10.3,"",IF10.3,1A1))
  401 format(X,1A1,1A3)
  410 format(X,A72)
  420 format(X,1A3,1F7.3,1A4)
  421 format(X,1A9,2R1)
  430 format(X,1A2,1F10.3,"",1F10.3,1A4)
  440 format(X,1A2,1F7.3,1A2)
  450 format(X,1A9,2F7.3,1A1)
  460 format(X,1A2,' contour ',1A6,R1)
  470 format(X,1A2,R1)
  480 format(X,24(/))
end

******************************************************************************

* subroutine menu
******************************************************************************

  subroutine menu(OPTION)
    integer*2 OPTION
    write(*,10)
    read(*,20) OPTION
    return
  end

10   format(9(/),8x,'MENU PLOT-3d :'//,24x,
      1'1' - plot on console '//,24x,
      2'2' - plot on plotter halsys',//,24x,
      3'3' - quit',8(/),8x,'Your choice is --> ',$,)
20   format(I2)
end

******************************************************************************

* subroutine scrint
******************************************************************************

  subroutine scrint(FILDES,OPTION)
include'usr/include/starbase.f1.h'
include'usr/include/starbase.f2.h'
character NULL
integer*2 OPTION
integer*4 FILDES
parameter (NULL=char(0))

if(OPTION.eq.1) then
   FILDES=gopen('/dev/crt'/NULL,OUTDEV,'hp98550'/NULL,INI)
   if(FILDES.eq.-1) then
      FILDES=gopen('/dev/crt'/NULL,OUTDEV,'hp3001'/NULL,INI)
   end if
end if
if(OPTION.eq.2) then
   FILDES=gopen('/dev/hp7550'/NULL,OUTDEV,'hpgl'/NULL,INI)
end if
call clear_view_surface(FILDES)
call mapping_mode(FILDES,1)
return
der

***************************************************************************
Mar 27 00:25 1990 loadvar Page 1

# *********************************************************
# Program LOADVAR
# This program load the file [shellcode]var for all parameters
# or for only some of them.
# Program made by Ch. Cartalas - augustus 1989
# Small modifications done by A.W.H. Klompe - october 1989
# Program uses program loadvar.f
# *********************************************************

fc -o /tmp/loadvar /users/datarad/odatarad/loadvar.o
/tmp/loadvar
rm /tmp/loadvar
# End of program loadvar
program loadvarf

**************************
Program LOADVAR

This program load the file [shellcode]var for all parameters
or for only some of them.

Program made by Ch. Cartalas - augustus 1989
Small modifications done by A.W.H. Klompe - october 1989

Program is called by loadvar

**************************

implicit integer*4 (A-Z)
character*77 STRING(400),ST,TEXT
character*35 A(400)
character*32 FILE
character*15 NFILE,NAME
character*15 NAME1
character*71 REP
character*17 TEXT1

I = 1
MARQ = 0
CHOICE = 1
K = 3
write(*,5)
read(*,14) NFILE
do while(NFILE(I:I) .ne. ' ' .and.I.lt.14)
   I=I+1
end do
I=I-1
FILE = '/users/data//'NFILE(I:I)//'var'
open (unit=20,status='old',file=FILE,iostat=IOS)
if (IOS.ne.0) then
   MARQ = 1
   open (unit=20,status='new',file=FILE,iostat=IOS)
   write(*,25) FILE(I:I+15),FILE(I:I+15)
   if (IOS .ne.0) write(*,26) FILE
   open (unit=15,status='old',file='/users/data/listvar')
end if

do while(CHOICE.le.4)
   write(*,50)
   read(*,55,err=13) CHOICE

Listing of parameters

if (CHOICE.eq.1) then
   rewind(unit=20,iostat=IOS)
   INDIC = 1
   do while(IOS.eq.0)
      read(20,14,iostat=IOS) A(INDIC)
      if( A(INDIC).ne."$end",and.
         1        A(INDIC)( :3).ne."$ ") INDIC = INDIC + 1
end do
write(*,80)
do L = 1,50
write(*,100) A(L),A(L+50)
end do
write(*,110)
read(*,*)
end if

