NUREG/CR-6357
ORNL/TM-13213

Evaluation and Field Validation of Eddy-Current Array Probes for Steam Generator Tube Inspection

Prepared by
C. V. Dodd, J. R. Pate

Oak Ridge National Laboratory

Prepared for
U.S. Nuclear Regulatory Commission

AVAILABILITY NOTICE

Availability of Reference Materials Cited in NRC Publications

Most documents cited in NRC publications will be available from one of the following sources:

1. The NRC Public Document Room, 2120 L Street, NW., Lower Level, Washington, DC 20555-0001

2. The Superintendent of Documents, U.S. Government Printing Office, P. O. Box 37082, Washington, DC 20402-9328

3. The National Technical Information Service, Springfield, VA 22161-0002

Although the listing that follows represents the majority of documents cited in NRC publications, it is not intended to be exhaustive.

Referenced documents available for inspection and copying for a fee from the NRC Public Document Room include NRC correspondence and internal NRC memoranda; NRC bulletins, circulars, information notices, inspection and investigation notices; licensee event reports; vendor reports and correspondence; Commission papers; and applicant and licensee documents and correspondence.

The following documents in the NUREG series are available for purchase from the Government Printing Office: formal NRC staff and contractor reports, NRC-sponsored conference proceedings, international agreement reports, grantee reports, and NRC booklets and brochures. Also available are regulatory guides, NRC regulations in the *Code of Federal Regulations,* and *Nuclear Regulatory Commission Issuances.*

Documents available from the National Technical Information Service include NUREG-series reports and technical reports prepared by other Federal agencies and reports prepared by the Atomic Energy Commission, forerunner agency to the Nuclear Regulatory Commission.

Documents available from public and special technical libraries include all open literature items, such as books, journal articles, and transactions. *Federal Register* notices, Federal and State legislation, and congressional reports can usually be obtained from these libraries.

Documents such as theses, dissertations, foreign reports and translations, and non-NRC conference proceedings are available for purchase from the organization sponsoring the publication cited.

Single copies of NRC draft reports are available free, to the extent of supply, upon written request to the Office of Administration, Distribution and Mail Services Section, U.S. Nuclear Regulatory Commission, Washington, DC 20555-0001.

Copies of industry codes and standards used in a substantive manner in the NRC regulatory process are maintained at the NRC Library, Two White Flint North, 11545 Rockville Pike, Rockville, MD 20852-2738, for use by the public. Codes and standards are usually copyrighted and may be purchased from the originating organization or, if they are American National Standards, from the American National Standards Institute, 1430 Broadway, New York, NY 10018-3308.

DISCLAIMER NOTICE

NUREG/CR–6357
ORNL/TM–13213

Evaluation and Field Validation of Eddy-Current Array Probes for Steam Generator Tube Inspection

Manuscript Completed: October 1994
Date Published: July 1996

Prepared by
C. V. Dodd, J. R. Pate

Oak Ridge National Laboratory
Managed by Lockheed Martin Energy Research Corporation

Oak Ridge National Laboratory
Oak Ridge, TN 37831

J. Muscara, NRC Project Manager

Prepared for
Division of Engineering Technology
Office of Nuclear Regulatory Research
U.S. Nuclear Regulatory Commission
Washington, DC 20555–0001
NRC Job Code B0417

Abstract

The objective of the Improved Eddy-Current ISI for Steam Generator Tubing program is to upgrade and validate eddy-current inspections, including probes, instrumentation, and data processing techniques for inservice inspection of new, used, and repaired steam generator tubes; to improve defect detection, classification, and characterization as affected by diameter and thickness variations, denting, probe wobble, tube sheet, tube supports, copper and sludge deposits, even when defect types and other variables occur in combination; to transfer this advanced technology to NRC's mobile NDE laboratory and staff. This report describes the design of specialized high-speed 16-coil eddy-current array probes. Both pancake and reflection coils are considered. Test results from inspections using the probes in working steam generators are given. Computer programs developed for probe calculations are also supplied.

Contents

Figures

Evaluation and Field Validation of Eddy-Current Array Probes for Steam Generator Tube Inspection

C. V. Dodd and J. R. Pate

Introduction

As the nation's steam generators have aged, new mechanisms for the degradation of steam generator tubes have appeared, and the frequency of forced outages due to major tube leak events has increased. To ensure that tube degradation is detected before it leads to a forced outage, it will be necessary to inspect during each outage a large number of the tubes in each steam generator using a technique which is sensitive to all forms of degradation. The primary method for the inspection of steam generator tubing is eddy-current testing, and while eddy-current testing can be used to perform a fast and reliable inspection, the eddy-current techniques currently being used for steam generator inspection are no longer adequate for the needs of the nuclear power industry.

The bobbin probe has traditionally been the primary method for the inspection of steam generator tubing. The probe consists of a pair of circumferentially-wound coils which are pulled through the tube at speeds of up to 40 inches per second, acquiring data as they travel. The probe's high speed and complete coverage of the tube circumference have made it a favorite of utilities with a tight outage schedule. However, since the probe induces eddy-currents in the tube wall with components only in the circumferential direction, the bobbin probe is not sensitive to circumferential cracks because these will not interrupt the flow of the induced currents. Furthermore, since the probe looks at the entire tube circumference at one time, the ability to use bobbin probe data to distinguish between cracks and volumetric flaws is limited.

To overcome the sensitivity limitations of the bobbin probe, the rotating pancake probe (RPC) was introduced. The RPC consists of one or more small coils pressed against the inner surface of the tube wall. In order to inspect the entire circumference of the tube, it is necessary to rotate the coil as it is pulled through the tube. Unfortunately, this means that the RPC is quite slow, with an inspection speed of only 0.2 in per second. This is clearly too slow to inspect the large numbers of tubes that should be inspected, and utilities avoid this type of inspection when at all possible. The result has been forced outages for tube leaks. Plants such as Palo Verde 2, have had outages lasting several months while long sections of tubes were inspected using the RPC.

Clearly a new probe design which combines the speed of the bobbin probe with the sensitivity of the RPC is needed to ensure the detection of all types of defects and to improve the ability to characterize defects. Toward this end, ORNL has developed a 16-coil eddy-current array probe. This probe consists of 16 small coils pressed against the inner surface of the tube and distributed about its circumference. The data obtained with this probe are similar in nature to those from the RPC, allowing detection of cracks of any orientation, but since it is not necessary to rotate the probe to achieve full coverage of the tube circumference, inspection speeds approaching those of the bobbin coil are possible.

Array Probe Design

The eddy-current array probes developed by ORNL consist of 16 small coils pressed against the inner surface of the tube as shown in Figure 1. The coils are divided into two

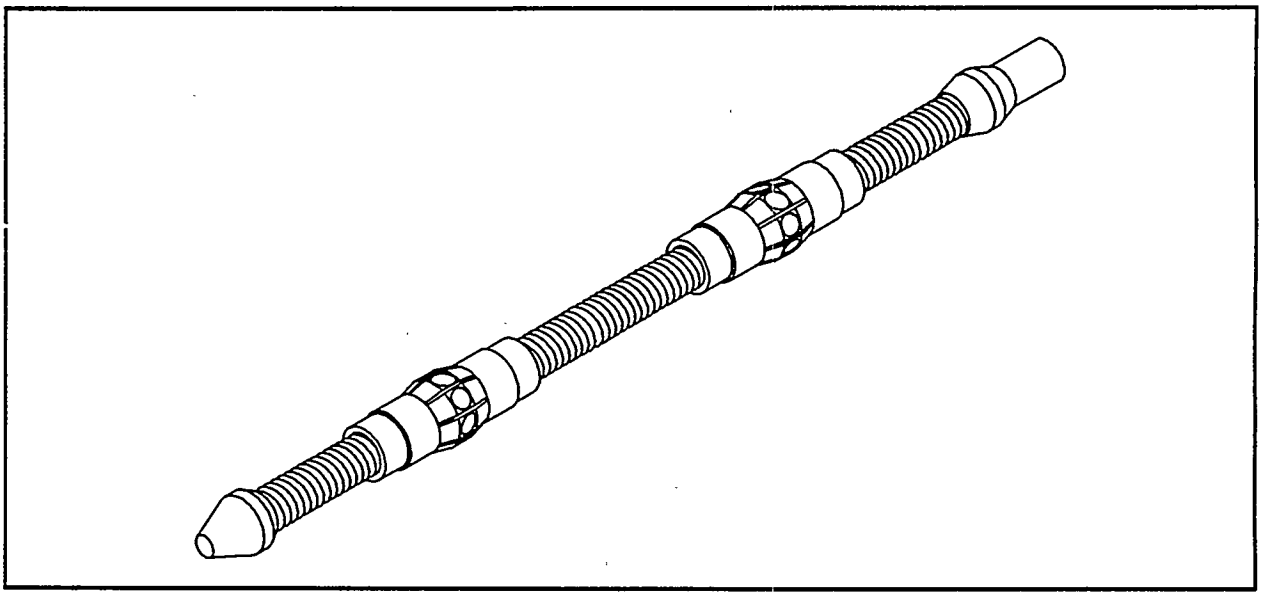


Figure 1 The 16-coil array probe

groups of eight coils each with the two groups being separated by a distance of approximately 4.5 inches in the axial direction. Each coil is mounted in an individual spring-loaded shoe to keep the coil pressed against the inner surface of the tube, which has a diameter of 0.775 inches, but if necessary the shoe will retract, allowing the probe to pass through a tube of diameter 0.720 inches. This makes it possible to inspect tubes with a significant amount of denting.

Figure 2 shows a cross section of one of the groups of eight coils. Neighboring coils in the same group are separated by 45 degrees in the circumferential direction. The coils in one group are centered in the regions between the coils in the other, resulting in there being 22.5 degrees between adjacent coils.

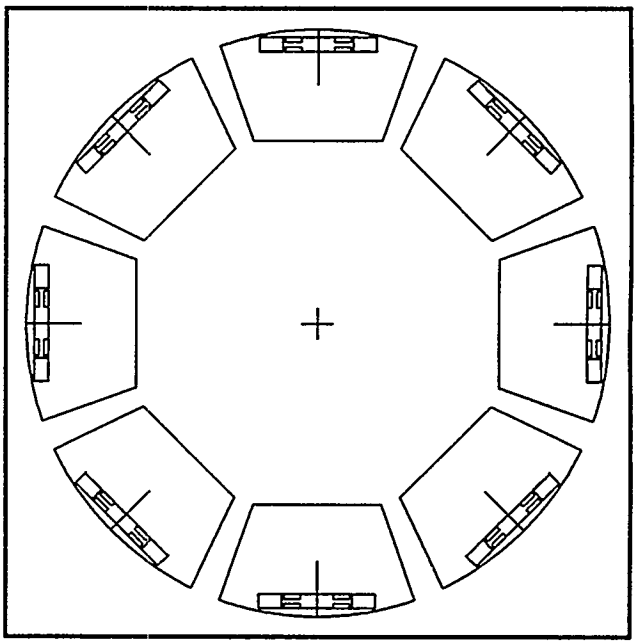


Figure 2 A cross section of the array probe

While the array probe has been designed for and tested in only straight sections of tubes with an outer diameter of 7/8 inches, Zetec, the company which manufactured the probe, feels that the probe can be constructed to inspect the U-bends of the tubes and that it can be scaled to inspect tubes with an outer diameter of 3/4 inches.

Individual Coil Design

The individual coils which make up the 16-coil array probe were designed to maximize sensitivity to small defects on the outer diameter of the tube. Two array probes with different types of coils were designed and constructed; one probe contained the commonly used pancake coil, and the other contained reflection coils. Figure 3 shows a pancake coil and its electrical circuit. The pancake coil is used as the test coil in a typical bridge circuit, and a second identical coil placed is in a reference tube. The voltage measured is the difference between the test and reference signals, and since the probes will be nearly balanced, the signal can be amplified, increasing the sensitivity to small changes in the part being tested.

In Figure 4 we show a reflection coil and its electrical connections. The reflection coil consists of a driver coil which is used to induce eddy currents in the part being tested, and two smaller pickup coils in which a voltage is induced by the eddy currents in the part. The reflection coil can be designed in such a way to offer better discrimination to lift-off, the distance between the coil and the test part, than can be obtained with the pancake coil, and it therefore has a significant advantage over the pancake coil for steam generator inspection. The reflection coil circuit is similar to an induction bridge which is widely used for many types of highly accurate measurements.

Design studies were done for both of the types of coils in order to maximize their

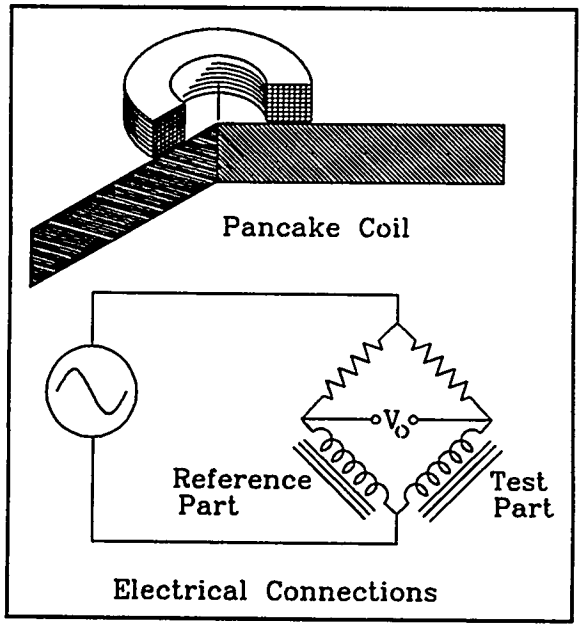


Figure 3 Pancake coil and electrical connections

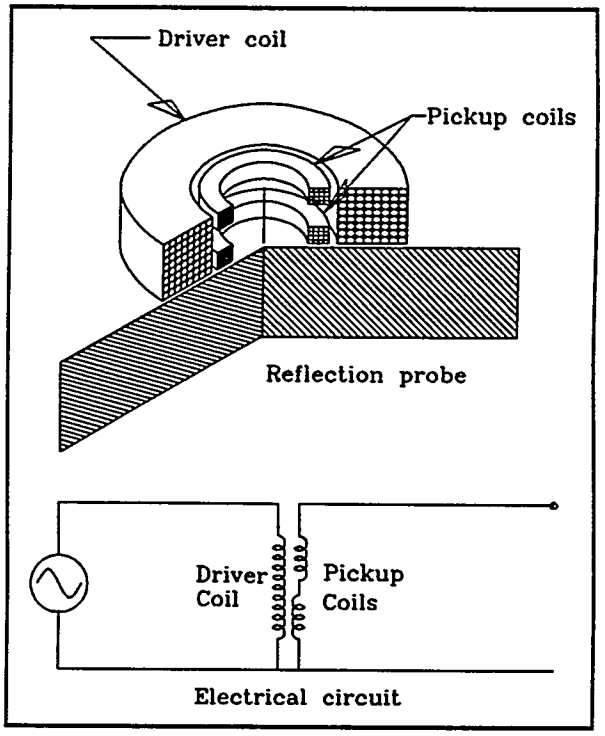


Figure 4 Reflection coil above a flat plate

sensitivity to small defects on the far side of the tube wall and to minimize the effects of other variables. For the purpose of the design study, we considered the tube wall to be a flat plate. The first consideration was to make the coils as flat as was practical. Figure 5 illustrates how the normalized coil impedance depends on the shape of the coil cross section. From the figure it is evident that for a given value of the quantity $\omega\mu\sigma r^2$, the product of the inspection frequency, tube conductivity, tube permeability, and coil mean radius squared, increasing coil flatness increases the coil impedance change from its air value. Therefore, the coil is more sensitive to the properties of the part being tested when its turns are as close to the part as possible. There is a limit on the coil flatness, however, since if the coil is made too thin, its dc resistance becomes too high. In addition, the small defects that we wish to measure are usually on the far side of the conductor, so that the absolute flatness is not critical. It is recognized that as coils become flatter, the sensitivity to lift-off, conductivity variations, and other conductor property variations will also increase, but other methods will be used to reduce these effects.

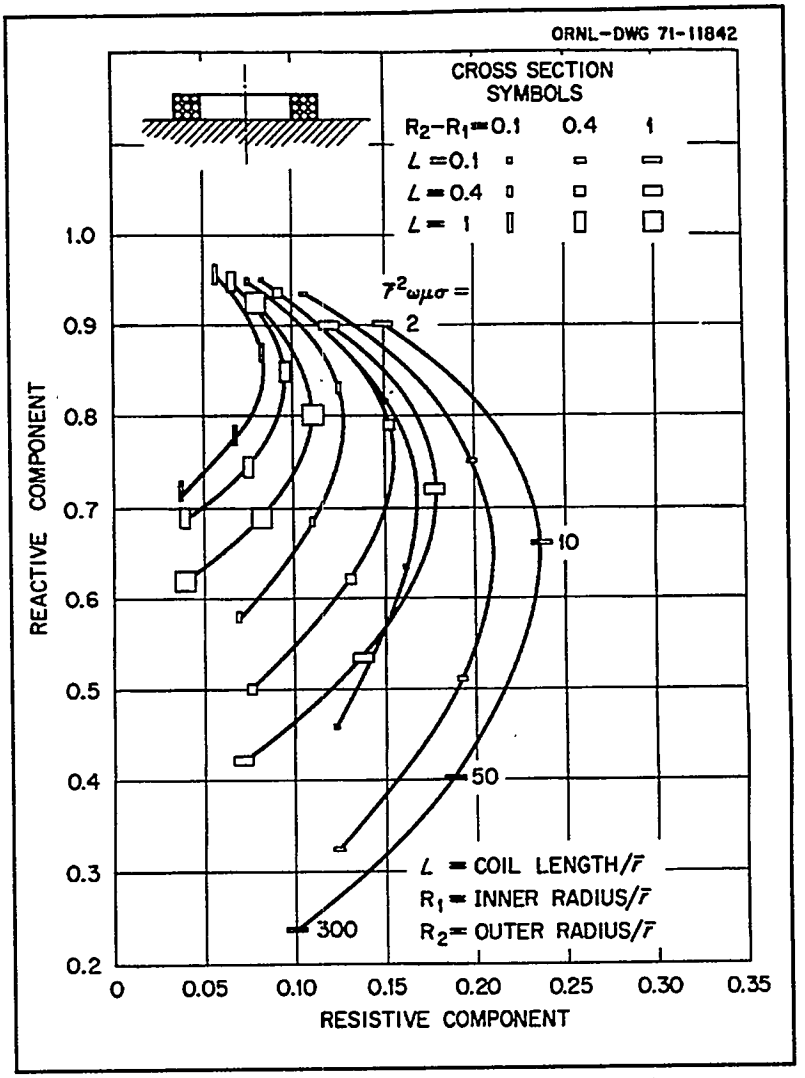


Figure 5 Variation in normalized coil impedance for different values of coil cross-section

A number of different pancake and reflection coil designs were tested using both analytical calculations and experimental measurements. The model used for most of the calculations was a point defect or a very small spherical defect. The reasoning was that if we could optimize for the smallest defects, detection of the larger ones would also be accomplished. Also, the mathematical model used for most of the calculations is only valid for small defects.

In Figure 6 we show the calculations of the impedance change for the P90 pancake coil due to a point defect on the far side of a conducting plate as a function of the defect radial location for several different test frequencies. Notice that there is an optimum frequency for maximum sensitivity and an optimum radius. This gives an indication of the field spread of the coil.

We show in Figure 7 a plot of the defect signal for different size coils as a function of the distance from the center of the coil. The outer diameter of the coils was fixed at 0.240 inches by the size of the shoe in which the coil was mounted, and the inner radius was varied from 0.0 to 0.16 inches, causing the mean radius of the coil to vary from 0.06 to 0.1 inches. Both the maximum sensitivity and the radial distance at which the maximum sensitivity occurs vary with coil size. The maximum sensitivity occurs for the P75 at a radius of 0.080 inches, and for the P90 it occurs at 0.0875 inches. While the maximum sensitivity of the P90 is slightly less than that of the P75, the greater field spread of the P90 means that it is to be preferred for use in the array coil. Both the P75 and P90 coils were built and tested, and the results were similar to those expected.

A comparison study was performed between the reflection and pancake coils to measure the defect depth in the presence of other property variations. The signals from the pancake and reflection coils were computed using the programs IMPTST and MULRFD, respectively. Property variations (in the file NRCPRO.DAT) include a lift-off variation of 0.010 inches, tube wall thickness values of 0.050 inches, 0.040 inches, and 0.030 inches, and defect depth values of 0.000 inches, 0.005 inches, 0.010 inches, and 0.015 inches, with copper deposits, tube supports, and magnetite deposits on the outside of the tube at various locations. The copper was placed at 15 locations, the tube support at 12 locations, and the magnetite at 12 locations, for a total of 39 OD artifact property values. For each of these artifact property values there are four defect depth

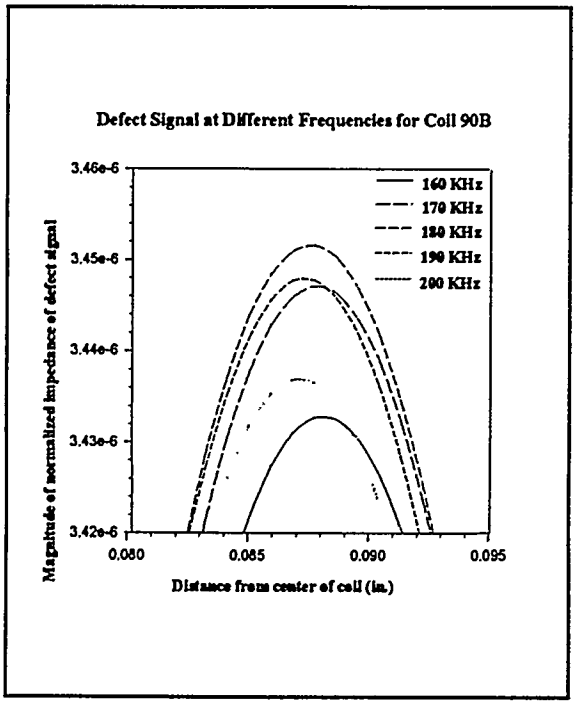


Figure 6 Defect signal for a far-side defect

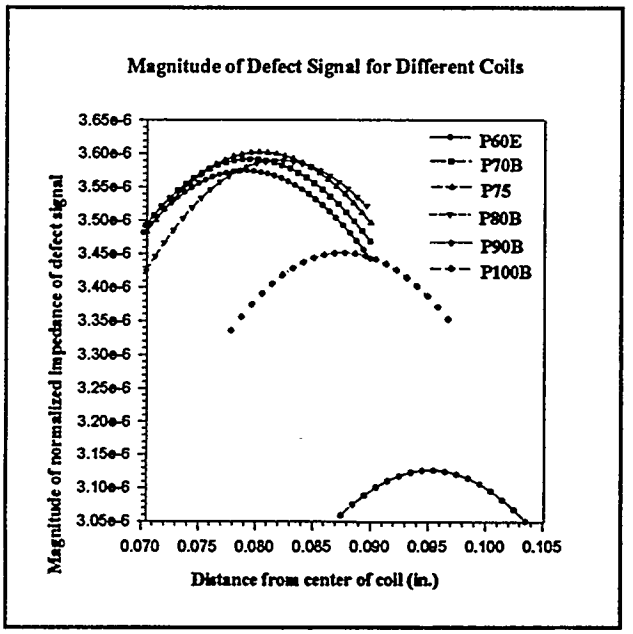


Figure 7 Magnitude of defect signals for coils of different sizes

values and five lift-off values, for a total of 780 property combinations. The frequencies were varied from 10 kHz to 1 MHz in a 1,2,5 sequence. The property variations for each of the three types of OD artifacts are summarized below.

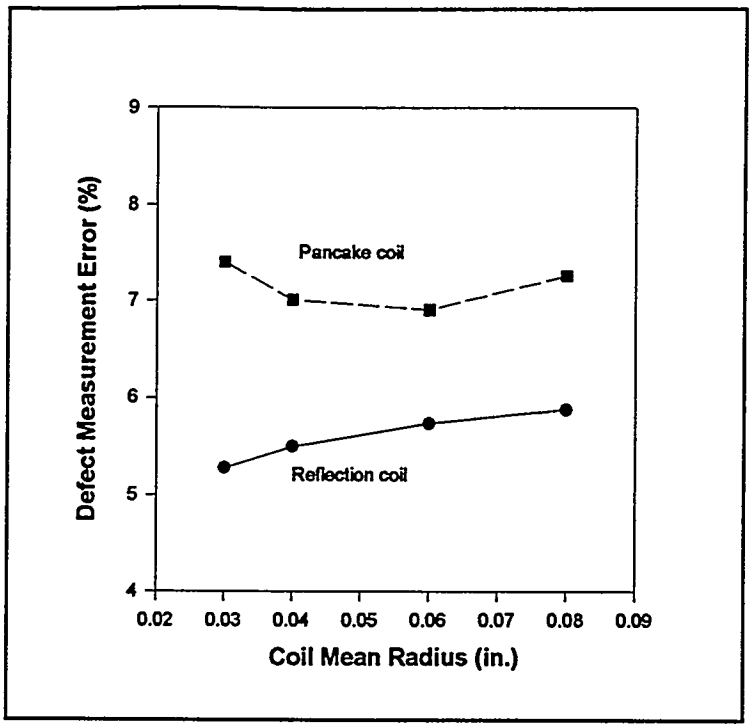
Region	Material	Thickness
1	Copper	Infinite
2	Air	0.5, 0.005, 0.003, 0.001
3	Inconel	0.050, 0.040, 0.030
4	Coil	

Region	Material	Thickness
1	Tube Support	Infinite
2	Air	0.005, 0.003, 0.0001
3	Inconel	0.050, 0.040, 0.030
4	Coil	

Region	Material	Thickness
1	Magnetite	Infinite
2	Air	0.005, 0.003, 0.0001
3	Inconel	0.050, 0.040, 0.030
4	Coil	

The best combination of three frequencies was chosen from the six frequencies used by running the program FINDFIT on the data for both coil types. Although this is only an approximation of the complexity of an actual problem, it does give an indication of how the multiple property fits can work without requiring an excessive amount of computer time. FINDFIT computes the best least squares fit of the property variations to the computed readings.

In Figure 8 we show the results of the error in the measurement of defect depth for both the pancake and reflection coils. The defect depth measurement error is the RMS of the fit error and the drift error, and it is measured in percent of wall thickness. For both types of coils, the fit error dominates. The neural net analysis techniques would probably give a better fit with no significant increase in the drift. From this comparison, the reflection coil seems to offer a significant advantage over the pancake coil.