* Modify one or several parameters

if (CHOICE.eq.2.or.CHOICE.eq.3) then
  rewind(unit=20,iostat=IOS)
  I = 1
  INDIC = 1
  write(*,60)
  read(*,14) NAME1
  do while(ichar(NAME1(I:I)).gt.33)
    if(ichar(NAME1(I:I)).gt.96) then
      NAME1(I:I)=char(ichar(NAME1(I:I))-32)
    end if
    I=I+1
  end do
  I = I-1
  NAME = ' '//NAME1(I:I)//'='
  if(I.le.0) I = 5
  TEXT(2:4) = "__"
  do while(TEXT(2:I+2).ne.NAME(2:I+2).and.IOS.eq.0)
    read(20,14,iostat=IOS) TEXT
  end do
  if(TEXT(2:I+1).eq.NAME(2:I+1)) then
    write(*,33) TEXT
  if(CHOICE.eq.2) write(*,61)
  if(CHOICE.eq.3) then
    if(I+2.eq.3) write(*,203) TEXT(I:I+1)
    if(I+2.eq.4) write(*,204) TEXT(I:I+1)
    if(I+2.eq.5) write(*,205) TEXT(I:I+1)
    if(I+2.eq.6) write(*,206) TEXT(I:I+1)
    if(I+2.eq.7) write(*,207) TEXT(I:I+1)
    if(I+2.eq.8) write(*,208) TEXT(I:I+1)
    if(I+2.eq.9) write(*,209) TEXT(I:I+1)
    if(I+2.eq.10) write(*,210) TEXT(I:I+1)
    if(I+2.eq.11) write(*,211) TEXT(I:I+1)
    if(I+2.eq.12) write(*,212) TEXT(I:I+1)
    if(I+2.eq.13) write(*,213) TEXT(I:I+1)
    if(I+2.eq.14) write(*,214) TEXT(I:I+1)
    if(I+2.eq.15) write(*,215) TEXT(I:I+1)
    if(I+2.eq.16) write(*,216) TEXT(I:I+1)
    if(I+2.eq.17) write(*,217) TEXT(I:I+1)
    if(I+2.eq.18) write(*,218) TEXT(I:I+1)
  end if
  read(*,14) REP
  do while(IOS.eq.0)
    read(20,14,iostat=IOS) STRING(INDIC)
    INDIC = INDIC + 1
  end do
do L = 1,INDIC
   backspace(unit=20)
end do
C=1
do while(NAME(C:C).ne."=".and.C.lt.16)
   C=C+1
end do
D=1
do while(REP(D:D+Z).ne." ".and.J.lt.72)
   D=D+1
end do
D=D-1
write(20,14) NAME(1:C)//REP(1:D)
if(CHOICE.eq.2) write(*,40) NAME1( :I)
K = 1
INDIC = INDIC - 2

*-----------------------------------------------------
* Modify several parameters
*-----------------------------------------------------
if (CHOICE.eq.3) write(*,20)
do while(REP.ne.'q'.and.K.lt.INDIC.and.CHOICE.eq.3)
   ST = STRING(K)
   if (ST(1:1).ne.'$'.and.ST( :6).ne." ")then
      N = 1
      do while ( ST(N:N).ne.'*')
         N = N + 1
      end do
      write(*,33) ST
      if(N.eq.3) write(*,203) ST(1:N-1)
      if(N.eq.4) write(*,204) ST(1:N-1)
      if(N.eq.5) write(*,205) ST(1:N-1)
      if(N.eq.6) write(*,206) ST(1:N-1)
      if(N.eq.7) write(*,207) ST(1:N-1)
      if(N.eq.8) write(*,208) ST(1:N-1)
      if(N.eq.9) write(*,209) ST(1:N-1)
      if(N.eq.10) write(*,210) ST(1:N-1)
      if(N.eq.11) write(*,211) ST(1:N-1)
      if(N.eq.12) write(*,212) ST(1:N-1)
      if(N.eq.13) write(*,213) ST(1:N-1)
      if(N.eq.14) write(*,214) ST(1:N-1)
      if(N.eq.15) write(*,215) ST(1:N-1)
      if(N.eq.16) write(*,216) ST(1:N-1)
      if(N.eq.17) write(*,217) ST(1:N-1)
      if(N.eq.18) write(*,217) ST(1:N)
   end do
read(*,14) REP
if (REP.ne.'q') then
   do while(REP(J:J+2).ne." ".and.J.lt.72)
      J=J+1
   end do
   write(20,34) ST( :N)//REP(1:J)
   K = K+1
end if
else
   if(ST(1:2).ne.'$e') write(*,14) ST
   write(20,14) ST
   K = K+1

end if
end do
do L = K, INDIC
   write(20,14) STRING(L)
end do
else
   write(*,41) NAME1( :I)
end if
end if