Another important factor is the ability of the coils to cover the entire circumference of the tube. To measure this ability the probes were used to scan an axial EDM notch of depth 40% of wall thickness and length 0.25 inches. The scan was conducted once with the center of the coil passing directly over the notch, and then it was repeated with an angle of 12 degrees between the center of the coil and the notch. The ratio of the response to the defect when the probe is at 12 degrees to the response when the probe is at 0 degrees is plotted in Figure 9. The figure shows that the sensitivity of the P90 is much more uniform than that of the P75 or the P60, and that the pancake coils have greater field uniformity than the reflection coils.

The zero lift-off, or the distance between the coil and the inner surface of tube, will also have an effect on the sensitivity to small defects on the far side of the tube wall. Both the sensitivity to and the resolution for small defects increases as the zero lift-off decreases. It is therefore best

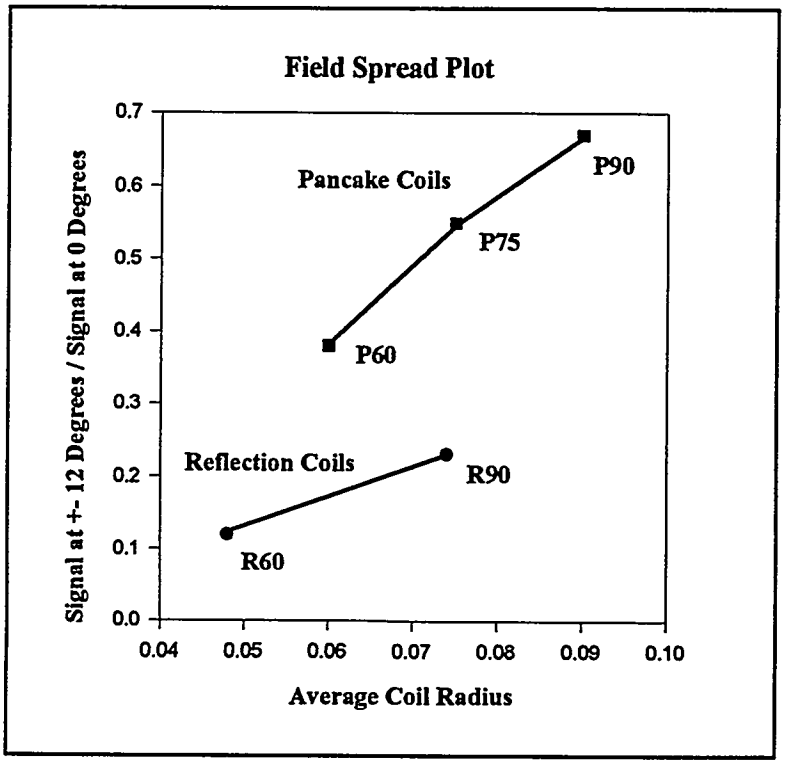


Figure 9 Field uniformity comparison

to get the coil as close to the tube wall as is practical. If the coil is flat, the curvature of the tube increases the distance between the coil and the tube wall. One method of reducing this distance is to contour the coil to conform to the curvature of the tube, as shown in Figure 10. Coils of two different shapes were used, one with the coil cylindrically shaped to fit the tube wall, and another with the coil spherically shaped. The cylindrically shaped coil has smaller zero lift-off, but it appears to be very slightly more susceptible to lift-off effects than the spherically shaped coil. The spherically shaped coil was also easier to fabricate. An attempt was made to build a cylindrically shaped reflection probe, but it was not successful.

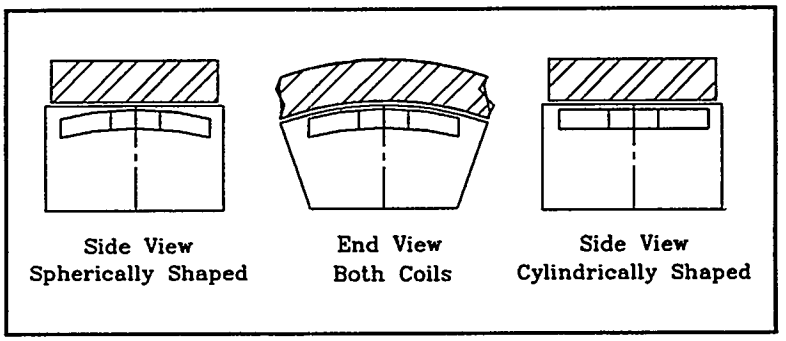


Figure 10 Shaped coils

As a final consideration the number of turns in the coil was determined. The number of turns is a compromise between the ratio of the dc coil impedance changes to the ac impedance changes and the effects of probe wire capacitance. It is desirable (but not always possible) to have a low cable capacitance between the instrument and the probe so that the probe operates well below its resonant frequency because near the coil resonance a 5 pF change in capacitance will lead to a signal change as large as that produced by a through-wall hole. For a 16-coil pancake array probe, 16 coaxial connections are needed, and for a 16-coil reflection array probe, 32 coaxial connections are needed. The initial reflection array probes were built using cable of capacitance 26 pF per foot. Since that time, cable with a capacitance of 20 pF per foot has been found for the reflection array, and cable with a capacitance of 17 pF per foot has been found for the pancake array probe. The inductance of the coils is about 50 to 90 microhenries so that with 110 feet of low-loss cable, the resonant frequency can be as high as 650 kHz, which is slightly above the highest test frequency we have used in the field validation of the probes. The number of layers in the coil, the number of turns per layer, and the wire gage are computed by the program AIRCO.

Instrumentation

The MIZ-30 eddy-current instrument was designed and constructed by Zetec specifically for array probe inspections. It is a high-speed multiple-frequency digital instrument that can drive 16 individual coils in either a bridge arrangement with a reference probe or in a transmit-receive mode for the reflection coils. It is capable of taking 500 readings per second for all 16 coils at four frequencies. The gain and hardware null are also software adjustable. A Hewlett Packard workstation is used to control the instrument over a local area network. The configuration of the instrument is set by the acquisition operator. The configuration can be entered manually or a data file containing the setup information can be transmitted over the LAN from any other workstation on the network and stored in the appropriate file. The configuration file can contain the setup

information for up to 16 different types of inspections. The probe is positioned on the face of the tubesheet (a Zetec SM-22 was used for the Prairie Island tests), and the probe pushed into the tube and then the probe pulled as the data are acquired and recorded. Up to 16 channels can be displayed so that the acquisition operator can verify that all the channels of the probe are working and can tell where the probe is in the tube. However, with the amount of data to be processed when all 16 channels were displayed, the computer lagged too far behind the actual probe motion so that it was much better to display only one channel most of the time. The acquisition computer also controlled the probe positioning and insertion, which lagged too far behind when the 16 channels were displayed. This problem is considerable when using a 700 series workstation.

Prairie Island test results for array probes - first test

The array probes were first tested at the Prairie Island Unit 2 steam generators on November 14, 1993. The system was assembled and tested on the Prairie Island mock-up the week before it was tested in the generators. The acquisition operators were not familiar with the MIZ-30, and some instructions were necessary before the acquisition programs could be successfully run. The probe pusher-pullers were designed to drive 3/8 inch diameter polyurethane cable into the tubing, and since the array probes use 1/2 inch diameter cable, a modification had to be made to the pusher-puller before it would drive the array probe cable. A special part had to be machined and another part had to be removed. One of the twister cables was discovered to be bad, and personnel from Conam, the company which performed the acquisition and primary analysis of the eddy-current data at Prairie Island, repaired it. Conam also suggested that we add enough extension cable (10 feet) to allow the MIZ-30 to be moved to an area with lower contamination, which we did.

The MIZ-30 and the cables were installed in the generator on Sunday morning, November 14. The pusher-puller in the steam generator had to be modified in the same manner as the one in the mock-up. Unfortunately, due to the contamination and other problems, it was much more difficult to modify, and the probe speed was limited to 15 inches per second by mechanical considerations.

A set of 48 tubes was selected for inspection based on the results of the regular eddy-current inspection which had been completed. The reflection array probe was tested first. The gain in the original design of the MIZ-30 was not as high as was needed to obtain the best results with this type of coil, and the signal-to-noise ratio for the reflection was thus about the same as that for the Zetec rotating pancake coil (RPC). The pancake array with spherically shaped P60 coils was then tested. The signal level for this probe was greater than that for the Zetec RPC by a factor of 5 to 10 for the same gain setting. The data from the various coils were reviewed by Gary Henry of EPRI, who determined that the signal-to-noise ratio of the pancake array probe was higher than that of the Zetec RPC by about a factor of 5.

In order to do improved signal analysis a circumferential groove standard had to be used to calibrate each coil in the probe. Figure 11 shows a drawing of the standard used. The grooves

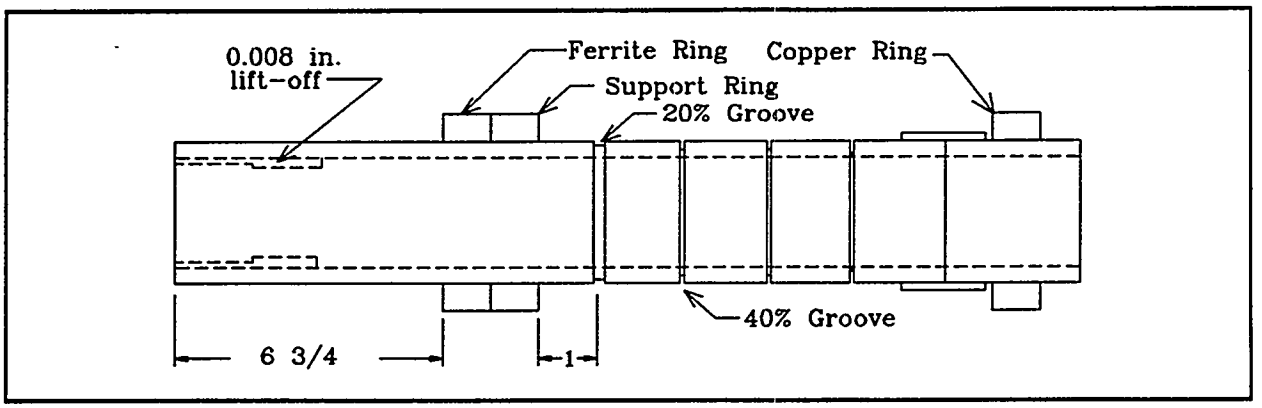


Figure 11 Circumferential groove standard

extend all the way around the circumference of the tube and have depths of 20%, 40%, 60%, 80% and 100% of wall thickness. In addition to the grooves the standard contained a ferrite ring (to simulate a magnetite deposit), a tube support ring, and a copper ring. Scanning the standard allowed us to perform mixes for the suppression of tube supports, magnetite, and copper. The data were analyzed at the site using the Zetec analysis software by analysts from Conam and Zetec. The data were also analyzed using the ORNL analysis software. Since the ORNL software was developed especially for array probe data, it is less cumbersome to use to perform mixes and other tasks. In general the array probes detected all indications that the bobbin coil detected. There were some calls made with the ORNL array probes that were not made with the bobbin coil, but these were all quite small and harmless. There were some indications at the tube supports, but none were judged to be serious. Figure 12 shows the bobbin coil scan of a section of tube R35C30 and Figure 13 shows a plot of the pancake array probe data from the same section of the tube. There appear to be three small defects just above the tubesheet.

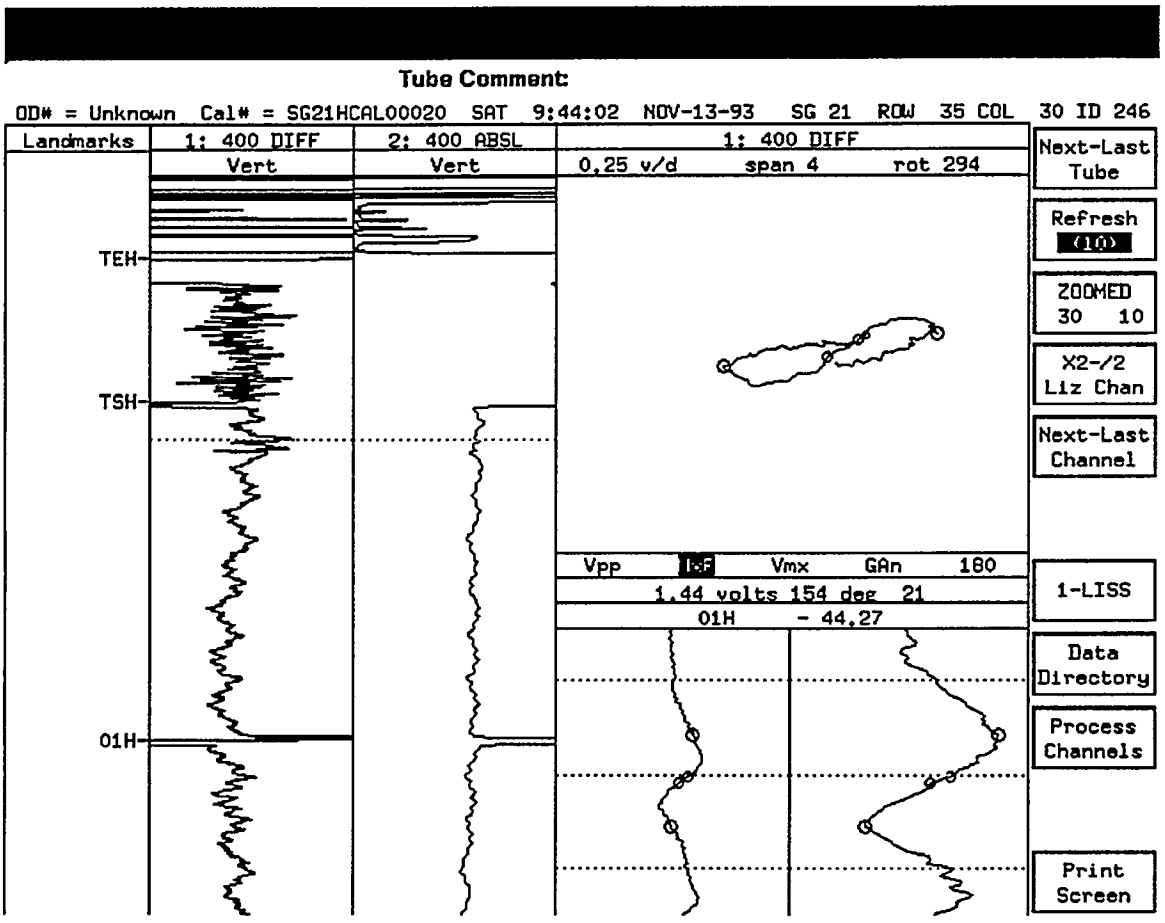


Figure 12 Bobbin probe scan of tube row 35 column 30 using Zetec Eddynet software

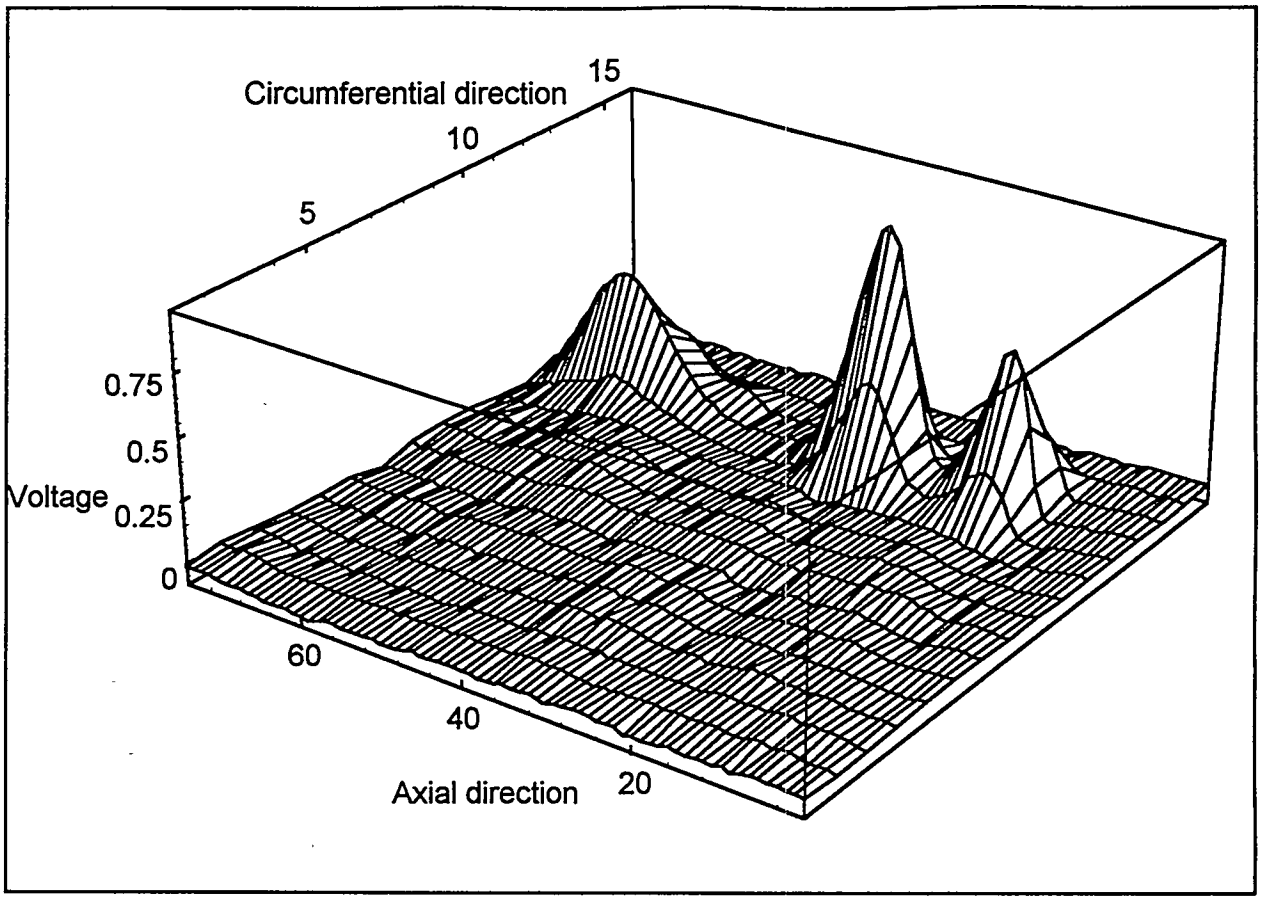


Figure 13 Pancake array scan of tube Row 35 Column 30 showing three indications

Speed trial studies

Near the end of the test, a special production test was run. In order to simulate the production inspection conditions, two adjacent tubes from the test group were selected. The probe was inserted in one tube after the other, alternating between the tubes. This simulated the conditions that would obtain if the tubes were being tested by going in a straight line down a row of tubes. A total of 17 inspections were performed at a rate of 68 seconds per tube or 53 tubes per hour for inspection of 30 feet of tubing. In these tests the insertion speed was limited to 15 inches per second due to mechanical problems. An insertion speed of 40 inches per second would improve the inspection rate by 20 percent to 68 tubes per hour. The array probes were tested at a pull speed of 15 inches per second, which is about 75 times faster then the pull speed (0.2 inches per second) of the RPC. At this pull speed there was more than adequate resolution.

Prairie Island test results for array probes - second test

The second test at Prairie Island was conducted on the steam generators of Unit 1 on Saturday, May 29, 1994. The plan was to perform an eddy-current test on selected tubes using the contaminated pancake and reflection array probes that had been stored at the plant since they were used in the inspection of Unit 2 in November 1993. Uncontaminated probes and cables were sent directly to Prairie Island from Zetec (where our MIZ-30 had been upgraded). Upon its arrival, the instrument was checked out with both types of array probes. The reflection probe exhibited a "touch effect", in which the signal changed if the connectors were touched. This had been noticed during development and checkout at Zetec, and it was found that if the connectors were wrapped in bubble padding, the effect disappeared.

One of the channels of the pancake array probe was not working. This probe had been tested at Zetec and all of the channels had been working properly then. The problem was traced to one of the extension cables (two are required for the pancake array probes) and this cable was repaired. A Conam technician examined the connectors and said that one of them was not wired well. He repaired the lead and broke two more in the process and then fixed those. The probes were then tested, and it was found that all channels were working. The data obtained on standard runs looked very good for both probes, and the signal-to-noise ratio appeared to be about the same for the reflection and pancake array probes.

There was a delay of several days due to problems in getting access to the generators, problems in setting up for the production eddy-current inspections, and problems in data acquisition over the network used for these tests. The problems were finally resolved, but only after a delay of several days. The ORNL inspection, which came at the end of the other eddy-current tests, was delayed from Monday until 3:00 a.m. Saturday.

During this time extensive testing and mixing with both types of array probes were performed. One important discovery was that a better mix was obtained if the ferrite ring was placed directly against the tube support ring, to simulate magnetite adjacent to the support. The mixes obtained using this showed very little residual and very little distortion of the 20% defect. Preliminary results using the Zetec mix showed that scans of the actual tube had very little

residual, and that small defects at the tube supports could be easily detected. The ORNL analysis program would not run on the data due to an upgrade that Zetec had made in their software.

Fifty tubes were selected to be tested with both the reflection and pancake array probes. Our acquisition window was only 12 hours, which included the time necessary to make modifications to the pusher-puller and the time required to run the cable. We used a ten foot twister cable and a 50 foot extension cable for both probes. The data were evaluated at Prairie Island by the Conam analysts. The data from Unit 1, which is older than Unit 2, showed many more magnetite deposits. The pancake array probe had one channel missing. It was not possible to tell if the missing channel was due to another broken connection in the extension cable or if the contaminated array probe used in the previous inspection had a broken coil. Due to the fact that we already had problems with the extension cable we decided not to contaminate our spare probe. Therefore, the test of the pancake array probe was run with only 15 of the 16 coils working. In spite of the high cost of extension cables, future inspections should be performed with at least one spare set of all cables and probes.

We did not experience as much difficulty in getting the pusher-puller to drive the cable as we had at the first inspection. In addition, the computer (a Hewlett Packard 700 series workstation) used was fast enough to take the data and send it to the analysis building in very near real time. While the program did abort twice at the start of the test, for the remainder of the test we had no problems. However, we experienced problems with the SM-22 positioner. It seemed to lose its location on the generator after about the first hour of testing. This required extensive time to locate plugs and other artifacts to verify the position of the tubes. It is possible that some of the tubes scanned were not the ones indicated, although the operators used every reasonable precaution.

Several valuable lessons were obtained from the tests at a small cost to our programs. The new MIZ-30 appears to give good, low-noise data. We were able to copy the setup file for the MIZ-30 acquisition which had been prepared earlier from an optical disk into the proper directory (named /miz30d) on the acquisition computer and rename it so that it was used for the acquisition. It would have been faster to do this over the LAN, but the operator who knew how to do it was not on duty at the time. Copying the setup file saved a great deal of time that would have been required to type in the proper settings and also reduced the chances of making an error. However, the setup for the individual coils in the array probe did not match since we used the old, contaminated probes that were used in the last outage rather than the similar but not identical ones that were used for testing before the inspection. The setup for the individual channels is obtained from the acquisition header, and if the correct individual values are used, that data analyst does not have to reset these. With 16 coils for each of the four frequencies, the setup time can be considerable using the Zetec software. Any mixes that are used require considerable time with the Zetec software. The scan of the tubing in the generator had a larger lift-off variation than we encountered in the scans of the standard, which made the data appear noisy at first. When the proper adjustments were made, the problem was corrected.

The ORNL analysis program was modified to match the new Zetec acquisition format and the data were analyzed at ORNL. In general, the array probes were able to detect all of the defects that the bobbin and RPC inspections showed. Figure 14 shows the vertical component of the reflection array probe data taken at 260 kHz at tube support 01H of tube R43C59 in steam

generator 12. The dark horizontal band across the plot is the tube support, and the bright spot near the center of the graph appears to be a defect. Figure 15 shows the same data with the tube support suppression mix applied. Here we see that the dark band representing the tube support has been eliminated while the bright spot has been amplified, indicating that it is indeed a defect.

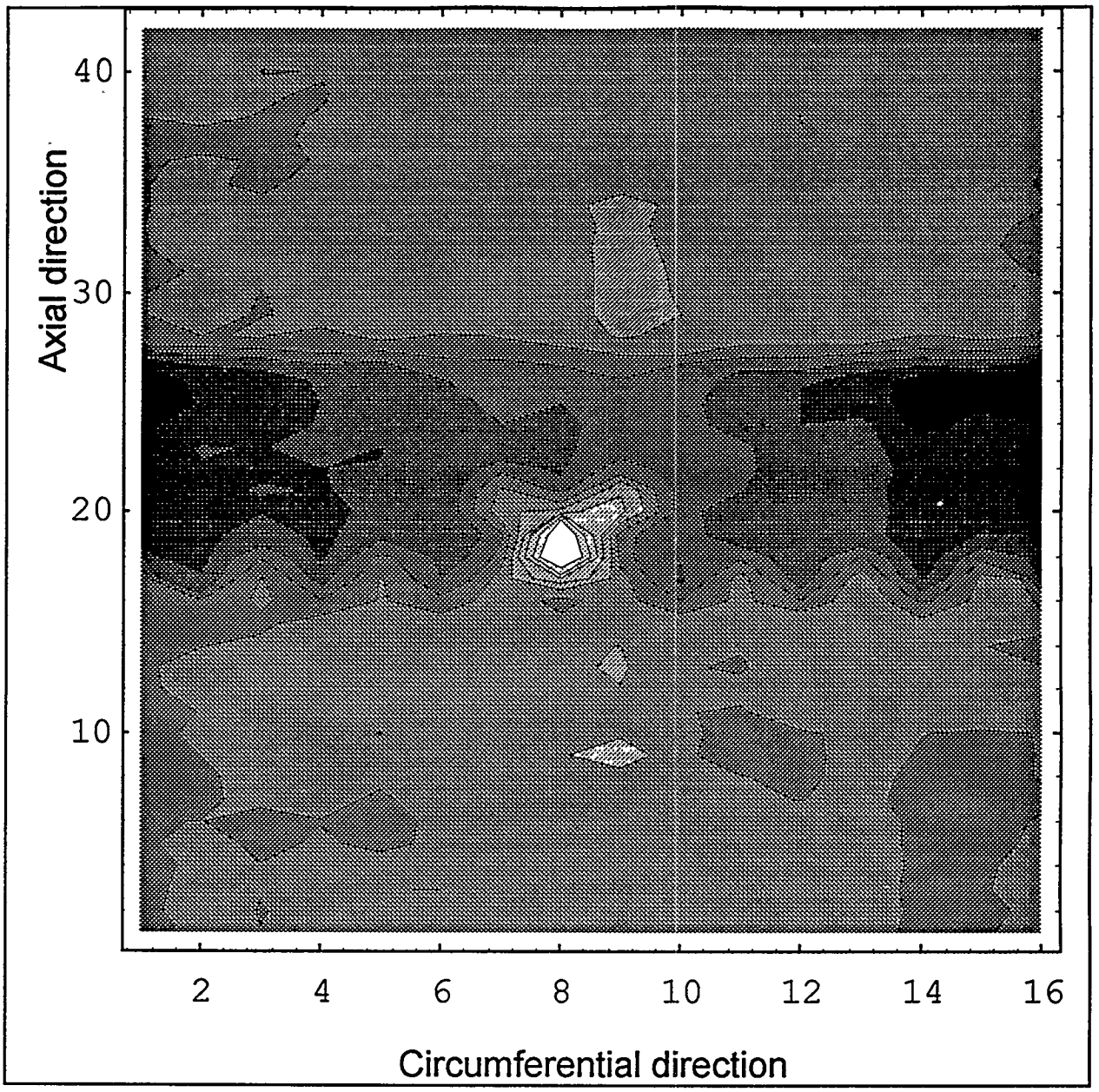


Figure 14 The vertical component of the 260 kHz data from the reflection array probe at tube support 01H in tube R43C59 in steam generator 12.

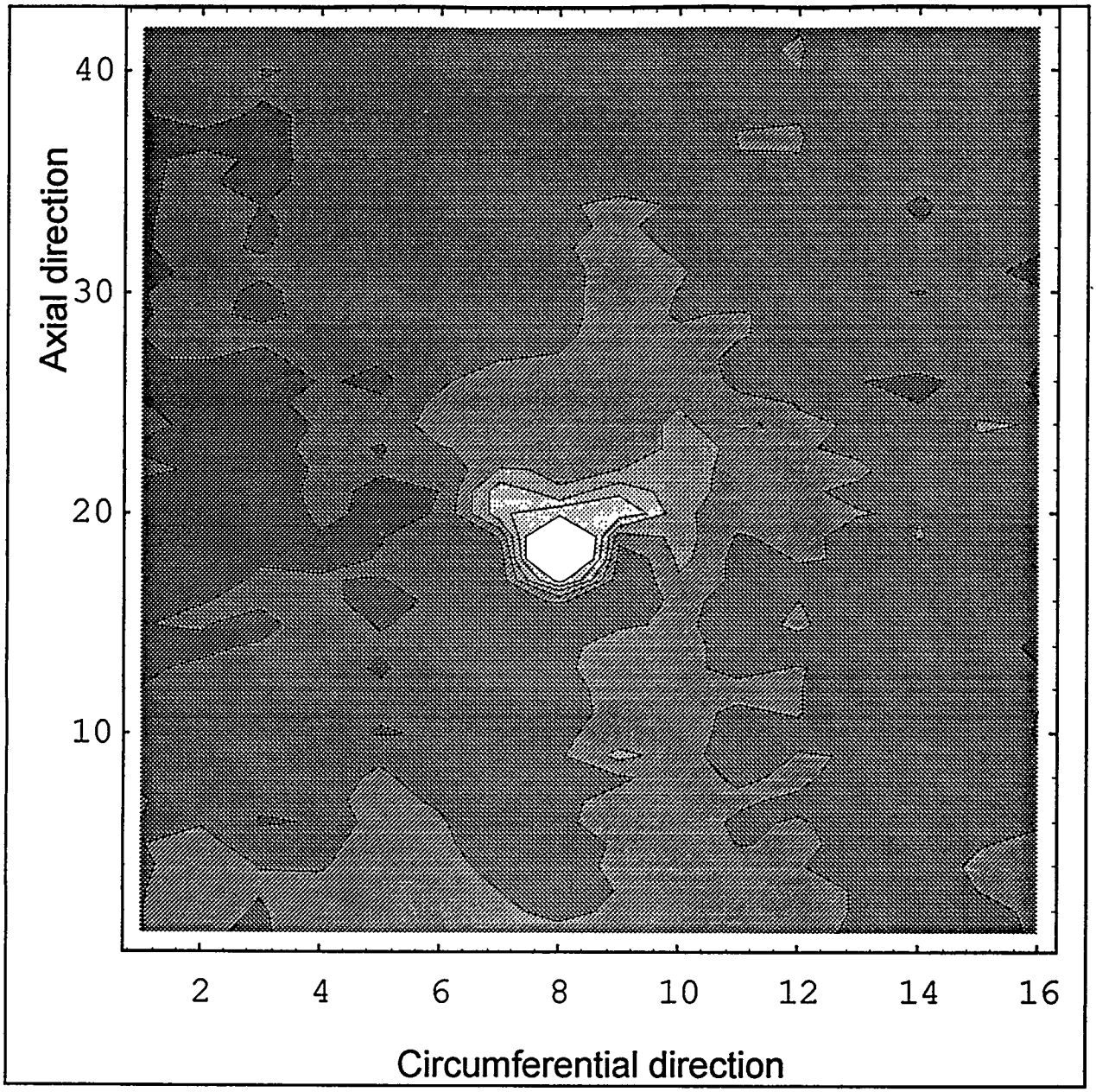


Figure 15 The vertical component of the 260 kHz reflection array probe data from tube support 01H in tube R43C59 of steam generator 12 with the tube support suppression mix applied.