--------------------------------------------------
* Create a new variable file
--------------------------------------------------
if (CHOICE.eq.4.and.MARQ.eq.0) then
   write(*,87) FILE
else if (CHOICE.eq.4) then
   rewind(unit=20,iostat=IOS)
   do while (IOS .eq. 0 .and.REP.ne.'q')
      read(15,14,iostat=IOS) TEXT1
      if(TEXT1(:,5).eq.'$list') then
         write(*,10)
         write(*,14) TEXT1
      end if
      if(TEXT1(:,5).eq.'$end'.or.TEXT1(1:4).eq.'$end')then
         write(20,14) TEXT1
      else
         write(*,15) TEXT1(Z: )
         read (*,14) REP
         I=1
         do while(TEXT1(I:1).ne."=".and.I.lt.18)
            I=I+1
         end do
         J=1
         do while(REP(J:J+2).ne." ".and.J.lt.72)
            J=J+1
         end do
         J=J-1
         if(REP.ne."q") write(20,35) TEXT1(1:I)//REP(1:J)
      end if
   end do
   if (IOS.gt.0) write(*,*)' reading error in unit 15'.
   if (REP.eq."q") then
      write(20,14) TEXT1(2: )
      do while(IOS.eq.0)
         read(15,14,iostat=IOS) TEXT1
         write(20,14) TEXT1
      end do
   end if
end if
end do

close(15)
close(20)

5 format(/,10x,'Enter the shell identification code ->',$)
10 format(/,50x,'Type 'q' to quit /)
14 format(A)
format(4x,A16,' ->',$)
format(13x,A16,$)
format(/,10x,'Variables following (type ''q'' to quit) :/',$)
format(/,3x,'File ','A,' not found, file ','A,' created')
format(/,5x,'Error opening file ','A,','//30x,'PROCESS CANCELED')
format(/,10x,'OLD value : ',A16)
format(16x,A16)
format(/,20x,'Parameter ','A,' up to date')
format(20x,'Parameter ','A,' not found')
format(9x,8x,'MENU LOADVAR : ',//,24x,'1 - List of parameters', 1, //,24x,'2 - modify one parameter',//,24x,'3 - modify several ', 2,'parameters',//,24x,'4 - create a new list file',//,24x, 3,'5 - quit',8(//),8x,'Your choice is -->',',$)
format(11)
format(/,10x,'Enter the name of parameter --> ',$,)
format(/,10x,'Enter the value --> ',$)
format(/,10x,'File ','A25,' is not new ,PROCESS forbidden')
format(/,10x,'List of parameters:')
format(/,13x,'Type any key to RETURN to menu',$)
format(22x,A2,' ->',$,)
format(22x,A3,' ->',$,)
format(22x,A4,' ->',$,)
format(22x,A5,' ->',$,)
format(22x,A6,' ->',$,)
format(22x,A7,' ->',$,)
format(22x,A8,' ->',$,)
format(22x,A9,' ->',$,)
format(22x,A10,' ->',$,)
format(22x,A11,' ->',$,)
format(22x,A12,' ->',$,)
format(22x,A13,' ->',$,)
format(22x,A14,' ->',$,)
format(22x,A15,' ->',$,)
format(22x,A16,' ->',$)
format(22x,A17,' ->',$)
end
Mar 27 16:05 1990  file_struct Page 1

***************
* Structure file a12114 *
***************

format(8f9.5)

<table>
<thead>
<tr>
<th>SCANNR</th>
<th>REF(1)</th>
<th>REF(2)</th>
<th>REF(3)</th>
<th>REF(4)</th>
<th>REF(5)</th>
<th>REF(6)</th>
<th>DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA</td>
<td>DATA</td>
<td>DATA</td>
<td>DATA</td>
<td>DATA</td>
<td>DATA</td>
<td>DATA</td>
<td>DATA</td>
</tr>
<tr>
<td>etc.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

201.000  -0.515  -0.885  1.225  0.435  -1.670  0.300  -5.768  
-5.764  -5.740  -5.725  -5.720  -5.710  -5.701  -5.696  -5.691  
-5.696  -5.691  -5.681  -5.681  -5.666  -5.671  -5.676  -5.681  
-5.676  -5.666  -5.656  -5.651  -5.646  -5.641  -5.641  -5.630  
-5.640  -5.645  -5.660  -5.675  -5.669  -5.659  -5.659  -5.634  