APPENDIX A

Appendix A: Computation of eddy-current signals for a pancake coil

The program IMPTST.F calculates the impedance change of a pancake coil in response to a defect in a flat plate. The program is very similar to ones described in an earlier report. The program is listed below.

```
c     PROGRAM TSTIMP
C
C     August 9, 1994  VERSION, CALCULATES IMPEDANCE FOR A SINGLE COIL.
C     CORRECTED FOR BURROWS DEFECT ERROR
C     MULTI-LAYER PROGRAMS ORNL/TM-6858 AND IN ORNL-TM-4107,
C     PP. 8-23. ALL COMPLEX OPERATIONS HAVE BEEN ELIMINATED, AND
C     THE PROGRAM NOW CALCULATES THE MAGNITUDES AND PHASES FOR ALL
C     NPT SETS OF PROPERTIES.  VARIATIONS IN THICKNESS,RESISTIVITY
C     AND DEFECTS ARE ALLOWED.  THE FREQUENCY CAN BE VARIED OVER NFT
C     AND THE LIFT-OFF VARIED OVER NLT DIFFERENT VALUES.  WHEN A DEFECT
C     IS INCLUDED THE MAGNITUDE AND PHASE WILL BE CALCULATED WITH AND
C     WITHOUT THE DEFECT.  NO ADDITIONAL SET OF PROPERTY VALUES (NPT)
C     WITHOUT THE DEFECT IS NEEDED, BUT NPTT IS INCREASED.  THE ACTUAL
C     NUMERICAL VALUE OF THE SUBSCRIPT MUST BE SUBSTITUTED IN FOR THE
C     SUBSCRIPT NAME IN THE DIMENSION STATEMENTS IN LINES 59-73.
c     defect depths are modified for far side defects with small c
c     all options have been removed.  Program generates the data for
c     pancake coils that are stored in pan.dat file.  The output can
c     be run by the FINDFIT.F program.
C
C     DFDP=DISTANCE OF DEFECT FROM SURFACE OF LAYER (IN INCHES)
C     DFLOC=DISTANCE OF DEFECT BELOW TOP OF THE REGION (IN INCHES)
C     DFRAD=DISTANCE OF DEFECT FROM AXIS (IN INCHES)
C     DFDIAM=DEFECT DIAMETER(OF EQUIVALENT SPHERE, IN INCHES)
C     DFVOL=NORMALIZED VOLUME OF DEFECT
C     IRDPRM=MAXIMUM NUMBER OF COEFFICIENTS IN EXPANSION
C     LI=LOGICAL INPUT UNIT
C     LOU=LOGICAL OUTPUT UNIT
C     NCOIL=POSITION INDEX OF COIL DATA IN FILE 28
C     NDFLOC=NUMBER OF DEFECT LOCATIONS IN A GIVEN SAMPLE
C     NDFSIZ=NUMBER OF DEFECT SIZES AT A GIVEN LOCATION
C     NDPS=NDFLOC*NDFSIZ=NUMBER OF DEFECTS PER SAMPLE
C     NFT=NUMBER OF FREQUENCIES
C     NLT=NUMBER OF LIFTOFF VALUES
C     NODF=TOTAL NUMBER OF DEFECTS
C     NPT=NUMBER OF SETS OF PROPERTIES
C     NPTT=NPT+NODF=NUMBER OF PROPERTIES+NUMBER OF DEFECTS
C     NPTT=NPT+NODF=NPT+NPT*NDPS=NPT*(1+NDFLOC*NDSIZ)
C     NRDF=NUMBER OF THE REGION WHERE DEFECT IS FOUND(=NRT IF NO DEFECT)
C     NRT=NUMBER OF MATERIAL REGIONS, NO 1 IS FAREST, NRT IS COIL REG.
C
      CHARACTER*6 NPROBE,COIL
      CHARACTER*4 PROPTY(7),CKTPAR(8),UNITS(7,2),UN(7)
      REAL L2,L3,L6
      DIMENSION CONVRT(2)
C
C           DIMENSIONS THAT ARE CHANGED:
C     DIMENSION RL1(NLT),RL2(NLT),WUSR(NRT,NPTT,NFT)
C     DIMENSION TMDFT(NFT),PHDFT(NFT),NPOL(6,NFT),COEF(IRDPRM)
C     DIMENSION BETAR(NRT),BETAI(NRT),COSF(NRT),SINF(NRT),XPON(NRT)
C     DIMENSION RES(NRT,NPT),PERM(NRT,NPT),THICK(NRT,NPT)
C     DIMENSION RHO(NRT,NPTT),U(NRT,NPTT),TH(NRT,NPTT)
C     DIMENSION DFDP(NDFLOC),DFV(NPTT),DFSIZ(NPTT),PRO(NPROPM)
C     DIMENSION NRDF(NPTT),DFRAD(NPTT),DFLOC(NPTT),FREQ(NFT),GAIN(NFT)
C     DIMENSION GAMAR(NPTT),GAMAI(NPTT),GAMADR(NPTT),GAMADI(NPTT)
C     DIMENSION TMAG(NLT,NPTT,NFT),PHASE(NLT,NPTT,NFT)
c     DIMENSION TMUTRE(NLT,NPTT,NFT),TMUTIM(NLT,NPTT,NFT)
C     DIMENSION DRIVRE(NLT,NPTT,NFT),DRIVIM(NLT,NPTT,NFT)
```

```
c     DIMENSION PICKRE(NLT,NPTT,NFT),PICKIM(NLT,NPTT,NFT)
C     DIMENSION DRVDR(NLT,NPTT,NFT),DRVDI(NLT,NPTT,NFT)
c     DIMENSION PICKDR(NLT,NPTT,NFT),PICKDI(NLT,NPTT,NFT)
C  THE APPROPRIATE NUMBERS SHOULD BE INSERTED IN THE FOLLOWING DIMENSION
C  STATEMENTS:
      DIMENSION RL1(5),RL2(5),WUSR(4,156,6)
c      DIMENSION TMDFT(6),PHDFT(6),NPOL(6,6),COEF(15)
      DIMENSION BETAR(4),BETAI(4),COSF(4),SINF(4),XPON(4)
      DIMENSION RES(4,39),PERM(4,39),THICK(4,39)
      DIMENSION RHO(4,156),U(4,156),TH(4,156)
      DIMENSION DFDP(3),DFV(156),DFSIZ(156),PRO(7)
      DIMENSION NRDF(156),DFRAD(156),DFLOC(156),FREQ(6),GAIN(6)
      DIMENSION GAMAR(156),GAMAI(156),GAMADR(156),GAMADI(156)
      DIMENSION TMAG(5,156,6),PHASE(5,156,6)
c     DIMENSION TMUTRE(5,156,6),TMUTIM(5,156,6)
      DIMENSION DRIVRE(5,156,6),DRIVIM(5,156,6)
c     DIMENSION PICKRE(5,156,6),PICKIM(5,156,6)
      DIMENSION DRVDR(5,156,6),DRVDI(5,156,6)
c     DIMENSION PICKDR(5,156,6),PICKDI(5,156,6)
      COMMON /B1/X,XX,XXXX
      COMMON /B2/R1,R2,L3,RBAR,VO,ZLDR,TNDR,RDCDR,RO,R9,CAPDR,CAPPU
      COMMON /B6/ L2,L6
      DATA TWOPI/6.28318531/,RAD/57.2957795/,CONVRT/1.,25.4/
      DATA PROPTY/'REST','PERM','THIK','L.O.','DLOC','DFSI','DF %'/
      DATA UNITS/'MOCM','REL.','INCH','INCH','INCH','CUIN','    ',
     *          'MOCM','REL.',' MM ',' MM ',' MM ','CUMM','    '/
      DATA NUNIT/1/
      DATA CKTPAR/'DR.T','PU.T','DR.R','PU.R','SE.R','SH.R',
     *'DR.C','PU.C'/
      DATA NRT/4/,NPT/39/,NLT/5/,NFT/6/,NDFLOC/3/,NDFSIZ/1/,NPROPM/7/
      DATA NPROBE/'P75   '/,IRDPRM/15/,ICOEF/1/
      DATA NCOE/21/,NPRINT/0/,LOU/9/,LO/0/,LI/5/,IR/1/,LOD/30/,INPR/41/
      DATA NRDF/156*3/
      DATA DFDP/.005,.010,.015/
      DATA DFRAD/156*1.0/,DFDIAM/.125/
C
C     THE DIFFERENT FREQUENCIES AND GAINS ARE NOW GIVEN.
C
      DATA FREQ/2.0E4,5.E4,1.0E5,2.E5,5.0E5,1.E6/
      DATA GAIN/1.,1.,1.,1.,1.,1./
C     OPEN PANCAKE COIL DATA FILE
      OPEN(28,FILE='pan.dat',STATUS='OLD')
C     OPEN OUTPUT FILE FOR READING DATA
      OPEN(LOD,FILE='p75imp.dat')
C     IN GAIN(NF) .EQ.1.0 GAIN WILL BE AUTOMATICALLY ADJUSTED
C     FOR TMAG(1,1,NF) = 4.5   THIS SHOULD BE LARGEST MAG VALUE
C
      NDPS=NDFLOC*NDFSIZ
      NDPS1=NDPS+1
      NODF=NPT*NDPS
      NPTT=NPT+NODF
      MSET=NLT*NPTT
C
C
C       THE MATERIAL PROPERTIES NOW FOLLOW,STARTING WITH THE FIRST
C       PROPERTY SET FOR THE LOWERMOST REGION. (RHO=RESISTIVITY IN
C       MICROHM-CM;U=RELATIVE PERMEABILITY;TH=THICKNESS IN INCHES)
C
      OPEN(INPR,FILE='nrcpro.dat',STATUS='OLD')
C     READ RESISTIVITY, PERMEABILITY AND THICKNESS DATA FOR EACH
C     REGION AND FOR EACH PROPERTY SET.
C     RES(NRT,NPT),PERM(NRT,NPT),THICK(NRT,NPT)
      DO 4 NP=1,NPT
      DO 3 NR=1,NRT

      READ(INPR,*)RES(NR,NP),PERM(NR,NP),THICK(NR,NP)
    3 CONTINUE
    4 CONTINUE
```

```
C     RES(NRT,NPT),PERM(NRT,NPT),THICK(NRT,NPT)
C     TIME AND DATE ARE PRINTED
   1  continue
c  1  CALL GETTIM(IHR,IMN,ISE,IFR)
c     CALL GETDAT(IYR,IMO,IDA)
c     IYR=IYR-1900
   2  FORMAT('TSTIMP      COIL ',A6)
      WRITE(LOU,2)NPROBE
      WRITE(LO,2)NPROBE
      WRITE(LO,865) NFT,MSET,NPROPM,NPTT
c     TIM1=FLOAT(IHR)*3600.+FLOAT(IMN)*60.+FLOAT(ISE)+FLOAT(IFR)/100.
C
C       THE INPUT DATA FOR THE PARAMETERS OF THE COILS
C       ARE READ FROM FILE 28. SEE FIG.2, P.4
C       AND FIG.4, P.7, ORNL-TM-4107, FOR DEFINITIONS.
C
  10 READ(28,11)COIL,RBAR,R1,R2,L3,L6,RDCDR,TNDR
  11 FORMAT(A6,5F8.4,F10.4,F8.1)
c 12  WRITE(0,11)COIL,RBAR,R1,R2,L3,L6,RDCDR,TNDR
      IF(COIL.EQ.'END   ')WRITE(0,*)' COIL NOT FOUND'
      IF(COIL.EQ.'END   ')GO TO 900
      IF(COIL.NE.NPROBE)GO TO 10
C     L2=NORMALIZED LIFTOFF INCREMENT.
      L2=.008/(RBAR*FLOAT(NLT-1))
C
C     THE CIRCUIT PARAMETERS ARE NOW GIVEN.
C     RO=DRIVER SERIES RESISTANCE(OHMS)
C     R9=PICKUP SHUNT RESISTANCE(OHMS)
C     CAPDR=DRIVER SHUNT CAPACITANCE(FARADS)
C     CAPPU=PICKUP SHUNT CAPACITANCE(FARADS)
C     VO=DRIVER OUTPUT VOLTAGE(RMS VOLTS)
C
      RO=40.
      R9=1.E6
      CAPDR=8.0E-10
      CAPPU=8.0E-10
      VO=3.535
C
C     SET UP PROPERTIES FOR THE INTEGRATION
C
      CNVT=CONVRT(NUNIT)
C     DINORM=DFDIAM/RBAR
C     VOLFAC=DINORM*DINORM*DINORM/8.
C     DFVOL=TWOPI*VOLFAC/1.5
      DO 15 NS=1,NPT
      DO 15 ND=1,NDPS1
      NP=(NS-1)*NDPS1+ND
      DO 15 NR=1,NRT
      RHO(NR,NP)=RES(NR,NS)
      TH(NR,NP)=THICK(NR,NS)
      U(NR,NP)=PERM(NR,NS)
      IF(NR.NE.NRDF(NP)) GO TO 15
      IF(ND.EQ.1) DFV(NP)=0.
      IF(ND.EQ.1) GO TO 15
c      DFLOC(NP)=DFDP(ND-1)
C     DEFECT IS LOCATED FROM THE BACK SIDE OF REGION NRDF(NP)
      DFLOC(NP)=TH(NR,NP)-DFDP(ND-1)
C      DFV(NP)=(2*DFLOC(NP)/RBAR)**3*TWOPI/12.
C     DEFECT IS A 0.125 DIA HOLE WITH SEN FACT CALCULATED AT 1/2 DEPTH
      DFV(NP)=(2*DFDP(ND-1)*.012272)/(RBAR)**3
C      WRITE(0,*)'DEFECT VOLUME =',NP,DFV(NP)
  15 CONTINUE
C
C       COMPUTE RECIPROCALS FOR LATER USE, TO AVOID DIVISIONS
C       INSIDE LOOPS.
C
  20 RRBAR = 1./RBAR
C
```

```
C     THE PERMEABILITY IS CHANGED TO SQRT(.5)/U(N) AND THE THICKNESS IS
C     CHANGED TO TH(N)*U(N)/RBAR.
C
      DO  30 NP=1,NPTT
      IF(NPRINT.NE.0) WRITE(LOU,25)
   25 FORMAT(1X)
      DO  30 NR=1,NRT
      TH(NR,NP)=U(NR,NP)*TH(NR,NP)*RRBAR
      U(NR,NP)=.707106781/U(NR,NP)
      IF(NR.NE.NRDF(NP)) GO TO 30
      DFLOC(NP)=.707106781*DFLOC(NP)*RRBAR/U(NR,NP)
C      DFRAD(NP)=DFRAD(NP)*RRBAR       DFRAD IS ALREADY NORMALIZED
   30 CONTINUE
C
C          THE INTEGRATION IS PERFORMED BY THE MIDPOINT METHOD,
C       EVALUATING AT THE CENTER OF THE INTERVAL; FOR X LARGE
C       THE INTEGRAL CONVERGES RAPIDLY,SO LARGER INTERVALS
C       ARE TAKEN.
C       IN THE INTEGRATION DRIVER IMPEDANCE AND AIR1 ARE CALCULATED.
C
      S1 = 0.01
      S2 = 5.0
      B1 = 0.0
      B2 = S2
C
C       INITIALIZE ALL SUMS TO ZERO AND CALCULATE THE VALUES OF
C       WUSR(NR,NP,NF).
C
      DO 50 NF=1,NFT
      RCON=.360198724*RBAR*RBAR*FREQ(NF)
      DO  50 NP=1,NPTT
      DO 40 NL=1,NLT
c          TMUTRE(NL,NP,NF)=0.
c          TMUTIM(NL,NP,NF)=0.
          DRIVRE(NL,NP,NF)=0.
          DRIVIM(NL,NP,NF)=0.
c          PICKRE(NL,NP,NF)=0.
c          PICKIM(NL,NP,NF)=0.
          DRVDR(NL,NP,NF)=0.
          DRVDI(NL,NP,NF)=0.
c          PICKDR(NL,NP,NF)=0.
c          PICKDI(NL,NP,NF)=0.
   40 CONTINUE
      DO 50 NR=1,NRT
          WUSR(NR,NP,NF)=RCON/(RHO(NR,NP)*U(NR,NP))
   50 CONTINUE
      AIR1=0.
c     AIR2=0.
C
   60 I1 = (B2 - B1)/S1
      X = B1 - S1*0.5
      DO 70 I=1,I1
        X = X + S1
        XX = X*X
        XXXX = XX*XX
      CALL DCOIL(S1,NLT,NPTT,NRT,NFT,RL1,RL2,GAMAR,GAMAI
     1,GAMADR,GAMADI,DRIVRE,DRIVIM,DRVDR,DRVDI,BETAR
     2,BETAI,COSF,SINF,XPON,WUSR,U,TH,NRDF,DFRAD,DFLOC,FREQ,AIR1)
C
   70  CONTINUE
       B1 = B2
       B2 = B2 + S2
       S1 = 0.05
       IF (X .LT. 9.) GO TO 60
       S1 = 0.1
       IF (X .LT. 29.) GO TO 60
       S1 = 0.2
       IF (X .LT. 39.) GO TO 60
```

```
      S1 = 0.5
      IF (X .LT. 79.) GO TO 60
C
C     THE INTEGRATION ENDS HERE.THE VALUES OF THICKNESS,U,DFLOC,DFRAD
C     ARE RESTORED TO ORIGIONAL VALUES,THEN CONVERTED TO DESIRED UNITS.
C
      DO 80 NP=1,NPTT
      DFSIZ(NP)=DFV(NP)*((RBAR*CNVT)**3)
      DO 80 NR=1,NRT
      U(NR,NP)=.707106781/U(NR,NP)
      TH(NR,NP)=CNVT*TH(NR,NP)/(U(NR,NP)*RRBAR)
      IF(NR.NE.NRDF(NP)) GO TO  80
      DFLOC(NP)=CNVT*DFLOC(NP)/(RRBAR*U(NR,NP))
      DFRAD(NP)=CNVT*DFRAD(NP)/RRBAR
   80 CONTINUE
      DO 85 NU=1,7
      UN(NU)=UNITS(NU,NUNIT)
   85 CONTINUE
c     CALL GETTIM(IHR,IMN,ISE,IFR)
c     TIM2=FLOAT(IHR)*3600.+FLOAT(IMN)*60.+FLOAT(ISE)+FLOAT(IFR)/100.
c     TIME=TIM2-TIM1
c     WRITE(LOU,86)TIME
c  86 FORMAT(' ELAPSED TIME (SEC) ',F14.2)
C
C     THE COIL VALUES ARE NOW PRINTED OUT
C
      CALL DVRCOL(LOU)
C
C     EACH SET OF PROPERTIES FOR EACH REGION IS NOW PRINTED OUT.
C
      WRITE(LOU,110)UN(3),UN(1),UN(2),UN(5),UN(5),UN(6)
  110 FORMAT(' PROP  REG   LAY TH      RESTVY     PERM  DEFECT:Z LOC',
     *' RAD LOC  SIZE(VOL)',/,'  SET',9X,A4,8X,A4,6X,A4,9X,A4,2(5X,A4))
      DO 120 NS=1,NPT
      WRITE(LOU,25)
      DO 120 ND=1,NDPS1
      NP=(NS-1)*NDPS1+ND
      DO 115 NR=1,NRT
      IF(NR.EQ.1.AND.DFV(NP).EQ.0)WRITE(LOU,130)NP,NR,TH(NR,NP)
     *,RHO(NR,NP),U(NR,NP)
      IF(NR.GT.1.AND.DFV(NP).EQ.0.0)WRITE(LOU,140)NR,TH(NR,NP)
     *,RHO(NR,NP),U(NR,NP)
      IF(NR.GT.1.AND.DFV(NP).GT.0.0.AND.NR.EQ.NRDF(NP))WRITE(LOU,150)
     *NP,NR,TH(NR,NP),RHO(NR,NP),U(NR,NP),DFLOC(NP),DFRAD(NP),DFSIZ(NP)
  115 CONTINUE
  120 CONTINUE
  130 FORMAT(2(I4),2(1PE12.4),0PF9.3)
  140 FORMAT(4X,I4,2(1PE12.4),0PF9.3)
  150 FORMAT(2(I4),2(1PE12.4),0PF9.3,3X,2(F9.4),1PE12.4)
      CALL DCIRK(TMAG,PHASE,TWOPI,RAD,DRIVRE,DRIVIM,DRVDR,DRVDI
     1,NRDF,FREQ,U,WUSR,GAIN,AIR1,NLT,NRT,NPTT,NFT,NODF,NPTT,DFV)
C
C     SET THE GAIN AT EACH FREQUENCY IF NEEDED
C
      DO 155 NF=1,NFT
      IF(GAIN(NF).EQ.1.0) GAIN(NF)=4.5/TMAG(1,1,NF)
  155 CONTINUE
C
C       NEXT THE PROPERTIES OF THE COILS ARE DETERMINED AND PRINTED
C
  160 CALL DCIRK(TMAG,PHASE,TWOPI,RAD,DRIVRE,DRIVIM,DRVDR,DRVDI
     1,NRDF,FREQ,U,WUSR,GAIN,AIR1,NLT,NRT,NPTT,NFT,NODF,NPTT,DFV)
C
C     THE CIRCUIT PARAMETERS ARE PRINTED OUT FOR THE REFLECTION
C     COIL CIRCUIT.
C
      CALL DVRCKT(LOU)
C
```

```
C WRITE INITIAL INFORMATION IN DIRECT ACCESS FILE LOD ON DISK
C
      WRITE(LOD,855)NPROBE
 855  FORMAT(A6)
      WRITE(LOD,865) NFT,MSET,NPROPM,NPTT
 865  FORMAT(4(1X,I5))
      WRITE(LOD,870)(FREQ(NF),NF=1,NFT)
      WRITE(LOD,870)(GAIN(NF),NF=1,NFT)
  870 FORMAT(6(1PE12.4))
      WRITE(LOD,875)(PROPTY(NPR),NPR=1,NPROPM)
 875  FORMAT(7(1X,A4))
C     SET REGION FOR WHICH WE WILL WRITE PROPERTY VALUES
      NREG=3
      DO 830 NP=1,NPTT
      DO 820 NL=1,NLT
      M=(NP-1)*NLT+NL
      PRO(1)=RHO(NREG,NP)
      PRO(2)=U(NREG,NP)
      PRO(3)=TH(NREG,NP)
      PRO(4)=(FLOAT(NL-1)*L2)*RBAR*CNVT
c     PRO(5)=DFLOC(NP)
C     FIT DEFECT DEPTH FROM THE BACKSIDE
      PRO(5)=TH(NREG,NP)-DFLOC(NP)
      IF(DFLOC(NP).EQ.0.)PRO(5)=0.
      PRO(6)=DFSIZ(NP)
c     PRO(7)=100*DFLOC(NP)/TH(NREG,NP)
      PRO(7)=100*(TH(NREG,NP)-DFLOC(NP))/TH(NREG,NP)
      IF(DFLOC(NP).EQ.0.)PRO(7)=0.
      WRITE(LOD,840)(TMAG(NL,NP,NF),PHASE(NL,NP,NF),NF=1,NFT),
     *(PRO(N),N=1,NPROPM)
  820 CONTINUE
  830 CONTINUE
C  840 FORMAT(3(F7.4,1X,F8.2,1X),7(F8.4,1X))
  840 FORMAT(6(F7.4,1X,F8.2,1X),7(F8.4,1X))
  900 STOP
      END
C
      SUBROUTINE DCIRK(TMAG,PHASE,TWOPI,RAD,DRIVRE,DRIVIM,DRVDR,DRVDI
     1,NRDF,FREQ,U,WUSR,GAIN,AIR1,NLT,NRT,NPT,NFT,NODF,NPTT,DFV)
C
C       COMPUTES MAGNITUDES AND PHASES OF OUTPUT VOLTAGE
C       FOR VARIOUS PROPERTIES,FREQUENCIES AND LIFTOFFS.
C
      DIMENSION TMAG(NLT,NPTT,NFT),PHASE(NLT,NPTT,NFT),
     *U(NRT,NPTT),WUSR(NRT,NPTT,NFT),NRDF(NPTT),FREQ(NFT),GAIN(NFT),
     *DRIVRE(NLT,NPTT,NFT),DRIVIM(NLT,NPTT,NFT),DRVDR(NLT,NPTT,NFT),
     *DRVDI(NLT,NPTT,NFT),DFV(NPTT)
      COMMON /B2/R1,R2,RL3,RBAR,V0,ZLDR,TNDR,RDCDR,R0,R9,CAPDR,CAPPU
      T1=TNDR/((R2-R1)*RL3)
c      T2=TNPU/((R4-R3)*RL4)
      COLFAC=1.0027518E-7*RBAR
      DVRFAC=COLFAC*T1*T1
cc    PICFAC=COLFAC*T2*T2
c     ZMUTFC=COLFAC*T1*T2
      ZLDR=DVRFAC*AIR1
c     ZLPU=PICFAC*AIR2
      DO 100 NF=1,NFT
      W=TWOPI*FREQ(NF)
      DVRF=DVRFAC*W
c     PICF=PICFAC*W
c     ZMUTF=ZMUTFC*W
      QT=V0*R9*GAIN(NF)
      X1=W*R0*CAPDR
      X2=W*R9*CAPPU
      Z1Z2RE=X1*X2-1.
      Z1Z2IM=-X1-X2
      DO 60 NP=1,NPTT
      DEF=-.1193662*DFV(NP)*WUSR(NRDF(NP),NP,NF)*U(NRDF(NP),NP)
```

```
      DO 50 NL=1,NLT
c     ZMUR=ZMUTF*(TMUTRE(NL,NP,NF)+DEF*(DRVDR(NL,NP,NF)*
c    *PICKDI(NL,NP,NF)+PICKDR(NL,NP,NF)*DRVDI(NL,NP,NF)))
c     ZMUI=ZMUTF*(TMUTIM(NL,NP,NF)-DEF*(DRVDR(NL,NP,NF)*
c    *PICKDR(NL,NP,NF)-DRVDI(NL,NP,NF)*PICKDI(NL,NP,NF)))
      ZDRR=-DVRF*(DRIVIM(NL,NP,NF)-DEF*(DRVDR(NL,NP,NF)*
     *DRVDR(NL,NP,NF)-DRVDI(NL,NP,NF)*DRVDI(NL,NP,NF)))
      ZDRI=DVRF*(DRIVRE(NL,NP,NF)+AIR1+2*DEF*
     *(DRVDR(NL,NP,NF)*DRVDI(NL,NP,NF)))
      TMAG(NL,NP,NF)=GAIN(NF)*SQRT(ZDRR*ZDRR+ZDRI*ZDRI)
      PHASE(NL,NP,NF)=RAD*(ATAN2(ZDRI,ZDRR))
   50 CONTINUE
   60 CONTINUE
  100 CONTINUE
      RETURN
      END
C
      SUBROUTINE PRFVLT(LOU,TMAG,PHASE,NRT,NPT,NFT,NLT,NODF,NPTT,RL1,
     *NRDF,FREQ,GAIN)
C
C     PRINTS OUT THE MAGNITUDES AND PHASES FOR THE DIFFERENT PROPERTY
C     SETS,FREQUENCIES AND DEFECTS.
C
      REAL L2,L6
      DIMENSION RL1(NLT),TMAG(NLT,NPTT,NFT),PHASE(NLT,NPTT,NFT)
      DIMENSION NRDF(NPTT),FREQ(NFT),GAIN(NFT)
      COMMON /B6/ L2,L6
      RL1(1)=L6
      DO 10 NL=2,NLT
      RL1(NL)=RL1(NL-1)+L2
   10 CONTINUE
      DO 110 NF=1,NFT
      WRITE(LOU,190)
      WRITE(LOU,150) FREQ(NF),GAIN(NF)
      WRITE(LOU,160)(RL1(NL),NL=1,NLT)
      WRITE(LOU,165)
      DO 100 NP=1,NPTT
      WRITE(LOU,170) NP,(TMAG(NL,NP,NF),NL=1,NLT)
      WRITE(LOU,180)(PHASE(NL,NP,NF),NL=1,NLT)
      WRITE(LOU,190)
  100 CONTINUE
  110 CONTINUE
  150 FORMAT(' FREQUENCY ',1PE13.5,'   GAIN ',1PE13.5)
  160 FORMAT('  PROP  LFT OF ',9(F9.4))
  165 FORMAT('   SET')
  170 FORMAT(1X,I5,'    MAG  ',9(F9.4))
  180 FORMAT('         PHA  ',9(F9.3))
  190 FORMAT(1X)
      RETURN
      END
C
C     SUBROUTINES FOR THE MULTI LAYER DESIGN PROGRAMS FOR COIL IMPEDANCE
C
C     SUBROUTINE GAMAML    (13 JULY 1977)
C
      SUBROUTINE GAMAML(NRT,NPT,NP,NFT,NF,BETAR,BETAI,COSF,SINF,XPON
     1,WUSR,U,TH,GAMAR,GAMAI,GAMADR,GAMADI,DFT,DFR,NRDF)
C
C     CALCULATES THE GAMMA FACTOR AND THE GAMAD FACTOR FOR ANY GIVEN SET
C     OF MATERIAL PROPERTIES CONSISTING FOR PLANER LAYERS WITH ARBITRARY
C     RESISTIVITIES,THICKNESSES,AND DEFECTS.
C
      DIMENSION BETAR(NRT),BETAI(NRT),COSF(NRT),SINF(NRT),XPON(NRT)
      DIMENSION WUSR(NRT,NPT,NFT),U(NRT,NPT),TH(NRT,NPT)
      DIMENSION GAMAR(NPT),GAMAI(NPT),GAMADR(NPT),GAMADI(NPT),DFT(NPT)
     1,DFR(NPT), NRDF(NPT)
C
C     A MATERIAL WITHOUT A DEFECT IS CALCULATED
```

```
C     AT THE SAME TIME AS THE MATERIAL WITH DEFECTS.
C
C      CALCULATE THE BETA VALUES THAT WILL BE USED AND THE LOWERMOST
C      REGION THAT WILL BE SEEN BY THE COIL.
C
      CALL BETAM(NSRT,NRT,NP,NPT,NF,NFT,BETAR,BETAI,COSF,SINF,XPON
     1,WUSR,U,TH,COSD,SIND,XPOND,DFT,NRDF)
C
C     CALCULATE THE INITIAL VALUES OF V1&V2 MATRICES AND THE DEFECT
C     MATRIX ,VD,IF THE DEFECT IS IN THE INITIAL REGION.IT IS SET TO 0
C     OTHERWISE.
C
      CALL VINITM(NSRT,NRT,NPT,NP,BETAR,BETAI,V1R,V1I,V2R,V2I
     1,VDR,VDI,NRDF,COSD,SIND,XPOND)
      NSRT=NSRT+1
      IF(NSRT.GE.NRT) GO TO 40
C
C     TRANSFORM FROM THE INITIAL REGION +1 TO THE LAST REGION,NRT.
C
      CALL VMATRM(NRT,NSRT,NRT,NPT,NP,BETAR,BETAI,COSF,SINF,XPON,
     1V1R,V1I,V2R,V2I,VDR,VDI,NRDF,COSD,SIND,XPOND)
C     THE GAMMA FACTORS ARE CALCULATED FROM THE V-MATRICES.
C
   40 CALL CPXQOT(GAMAR(NP),GAMAI(NP),V1R,V1I,V2R,V2I)
      CALL CPXQOT(GAMADR(NP),GAMADI(NP),VDR,VDI,V2R,V2I)
      RETURN
      END
C
C      VINITM SUBROUTINE   (13 JULY 1977)
C
      SUBROUTINE VINITM(NSRT,NRT,NPT,NP,BETAR,BETAI,V1R,V1I,V2R,V2I
     1,VDR,VDI,NRDF,COSD,SIND,XPOND)
C     SUBROUTINE TO CALCULATE THE INITIAL VIJ MATRIX
C     AND TRANSFORM TO REGION NSRT+1.
      DIMENSION NRDF(NPT),BETAR(NRT),BETAI(NRT)
      NR=NSRT
      NSC=NR+1
      V1R=BETAR(NSC)-BETAR(NR)
      V1I=BETAI(NSC)-BETAI(NR)
      V2R=BETAR(NSC)+BETAR(NR)
      V2I=BETAI(NSC)+BETAI(NR)
      IF(NR.EQ.NRDF(NP)) GO TO 50
      VDR=0.0
      VDI=0.0
      RETURN
   50 VDR=(BETAR(NSC)*COSD+BETAI(NSC)*SIND)*2/XPOND
      VDI=(BETAI(NSC)*COSD-BETAR(NSC)*SIND)*2/XPOND
      RETURN
      END
C          VMATRM SUBROUTINE   (13 JULY 1977)
C
      SUBROUTINE VMATRM(NSTP,NSRT,NRT,NPT,NP,BETAR,BETAI,COSF,SINF,XPON,
     1V1R,V1I,V2R,V2I,VDR,VDI,NRDF,COSD,SIND,XPOND)
C     TRANSFORMATION FROM REGION NSRT TO REGION NSTP
      DIMENSION BETAR(NRT),BETAI(NRT),COSF(NRT),SINF(NRT),XPON(NRT)
      DIMENSION NRDF(NPT)
      NR=NSRT
C     MAIN LOOP
C     DEFINE OLD VALUES AS THE CURRENT VALUE OF VJ2(NP)
   20 V1RO=V1R
      V1IO=V1I
      V2RO=V2R
      V2IO=V2I
      NSC=NR+1
C     DEFINE THE BETA FUNCTIONS AND EXPONENTIAL FUNCTIONS USED IN THE
C     TRANSFORMATION CALCULATIONS BETWEEN NR AND NR+1.
      B1=BETAR(NSC)+BETAR(NR)
      B2=BETAI(NSC)+BETAI(NR)
```

```
      B3=BETAR(NSC)-BETAR(NR)
      B4=BETAI(NSC)-BETAI(NR)
      XP1=COSF(NR)/XPON(NR)
      XP2=SINF(NR)/XPON(NR)
      XP3=COSF(NR)*XPON(NR)
      XP4=SINF(NR)*XPON(NR)
C     THE REAL & IM PARTS OF THE TRANSFORMATION MATRIX,TIJ,ARE NOW
C     CALCULATED.
      T11R=B1*XP1+B2*XP2
      T11I=B2*XP1-B1*XP2
      T12R=B3*XP3-B4*XP4
      T12I=B4*XP3+B3*XP4
      T21R=B3*XP1+B4*XP2
      T21I=B4*XP1-B3*XP2
      T22R=B1*XP3-B2*XP4
      T22I=B2*XP3+B1*XP4
C     TRANSFORM FROM VJ2(NR) TO VI2(NR+1)
      V1R=T11R*V1RO-T11I*V1IO+T12R*V2RO-T12I*V2IO
      V1I=T11I*V1RO+T11R*V1IO+T12I*V2RO+T12R*V2IO
      V2R=T21R*V1RO-T21I*V1IO+T22R*V2RO-T22I*V2IO
      V2I=T21I*V1RO+T21R*V1IO+T22I*V2RO+T22R*V2IO
      IF(NR.LT.NRDF(NP)) GO TO 50
      IF(NR.GT.NRDF(NP)) GO TO 40
C     INITIAL VDR,VDI CALCULATION IN THE DEFECT REGION
      TGF1=COSF(NR)*COSD+SINF(NR)*SIND
      TGF2=COSF(NR)*SIND-SINF(NR)*COSD
      XP1=XPOND/XPON(NR)
      VDR=(V1RO*TGF1-V1IO*TGF2)*XP1+(V2RO*TGF1+V2IO*TGF2)/XP1
      VDI=(V1RO*TGF2+V1IO*TGF1)*XP1-(V2RO*TGF2-V2IO*TGF1)/XP1
C     CALCULATIONS FOR REGIONS ABOVE THE DEFECT.
   40 VDRO=VDR
      VDIO=VDI
      VDR=2*(BETAR(NSC)*VDRO-BETAI(NSC)*VDIO)
      VDI=2*(BETAI(NSC)*VDRO+BETAR(NSC)*VDIO)
C INCREMENT REGION COUNT &EXIT IF WE HAVE REACHED THE STOP (NSTP) REG.
   50 NR=NR+1
      IF (NR.LT.NSTP) GO TO 20
      RETURN
      END
C
C     SUBROUTINE BETAM (13 JULY 1977)
C
      SUBROUTINE BETAM(NR,NRT,NP,NPT,NF,NFT,BETAR,BETAI,COSF,SINF,XPON
     1,WUSR,U,TH,COSD,SIND,XPOND,DFT,NRDF)
      DIMENSION BETAR(NRT),BETAI(NRT),COSF(NRT),SINF(NRT),XPON(NRT)
      DIMENSION WUSR(NRT,NPT,NFT),U(NRT,NPT),TH(NRT,NPT)
      DIMENSION DFT(NPT),NRDF(NPT)
      COMMON /B1/X,XX,XXXX
C     START BETA CALCULATIONS AT UPPERMOST REGION AND WORK DOWN
C     AWAY FROM COIL.
      NR=NRT
      BETAR(NR)=X
      BETAI(NR)=0.
C     SET EXPONENT SUM TO ZERO AND START CALCULATING BETAS.
      SUMEXP=0.
   20 NR=NR-1
      BETAR(NR)=U(NR,NP)*SQRT(XX+SQRT(XXXX+WUSR(NR,NP,NF)*WUSR(NR,NP,NF
     1)))
      BETAI(NR)=U(NR,NP)*U(NR,NP)*WUSR(NR,NP,NF)/BETAR(NR)
      IF (NR.EQ.1) GO TO 50
      XTH=BETAR(NR)*TH(NR,NP)
      SUMEXP=SUMEXP+XTH
      IF (SUMEXP.GT.20.) GO TO 40
      XPON(NR)=EXP(XTH)
      COSF(NR)=COS(BETAI(NR)*TH(NR,NP))
      SINF(NR)=SIN(BETAI(NR)*TH(NR,NP))
      IF(NRDF(NP).NE.NR) GO TO 20
   25 XPOND=EXP(BETAR(NR)*DFT(NP))
```

```
      COSD=COS(BETAI(NR)*DFT(NP))
      SIND=SIN(BETAI(NR)*DFT(NP))
      IF(NR.NE.1) GO TO 20
   40 RETURN
C     CALCULATE THE DEFECT VALUES IF IT IS IN REGION 1.
   50 IF(NRDF(NP).EQ.1)GO TO 25
      RETURN
      END
C
C        DCOIL (June 23, 1988)
C
      SUBROUTINE DCOIL(S1,NLT,NPT,NRT,NFT,RL1,RL2,GAMAR,GAMAI
     1,GAMADR,GAMADI,DRIVRE,DRIVIM,DRVDR,DRVDI,BETAR
     2,BETAI,COSF,SINF,XPON,WUSR,U,TH,NRDF,DFR,DFT,FREQ,AIR1)
      DIMENSION BETAR(NRT),BETAI(NRT),COSF(NRT),SINF(NRT),XPON(NRT)
      DIMENSION WUSR(NRT,NPT,NFT),U(NRT,NPT),TH(NRT,NPT)
      DIMENSION RL1(NLT),RL2(NLT),NRDF(NPT),DFR(NPT),DFT(NPT),FREQ(NFT)
      DIMENSION GAMAR(NPT),GAMAI(NPT),GAMADR(NPT),GAMADI(NPT)
     1,DRIVRE(NLT,NPT,NFT),DRIVIM(NLT,NPT,NFT),DRVDR(NLT,NPT,NFT)
     2,DRVDI(NLT,NPT,NFT)
      REAL L3,L2,L6
      COMMON /B1/X,XX,XXXX
      COMMON /B2/R1,R2,L3,RBAR,V0,ZLDR,TNDR,RDCDR,R0,R9,CAPDR,CAPPU
      COMMON /B6/ L2,L6
C
C      CALCULATION OF THE COIL PART OF THE INTEGRAND. SUBPROGRAM FOR
C      FOR PANCAKE TYPE COILS.
C
C
C       SUBROUTINE BESSEL EVALUATES THE INTEGRAL OF
C       THE PRODUCT OF THE BESSEL FUNCTION J1(X) AND ITS
C       ARGUMENT, X.
C
      CALL BESSEL(XJR2,X,R2)
      CALL BESSEL(XJR1,X,R1)
C      CALL BESSEL(XJR4,X,R4)
C      CALL BESSEL(XJR3,X,R3)
      D21 = XJR2 - XJR1
c      D43 = XJR4 - XJR3
c      S6 = S1*D43
c      S3 = S6*D21
      S7=S1*D21
      S4=S1*D21*D21
c      S5 = S6*D43
      EX3 = EXP(-X*L3)
      W1 = 1. - EX3
c      EX4 = EXP(-X*L4)
c      W2 = 1. - EX4
c      EX5 = EXP(-X*L5)
c      W3 = EX3/(EX4*EX4*EX5*EX5)
C
C     UPDATE OF AIR VALUES
C
      AIR1=AIR1+(S4+S4)*(X*L3-W1)
c      Q3=X*L4-W2
c     AIR2=AIR2+(S5+S5)*(Q3+Q3-W2*W2*W3)
C
C       BYPASS THE UPDATE OF MUTUAL INDUCTANCE QUANTITIES
C       FOR LARGE X.
C
      IF (X .GT. 30.0) GO TO 200
      EX1 = EXP(-X*L2)
      EX7 = EXP(-X*L6)
      EX2=EX1*EX1
      EX6=EX7*EX7
c      W4 = 1. - EX4*W3
      W6 = EX6*W1
c      W7 = EX5*W2*W4
```

```
c      W8=W7*EX7
       W9=W1*EX7
c      W5 = W7*W6
       W6 = W6*W1
c      W7 = EX6*W7*W7
       RL1(1)=1.0
       RL2(1) = 1.0
       DO  50 NL=2,NLT
       LMINUS=NL-1
       RL1(NL)=EX1*RL1(LMINUS)
       RL2(NL)=EX2*RL2(LMINUS)
   50 CONTINUE
c      TMUT = S3*W5
       DVR = S4*W6
c      PIC = S5*W7
       DVRD=S7*W9
c      PICD=S6*W8
C
C
C      LOOP OVER ALL NFT FREQUENCIES.
C
       DO 160 NF=1,NFT
C
C       LOOP OVER ALL THE NPT DIFFERENT SETS OF PROPERTIES,CALCULATING
C      THE VARIOUS GAMMA AND DEFECT FACTORS AS WE GO.
C
       DO 155 NP=1,NPT
       CALL GAMAML(NRT,NPT,NP,NFT,NF,BETAR,BETAI,COSF,SINF,XPON
      1,WUSR,U,TH,GAMAR,GAMAI,GAMADR,GAMADI,DFT,DFR,NRDF)
c        TMUR = TMUT*GAMAR(NP)
c        TMUI = TMUT*GAMAI(NP)
         DVRR = DVR*GAMAR(NP)
         DVRI = DVR*GAMAI(NP)
c        PICR = PIC*GAMAR(NP)
c        PICI = PIC*GAMAI(NP)
       IF(NRDF(NP).GT.NRT) GO TO 100
       Q1=X*DFR(NP)
       CALL BESEL1(Q1,RJ1)
       DRDR=DVRD*RJ1*GAMADR(NP)
       DRDI=DVRD*RJ1*GAMADI(NP)
c      PICDR=PICD*RJ1*GAMADR(NP)
c      PICDI=PICD*RJ1*GAMADI(NP)
  100 CONTINUE
         DO 150 NL=1,NLT
c          TMUTRE(NL,NP,NF) = TMUTRE(NL,NP,NF) + TMUR*RL2(NL)
c          TMUTIM(NL,NP,NF) = TMUTIM(NL,NP,NF) + TMUI*RL2(NL)
           DRIVRE(NL,NP,NF) = DRIVRE(NL,NP,NF) + DVRR*RL2(NL)
           DRIVIM(NL,NP,NF) = DRIVIM(NL,NP,NF) + DVRI*RL2(NL)
c          PICKRE(NL,NP,NF) = PICKRE(NL,NP,NF) + PICR*RL2(NL)
c          PICKIM(NL,NP,NF) = PICKIM(NL,NP,NF) + PICI*RL2(NL)
          IF(NRDF(NP).GT.NRT) GO TO 150
           DRVDR(NL,NP,NF)=DRVDR(NL,NP,NF)+DRDR*RL1(NL)
           DRVDI(NL,NP,NF)=DRVDI(NL,NP,NF)+DRDI*RL1(NL)
c          PICKDR(NL,NP,NF)=PICKDR(NL,NP,NF)+PICDR*RL1(NL)
c          PICKDI(NL,NP,NF)=PICKDI(NL,NP,NF)+PICDI*RL1(NL)
  150    CONTINUE
  155  CONTINUE
  160 CONTINUE
C
  200  CONTINUE
       RETURN
       END
C
C      SUBROUTINE BESEL1    (19 JULY 1977)
C
       SUBROUTINE BESEL1(Q1,RJ1)
C
C      CALCULATES J1(Q1) .
```

```
C
      IF(Q1.GT.3) GO TO 20
      Q1S=Q1*Q1
      Q2S=((2.1E-11*Q1S-5.38E-9)*Q1S+6.757E-7)*Q1S-5.42443E-5
      Q2S=((Q2S*Q1S+2.60415E-3)*Q1S-6.25E-2)*Q1S+.5
      RJ1=Q1*Q2S
      RETURN
   20 Q3S=(((-.14604057/Q1+.27617679)/Q1-.20210391)/Q1+4.61835E-3)/Q1
      Q3S=((Q3S+.14937)/Q1+4.68E-6)/Q1+.79788456
      Q4S=(((-.21262014/Q1+.19397232)/Q1+6.022188E-2)/Q1-.17222733)/Q1
      Q4S=((Q4S+5.085E-4)/Q1+.37498836)/Q1-2.35619449+Q1
      RJ1=Q3S*COS(Q4S)/SQRT(Q1)
      RETURN
      END
C
C     SUBROUTINE DVRCOL        (20 JULY 1977)
C
      SUBROUTINE DVRCOL(LOU)
C
C  SUBROUTINE TO PRINT OUT THE REFLECTION COIL PARAMETERS.(20 JULY 77)
C
      REAL L2,L3,L6
      COMMON /B2/R1,R2,L3,RBAR,V0,ZLDR,TNDR,RDCDR,R0,R9,CAPDR,CAPPU
      COMMON /B6/ L2,L6
      WRITE(LOU,50)RBAR
      WRITE(LOU,60)
      WRITE(LOU,70)R1,R2,L3,TNDR,L6
c      WRITE(LOU,80)R3,R4,L4,TNPU,L5
      WRITE(LOU,90)
   50 FORMAT('MEAN RADIUS ',F12.5,' INCHES; NORMALIZED COIL DIMENSIONS')
   60 FORMAT('    COIL     INN. RAD    OUT. RAD     LENGTH     TURNS EA
     1   O LIFT OFF')
   70 FORMAT(' DRIVER  ',3(F12.4),F12.1,F14.4)
C   80 FORMAT(' PICK-UP ',3(F12.4),F12.1,F14.4)
   90 FORMAT(1X)
      RETURN
      END
C
C     SUBROUTINE DVRCKT   (20 JULY 1977)
C
      SUBROUTINE DVRCKT(LOU)
      REAL L3
      COMMON /B2/R1,R2,L3,RBAR,V0,ZLDR,TNDR,RDCDR,R0,R9,CAPDR,CAPPU
      WLCDR=1./SQRT(ZLDR*CAPDR)
c     WLCPU=1./SQRT(ZLPU*CAPPU)
      WRITE(LOU,50)
      WRITE(LOU,60)R0,RDCDR,CAPDR,ZLDR,WLCDR,V0
c     WRITE(LOU,70)R9,RDCPU,CAPPU,ZLPU,WLCPU
      WRITE(LOU,80)
   50 FORMAT('           SER/SHT RES COIL DC RES SHUNT CAP. COIL INDUCT
     1 RES.FREQ   DRIV VOLT ')
   60 FORMAT(' DVR CKT ',2(F12.3),3(1PE11.4),0PF11.4)
c  70 FORMAT(' PICK CKT',2(F12.3),3(1PE11.4),0PF11.4)
   80 FORMAT(1X)
      RETURN
      END
C
C      CPXQOT SUBROUTINE       (18 MAY 1977)
C
      SUBROUTINE CPXQOT(GAMR,GAMI,V12R,V12I,V22R,V22I)
C     CALCULATES THE COMPLEX QUOTIENT OF V12/V22,WHICH IS THE GAMMA
C     FACTOR
      Q4=V22I/V22R
      Q3=1./(V22R+V22I*Q4)
      Q4=-Q3*Q4
      GAMR=V12R*Q3-V12I*Q4
      GAMI=V12R*Q4+V12I*Q3
      RETURN
```

```
      END
C
C
C       BESSEL.FOR     (17 FEB. 1977)
C
      SUBROUTINE BESSEL(XJ1,X,R)
C
C       COMPUTES THE INTEGRAL OF THE BESSEL FUNCTION J1(X)
C       TIMES X FROM ZERO TO Z. USES POWER SERIES SUMMATION FOR
C       Z .LE. 5, AND AN ASYMPTOTIC SERIES FOR Z .GT. 5.
C       SEE ORNL-TM-3295, PP. 258-261.
C
      DATA PI04 /.785398163/
      Z=X*R
      IF (Z .GT. 5.0) GO TO 1010
      K = Z + Z
      L = K + 3
      F1 = 0.5*R*R*R
      XJ1 = F1/3.0
      T = -.25*Z*Z
      D1 = 2.
      D2 = 5.
      E = 4.
      DO 1000 I=1,L
        F1 = F1*T/D1
        XJ1 = XJ1 + F1/D2
        D1 = D1 + E
        E = E + 2.
        D2 = D2 + 2.
 1000 CONTINUE
      GO TO 1020
C
C       ASYMPTOTIC SERIES FOR Z .GT. 5
C
 1010 T = 1./Z
      X0=(((((-188.1357*T + 109.1142)*T - 23.79333)*T + 2.050931)*T
     1     - 0.1730503)*T + 0.7034845)*T - 0.064109E-3
      X1 = ((((-5.817517*T + 2.105874)*T - .6896196)*T +.4952024)*T
     1    - 0.187344E-2)*T + 0.7979095
      ARG = Z - PI04
      XJ1 = (1. - SQRT(Z)*(X1*COS(ARG) - X0*SIN(ARG)))/(X**3)
 1020 RETURN
      END
```

APPENDIX B

Appendix B: Computation of eddy-current signals for a reflection coil

Program MULRFD.F calculates the voltage change induced in a reflection coil by a defect in a flat plate. The program has been described in an earlier report. It is listed below.

```
      PROGRAM NRCRFD1
C
C     September 21, 1994 VERSION, CORRECTED FOR BURROWS DEFECT ERROR
C     MULTI-LAYER PROGRAMS ORNL/TM-6858 AND IN ORNL-TM-4107,
C     PP. 8-23. ALL COMPLEX OPERATIONS HAVE BEEN ELIMINATED, AND
C     THE PROGRAM NOW CALCULATES THE MAGNITUDES AND PHASES FOR ALL
C     NPT SETS OF PROPERTIES.  VARIATIONS IN THICKNESS,RESISTIVITY
C     AND DEFECTS ARE ALLOWED.  THE FREQUENCY CAN BE VARIED OVER NFT
C     AND THE LIFT-OFF VARIED OVER NLT DIFFERENT VALUES.  WHEN A DEFECT
C     IS INCLUDED THE MAGNITUDE AND PHASE WILL BE CALCULATED WITH AND
C     WITHOUT THE DEFECT.  NO ADDITIONAL SET OF PROPERTY VALUES (NPT)
C     WITHOUT THE DEFECT IS NEEDED, BUT NPTT IS INCREASED.  THE ACTUAL
C     NUMERICAL VALUE OF THE SUBSCRIPT MUST BE SUBSTITUTED IN FOR THE
C     SUBSCRIPT NAME IN THE DIMENSION STATEMENTS IN LINES 59-73.
c     defect depths are modified for far side defects with small c
c     all options have been removed, and calculated data is stored.
c     properties are read from a data file rather than put in program
C
C     DFDP=DISTANCE OF DEFECT FROM SURFACE OF LAYER (IN INCHES)
C     DFLOC=DISTANCE OF DEFECT BELOW TOP OF THE REGION (IN INCHES)
C     DFRAD=DISTANCE OF DEFECT FROM AXIS (IN INCHES)
C     DFDIAM=DEFECT DIAMETER(OF EQUIVALENT SPHERE, IN INCHES)
C     DFVOL=NORMALIZED VOLUME OF DEFECT
C     LI=LOGICAL INPUT UNIT
C     LOU=LOGICAL OUTPUT UNIT
C     NCOIL=POSITION INDEX OF COIL DATA IN FILE 28
C     NDFLOC=NUMBER OF DEFECT LOCATIONS IN A GIVEN SAMPLE
C     NDFSIZ=NUMBER OF DEFECT SIZES AT A GIVEN LOCATION
C     NDPS=NDFLOC*NDFSIZ=NUMBER OF DEFECTS PER SAMPLE
C     NFT=NUMBER OF FREQUENCIES
C     NLT=NUMBER OF LIFTOFF VALUES
C     NODF=TOTAL NUMBER OF DEFECTS
C     NPT=NUMBER OF SETS OF PROPERTIES
C     NPTT=NPT+NODF=NUMBER OF PROPERTIES+NUMBER OF DEFECTS
C     NPTT=NPT+NODF=NPT+NPT*NDPS=NPT*(1+NDFLOC*NDSIZ)
C     NRDF=NUMBER OF THE REGION WHERE DEFECT IS FOUND(=NRT IF NO DEFECT)
C     NRT=NUMBER OF MATERIAL REGIONS, NO 1 IS FAREST, NRT IS COIL REG.
C
      CHARACTER*6 NPROBE,COIL
      CHARACTER*4 PROPTY(7),CKTPAR(8),UNITS(7,2),UN(7)
      REAL L2,L3,L4,L5,L6
      DIMENSION CONVRT(2)
C
C             DIMENSIONS THAT ARE CHANGED:
C     DIMENSION RL1(NLT),RL2(NLT),WUSR(NRT,NPTT,NFT)
C     DIMENSION TMDFT(NFT),PHDFT(NFT),NPOL(6,NFT)
C     DIMENSION BETAR(NRT),BETAI(NRT),COSF(NRT),SINF(NRT),XPON(NRT)
C     DIMENSION RES(NRT,NPT),PERM(NRT,NPT),THICK(NRT,NPT)
C     DIMENSION RHO(NRT,NPTT),U(NRT,NPTT),TH(NRT,NPTT)
C     DIMENSION DFDP(NDFLOC),DFV(NPTT),DFSIZ(NPTT),PRO(NPROPM)
C     DIMENSION NRDF(NPTT),DFRAD(NPTT),DFLOC(NPTT),FREQ(NFT),GAIN(NFT)
C     DIMENSION GAMAR(NPTT),GAMAI(NPTT),GAMADR(NPTT),GAMADI(NPTT)
C     DIMENSION TMAG(NLT,NPTT,NFT),PHASE(NLT,NPTT,NFT)
C     DIMENSION TMUTRE(NLT,NPTT,NFT),TMUTIM(NLT,NPTT,NFT)
C     DIMENSION DRIVRE(NLT,NPTT,NFT),DRIVIM(NLT,NPTT,NFT)
C     DIMENSION PICKRE(NLT,NPTT,NFT),PICKIM(NLT,NPTT,NFT)
C     DIMENSION DRVDR(NLT,NPTT,NFT),PICKDR(NLT,NPTT,NFT)
C     DIMENSION DRVDI(NLT,NPTT,NFT),PICKDI(NLT,NPTT,NFT)
```

```
C  THE APPROPRIATE NUMBERS SHOULD BE INSERTED IN THE FOLLOWING DIMENSION
C  STATEMENTS:
      DIMENSION RL1(5),RL2(5),WUSR(4,156,6)
c      DIMENSION TMDFT(6),PHDFT(6),NPOL(6,6)
      DIMENSION BETAR(4),BETAI(4),COSF(4),SINF(4),XPON(4)
      DIMENSION RES(4,39),PERM(4,39),THICK(4,39)
      DIMENSION RHO(4,156),U(4,156),TH(4,156)
      DIMENSION DFDP(3),DFV(156),DFSIZ(156),PRO(7)
      DIMENSION NRDF(156),DFRAD(156),DFLOC(156),FREQ(6),GAIN(6)
      DIMENSION GAMAR(156),GAMAI(156),GAMADR(156),GAMADI(156)
      DIMENSION TMAG(5,156,6),PHASE(5,156,6)
      DIMENSION TMUTRE(5,156,6),TMUTIM(5,156,6)
      DIMENSION DRIVRE(5,156,6),DRIVIM(5,156,6)
      DIMENSION PICKRE(5,156,6),PICKIM(5,156,6)
      DIMENSION DRVDR(5,156,6),PICKDR(5,156,6)
      DIMENSION DRVDI(5,156,6),PICKDI(5,156,6)
      COMMON /B1/X,XX,XXXX
      COMMON /B2/R1,R2,R3,R4,L3,L4,RBAR,V0,ZLDR,ZLPU,
     *TNDR,TNPU,RDCDR,RDCPU,R0,R9,CAPDR,CAPPU
      COMMON /B6/ L2,L5,L6
      DATA TWOPI/6.28318531/,RAD/57.2957795/,CONVRT/1.,25.4/
      DATA PROPTY/'REST','PERM','THIK','L.O.','DLOC','DFSI','DF %'/
      DATA UNITS/'MOCM','REL.','INCH','INCH','INCH','CUIN','    ',
     *           'MOCM','REL.',' MM ',' MM ',' MM ','CUMM','    '/
      DATA NUNIT/1/
      DATA CKTPAR/'DR.T','PU.T','DR.R','PU.R','SE.R','SH.R',
     *'DR.C','PU.C'/
      DATA NRT/4/,NPT/39/,NLT/5/,NFT/6/,NDFLOC/3/,NDFSIZ/1/,NPROPM/7/
      DATA NPROBE/'R83D  '/
      DATA NPRINT/0/,LOU/0/,LO/0/,LI/5/,IR/1/,LOD/30/,INPR/41/
      DATA NRDF/156*3/
      DATA DFDP/.005,.010,.015/
      DATA DFRAD/156*0.832/,DFDIAM/.125/
C
C     THE DIFFERENT FREQUENCIES AND GAINS ARE NOW GIVEN.
C
      DATA FREQ/2.0E4,5.E4,1.0E5,2.0E5,5.0E5,1.0E6/
      DATA GAIN/1.,1.,1.,1.,1.,1./
      OPEN(28,FILE='ref.dat',STATUS='OLD')
C     OPEN OUTPUT FILE FOR WRITING DATA
      OPEN(LOD,FILE='r83dref.dat')
C     IN GAIN(NF) .EQ.1.0 GAIN WILL BE AUTOMATICALLY ADJUSTED
C     FOR TMAG(1,1,NF) = 4.5   THIS SHOULD BE LARGEST MAG VALUE
C
      NDPS=NDFLOC*NDFSIZ
      NDPS1=NDPS+1
      NODF=NPT*NDPS
      NPTT=NPT+NODF
      MSET=NLT*NPTT
C
C
C       THE MATERIAL PROPERTIES NOW FOLLOW,STARTING WITH THE FIRST
C       PROPERTY SET FOR THE LOWERMOST REGION. (RHO=RESISTIVITY IN
C       MICROHM-CM;U=RELATIVE PERMEABILITY;TH=THICKNESS IN INCHES)
C
      OPEN(INPR,FILE='nrcpro.dat',STATUS='OLD',ERR=1)
C     READ RESISTIVITY, PERMEABILITY AND THICKNESS DATA FOR EACH
C     REGION AND FOR EACH PROPERTY SET.
C     RES(NRT,NPT),PERM(NRT,NPT),THICK(NRT,NPT)
      DO 4 NP=1,NPT
      DO 3 NR=1,NRT
      READ(INPR,*)RES(NR,NP),PERM(NR,NP),THICK(NR,NP)
      WRITE(LOU,*)NP,NR,RES(NR,NP),PERM(NR,NP),THICK(NR,NP)
    3 CONTINUE
    4 CONTINUE
C     PROGRAM NAME AND COIL TYPE ARE PRINTED OUT
 1    CONTINUE
   2  FORMAT('NRCRFD1      COIL ',A6)
```

```
      WRITE(LOU,2)NPROBE
      WRITE(LO,2)NPROBE
      WRITE(LO,865) NFT,MSET,NPROPM,NPTT
C
C       THE INPUT DATA FOR THE PARAMETERS OF THE COILS
C       ARE READ FROM FILE 28. SEE FIG.2, P.4
C       AND FIG.4, P.7, ORNL-TM-4107, FOR DEFINITIONS.
C
   10 READ(28,11)COIL,RBAR,R1,R2,L3,R3,R4,L4,L5,L6
     *,RDCDR,RDCPU,TNDR,TNPU
   11 FORMAT(A6,9F8.4,F10.4,F11.4,2F8.1)
C 12  WRITE(0,11)COIL,RBAR,R1,R2,L3,R3,R4,L4,L5,L6,RDCDR,RDCPU,TNDR,TNPU
      IF(COIL.EQ.'END   ')WRITE(0,*)' COIL NOT FOUND'
      IF(COIL.EQ.'END   ')GO TO 900
      IF(COIL.NE.NPROBE)GO TO 10
C     L2=NORMALIZED LIFTOFF INCREMENT.
      L2=.0100/(RBAR*FLOAT(NLT-1))
C
C     THE CIRCUIT PARAMETERS ARE NOW GIVEN.
C     RO=DRIVER SERIES RESISTANCE(OHMS)
C     R9=PICKUP SHUNT RESISTANCE(OHMS)
C     CAPDR=DRIVER SHUNT CAPACITANCE(FARADS)
C     CAPPU=PICKUP SHUNT CAPACITANCE(FARADS)
C     VO=DRIVER OUTPUT VOLTAGE(RMS VOLTS)
C
      RO=40.
      R9=1.E6
      CAPDR=2.288E-9
      CAPPU=2.288E-9
      VO=3.535
C
C     SET UP PROPERTIES FOR THE INTEGRATION
C
      CNVT=CONVRT(NUNIT)
C     DINORM=DFDIAM/RBAR
C     VOLFAC=DINORM*DINORM*DINORM/8.
C     DFVOL=TWOPI*VOLFAC/1.5
      DO 15 NS=1,NPT
      DO 15 ND=1,NDPS1
      NP=(NS-1)*NDPS1+ND
      DO 15 NR=1,NRT
      RHO(NR,NP)=RES(NR,NS)
      TH(NR,NP)=THICK(NR,NS)
      U(NR,NP)=PERM(NR,NS)
      IF(NR.NE.NRDF(NP)) GO TO 15
      IF(ND.EQ.1) DFV(NP)=0.
      IF(ND.EQ.1) GO TO 15
c      DFLOC(NP)=DFDP(ND-1)
C     DEFECT IS LOCATED FROM THE BACK SIDE OF REGION NRDF(NP)
      DFLOC(NP)=TH(NR,NP)-DFDP(ND-1)
C      DFV(NP)=(2*DFLOC(NP)/RBAR)**3*TWOPI/12.
C     DEFECT IS A 0.125 DIA HOLE WITH SEN FACT CALCULATED AT 1/2 DEPTH
      DFV(NP)=(2*DFDP(ND-1)*.012272)/(RBAR)**3
C      WRITE(0,*)'DEFECT VOLUME =',NP,DFV(NP)
   15 CONTINUE
C
C       COMPUTE RECIPROCALS FOR LATER USE, TO AVOID DIVISIONS
C       INSIDE LOOPS.
C
   20 RRBAR = 1./RBAR
C
C     THE PERMEABILITY IS CHANGED TO SQRT(.5)/U(N) AND THE THICKNESS IS
C     CHANGED TO TH(N)*U(N)/RBAR.
C
      DO  30 NP=1,NPTT
      IF(NPRINT.NE.0) WRITE(LOU,25)
   25 FORMAT(1X)
      DO  30 NR=1,NRT
```

```
      TH(NR,NP)=U(NR,NP)*TH(NR,NP)*RRBAR
      U(NR,NP)=.707106781/U(NR,NP)
      IF(NR.NE.NRDF(NP)) GO TO 30
      DFLOC(NP)=.707106781*DFLOC(NP)*RRBAR/U(NR,NP)
C      DFRAD(NP)=DFRAD(NP)*RRBAR       DFRAD IS ALREADY NORMALIZED
   30 CONTINUE
C
C          THE INTEGRATION IS PERFORMED BY THE MIDPOINT METHOD,
C       EVALUATING AT THE CENTER OF THE INTERVAL; FOR X LARGE
C       THE INTEGRAL CONVERGES RAPIDLY,SO LARGER INTERVALS
C       ARE TAKEN.
C          IN THE INTEGRATION TMUT, DRIVER, PICKUP, AIR1, AND
C       AIR2 ARE CALCULATED.
C
      S1 = 0.01
      S2 = 5.0
      B1 = 0.0
      B2 = S2
C
C       INITIALIZE ALL SUMS TO ZERO AND CALCULATE THE VALUES OF
C       WUSR(NR,NP,NF).
C
      DO 50 NF=1,NFT
      RCON=.360198724*RBAR*RBAR*FREQ(NF)
      DO  50 NP=1,NPTT
      DO 40 NL=1,NLT
          TMUTRE(NL,NP,NF)=0.
          TMUTIM(NL,NP,NF)=0.
          DRIVRE(NL,NP,NF)=0.
          DRIVIM(NL,NP,NF)=0.
          PICKRE(NL,NP,NF)=0.
          PICKIM(NL,NP,NF)=0.
          DRVDR(NL,NP,NF)=0.
          DRVDI(NL,NP,NF)=0.
          PICKDR(NL,NP,NF)=0.
          PICKDI(NL,NP,NF)=0.
   40 CONTINUE
      DO 50 NR=1,NRT
          WUSR(NR,NP,NF)=RCON/(RHO(NR,NP)*U(NR,NP))
   50 CONTINUE
      AIR1=0.
      AIR2=0.
C
   60 I1 = (B2 - B1)/S1
      X = B1 - S1*0.5
      DO 70 I=1,I1
        X = X + S1
        XX = X*X
        XXXX = XX*XX
      CALL RFCOLM(S1,NLT,NPTT,NRT,NFT,RL1,RL2,GAMAR,GAMAI
     1,GAMADR,GAMADI,TMUTRE,TMUTIM,DRIVRE,DRIVIM,PICKRE,PICKIM
     2,DRVDR,DRVDI,PICKDR,PICKDI,BETAR,BETAI,COSF,SINF,XPON
     3,WUSR,U,TH,NRDF,DFRAD,DFLOC,AIR1,AIR2)
C
   70 CONTINUE
      B1 = B2
      B2 = B2 + S2
      S1 = 0.05
      IF (X .LT. 9.) GO TO 60
      S1 = 0.1
      IF (X .LT. 29.) GO TO 60
      S1 = 0.2
      IF (X .LT. 39.) GO TO 60
      S1 = 0.5
      IF (X .LT. 79.) GO TO 60
C
C     THE INTEGRATION ENDS HERE.THE VALUES OF THICKNESS,U,DFLOC,DFRAD
C     ARE RESTORED TO ORIGIONAL VALUES,THEN CONVERTED TO DESIRED UNITS.
```

```
C
      DO 80 NP=1,NPTT
      DFSIZ(NP)=DFV(NP)*((RBAR*CNVT)**3)
      DO 80 NR=1,NRT
      U(NR,NP)=.707106781/U(NR,NP)
      TH(NR,NP)=CNVT*TH(NR,NP)/(U(NR,NP)*RRBAR)
      IF(NR.NE.NRDF(NP)) GO TO  80
      DFLOC(NP)=CNVT*DFLOC(NP)/(RRBAR*U(NR,NP))
      DFRAD(NP)=CNVT*DFRAD(NP)/RRBAR
   80 CONTINUE
      DO 85 NU=1,7
      UN(NU)=UNITS(NU,NUNIT)
   85 CONTINUE
C
C     THE COIL VALUES ARE NOW PRINTED OUT
C
      CALL PRFCOL(LOU)
C
C     EACH SET OF PROPERTIES FOR EACH REGION IS NOW PRINTED OUT.
C
      WRITE(LOU,110)UN(3),UN(1),UN(2),UN(5),UN(5),UN(6)
  110 FORMAT(' PROP  REG   LAY TH      RESTVY     PERM  DEFECT:Z LOC',
     *' RAD LOC  SIZE(VOL)',/,'  SET',9X,A4,8X,A4,6X,A4,9X,A4,2(5X,A4))
      DO 120 NS=1,NPT
      WRITE(LOU,25)
      DO 120 ND=1,NDPS1
      NP=(NS-1)*NDPS1+ND
      DO 115 NR=1,NRT
      IF(NR.EQ.1.AND.DFV(NP).EQ.0)WRITE(LOU,130)NP,NR,TH(NR,NP)
     *,RHO(NR,NP),U(NR,NP)
      IF(NR.GT.1.AND.DFV(NP).EQ.0.0)WRITE(LOU,140)NR,TH(NR,NP)
     *,RHO(NR,NP),U(NR,NP)
      IF(NR.GT.1.AND.DFV(NP).GT.0.0.AND.NR.EQ.NRDF(NP))WRITE(LOU,150)
     *NP,NR,TH(NR,NP),RHO(NR,NP),U(NR,NP),DFLOC(NP),DFRAD(NP),DFSIZ(NP)
  115 CONTINUE
  120 CONTINUE
  130 FORMAT(2(I4),2(1PE12.4),0PF9.3)
  140 FORMAT(4X,I4,2(1PE12.4),0PF9.3)
  150 FORMAT(2(I4),2(1PE12.4),0PF9.3,3X,2(F9.4),1PE12.4)
      CALL RFCIRK(TMAG,PHASE,TWOPI,RAD,TMUTRE,TMUTIM,DRIVRE
     1,DRIVIM,PICKRE,PICKIM,DRVDR,DRVDI,PICKDR,PICKDI,NRDF,FREQ
     2,U,WUSR,GAIN,AIR1,AIR2,NLT,NRT,NFT,NPTT,DFV)
C
C     SET THE GAIN AT EACH FREQUENCY IF NEEDED
C
      DO 155 NF=1,NFT
      IF(GAIN(NF).EQ.1.0) GAIN(NF)=4.5/TMAG(1,1,NF)
  155 CONTINUE
C
C       NEXT THE PROPERTIES OF THE COILS ARE DETERMINED AND PRINTED
C
  160 CALL RFCIRK(TMAG,PHASE,TWOPI,RAD,TMUTRE,TMUTIM,DRIVRE
     1,DRIVIM,PICKRE,PICKIM,DRVDR,DRVDI,PICKDR,PICKDI,NRDF,FREQ
     2,U,WUSR,GAIN,AIR1,AIR2,NLT,NRT,NFT,NPTT,DFV)
C
C     THE CIRCUIT PARAMETERS ARE PRINTED OUT FOR THE REFLECTION
C     COIL CIRCUIT.
C
      CALL PRFCKT(LOU)
C

C WRITE INITIAL INFORMATION IN DIRECT ACCESS FILE LOD ON DISK
C
      WRITE(LOD,855)NPROBE
 855  FORMAT(A6)
      WRITE(LOD,865) NFT,MSET,NPROPM,NPTT
 865  FORMAT(4(1X,I5))
      WRITE(LOD,870)(FREQ(NF),NF=1,NFT)
```

```
      WRITE(LOD,870)(GAIN(NF),NF=1,NFT)
  870 FORMAT(6(1PE12.4))
      WRITE(LOD,875)PROPTY(3),PROPTY(4)
  875 FORMAT(7(1X,A4))
C     SET REGION FOR WHICH WE WILL WRITE PROPERTY VALUES
      NREG=3
      DO 830 NP=1,NPTT
      DO 820 NL=1,NLT
      M=(NP-1)*NLT+NL
      PRO(1)=RHO(NREG,NP)
      PRO(2)=U(NREG,NP)
      PRO(3)=TH(NREG,NP)
      PRO(4)=(FLOAT(NL-1)*L2)*RBAR*CNVT
c     PRO(5)=DFLOC(NP)
C     FIT DEFECT DEPTH FROM THE BACKSIDE
      PRO(5)=TH(NREG,NP)-DFLOC(NP)
      IF(DFLOC(NP).EQ.0.)PRO(5)=0.
      PRO(6)=DFSIZ(NP)
c     PRO(7)=100*DFLOC(NP)/TH(NREG,NP)
      PRO(7)=100*(TH(NREG,NP)-DFLOC(NP))/TH(NREG,NP)
      IF(DFLOC(NP).EQ.0.)PRO(7)=0.
      WRITE(LOD,840)(TMAG(NL,NP,NF),PHASE(NL,NP,NF),NF=1,NFT),
     *(PRO(N),N=1,NPROPM)
  820 CONTINUE
  830 CONTINUE
C  840 FORMAT(3(F7.4,1X,F8.2,1X),7(F8.4,1X))
  840 FORMAT(6(F7.4,1X,F8.2,1X),7(F8.4,1X))
  900 STOP
      END
C
      SUBROUTINE RFCIRK(TMAG,PHASE,TWOPI,RAD,TMUTRE,TMUTIM,DRIVRE,
     *DRIVIM,PICKRE,PICKIM,DRVDR,DRVDI,PICKDR,PICKDI,NRDF,FREQ,
     *U,WUSR,GAIN,AIR1,AIR2,NLT,NRT,NFT,NPTT,DFV)
C
C       COMPUTES MAGNITUDES AND PHASES OF OUTPUT VOLTAGE
C       FOR VARIOUS PROPERTIES,FREQUENCIES AND LIFTOFFS.
C
      DIMENSION TMAG(NLT,NPTT,NFT),PHASE(NLT,NPTT,NFT),
     *U(NRT,NPTT),WUSR(NRT,NPTT,NFT),NRDF(NPTT),FREQ(NFT),GAIN(NFT),
     *TMUTRE(NLT,NPTT,NFT),TMUTIM(NLT,NPTT,NFT),DRIVRE(NLT,NPTT,NFT),
     *DRIVIM(NLT,NPTT,NFT),PICKRE(NLT,NPTT,NFT),PICKIM(NLT,NPTT,NFT)
      DIMENSION DRVDR(NLT,NPTT,NFT),PICKDR(NLT,NPTT,NFT),DFV(NPTT)
      DIMENSION DRVDI(NLT,NPTT,NFT),PICKDI(NLT,NPTT,NFT)
      COMMON /B2/R1,R2,R3,R4,RL3,RL4,RBAR,V0,ZLDR,ZLPU,
     *          TNDR,TNPU,RDCDR,RDCPU,R0,R9,CAPDR,CAPPU
      T1=TNDR/((R2-R1)*RL3)
      T2=TNPU/((R4-R3)*RL4)
      COLFAC=1.0027518E-7*RBAR
      DVRFAC=COLFAC*T1*T1
      PICFAC=COLFAC*T2*T2
      ZMUTFC=COLFAC*T1*T2
      ZLDR=DVRFAC*AIR1
      ZLPU=PICFAC*AIR2
      DO 100 NF=1,NFT
      W=TWOPI*FREQ(NF)
      DVRF=DVRFAC*W
      PICF=PICFAC*W
      ZMUTF=ZMUTFC*W
      QT=V0*R9*GAIN(NF)
      X1=W*R0*CAPDR
      X2=W*R9*CAPPU
      Z1Z2RE=X1*X2-1.
      Z1Z2IM=-X1-X2
      DO 60 NP=1,NPTT
      DEF=-.1193662*DFV(NP)*WUSR(NRDF(NP),NP,NF)*U(NRDF(NP),NP)
      DO 50 NL=1,NLT
      ZMUR=ZMUTF*(TMUTRE(NL,NP,NF)+DEF*(DRVDR(NL,NP,NF)*
     *PICKDI(NL,NP,NF)+PICKDR(NL,NP,NF)*DRVDI(NL,NP,NF)))
```

```
      ZMUI=ZMUTF*(TMUTIM(NL,NP,NF)-DEF*(DRVDR(NL,NP,NF)*
     *PICKDR(NL,NP,NF)-DRVDI(NL,NP,NF)*PICKDI(NL,NP,NF)))