***************
* Structure file a12114b *
***************

SCANNR  format(115)
REF(i)  format(8f9.5)
DATA    format(8f9.5)

SCANNR

<table>
<thead>
<tr>
<th>SCANNR</th>
<th>REF(1)</th>
<th>REF(2)</th>
<th>REF(3)</th>
<th>REF(4)</th>
<th>REF(5)</th>
<th>REF(6)</th>
<th>DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA</td>
<td>DATA</td>
<td>DATA</td>
<td>DATA</td>
<td>DATA</td>
<td>DATA</td>
<td>DATA</td>
<td>DATA</td>
</tr>
<tr>
<td>etc.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

201
-0.51500  -0.88500  1.22500  0.43500  -1.67000  0.30000
-1.17057  -1.16557  -1.14557  -1.12657  -1.11157  -1.10257  -1.09757
-1.08257  -1.07357  -1.08257  -1.08257  -1.06757  -1.07257  -1.07757
-1.08257  -1.07757  -1.06757  -1.05757  -1.05257  -1.04757  -1.04257  -1.04257
-1.03157  -1.04157  -1.04557  -1.06157  -1.07657  -1.07057  -1.06057  -1.06057
Mar 27 16:05 1990  file_struct Page 2

************************************************
* Structure file ai2114c, ai2114f and file ai2114k *
************************************************

SCANNER format(1i5)
DATA format(8f9.5)

SCANNER   DATA   DATA   DATA   DATA   DATA   DATA   DATA   DATA   DATA
    etc.

201
.13776  .13824  .15775  .16817  .16853  .17399  .17859  .17908
.17844  .16999  .17035  .17580  .17120  .18174  .17205  .16260
.15303  .15339  .15808  .16454  .16490  .16538  .16563  .16126
.16769  .15315  .14345  .12395  .10444  .10589  .11135  .10674
.12734  .12370  .11922  .12467  .12007  .11159  .10698  .10044

************************************************
* Structure file ai2114hc *
************************************************

IDENTIFICATION format(1A25)
MEASUREDATA format(1A25)
K1 K2 L1 L2 format(1I5)
COEFFICIENTS format(8F9.5)

IDENTIFICATION MEASUREDATA
K1 K2 L1 L2
A(0,0) A(0,1) A(0,2) A(0,3) A(0,4) A(0,5) A(0,6) A(0,7)
   etc. until A(K2,L2)
B(0,1) B(0,2) B(0,3) B(0,4) B(0,5) B(0,6) B(0,7) B(0,8)
   etc. until B(K2,L2)

ai2114 fields
06-01-1983

  0  30  0  40
.00453  -.00011  .02883  -.03040  -.10491  .11710  .12718  .00881
-0.4025  .01608  -.09886  .00059  -.00135  -.03267  .00670  .00773
.00371  -.03036  -.01322  -.00717  .01508  -.00547  .00045  -.00632
.00230  -.00493  .00294  -.00612  -.00290  .00060  .00351  .00467
.00126  .01956  -.00582  -.00266  -.01281  .00168  .00765  .00254
Mar 27 16:05 1990  file_struct Page 3

* Structure file al2114hs *

IDENTIFICATION format(1A25)
MEASUREDATA  format(1A25)
K1 K2 L1 L2  format(4I5)
COEFFICIENTS  format(8F9.5)

IDENTIFICATION
MEASUREDATE
K1 K2 L1 L2
C(1,0) C(1,1) C(1,2) C(1,3) C(1,4) C(1,5) C(1,6) C(1,7)
etc. until C(K2,L2)
D(1,1) D(1,2) D(1,3) D(1,4) D(1,5) D(1,6) D(1,7) D(1,8)
etc. until D(K2,L2)

al2114 fields
06-01-1983
0 30 0 40
-.00592  .00129  .04647  -.04989  -.15067  .15325  .17885  .02466
-.05044  .01834  -.15978  .00557  .00031  -.06307  .01148  .01928
 .00259  -.04797  -.02933  -.00305  .01810  -.00922  -.00293  -.01090
 .00655  -.00300  .00516  -.01115  .00027  -.00088  .00850  .00517
 .00042  .03383  -.00948  -.00821  -.01456  .00397  .00952  .00108