      ZDRR=-DVRF*(DRIVIM(NL,NP,NF)-DEF*(DRVDR(NL,NP,NF)*
     *DRVDR(NL,NP,NF)-DRVDI(NL,NP,NF)*DRVDI(NL,NP,NF)))
      ZDRI=DVRF*(DRIVRE(NL,NP,NF)+AIR1+2*DEF*
     *(DRVDR(NL,NP,NF)*DRVDI(NL,NP,NF)))
      ZPUR=-PICF*(PICKIM(NL,NP,NF)-DEF*(PICKDR(NL,NP,NF)*
     *PICKDR(NL,NP,NF)-PICKDI(NL,NP,NF)*PICKDI(NL,NP,NF)))
      ZPUI=PICF*(PICKRE(NL,NP,NF)+AIR2+2*DEF*
     *(PICKDR(NL,NP,NF)*PICKDI(NL,NP,NF)))
      ZPR=ZDRR+RDCDR
      C1=ZPR*X1+ZDRI
      C2=X1*ZDRI-R0-ZPR
      ZPR=ZPUR+RDCPU
      C3=ZPR*X2+ZPUI
      C4=X2*ZPUI-R9-ZPR
      Z5SQRE=(ZMUR+ZMUI)*(ZMUR-ZMUI)
      ZPR=ZMUR*ZMUI
      Z5SQIM=ZPR+ZPR
      DENRE=Z1Z2RE*Z5SQRE-Z1Z2IM*Z5SQIM+C1*C3-C2*C4
      DENIM=Z1Z2RE*Z5SQIM+Z1Z2IM*Z5SQRE+C1*C4+C2*C3
      TNMRE=QT*ZMUI
      TNMIM=-QT*ZMUR
      TMAG(NL,NP,NF)=SQRT((TNMRE**2+TNMIM**2)/(DENRE**2+DENIM**2))
      PHASE(NL,NP,NF)=RAD*(ATAN2(TNMIM*DENRE-TNMRE*DENIM,
     *                           TNMRE*DENRE+TNMIM*DENIM))
   50 CONTINUE
   60 CONTINUE
  100 CONTINUE
      RETURN
      END
C
      SUBROUTINE PRFVLT(LOU,TMAG,PHASE,NFT,NLT,NPTT,RL1,
     *FREQ,GAIN)
C
C     PRINTS OUT THE MAGNITUDES AND PHASES FOR THE DIFFERENT PROPERTY
C     SETS,FREQUENCIES AND DEFECTS.
C
      REAL L2,L5,L6
      DIMENSION RL1(NLT),TMAG(NLT,NPTT,NFT),PHASE(NLT,NPTT,NFT)
      DIMENSION FREQ(NFT),GAIN(NFT)
      COMMON /B6/ L2,L5,L6
      RL1(1)=L6
      DO 10 NL=2,NLT
      RL1(NL)=RL1(NL-1)+L2
   10 CONTINUE
      DO 110 NF=1,NFT
      WRITE(LOU,190)
      WRITE(LOU,150) FREQ(NF),GAIN(NF)
      WRITE(LOU,160)(RL1(NL),NL=1,NLT)
      WRITE(LOU,165)
      DO 100 NP=1,NPTT
      WRITE(LOU,170) NP,(TMAG(NL,NP,NF),NL=1,NLT)
      WRITE(LOU,180)(PHASE(NL,NP,NF),NL=1,NLT)
      WRITE(LOU,190)
  100 CONTINUE
  110 CONTINUE
  150 FORMAT(' FREQUENCY ',1PE13.5,'   GAIN ',1PE13.5)
  160 FORMAT('  PROP  LFT OF ',9(F9.4))
  165 FORMAT('   SET')
  170 FORMAT(1X,I5,'    MAG  ',9(F9.4))
  180 FORMAT('         PHA  ',9(F9.3))
  190 FORMAT(1X)
      RETURN
      END
C
C     SUBROUTINES FOR THE MULTI LAYER DESIGN PROGRAMS FOR REFLECTION
C     TYPE COILS.
```

```
C
C     SUBROUTINE GAMAML    (13 JULY 1977)
C
      SUBROUTINE GAMAML(NRT,NPT,NP,NFT,NF,BETAR,BETAI,COSF,SINF,XPON
     1,WUSR,U,TH,GAMAR,GAMAI,GAMADR,GAMADI,DFT,NRDF)
C
C     CALCULATES THE GAMMA FACTOR AND THE GAMAD FACTOR FOR ANY GIVEN SET
C     OF MATERIAL PROPERTIES CONSISTING FOR PLANER LAYERS WITH ARBITRARY
C     RESISTIVITIES,THICKNESSES,AND DEFECTS.
C
      DIMENSION BETAR(NRT),BETAI(NRT),COSF(NRT),SINF(NRT),XPON(NRT)
      DIMENSION WUSR(NRT,NPT,NFT),U(NRT,NPT),TH(NRT,NPT)
      DIMENSION GAMAR(NPT),GAMAI(NPT),GAMADR(NPT),GAMADI(NPT),DFT(NPT)
     1, NRDF(NPT)
C
C     A MATERIAL WITHOUT A DEFECT IS CALCULATED
C     AT THE SAME TIME AS THE MATERIAL WITH DEFECTS.
C
C      CALCULATE THE BETA VALUES THAT WILL BE USED AND THE LOWERMOST
C      REGION THAT WILL BE SEEN BY THE COIL.
C
      CALL BETAM(NSRT,NRT,NP,NPT,NF,NFT,BETAR,BETAI,COSF,SINF,XPON
     1,WUSR,U,TH,COSD,SIND,XPOND,DFT,NRDF)
C
C     CALCULATE THE INITIAL VALUES OF V1&V2 MATRICES AND THE DEFECT
C     MATRIX ,VD,IF THE DEFECT IS IN THE INITIAL REGION.IT IS SET TO 0
C     OTHERWISE.
C
      CALL VINITM(NSRT,NRT,NPT,NP,BETAR,BETAI,V1R,V1I,V2R,V2I
     1,VDR,VDI,NRDF,COSD,SIND,XPOND)
      NSRT=NSRT+1
      IF(NSRT.GE.NRT) GO TO 40
C
C     TRANSFORM FROM THE INITIAL REGION +1 TO THE LAST REGION,NRT.
C
      CALL VMATRM(NRT,NSRT,NRT,NPT,NP,BETAR,BETAI,COSF,SINF,XPON,
     1V1R,V1I,V2R,V2I,VDR,VDI,NRDF,COSD,SIND,XPOND)
C     THE GAMMA FACTORS ARE CALCULATED FROM THE V-MATRICES.
C
   40 CALL CPXQOT(GAMAR(NP),GAMAI(NP),V1R,V1I,V2R,V2I)
      CALL CPXQOT(GAMADR(NP),GAMADI(NP),VDR,VDI,V2R,V2I)
      RETURN
      END
C
C      VINITM SUBROUTINE   (13 JULY 1977)
C
      SUBROUTINE VINITM(NSRT,NRT,NPT,NP,BETAR,BETAI,V1R,V1I,V2R,V2I
     1,VDR,VDI,NRDF,COSD,SIND,XPOND)
C     SUBROUTINE TO CALCULATE THE INITIAL VIJ MATRIX
C     AND TRANSFORM TO REGION NSRT+1.
      DIMENSION NRDF(NPT),BETAR(NRT),BETAI(NRT)
      NR=NSRT
      NSC=NR+1
      V1R=BETAR(NSC)-BETAR(NR)
      V1I=BETAI(NSC)-BETAI(NR)
      V2R=BETAR(NSC)+BETAR(NR)
      V2I=BETAI(NSC)+BETAI(NR)
      IF(NR.EQ.NRDF(NP)) GO TO 50
      VDR=0.0
      VDI=0.0
      RETURN
   50 VDR=(BETAR(NSC)*COSD+BETAI(NSC)*SIND)*2/XPOND
      VDI=(BETAI(NSC)*COSD-BETAR(NSC)*SIND)*2/XPOND
      RETURN
      END
C          VMATRM SUBROUTINE   (13 JULY 1977)
C
      SUBROUTINE VMATRM(NSTP,NSRT,NRT,NPT,NP,BETAR,BETAI,COSF,SINF,XPON,
```

```
     1V1R,V1I,V2R,V2I,VDR,VDI,NRDF,COSD,SIND,XPOND)
C     TRANSFORMATION FROM REGION NSRT TO REGION NSTP
      DIMENSION BETAR(NRT),BETAI(NRT),COSF(NRT),SINF(NRT),XPON(NRT)
      DIMENSION NRDF(NPT)
      NR=NSRT
C     MAIN LOOP
C     DEFINE OLD VALUES AS THE CURRENT VALUE OF VJ2(NP)
   20 V1RO=V1R
      V1IO=V1I
      V2RO=V2R
      V2IO=V2I
      NSC=NR+1
C     DEFINE THE BETA FUNCTIONS AND EXPONENTIAL FUNCTIONS USED IN THE
C     TRANSFORMATION CALCULATIONS BETWEEN NR AND NR+1.
      B1=BETAR(NSC)+BETAR(NR)
      B2=BETAI(NSC)+BETAI(NR)
      B3=BETAR(NSC)-BETAR(NR)
      B4=BETAI(NSC)-BETAI(NR)
      XP1=COSF(NR)/XPON(NR)
      XP2=SINF(NR)/XPON(NR)
      XP3=COSF(NR)*XPON(NR)
      XP4=SINF(NR)*XPON(NR)
C     THE REAL & IM PARTS OF THE TRANSFORMATION MATRIX,TIJ,ARE NOW
C     CALCULATED.
      T11R=B1*XP1+B2*XP2
      T11I=B2*XP1-B1*XP2
      T12R=B3*XP3-B4*XP4
      T12I=B4*XP3+B3*XP4
      T21R=B3*XP1+B4*XP2
      T21I=B4*XP1-B3*XP2
      T22R=B1*XP3-B2*XP4
      T22I=B2*XP3+B1*XP4
C     TRANSFORM FROM VJ2(NR) TO VI2(NR+1)
      V1R=T11R*V1RO-T11I*V1IO+T12R*V2RO-T12I*V2IO
      V1I=T11I*V1RO+T11R*V1IO+T12I*V2RO+T12R*V2IO
      V2R=T21R*V1RO-T21I*V1IO+T22R*V2RO-T22I*V2IO
      V2I=T21I*V1RO+T21R*V1IO+T22I*V2RO+T22R*V2IO
      IF(NR.LT.NRDF(NP)) GO TO 50
      IF(NR.GT.NRDF(NP)) GO TO 40
C     INITIAL VDR,VDI CALCULATION IN THE DEFECT REGION
      TGF1=COSF(NR)*COSD+SINF(NR)*SIND
      TGF2=COSF(NR)*SIND-SINF(NR)*COSD
      XP1=XPOND/XPON(NR)
      VDR=(V1RO*TGF1-V1IO*TGF2)*XP1+(V2RO*TGF1+V2IO*TGF2)/XP1
      VDI=(V1RO*TGF2+V1IO*TGF1)*XP1-(V2RO*TGF2-V2IO*TGF1)/XP1
C     CALCULATIONS FOR REGIONS ABOVE THE DEFECT.
   40 VDRO=VDR
      VDIO=VDI
      VDR=2*(BETAR(NSC)*VDRO-BETAI(NSC)*VDIO)
      VDI=2*(BETAI(NSC)*VDRO+BETAR(NSC)*VDIO)
C INCREMENT REGION COUNT &EXIT IF WE HAVE REACHED THE STOP (NSTP) REG.
   50 NR=NR+1
      IF (NR.LT.NSTP) GO TO 20
      RETURN
      END
C
C     SUBROUTINE BETAM (13 JULY 1977)
C
      SUBROUTINE BETAM(NR,NRT,NP,NPT,NF,NFT,BETAR,BETAI,COSF,SINF,XPON
     1,WUSR,U,TH,COSD,SIND,XPOND,DFT,NRDF)
      DIMENSION BETAR(NRT),BETAI(NRT),COSF(NRT),SINF(NRT),XPON(NRT)
      DIMENSION WUSR(NRT,NPT,NFT),U(NRT,NPT),TH(NRT,NPT)
      DIMENSION DFT(NPT),NRDF(NPT)
      COMMON /B1/X,XX,XXXX
C     START BETA CALCULATIONS AT UPPERMOST REGION AND WORK DOWN
C     AWAY FROM COIL.
      NR=NRT
      BETAR(NR)=X
```

```
      BETAI(NR)=0.
C     SET EXPONENT SUM TO ZERO AND START CALCULATING BETAS.
      SUMEXP=0.
   20 NR=NR-1
      BETAR(NR)=U(NR,NP)*SQRT(XX+SQRT(XXXX+WUSR(NR,NP,NF)*WUSR(NR,NP,NF
     1)))
      BETAI(NR)=U(NR,NP)*U(NR,NP)*WUSR(NR,NP,NF)/BETAR(NR)
      IF (NR.EQ.1) GO TO 50
      XTH=BETAR(NR)*TH(NR,NP)
      SUMEXP=SUMEXP+XTH
      IF (SUMEXP.GT.20.) GO TO 40
      XPON(NR)=EXP(XTH)
      COSF(NR)=COS(BETAI(NR)*TH(NR,NP))
      SINF(NR)=SIN(BETAI(NR)*TH(NR,NP))
      IF(NRDF(NP).NE.NR) GO TO 20
   25 XPOND=EXP(BETAR(NR)*DFT(NP))
      COSD=COS(BETAI(NR)*DFT(NP))
      SIND=SIN(BETAI(NR)*DFT(NP))
      IF(NR.NE.1) GO TO 20
   40 RETURN
C     CALCULATE THE DEFECT VALUES IF IT IS IN REGION 1.
   50 IF(NRDF(NP).EQ.1)GO TO 25
      RETURN
      END
C
C        RFCOLM (13 JULY 1977)
C
      SUBROUTINE RFCOLM(S1,NLT,NPT,NRT,NFT,RL1,RL2,GAMAR,GAMAI
     1,GAMADR,GAMADI,TMUTRE,TMUTIM,DRIVRE,DRIVIM,PICKRE,PICKIM
     2,DRVDR,DRVDI,PICKDR,PICKDI,BETAR,BETAI,COSF,SINF,XPON
     3,WUSR,U,TH,NRDF,DFR,DFT,AIR1,AIR2)
      DIMENSION BETAR(NRT),BETAI(NRT),COSF(NRT),SINF(NRT),XPON(NRT)
      DIMENSION WUSR(NRT,NPT,NFT),U(NRT,NPT),TH(NRT,NPT)
      DIMENSION RL1(NLT),RL2(NLT),NRDF(NPT),DFR(NPT),DFT(NPT)
      DIMENSION GAMAR(NPT),GAMAI(NPT),GAMADR(NPT),GAMADI(NPT)
     1,TMUTRE(NLT,NPT,NFT),TMUTIM(NLT,NPT,NFT),DRIVRE(NLT,NPT,NFT)
     2,DRIVIM(NLT,NPT,NFT),PICKRE(NLT,NPT,NFT),PICKIM(NLT,NPT,NFT)
      DIMENSION DRVDR(NLT,NPT,NFT),PICKDR(NLT,NPT,NFT)
      DIMENSION DRVDI(NLT,NPT,NFT),PICKDI(NLT,NPT,NFT)
      REAL L3,L4,L2,L5,L6
      COMMON /B1/X,XX,XXXX
      COMMON /B2/R1,R2,R3,R4,L3,L4,RBAR,V0,ZLDR,ZLPU,
     1   TNDR,TNPU,RDCDR,RDCPU,R0,R9,CAPDR,CAPPU
      COMMON /B6/ L2,L5,L6
C
C      CALCULATION OF THE COIL PART OF THE INTEGRAND. SUBPROGRAM FOR
C      FOR REFLECTION TYPE COILS.
C
C
C       SUBROUTINE BESSEL EVALUATES THE INTEGRAL OF
C       THE PRODUCT OF THE BESSEL FUNCTION J1(X) AND ITS
C       ARGUMENT, X.
C
      CALL BESSEL(XJR2,X,R2)
      CALL BESSEL(XJR1,X,R1)
      CALL BESSEL(XJR4,X,R4)
      CALL BESSEL(XJR3,X,R3)
      D21 = XJR2 - XJR1
      D43 = XJR4 - XJR3
      S6 = S1*D43
      S3 = S6*D21
      S7=S1*D21
      S4=S7*D21
      S5 = S6*D43
      EX3 = EXP(-X*L3)
      W1 = 1. - EX3
      EX4 = EXP(-X*L4)
      W2 = 1. - EX4
```

```
      EX5 = EXP(-X*L5)
      W3 = EX3/(EX4*EX4*EX5*EX5)
C
C     UPDATE OF AIR VALUES
C
      AIR1=AIR1+(S4+S4)*(X*L3-W1)
      Q3=X*L4-W2
      AIR2=AIR2+(S5+S5)*(Q3+Q3-W2*W2*W3)
C
C       BYPASS THE UPDATE OF MUTUAL INDUCTANCE QUANTITIES
C       FOR LARGE X.
C
      IF (X .GT. 30.0) GO TO 200
      EX1 = EXP(-X*L2)
      EX7 = EXP(-X*L6)
      EX2=EX1*EX1
      EX6=EX7*EX7
      W4 = 1. - EX4*W3
      W6 = EX6*W1
      W7 = EX5*W2*W4
      W8=W7*EX7
      W9=W1*EX7
      W5 = W7*W6
      W6 = W6*W1
      W7 = EX6*W7*W7
      RL1(1)=1.0
      RL2(1) = 1.0
      DO  50 NL=2,NLT
      LMINUS=NL-1
      RL1(NL)=EX1*RL1(LMINUS)
      RL2(NL)=EX2*RL2(LMINUS)
   50 CONTINUE
      TMUT = S3*W5
      DVR = S4*W6
      PIC = S5*W7
      DVRD=S7*W9
      PICD=S6*W8
C
C
C     LOOP OVER ALL NFT FREQUENCIES.
C
      DO 160 NF=1,NFT
C
C      LOOP OVER ALL THE NPT DIFFERENT SETS OF PROPERTIES,CALCULATING
C     THE VARIOUS GAMMA AND DEFECT FACTORS AS WE GO.
C
      DO 155 NP=1,NPT
      CALL GAMAML(NRT,NPT,NP,NFT,NF,BETAR,BETAI,COSF,SINF,XPON
     1,WUSR,U,TH,GAMAR,GAMAI,GAMADR,GAMADI,DFT,NRDF)
        TMUR = TMUT*GAMAR(NP)
        TMUI = TMUT*GAMAI(NP)
        DVRR = DVR*GAMAR(NP)
        DVRI = DVR*GAMAI(NP)
        PICR = PIC*GAMAR(NP)
        PICI = PIC*GAMAI(NP)
      IF(NRDF(NP).GT.NRT) GO TO 100
      Q1=X*DFR(NP)
      CALL BESEL1(Q1,RJ1)
      DRDR=DVRD*RJ1*GAMADR(NP)
      DRDI=DVRD*RJ1*GAMADI(NP)
      PICDR=PICD*RJ1*GAMADR(NP)
      PICDI=PICD*RJ1*GAMADI(NP)
  100 CONTINUE
        DO 150 NL=1,NLT
          TMUTRE(NL,NP,NF) = TMUTRE(NL,NP,NF) + TMUR*RL2(NL)
          TMUTIM(NL,NP,NF) = TMUTIM(NL,NP,NF) + TMUI*RL2(NL)
          DRIVRE(NL,NP,NF) = DRIVRE(NL,NP,NF) + DVRR*RL2(NL)
          DRIVIM(NL,NP,NF) = DRIVIM(NL,NP,NF) + DVRI*RL2(NL)
```

```
          PICKRE(NL,NP,NF) = PICKRE(NL,NP,NF) + PICR*RL2(NL)
          PICKIM(NL,NP,NF) = PICKIM(NL,NP,NF) + PICI*RL2(NL)
         IF(NRDF(NP).GT.NRT) GO TO 150
          DRVDR(NL,NP,NF)=DRVDR(NL,NP,NF)+DRDR*RL1(NL)
          DRVDI(NL,NP,NF)=DRVDI(NL,NP,NF)+DRDI*RL1(NL)
          PICKDR(NL,NP,NF)=PICKDR(NL,NP,NF)+PICDR*RL1(NL)
          PICKDI(NL,NP,NF)=PICKDI(NL,NP,NF)+PICDI*RL1(NL)
  150    CONTINUE
  155   CONTINUE
   160 CONTINUE
C
  200   CONTINUE
        RETURN
        END
C
C       SUBROUTINE BESEL1    (19 JULY 1977)
C
        SUBROUTINE BESEL1(Q1,RJ1)
C
C       CALCULATES J1(Q1) .
C
        IF(Q1.GT.3) GO TO 20
        Q1S=Q1*Q1
        Q2S=((2.1E-11*Q1S-5.38E-9)*Q1S+6.757E-7)*Q1S-5.42443E-5
        Q2S=((Q2S*Q1S+2.60415E-3)*Q1S-6.25E-2)*Q1S+.5
        RJ1=Q1*Q2S
        RETURN
    20 Q3S=(((-.14604057/Q1+.27617679)/Q1-.20210391)/Q1+4.61835E-3)/Q1
        Q3S=((Q3S+.14937)/Q1+4.68E-6)/Q1+.79788456
        Q4S=(((-.21262014/Q1+.19397232)/Q1+6.022188E-2)/Q1-.17222733)/Q1
        Q4S=((Q4S+5.085E-4)/Q1+.37498836)/Q1-2.35619449+Q1
        RJ1=Q3S*COS(Q4S)/SQRT(Q1)
        RETURN
        END
C
C       SUBROUTINE PRFCOL        (20 JULY 1977)
C
        SUBROUTINE PRFCOL(LOU)
C
C  SUBROUTINE TO PRINT OUT THE REFLECTION COIL PARAMETERS.(20 JULY 77)
C
        REAL L2,L3,L4,L5,L6
        COMMON /B2/R1,R2,R3,R4,L3,L4,RBAR,V0,ZLDR,ZLPU,
       1   TNDR,TNPU,RDCDR,RDCPU,R0,R9,CAPDR,CAPPU
        COMMON /B6/ L2,L5,L6
        WRITE(LOU,50)RBAR
        WRITE(LOU,60)
        WRITE(LOU,70)R1,R2,L3,TNDR,L6
        WRITE(LOU,80)R3,R4,L4,TNPU,L5
        WRITE(LOU,90)
    50 FORMAT('MEAN RADIUS ',F12.5,' INCHES; NORMALIZED COIL DIMENSIONS')
    60 FORMAT('    COIL     INN. RAD    OUT. RAD     LENGTH     TURNS EA
      1   0 L.O./RCES ')
    70 FORMAT(' DRIVER  ',3(F12.4),F12.1,F14.4)
    80 FORMAT(' PICK-UP ',3(F12.4),F12.1,F14.4)
    90 FORMAT(1X)
        RETURN
        END
C
C       SUBROUTINE PRFCKT   (20 JULY 1977)
C
        SUBROUTINE PRFCKT(LOU)
        REAL L3,L4
        COMMON /B2/R1,R2,R3,R4,L3,L4,RBAR,V0,ZLDR,ZLPU,
       1   TNDR,TNPU,RDCDR,RDCPU,R0,R9,CAPDR,CAPPU
        WLCDR=1./SQRT(ZLDR*CAPDR)
        WLCPU=1./SQRT(ZLPU*CAPPU)
        WRITE(LOU,50)
```

```
      WRITE(LOU,60)RO,RDCDR,CAPDR,ZLDR,WLCDR,VO
      WRITE(LOU,70)R9,RDCPU,CAPPU,ZLPU,WLCPU
      WRITE(LOU,80)
   50 FORMAT('           SER/SHT RES COIL DC RES SHUNT CAP. COIL INDUCT
     1 RES.FREQ   DRIV VOLT ')
   60 FORMAT(' DVR CKT ',2(F12.3),3(1PE11.4),0PF11.4)
   70 FORMAT(' PICK CKT',2(F12.3),3(1PE11.4),0PF11.4)
   80 FORMAT(1X)
      RETURN
      END
C
C      CPXQOT SUBROUTINE       (18 MAY 1977)
C
      SUBROUTINE CPXQOT(GAMR,GAMI,V12R,V12I,V22R,V22I)
C     CALCULATES THE COMPLEX QUOTIENT OF V12/V22,WHICH IS THE GAMMA
C     FACTOR
      Q4=V22I/V22R
      Q3=1./(V22R+V22I*Q4)
      Q4=-Q3*Q4
      GAMR=V12R*Q3-V12I*Q4
      GAMI=V12R*Q4+V12I*Q3
      RETURN
      END
C
C
C        BESSEL.FOR     (17 FEB. 1977)
C
      SUBROUTINE BESSEL(XJ1,X,R)
C
C       COMPUTES THE INTEGRAL OF THE BESSEL FUNCTION J1(X)
C       TIMES X FROM ZERO TO Z. USES POWER SERIES SUMMATION FOR
C       Z .LE. 5, AND AN ASYMPTOTIC SERIES FOR Z .GT. 5.
C       SEE ORNL-TM-3295, PP. 258-261.
C
      DATA PI04 /.785398163/
      Z=X*R
      IF (Z .GT. 5.0) GO TO 1010
      K = Z + Z
      L = K + 3
      F1 = 0.5*R*R*R
      XJ1 = F1/3.0
      T = -.25*Z*Z
      D1 = 2.
      D2 = 5.
      E = 4.
      DO 1000 I=1,L
        F1 = F1*T/D1
        XJ1 = XJ1 + F1/D2
        D1 = D1 + E
        E = E + 2.
        D2 = D2 + 2.
 1000 CONTINUE
      GO TO 1020
C
C       ASYMPTOTIC SERIES FOR Z .GT. 5
C
 1010 T = 1./Z
      X0=(((((-188.1357*T + 109.1142)*T - 23.79333)*T + 2.050931)*T
     1     - 0.1730503)*T + 0.7034845)*T - 0.064109E-3
      X1 = ((((-5.817517*T + 2.105874)*T - .6896196)*T +.4952024)*T
     1    - 0.187344E-2)*T + 0.7979095
      ARG = Z - PI04
      XJ1 = (1. - SQRT(Z)*(X1*COS(ARG) - X0*SIN(ARG)))/(X**3)
 1020 RETURN
      END
```

APPENDIX C

Appendix C: Least squares program

Program FINDFIT.F finds the best polynomial to use to perform a fit of eddy-current data to some property of the part being tested. The techniques used have been described in earlier reports. The program is listed below.

```
      PROGRAM FINDFITN
C     VERSION August 9, 1994
C     PROGRAM TO PERFORM A LEAST SQUARES FIT TO DATA READ INTO A
C     DISK FILE BY NRCRFD, A VERSION OF MULRFD. THIS PROGRAM WILL CYCLE
C     THROUGN A SET OF FIT VALUES AND PRINT THE BEST FIT AND RMS ERROR
C     AT EACH COMBINATION OF FREQUENCIES. FAST VERSION, STORES READINGS
C     AND PROPS IN BIG ARRAY RATHER THAN READ FROM FILE EACH TIME.
C     COMPILE WITH FB (BIG ARRAY) OPTION.
C
C     IRDPRM=MAXIMUM NUMBER OF COEFFICIENTS IN EXPANSION
C     LITEK=LOGICAL INPUT UNIT
C     LOTEK=OPERATOR OUTPUT UNIT FOR PROMPTING AND DISPLAY
C     LPT=LOGICAL OUTPUT UNIT FOR PERMANENT RECORD
C     NF=FREQUENCY INDEX
C     NFT=NUMBER OF FREQUENCIES USED IN FITTING EQUATION
C     NFTT=TOTAL NUMBER OF FREQUENCIES MEASURED IN RDGANA
C     NP=PROPERTY INDEX
C     NPROPM=MAXIMUM NUMBER OF PROPERTIES CALCULATED(=7 NOW)
C     NPT=TOTAL NUMBER OF SETS OF PROPERTY VALUES USED FOR THE FIT
C     NPTT=VALUE OF NPT READ FROM FILE 30, TOTAL POINTS FROM BIGRDG
C          SOME SETS OF RDGS FROM FILE 30 MAY NOT BE USED; NPTT.GE.NPT
C     NPRINT=PRINT AND TRANSFER INDEX.
C     NSTOP=INDEX TO STOP THE INSTRUMENT READINGS
C     VAR NAME     NPOL(X,NF)     FUNCTION IS
C     NMFT           1            MAGNITUDE FUNCTION NUMBER
C     NMPT           2            MAGNITUDE POLYNOMIAL DEGREE
C     NPFT           4            PHASE FUNCTION NUMBER
C     NPPT           5            PHASE POLYNOMIAL DEGREE
C     NPCT           6            PHASE-MAG CROSS TERMS - 1
C
      CHARACTER CONDITIONS*40,PRONAM(7)*4,NPROBE*6
      CHARACTER*79 FIT1,FIT2,ERR1,ERR2
C
C  THE APPROPRIATE NUMBERS SHOULD BE INSERTED IN THE FOLLOWING
C  DIMENSION STATEMENTS;
C
      DIMENSION READNG(781,16),PRO(780),PROP(7),NBITF(6)
      DIMENSION TMAG(780,6),PHASE(780,6),FREQ(6),GAIN(6)
      DIMENSION TMDFT(6),PHDFT(6),COE(15)
      DIMENSION RDG1(15),NPOL(6,6)
      DIMENSION TMAG1(6),PHASE1(6)
C
C
C  DATA THAT MAY NEED TO BE CHANGED:
      DATA NPT/780/,NPRINT/0/,LPT/8/,LOTEK/6/,NRDG/30/,LID/37/,LOD/42/
      DATA IRDPRM/15/,IR/1/,NCHS/6/,NPROPM/7/,NPROPT/1/
C     DATA CONDITIONS/'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'/
      DATA CONDITIONS/'NRC TUBE W TUBE-SHT, MAG, CU, DEF,   THK'/
      DATA NFTT/6/,NFT/3/,NPROP/5/
      DATA NPCT,NPPT,NPFT,NMPT,NMFT/2,2,1,2,1/
      DATA SDIFM/0.100/,SDRIFM/0.15/,ERMIN0/.2/,DIFMIN0/.2/
      DATA NFREQ0/1/,NFIT/64/,DIFMIN/.2/,ERMIN/.2/
C
C PRINT TITLE AND DATE
C     TIME AND DATE ARE PRINTED
c     CALL GETTIM(IHR,IMN,ISE,IFR)
c     CALL GETDAT(IYR,IMO,IDA)
c     IYR=IYR-1900
```

```
  2   FORMAT('FINDFITN   ',A40)
      WRITE(LPT,2)CONDITIONS
C
   30 FORMAT(1X)
C
C OPEN  FILE FOR INPUT DATA FROM TSTIMP
      OPEN(unit=LID,file='r83dref.dat')
      OPEN(unit=LPT,file='r83dc')
C
C READ INITIAL INFORMATION IN DIRECT ACCESS FILE LID ON DISK
C
      READ(LID,55)NPROBE
  55  FORMAT(A6)
      READ(LID,65) NDM,NPTT,NPROPM
      WRITE(LOTEK,65) NFT,NPTT,NPROPM
      WRITE(LOTEK,140)IHR,IMN,ISE,IMO,IDA,IYR,NPTT,NPT
  65  FORMAT(3(1X,I5))
      READ(LID,70)(FREQ(NF),NF=1,NFTT)
      READ(LID,70)(GAIN(NF),NF=1,NFTT)
  70  FORMAT(6(1PE12.4))
      READ(LID,75)(PRONAM(NPR),NPR=1,NPROPM)
  75  FORMAT(7(1X,A4))
      WRITE(LOTEK,75)(PRONAM(NPR),NPR=1,NPROPM)
C     THE SCREENING OF PROPERTIES IS BE DONE AT THIS POINT.  WE WILL
C     OVERWRITE THE DATA STORED IN TMAG, PHASE AND PROP IF THE PROPERTY
C     IS NOT ONE THAT WE SELECT.
      NP=1
      DO 85 NPP=1,NPTT
  82  READ(LID,*)(TMAG(NP,NF),PHASE(NP,NF),NF=1,NFTT),
     *(PROP(NPR),NPR=1,NPROPM)
C     JUMP AROUND IF WE DO NOT WANT THIS PROPERTY AND NEXT RDGS WILL
C     OVERWRITE THE DATA IN TMAG, PHASE AND PROP ARRAYS
C     IF(PROP(3).NE.0.050) GO TO 85
      PRO(NP)=PROP(NPROP)
      NP=NP+1
C      WRITE(0,1297)NP,(TMAG(NP,NF),PHASE(NP,NF),NF=1,NFTT),PRO(NP)
1297  FORMAT(I3,15(F9.3,1X))
  85  CONTINUE
      NP=NP-1
      IF(NP.NE.NPT)WRITE(0,90)NP,NPT
   90 FORMAT(' DATA SCREENING ERROR; ',I6,' POINTS ',
     *'RATHER THAN ',I6,' POINTS USED')
C
C END OF SECTION WHICH READS DIRECT ACCESS FILE
C
C      THE DATE AND TIME THE DATA WAS TAKEN ARE PRINTED
C
      WRITE(LPT,140)IHR,IMN,ISE,IMO,IDA,IYR,NPTT,NPT,NPROBE
C      WRITE(LOD,140)IHR,IMN,ISE,IMO,IDA,IYR,NPTT,NPT,NPROBE
  140 FORMAT('DATA COMPUTED  ',I2,':',I2,':',I2,' DATE '
     *,I2,'/',I2,'/',I2,' POINTS:TOTAL=',I5,' USED=',I5,' PROBE ',A6)
  160 FORMAT(1X)
      WRITE(LPT,170)(FREQ(NF),NF=1,NFTT)
      WRITE(LPT,180)(GAIN(NF),NF=1,NFTT)
      WRITE(LPT,30)
C      WRITE(LOD,170)(FREQ(NF),NF=1,NFTT)
 170  FORMAT('FREQS ',6(1PE12.4))
 180  FORMAT('GAIN  ',6(1PE12.4))
C
C     LEAST SQUARES DESIGN SECTION.
C
C     SET UP PROPERTY ARRAYS
C     EXPAND THE RAW READINGS INTO IRDPR READINGS.
C
  200 NPTT1=NPT+1
      NNFNT=(NPCT*NPPT*NPFT*NMPT*NMFT)**NFT-1
      NNFFT=(2**NFTT)-1
C     START NNFF AT A VALUE GREATER THAN 1 TO SKIP FREQUENCY SETS
```

```
      DO 1005 NNFF=NFREQ0,NNFFT
      MFF=NNFF
      NSUM=0
      DO 235 N=1,NFTT
      NBITF(N)=MOD(MFF,2)
      MFF=MFF/2
      NSUM=NSUM+NBITF(N)
  235 CONTINUE
      IF(NSUM.NE.NFT)GO TO 1005
      IF(NBITF(NF).EQ.1)NFF=NFF+1
C     CYCLE OVER THE OFFSET COUNTER, NOFS=1,1 FOR NO OFFSET, NOFS=1,2
      DO 1004 NOFS=1,2
      IOFSET=NOFS-1
      DO 250 NF=1,NFTT
      DO 250 NC=1,6
      NPOL(NC,NF)=0
  250 CONTINUE
C     NNFN CAN BE STARTED AT THE LAST VALUE RUN TO FINISH AN
C     INTERUPTED RUN
      DO 1000 NNFN=NFIT,NNFNT
      NF=0
      MNFN=NNFN
  300 NF=NF+1
      IF(NBITF(NF).NE.1)GO TO 310
      NPC=MOD(MNFN,NPCT)
      MNFN=MNFN/NPCT
      NPP=MOD(MNFN,NPPT)+1
      MNFN=MNFN/NPPT
      NPF=2*MOD(MNFN,NPFT)+1
      MNFN=MNFN/NPFT
      NMP=MOD(MNFN,NMPT)+1
      MNFN=MNFN/NMPT
      NMF=MOD(MNFN,NMFT)+1
      MNFN=MNFN/NMFT
      NPOL(1,NF)=NMF
      NPOL(2,NF)=NMP
      NPOL(4,NF)=NPF
      NPOL(5,NF)=NPP
      NPOL(6,NF)=NPC
  310 IF(MNFN.GT.0)GO TO 300
      IRDPR=IOFSET
      DO 320 NF=1,NFTT
      IRDPR=NPOL(2,NF)+NPOL(5,NF)+NPOL(6,NF)+IRDPR
  320 CONTINUE
      IF(IRDPRM.LT.IRDPR)GO TO 1000
      IRDPR1=IRDPR+1
  470 DO 480 JF=1,NFTT
      TMDFT(JF)=0.
      PHDFT(JF)=0.
  480 CONTINUE
      NR=1
      DO 490 NP=1,NPT
      DO 481 JF=1,NFTT
      TMAG1(JF)=TMAG(NP,JF)
      PHASE1(JF)=PHASE(NP,JF)
 481  CONTINUE
C
      CALL RDGEXP(RDG1,TMAG1,PHASE1,NPOL,IOFSET,TMDFT,PHDFT,IRDPRM,NFTT)
C      WRITE(0,1298)NP,(RDG1(IR),IR=1,IRDPRM)
1298  FORMAT(I3,15(F9.3,1X))
      DO 485 IRD=1,IRDPRM
      READNG(NR,IRD)=RDG1(IRD)
 485  CONTINUE
      NR=NR+1
  490 CONTINUE
      NRF=NR-1
C
C     DO THE LEAST SQUARES FIT OF THE READINGS TO THE PROPERTIES.
```

```
C
      CALL ALSQS(READNG,PRO,COE,RSOS,NPT,IRDPR,NPTT1,IRDPR1)
C
C     CALCULATE THE DIFFERENCES IN THE FIT AND THE MAXIMUM DRIFTS
C
  500 SSDRIF=0.
      SSDIFF=0.
      NR=1
      DO 570 NP=1,NPT
      DRIFT=0.
      DO 515 JF=1,NFTT
      TMAG1(JF)=TMAG(NP,JF)
      PHASE1(JF)=PHASE(NP,JF)
  515 CONTINUE
C
      DO 540 JF=1,NFTT
      IF(NBITF(JF).EQ.0)GO TO 540
      DO 530 NC=1,2
C
C  ONE MAGNITUDE OR PHASE DRIFT IS SET ON AT A TIME.
C  TYPICAL ERROR IS 0.005 IN THE MAGNITUDE AND 0.01 IN THE PHASE
C
      IF(NC.EQ.1)TMDFT(JF)=0.005
      IF(NC.EQ.2)PHDFT(JF)=.01
      CALL RDGEXP(RDG1,TMAG1,PHASE1,NPOL,IOFSET,TMDFT,PHDFT,IRDPRM,NFTT)
C
C     THE POLYNOMIAL IS CALCULATED
C
      SUM=0.
      DO 520 IR=1,IRDPR
      SUM=SUM+COE(IR)*RDG1(IR)
  520 CONTINUE
      DRIFT=DRIFT+ABS(READNG(NR,IRDPR1)-SUM)
      TMDFT(JF)=0.
      PHDFT(JF)=0.
  530 CONTINUE
  540 CONTINUE
      DIFF=PRO(NR)-READNG(NR,IRDPR1)
      SSDIFF=SSDIFF+DIFF*DIFF
      SSDRIF=SSDRIF+DRIFT*DRIFT
  565 NR=NR+1
  570 CONTINUE
      NRD=NR-1
      IF(NPT.NE.NRF.OR.NPT.NE.NRD)WRITE(LOTEK,*)'READING ERROR:
     * ASSUMED=',NPT,'  FIT =',NRF,'  DRIFT=',NRD
      SDRIF=SQRT(SSDRIF/FLOAT(NPT))
      SDIFF=SQRT(SSDIFF/FLOAT(NPT))
      ERROR=SQRT(SDRIF*SDRIF+SDIFF*SDIFF)
      IF(ERROR.GT.ERMIN.AND.SDIFF.GT.DIFMIN) GO TO 575
      IF(ERROR.LT.ERMIN.AND.SDIFF.LT.DIFMIN) GO TO 571
      IF(SDIFF.LT.DIFMIN) WRITE(FIT1,585)IOFSET,PRONAM(NPROP)
     *,SDIFF,SDRIF,(NPOL(2,JF),NPOL(5,JF),JF=1,NFTT)
      IF(SDIFF.GT.DIFMIN) WRITE(ERR1,586)IOFSET,PRONAM(NPROP)
     *,SDIFF,SDRIF,(NPOL(2,JF),NPOL(5,JF),JF=1,NFTT)
      IF(ERROR.LT.ERMIN) WRITE(ERR2,587)NNFN,NNFF,
     *ERROR,(NPOL(3,JF),NPOL(6,JF),JF=1,NFTT)
      IF(ERROR.GT.ERMIN) WRITE(FIT2,588)NNFN,NNFF,
     *ERROR,(NPOL(3,JF),NPOL(6,JF),JF=1,NFTT)
      GO TO 574
  571 WRITE(FIT1,585)IOFDSET,PRONAM(NPROP)
     *,SDIFF,SDRIF,(NPOL(2,JF),NPOL(5,JF),JF=1,NFTT)
      WRITE(ERR1,585)IOFSET,PRONAM(NPROP)
     *,SDIFF,SDRIF,(NPOL(2,JF),NPOL(5,JF),JF=1,NFTT)
      WRITE(FIT2,587)NNFN,NNFF,
     *ERROR,(NPOL(3,JF),NPOL(6,JF),JF=1,NFTT)
      WRITE(ERR2,587)NNFN,NNFF,
     *ERROR,(NPOL(3,JF),NPOL(6,JF),JF=1,NFTT)
  574 IF(ERROR.LT.ERMIN)ERMIN=ERROR
```

```
      IF(SDIFF.LT.DIFMIN)DIFMIN=SDIFF
  575 CONTINUE
c     WRITE(LOTEK,586)IOFSET,PRONAM(NPROP),SDIFF,SDRIF,
c    *(NPOL(2,JF),NPOL(5,JF),JF=1,NFTT)
c     WRITE(LOTEK,588)NNFN,NNFF,ERROR,(NPOL(3,JF),
c    *NPOL(6,JF),JF=1,NFTT)
c     WRITE(LOTEK,30)
  580 FORMAT(' RMS DIF IN ',A4,'=',F10.5,2X,'DRIFT=',F10.5,3X
     *,'FCTN',12I2)
  585 FORMAT(I2,' CONSTANT  ',A4,'=',F10.5,'** DRIFT=',F9.5,
     *'   POL ',12I2)
  586 FORMAT(I2,' CONSTANT  ',A4,'=',F10.5,'   DRIFT=',F9.5,
     *'   POL ',12I2)
  587 FORMAT(' INDEX;FIT',I6,' FREQ',I3,'       ERROR=',F9.5,
     *'** XTRM',12I2)
  588 FORMAT(' INDEX;FIT',I6,' FREQ',I3,'       ERROR=',F9.5,
     *'   XTRM',12I2)
  630 CONTINUE
 1000 CONTINUE
      WRITE(LPT,1006)ERR1
      WRITE(LPT,1006)ERR2
      WRITE(LPT,1006)FIT1
      WRITE(LPT,1006)FIT2
      WRITE(LPT,30)
      NFIT=64
      ERMIN=ERMIN0
      DIFMIN=DIFMIN0
 1004 CONTINUE
 1005 CONTINUE
 1006 FORMAT(A79)
      GO TO 900
  991 WRITE(LOTEK,*)'ERROR IN OPENING INPUT DATA FILE'
      GO TO 900
  996 WRITE(LOTEK,*)'ERROR IN READING DATA FILE'
      GO TO 900
  900 STOP
      END
      SUBROUTINE RDGEXP(READNG,TMAG,PHASE,NPOL,IOFSET,TMDFT,PHDFT
     1,IRDPRM,NFT)
C     REAL*8 READNG,RDG
      DIMENSION READNG(IRDPRM),NPOL(6,NFT),TMDFT(NFT),PHDFT(NFT)
      DIMENSION TMAG(NFT),PHASE(NFT)
C
C     NPOL CONTAINS A NUMBER FOR THE FUNCTION TYPE, THE POLYNOMIAL
C     DEGREE, AND THE NUMBER OF CROSS TERMS
C     FOR THE MAGNITUDE AND PHASE AT EACH FREQUENCY,STORED AS NPOL
C     (NF;1-MAG FUN, 2-MAG POL,3-MAG #CROSS TERMS,4-PH FUN,5-PH POL,
C     6-PH # CROSS TERMS).  IF IOFSET=0, NO OFF-SET
C     WILL BE INCLUDED,=1 OFF-SET IS INCLUDED.THE VALUES OF TMDFT(NF)&
C     PHDFT(NF) GIVE THE AMOUNT OF DRIFT IN THE MAGNITUDE AND PHASE.IF
C      NPOL(NCP,NF) =0,THAT PARTICULAR MAGNITUDE AND PHASE FOR THAT
C     FREQUENCY WILL BE SKIPPED.
C
      READNG(1)=1.
      N=1
      IF(IOFSET.EQ.1)N=2
      DO 210 NF=1,NFT
      DO 200 NC=1,2
      NCC=NC*3
      NCP=NCC-1
      NCF=NCP-1
      ROLD=RDG
      IF(NPOL(NCP,NF).EQ.0) GO TO 200
      IF(NC.EQ.1) RDG=TMAG(NF)+TMDFT(NF)
      IF(NC.EQ.2) RDG=PHASE(NF)+PHDFT(NF)
C
C     THE TYPE OF FUNCTION IS SELECTED
C
```

```
      IF(NPOL(NCF,NF).EQ.1)RDG=RDG
      IF(NPOL(NCF,NF).EQ.2)RDG=ALOG(RDG)
      IF(NPOL(NCF,NF).EQ.3)RDG=EXP(RDG)
      IF(NPOL(NCF,NF).EQ.4)RDG=1./RDG
C
C     THE TYPE OF POLYNOMIAL IS SELECTED
C     AND THE POLYNOMIAL VALUES ARE CONSTRUCTED.
C
      READNG(N)=RDG
      N=N+1
      NDEG=NPOL(NCP,NF)-1
      IF(NDEG.LT.1) GO TO 15
      DO 10 I=1,NDEG
      READNG(N)=RDG*READNG(N-1)
      N=N+1
   10 CONTINUE
C
C     CROSS TERMS ARE CONSTRUCTED
C
15    IF(NPOL(NCC,NF).EQ.0) GO TO 200
      RDY=ROLD
      NCTERM=NPOL(NCC,NF)
      DO 20 I=1,NCTERM
      RDY=ROLD*RDY
   20 CONTINUE
      IF(RDY.NE.0) RINV=RDG/ROLD
      IF(RDY.EQ.0) RINV=0.
      DO 30 I=1,NCTERM
      RDY=RDY*RINV
      READNG(N)=RDY
      N=N+1
   30 CONTINUE
C
  200 CONTINUE
  210 CONTINUE
      RETURN
      END
      SUBROUTINE ALSQS(A,Y,B,R2,NN,MM,NA,NB)
C
C     ALSQS IS A FORTRAN IV SUBROUTINE TO SOLVE THE LINEAR LEAST
C     SQUARES PROBLEM  NORM(AB - Y) = MIN.   CALLING SEQUENCE IS
C           CALL ALSQS(A,Y,B,R2,N,M,NA)
C     WHERE
C      A   IS AN ARRAY CONTAINING THE LEAST SQUARES MATRIX.
C          UPON RETURN THE (M+1)-TH COLUMN CONTAINS THE
C          APPROXIMATING VECTOR AB.
C      Y   IS THE VECTOR TO BE FIT
C      B   CONTAINS UPON RETURN THE COEFFICIENTS OF THE FIT.
C      R2  CONTAINS UPON RETURN THE RESIDUAL SUM OF SQUARES.
C      N   IS THE NUMBER OF ROWS IN THE LEAST SQUARES MATRIX.
C      M   IS THE NUMBER OF COLUMNS IN THE LEAST SQUARES
C          MATRIX.
C      NA  IS THE FIRST DIMENSION OF THE ARRAY A.
C
C     CALL ALSQS(READNG,PRO,COE,RSOS,NPT,IRDPR,NPTT1,IRDPR1)
C     CALL ALSQS(     A,  Y,  B,  R2, NN,   MM,   NA,    NB)
C
      DIMENSION A(NA,NB),Y(NN),B(MM)
      N = NN
      N1 = N+1
      M = MM
      M1 = M+1
      MM1 = M-1
C
C     REDUCE THE LEAST SQUARES MATRIX TO UPPER TRIANGULAR FORM
C
      DO 60 L=1,M
       SS = 0.
```

```
      DO 10 I=L,N
   10    SS = SS + A(I,L)**2
      S2 = SS
      S =SQRT(S2)
      IF (A(L,L).LT.0.) S=-S
      D = S2 + S*A(L,L)
      A(L,L) = A(L,L) + S
      IF (L.EQ.M) GO TO 50
      L1 = L+1
      DO 30 J=L1,M
         PP = 0.
         DO 20 I=L,N
   20       PP = PP + A(I,L)*A(I,J)
   30    A(N1,J) = PP/D
      DO 40 J=L1,M
         DO 40 I=L,N
   40       A(I,J) = A(I,J) - A(I,L)*A(N1,J)
   50 A(N1,L) = -S
   60 CONTINUE
C
C     REDUCE THE VECTOR Y
C
      DO 80 I=1,N
   80 A(I,M1) = Y(I)
      DO 100 L=1,M
       PP = 0.
       DO 90 I=L,N
   90    PP = PP + A(I,L)*A(I,M1)
       D = PP/(-A(L,L)*A(N1,L))
       DO 100 I=L,N
  100    A(I,M1) = A(I,M1) - D*A(I,L)
C
C     CALCULATE THE COEFFICIENT VECTOR B
C
      B(M) = A(M,M1)/A(N1,M)
      IF (M.EQ.1) GO TO 130
      DO 120 LL=1,MM1
       L = M-LL
       L1 = L+1
       PP = A(L,M1)
       DO 110 I=L1,M
  110    PP = PP - A(L,I)*B(I)
  120  B(L) = PP/A(N1,L)
C
C     CALCULATE R2
C
  130 SS = 0.
      MP1 = M+1
      DO 140 I=MP1,N
       SS = SS + A(I,M1)**2
  140  A(I,M1) = 0.
      R2 = SS
C
C     PERFORM THE BACK CALCULATIONS
C
      DO 170 LL=1,M
       L = M-LL+1
       PP = 0.
       DO 150 I=L,N
  150    PP = PP + A(I,L)*A(I,M1)
       D = PP/(-A(L,L)*A(N1,L))
       DO 160 I=L,N
  160    A(I,M1) = A(I,M1) - D*A(I,L)
  170  CONTINUE
      RETURN
      END
```

APPENDIX D

Appendix D: Neural network program

This appendix describes the neural net program used for the automatic analysis of eddy-current data from the array probe. The program, when presented with training data taken on samples with known properties (such as defect depth or wall thickness), extracts the features of the data associated with the property of interest and learns the relationship between the data and the properties so that, when given an unfamiliar set of data, the program can respond by giving the value of the property most likely associated with that set of data. For the neural net program to be able to perform the analysis of data from unknown tubes adequately, the data in the training set must include examples of every type of signal that it is likely to be presented to the neural net. If, for example, the neural net is to be taught to recognize the defect depth in a tube, the training set must include scans of many types of defects such as notches and holes, and will also preferably include scans of tubes with more realistic degradation such as IGA and ODSCC. It is necessary in the training set for each type of degradation to be scanned in the presence of other artifacts which may occur near defects in the unknown tubes; therefore tube supports, magnetite deposits, and copper deposits must be placed at different locations relative to the defects in the training samples. The neural net can be expected to interpolate successfully between data sets given it in training so that if it is trained on defects of depth 40% and 60% of wall thickness, it should be able to recognize defects of depth 50% of wall thickness, but the neural net cannot be expected, for example, to recognize or be able to characterize a defect in the presence of a deposit of a type not included in the training set. The algorithm used by the neural net is the backpropagation algorithm, which is described in many books.

The first step in training the neural net is to acquire data on the training samples using programs SCAN18.C or SCAN30.C described in an earlier report. The neural net program is designed to read files written in the format of these programs. The number of training data files must be entered in the program variable nfiles, and the names of these files must be assigned to the elements of the file array. The number of position coordinates in each file must be given in the naxes array and a number identifying which tube was scanned to obtain the data should be entered in the standard array. Finally a set of positions for each axis at which a software null may be performed should be given in the nullpos array. In addition to the data which are contained in the file, it is necessary to specify the value of the property to be learned at each point in the tube at which data were taken. While this information can be, and in the past has been, included in the data file itself, it is usually more convenient to have a subroutine in the program which knows the value of the property at each point in each tube and which can thus calculate the actual value of the property based on the position and the tube identification.

After the modifications described above have been made in the neural net program, it can be compiled by executing a script with the following commands:

```
PROG=$1
cc -c -Aa +O2 $PROG.c
LDOPTS= cc -Aa -L /usr/lib/X11R4 -ldld -o $PROG $PROG.o -lXwindow -lsb
        -lXhp11 -lX11 -lm -lcurses
```

When the program is run it will first ask whether the user wants to restore a previously saved weights file to begin training. This is useful if it is necessary to interrupt and then resume training at a later time. If the user tells the program that he does not want to restore a file, the program will automatically assign small random numbers to all weights. The program then asks how many layers the user wishes to include in the neural net. The usual response is 3. The next question asked by the program is the number of elements for the input layer. The correct response to enter here is one greater than the number of readings in each data point. For example, if the data used for training contains readings taken at four frequencies with two components at each frequency, there are eight readings at each data point, so the correct resoponse to the question would be nine. The program next asks how many elements should be included in the hidden layer of the network. The usual response here is a number from four to nine. The program then asks how many elements should be in the output layer. If the user is training the network to output a single property value, the correct response here is two. The program will then ask for a value for the momentum parameter and for the learning rate. Any number less than one can be entered in response to these questions.

At this point the program will read the input data files and begin training. At the end of a number of training attempts determined by the program variable nna, the program will print out the average rms error over the last nna training passes. While the program is running it accepts the commands described below.

Command	Action
e	At the end of each nna training attempts the program will, in addition to calculating the average rms error over the last training cycle, make a pass through the entire data set and calculate the true rms error with the weights in their state at that point. If the rms error has reached a new low, the weights are automatically saved in a disk file.
l	The program will prompt the user for a new value for the learning rate.
m	The program will prompt the user for a new value for the momentum parameter.
q	The program will halt execution.
s	The program will save the weights to a disk file.

After the training has been completed, the weight files stored in the format of the neural net program can be read by the analysis program SECORT described in an earlier report to apply the results of the training to eddy-current data not included in the training set.

The program is listed below.

```
/*
 * zgbpn.c
 * Version 20 October 1994
 */
#include <stdio.h>
#include <stdlib.h>
```

```
#include <string.h>
#include <curses.h>
#include <math.h>

float set_depth_z11954(float *);
float set_depth_z11956(float *);
int find_null(int,FILE *);

int nfiles = 11;

char file[][80] = {
  "../eddynet/vbd001.dat",
  "../eddynet/vbd002.dat",
  "../eddynet/vbd003.dat",
  "../eddynet/vbd004.dat",
  "../eddynet/vbd005.dat",
  "../eddynet/vbd006.dat",
  "../eddynet/vbd007.dat",
  "../eddynet/vbe001.dat",
  "../eddynet/vbe002.dat",
  "../eddynet/vbe006.dat",
  "../eddynet/vbe007.dat"
  };

int naxes[] = {
  1,1,1,1,1,
  1,1,1,1,1,
  1};

int standard[] = {
  11956,11956,11956,11956,11956,
  11956,11956,11956,11956,11956,
  11956};

float nullpos[][3] = {
  {3.88,0.,0.},
  {4.63,0.,0.},
  {4.63,0.,0.},
  {3.13,0.,0.},
  {3.13,0.,0.},
  {3.88,0.,0.},
  {3.88,0.,0.},
  {3.88,0.,0.},
  {4.63,0.,0.},
  {3.88,0.,0.},
  {3.88,0.,0.}};

float nullval[4][2];

void main(void)
{
  FILE *strdat, *strwt, *strout;
  char wtsin[80], wtsout[80];
  int found_null;
  int eoftst;
  float x[3];
  int is_null_pos;
  int npt = 0;
  int i, j, k, l;
  int nlayers;
  int ninput, nhidden, noutput;
  int restore_from_file;
  int pattern;
  int ch;
  int tmpdatr,tmpdati;
  int data[4][2];
  int print_period = 1;
  int calculate_error = 0;
```

```
  int undo_bad_step = 0;
  long nn = 0;
  double w0[20][33], w1[2][20];
  double d0[20][33], d1[2][20];
  double last_w0[20][33], last_w1[2][20];
  double prev_epoch_w0[20][33], prev_epoch_w1[2][20];
  double prev_epoch_d0[20][33], prev_epoch_d1[2][20];
  float input[100000][33];
  float pos;
  float *p0_input, *p1_input;
  double hidden[20], output[2];
  float actual_depth[100000];
  double sserr, sum, rmserr;
  double prev_epoch_err = 1000.;
  double lowest_error = 15.;
  int lowest;
  double hidden_error[20], output_error[2];
  float desired_value[100000][2];
  double alpha, eta0;
  float caldep;
  float jog_amt;
  float datr,dati;
  float rdatr,rdati;
  float datm,datp;
  int use;

/* Pancake coil */

  float normfac[4] = {602.,888.,877.,334.};
  float phase[4] = {-57.2,-18.2,-5.5,0.};
  float rotation[4] = {116.,-18.,-91.,-133.};

/* Reflection coil */
#if(0)
  float normfac[4] = {625.,886.,707.,254.};
  float phase[4] = {263.5,-30.2,-6.5,3.1};
#endif

/* TEMPORARY */

  normfac[1] *= 1.;
  normfac[2] *= 4.;
  normfac[3] *= 8.;

  initscr();
  scrollok(stdscr, TRUE);
  move(0, 0);
  clrtobot();
  erase();
  refresh();

  for(i=0;i<4;i++) {
    phase[i] *= 3.1415927/180.;
    rotation[i] *= 3.1415927/180.;
  }

#if(0)
  if((strdat = fopen("/users/jrp/bpn30/z11956.dat", "r")) == NULL) {
    printw("Error:  failure to open input data file.\n");
    refresh();
    exit(1);
  }
#endif

  printw("Enter 1 to restore from file, 0 otherwise  ");
  refresh();
  scanw("%d", &restore_from_file);
```

```
  if(restore_from_file) {
    printw("Enter filename  ");
    refresh();
    scanw("%s", &wtsin[0]);
    if((strwt = fopen(&wtsin[0], "r")) == NULL) {
      printw("Error:  failure to open input data file.\n");
      refresh();
      exit(1);
    }
    fscanf(strwt, "%ld", &nn);
    fscanf(strwt, "%d", &nlayers);
    if(nlayers != 3) {
      printw("Error: Number of layers in file not equal to 3.\n");
      refresh();
      exit(1);
    }
    fscanf(strwt, "%d %d %d", &ninput, &nhidden, &noutput);
    fscanf(strwt, "%lf %lf", &alpha, &eta0);

    for(i=0; i<ninput; i++) {
      for(j=1; j<nhidden; j++) {
        fscanf(strwt, "%lf", &w0[j][i]);
        last_w0[j][i] = w0[j][i];
        d0[j][i] = 0.;
      }
    }
    for(i=0; i<nhidden; i++) {
      for(j=1; j<noutput; j++) {
        fscanf(strwt, "%lf", &w1[j][i]);
        last_w1[j][i] = w1[j][i];
        d1[j][i] = 0.;
      }
    }
    fclose(strwt);
  }
  else {
    printw("Enter number of elements in input layer ");
    refresh();
    scanw("%d", &ninput);
    printw("Enter number of elements in hidden layer ");
    refresh();
    scanw("%d", &nhidden);
    printw("Enter number of elements in output layer ");
    refresh();
    scanw("%d", &noutput);

    printw("Enter momentum  ");
    refresh();
    scanw("%lf", &alpha);
    printw("Enter learning rate  ");
    refresh();
    scanw("%lf", &eta0);
printw("alpha = %f, eta0 = %f\n", alpha, eta0);

    for(i=0; i<500; i++) {
      j = rand();
    }

    for(i=0; i<ninput; i++) {
      for(j=1; j<nhidden; j++) {
        w0[j][i] = 0.1 * (1./32767.) * rand() - 0.5;
        last_w0[j][i] = w0[j][i];
        d0[j][i] = 0.;
      }
    }
    for(i=0; i<nhidden; i++) {
      for(j=1; j<noutput; j++) {
        w1[j][i] = (1./32767.) * rand() - 0.5;
```

```
        last_w1[j][i] = w1[j][i];
        d1[j][i] = 0.;
      }
    }
  }

  nodelay(stdscr, TRUE);

  strout = fopen("vbcbpn.out","w");
  if(strout == NULL) {
    printf("Unable to open output data file.\n");
    exit(-1);
  }

  i = 0;
  for(k=0;k<nfiles;k++) {
    printw("Reading file %d,%s\n",k,file[k]);
    refresh();
    if((strdat = fopen(file[k], "r")) == NULL) {
      printw("Error:  failure to open input data file %s.\n",file[k]);
      refresh();
      exit(1);
    }

/* Read null values */

    found_null = find_null(k,strdat);

    if(!found_null) {
      printw("Null position not found in file %s\n",file[k]);
      printw("Null position = (%f, %f, %f)\n",nullpos[0],
             nullpos[1],nullpos[2]);
      refresh();
      exit(-1);
    }
    else {
      printw("Found null in file %d\n",k);
      refresh();
    }
    rewind(strdat);

    eoftst = fscanf(strdat,"%d %d",&data[0][0],&data[0][1]);

    while(eoftst != EOF) {
      for(j=0;j<3;j++) {
        fscanf(strdat,"%d %d",&data[j+1][0],&data[j+1][1]);
      }

      for(j=0;j<naxes[k];j++) {
        fscanf(strdat,"%f",&x[j]);
      }

/* Discard data points we don't want */

      if(standard[k] == 11954) {
        if(fabs(x[0] - 2.50) < 0.005) {
          use = 0;
        }
        else if(fabs(x[0] - 3.50) < 0.005) {
          use = 0;
        }
        else if(x[0] < 2.5) {
          use = 0;
        }
        else if(x[0] > 4.5) {
          use = 0;
        }
        else {
```

```
        use = 1;
      }

    }
    else {
      use = 1;
    }

    if(use) {
      i++;
      input[i][0] = 1.0;
      for(j=0;j<4;j++) {
        datr = ((data[j][0] - nullval[j][0]) * cos(phase[j])
                - (data[j][1] - nullval[j][1]) * sin(phase[j]))
          /normfac[j];
        dati = ((data[j][1] - nullval[j][1]) * cos(phase[j])
                + (data[j][0] - nullval[j][0]) * sin(phase[j]))
          /normfac[j];

        rdatr = datr*cos(rotation[j]) - dati*sin(rotation[j]);
        rdati = dati*cos(rotation[j]) + datr*sin(rotation[j]);

        if(j == 0) {
          rdatr = 0.;
          rdati = 0.;
        }

        if(j<4) {
          rdatr = rdati*rdati;
        }

        input[i][3*j+1] = rdatr;
        input[i][3*j+2] = rdati;
        input[i][3*j+3] = rdati*rdati*rdati;
/*
        if(rdatr > 1.) {
          input[i][4*j+3] = rdatr;
        }
        else {
          input[i][4*j+3] = rdatr;
        }
        if(rdati > 1.) {
          input[i][4*j+4] = rdati;
        }
        else {
          input[i][4*j+4] = rdati;
        }
*/
        fprintf(strout,"%f %f ",datr,dati);
      }

      if(standard[k] == 11956) {
        actual_depth[i] = set_depth_z11956(x);
      }
      else if(standard[k] == 11954) {
        actual_depth[i] = set_depth_z11954(x);
      }
      else {
        printf("Unknown standard.\n");
        exit(-1);
      }
      fprintf(strout,"%f\n",actual_depth[i]);

      desired_value[i][0] = 0.;
      desired_value[i][1] = 0.006 * actual_depth[i] + 0.2;
    }
    eoftst = fscanf(strdat,"%d %d",&data[0][0],&data[0][1]);
```

```
    }
    fclose(strdat);
  }

  fclose(strout);

  npt = i;
  printw("npt = %d\n",npt);
  refresh();

lbl1:
  sserr = 0.;
  hidden[0] = 1.0;

   for(l=0; l<1000000; l++) {
      pattern = (int)(npt/32767. * rand());
      if(pattern >= npt) {
         pattern = npt - 1;
      }

      p0_input = &input[pattern][0];

      for(j=1; j<nhidden; j++) {
         p1_input = p0_input;
         sum = 0.;
         for(k=0; k<ninput; k++) {
            sum += *(p1_input) * w0[j][k];
            p1_input++;
         }
         hidden[j] = 1.0/(1.0 + exp(-sum));
      }

      for(i=1; i<noutput; i++) {
         sum = 0.;
         for(j=0; j<nhidden; j++) {
            sum += hidden[j] * w1[i][j];
         }
         output[i] = 1.0/(1.0 + exp(-sum));
         output_error[i] = output[i] * (1.0 - output[i])
                           * (desired_value[pattern][i] - output[i]);
      }

      for(j=0; j<nhidden; j++) {
         hidden_error[j] = 0.;
         for(i=1; i<noutput; i++) {
            hidden_error[j] += output_error[i] * w1[i][j];
         }
         hidden_error[j] *= hidden[j] * (1.0 - hidden[j]);
      }

      for(j=1; j<nhidden; j++) {
         p1_input = p0_input;
         for(k=0; k<ninput; k++) {
            w0[j][k] += *(p1_input) * eta0 * hidden_error[j]
                        + alpha * d0[j][k];
            d0[j][k] = w0[j][k] - last_w0[j][k];
            last_w0[j][k] = w0[j][k];
            p1_input++;
         }
      }

      for(i=1; i<noutput; i++) {
         for(j=0; j<nhidden; j++) {
            w1[i][j] += hidden[j] * eta0 * output_error[i]
                        + alpha * d1[i][j];
            d1[i][j] = w1[i][j] - last_w1[i][j];
            last_w1[i][j] = w1[i][j];
         }
```

```
}

caldep = (output[1] - 0.2)/0.006;

    sserr += (caldep - actual_depth[pattern])
          * (caldep - actual_depth[pattern]);
}
rmserr = sqrt(sserr/1000000.);
nn += 1;
if((nn % print_period) == 0) {
    printw("%ld %8.5f\n", nn, rmserr);
    refresh();
    if(calculate_error) {
        rmserr = 0.;
        for(pattern = 0; pattern<npt; pattern++) {
            for(j=1; j<nhidden; j++) {
                sum = 0.;
                for(k=0; k<ninput; k++) {
                    sum += input[pattern][k] * w0[j][k];
                }
                hidden[j] = 1.0/(1.0 + exp(-sum));
            }
            for(i=1; i<noutput; i++) {
                sum = 0.;
                for(j=0; j<nhidden; j++) {
                    sum += hidden[j] * w1[i][j];
                }
                output[i] = 1.0/(1.0 + exp(-sum));
            }

            rmserr += (desired_value[pattern][1] - output[1])
                   * (desired_value[pattern][1] - output[1]);
            }

        rmserr = sqrt(rmserr/npt)/0.006;
        if(rmserr < lowest_error) {
           lowest_error = rmserr;
           lowest = 1;
           printw("True rms error = %f**\n", rmserr);
        }
        else {
           lowest = 0;
           printw("True rms error = %f, best = %f\n"
                , rmserr, lowest_error);
        }
        if(undo_bad_step) {
        if(rmserr < prev_epoch_err) {
           prev_epoch_err = rmserr;
           for(i=1; i<noutput; i++) {
              for(j=0; j<nhidden; j++) {
                 prev_epoch_w1[i][j] = w1[i][j];
                 prev_epoch_d1[i][j] = d1[i][j];
              }
           }
           for(j=1; j<nhidden; j++) {
              for(k=0; k<ninput; k++) {
                 prev_epoch_w0[j][k] = w0[j][k];
                 prev_epoch_d0[j][k] = d0[j][k];
              }
           }
        }
        else {
           printw("Last epoch changes reversed.\n");
           for(i=1; i<noutput; i++) {
              for(j=0; j<nhidden; j++) {
                 w1[i][j] = prev_epoch_w1[i][j];
                 last_w1[i][j] = w1[i][j];
                 d1[i][j] = prev_epoch_d1[i][j];
```

```
                    }
                  }
                  for(j=1; j<nhidden; j++) {
                    for(k=0; k<ninput; k++) {
                      w0[j][k] = prev_epoch_w0[j][k];
                      last_w0[j][k] = w0[j][k];
                      d0[j][k] = prev_epoch_d0[j][k];
/*printw("j k w0 %d %d %f", j, k, w0[j][k]);*/
                    }
                  }
                }
                }
                refresh();
              }
            }
  ch = getch();
  if((ch == 's') || (lowest)) {
    if(ch == 's') {
      nodelay(stdscr, FALSE);
      printw("\nEnter filename  ");
      refresh();
      scanw("%s", &wtsout[0]);
    }
    else {
      strcpy(&wtsout[0], "vbcbest.wts");
    }
    if((strout = fopen(&wtsout[0], "w")) == NULL) {
      printw("Error:  failure to open output data file.\n");
      refresh();
      exit(1);
    }
    fprintf(strout, "%ld\n", nn);
    fprintf(strout, "3 ");
    fprintf(strout, "%d %d %d\n", ninput, nhidden, noutput);
    fprintf(strout, "%f %f\n", alpha, eta0);
    for(i=0; i<ninput; i++) {
      for(j=1; j<nhidden; j++) {
        fprintf(strout, "%f ", w0[j][i]);
      }
      fprintf(strout, "\n");
    }
    for(i=0; i<nhidden; i++) {
      for(j=1; j<noutput; j++) {
        fprintf(strout, "%f ", w1[j][i]);
      }
      fprintf(strout, "\n");
    }
    fclose(strout);
    nodelay(stdscr, TRUE);
    goto lbl1;
  }
  else if(ch == 'j') {
    nodelay(stdscr, FALSE);
    printw("\nEnter amount to jog weights  ");
    refresh();
    scanw("%f", &jog_amt);

    for(i=1; i<noutput; i++) {
      for(j=0; j<nhidden; j++) {
        w1[i][j] += (jog_amt * (rand() - 16384))/16384.;
        d1[i][j] = 0.;
        last_w1[i][j] = w1[i][j];
      }
    }

    for(j=1; j<nhidden; j++) {
      for(k=0; k<ninput; k++) {
        w0[j][k] += (jog_amt * (rand() - 16384))/16384.;
```

```
      d0[j][k] = 0.;
      last_w0[j][k] = w0[j][k];
    }
  }

  nodelay(stdscr, TRUE);
  goto lbl1;
}
else if(ch == 'q') {
  goto lbl2;
}
else if(ch == 'l') {
  nodelay(stdscr, FALSE);
  printw("\nCurrent learning rate = %f\n", eta0);
  printw("Enter learning rate  ");
  refresh();
  scanw("%lf", &eta0);

  for(i=1; i<noutput; i++) {
    for(j=0; j<nhidden; j++) {
      d1[i][j] = 0.;
    }
  }

  for(j=1; j<nhidden; j++) {
    for(k=0; k<ninput; k++) {
      d0[j][k] = 0.;
    }
  }

  nodelay(stdscr, TRUE);
  goto lbl1;
}
else if(ch == 'm') {
  nodelay(stdscr, FALSE);
  printw("\nCurrent momentum = %f\n", alpha);
  printw("Enter momentum  ");
  refresh();
  scanw("%lf", &alpha);
  nodelay(stdscr, TRUE);
  goto lbl1;
}
else if(ch == 'c') {
  printw("\nConcise mode\n");
  refresh();
  print_period = 5;
  goto lbl1;
}
else if(ch == 'v') {
  printw("\nVerbose mode\n");
  refresh();
  print_period = 1;
  goto lbl1;
}
else if(ch == 'e') {
  calculate_error++;
  calculate_error = calculate_error % 2;
  printw("\n");
  goto lbl1;
}
else if(ch == 'u') {
  undo_bad_step++;
  undo_bad_step = undo_bad_step %2;
  printw("\n");
  goto lbl1;
}
else if(ch == 'p') {
  strout = fopen("bpntmp.dat","w");
```

```
    if(strout == NULL) {
      printw("Unable to open output data file.\n");
      refresh();
      goto lbl1;
    }
    for(k=0;k<nfiles;k++) {
      strdat = fopen(file[k],"r");
      if(strdat == NULL) {
        printw("Unable to open input data file %s\n",file[k]);
        refresh();
        goto lbl1;
      }
      fclose(strdat);
    }
    fclose(strout);
  }
  else {
    goto lbl1;
  }

 lbl2:
  endwin();
}

float set_depth_z11956(float *xp)
{
/* Set actual depths for standard Z-11956 */
  float depth;
  float x,y,z;

  x = xp[0];
  y = xp[1];
  z = xp[2];

  if(fabs(x - 3.5) < 0.15) {
    if(fabs(x - 3.5) < 0.10) {
      depth  = 20.;
    }
    else {
      depth = (0.15 - fabs(3.5 - x)) * 20./0.05;
    }
  }
  else if(fabs(x - 4.25) < 0.15) {
    if(fabs(x - 4.25) < 0.10) {
      depth = 40.;
    }
    else {
      depth = (0.15 - fabs(4.25 - x)) * 40./0.05;
    }
  }
  else if(fabs(x - 5.00) < 0.15) {
    if(fabs(x - 5.00) < 0.10) {
      depth = 60.;
    }
    else {
      depth = (0.15 - fabs(5.00 - x)) * 60./0.05;
    }
  }
  else if(fabs(x - 5.75) < 0.15) {
    if(fabs(x - 5.75) < 0.10) {
      depth = 80.;
    }
    else {
      depth = (0.15 - fabs(5.75 - x)) * 80./0.05;
    }
  }
  else {
    depth = 0.;
```

```
  }

  return depth;
}

float set_depth_z11954(float *xp)
{
/* Set actual depths for standard Z-11954 */
  float depth;
  float x,y,z;
  float dist;
  float xdist, zdist;

  x = xp[0];
  y = xp[1];
  z = xp[2];

  if(z < 0.) {
    zdist = 0.;
  }
  else {
    zdist = z * 0.4375 * 3.1415927/180.;
  }

  if(fabs(x - 2.00) < 0.15) {
    xdist = fabs(x - 2.00);
    dist = sqrt(xdist*xdist + zdist*zdist);
    if(dist < 0.10) {
      depth = 20.;
    }
    else if(dist < 0.15) {
      depth = (0.15 - dist) * 20./0.05;
    }
    else {
      depth = 0.;
    }
  }
  else if(fabs(x - 3.00) < 0.15) {
    xdist = fabs(x - 3.00);
    dist = sqrt(xdist*xdist + zdist*zdist);
    if(dist < 0.10) {
      depth = 40.;
    }
    else if(dist < 0.15) {
      depth = (0.15 - dist) * 40./0.05;
    }
    else {
      depth = 0.;
    }
  }
  else if(fabs(x - 4.00) < 0.15) {
    xdist = fabs(x - 4.00);
    dist = sqrt(xdist*xdist + zdist*zdist);
    if(dist < 0.10) {
      depth = 60.;
    }
    else if(dist < 0.15) {
      depth = (0.15 - dist) * 60./0.05;
    }
    else {
      depth = 0.;
    }
  }
  else if(fabs(x - 5.00) < 0.15) {
    xdist = fabs(x - 5.00);
    dist = sqrt(xdist*xdist + zdist*zdist);
    if(dist < 0.10) {
      depth = 80.;
```

```
    }
    else if(dist < 0.15) {
      depth = (0.15 - dist) * 80./0.05;
    }
    else {
      depth = 0.;
    }
  }
  else if(fabs(x - 6.00) < 0.15) {
    xdist = fabs(x - 6.00);
    dist = sqrt(xdist*xdist + zdist*zdist);
    if(dist < 0.10) {
      depth = 100.;
    }
    else if(dist < 0.15) {
      depth = (0.15 - dist) * 100./0.05;
    }
    else {
      depth = 0.;
    }
  }
  else {
    depth = 0.;
  }

  return depth;
}

int find_null(int k,FILE *strdat)
{
  int data[4][2];
  float x[3];
  int found_null = 0;
  int is_null_pos;
  int eoftst;
  int j;

  eoftst = fscanf(strdat,"%d %d",&data[0][0],&data[0][1]);

  while((eoftst != EOF) && (!found_null)) {
    for(j=0;j<3;j++) {
      fscanf(strdat,"%d %d",&data[j+1][0],&data[j+1][1]);
    }
    for(j=0;j<naxes[k];j++) {
      fscanf(strdat,"%f",&x[j]);
    }
    is_null_pos = 0;
    for(j=0;j<naxes[k];j++) {
      if(fabs(nullpos[k][j] - x[j]) < 0.005) {
        is_null_pos++;
      }
    }

    if(is_null_pos == naxes[k]) {
      found_null = 1;
      for(j=0;j<4;j++) {
        nullval[j][0] = data[j][0];
        nullval[j][1] = data[j][1];
      }
    }
    eoftst = fscanf(strdat,"%d %d",&data[0][0],&data[0][1]);
  }
  return found_null;
}
```

INTERNAL DISTRIBUTION

1.	W. R. Corwin	19.	C. E. Pugh
2.	L. D. Chitwood	20.	W. A. Simpson
3.	D. F. Craig	21-22.	M&C Records Office
4.	S. A. David	23.	ORNL Patent Section
5-9.	C. V. Dodd	24.	Central Research Library
10.	H. W. Hayden, Jr.	25.	Document Reference Section
11-16.	D. J. McGuire	26-27.	Laboratory Records Department
17.	R. K. Nanstad	28.	Laboratory Records (RC)
18.	J. R. Pate	29.	Nuclear Safety Information Center

EXTERNAL DISTRIBUTION

30-31. NRC, OFFICE OF NUCLEAR REGULATORY RESEARCH, Washington, DC 20555

J. Craig
M. E. Mayfield
J. Muscara
C. Z. Serpan

32.-33. ELECTRIC POWER RESEARCH INSTITUTE, P.O. Box 217097, Charlotte, NC 28221

M. Elmo
K. Kryzwosz

34. BATTELLE PACIFIC NORTHWEST LABORATORIES, P.O. Box 999, Richland, WA 99352

B. Ferris

35. Steven D. Brown, 23 Greensburg Street, Delmont Borough, PA 15626

36. DOE, OAK RIDGE OPERATIONS OFFICE, Oak Ridge, TN 37831-8600

Office of Assistant Manager for Energy Research and Development

37. DOE, OFFICE OF SCIENTIFIC AND TECHNICAL INFORMATION, P. O. Box 62, Oak Ridge, TN 37831

38-286. Given distribution as shown in category RF (NTIS-10)

NRC FORM 335 (2-89) NRCM 1102, 3201, 3202

U.S. NUCLEAR REGULATORY COMMISSION

BIBLIOGRAPHIC DATA SHEET

(See instructions on the reverse)

1. REPORT NUMBER (Assigned by NRC, Add Vol., Supp., Rev., and Addendum Numbers, if any.)

NUREG/CR-6357
ORNL/TM-13213

2. TITLE AND SUBTITLE

Evaluation and Field Validation of Eddy-Current Array Probes for Steam Generator Tube Inspection

3. DATE REPORT PUBLISHED

MONTH: July | YEAR: 1996

4. FIN OR GRANT NUMBER

B0417

5. AUTHOR(S)

C. V. Dodd and J. R. Pate

6. TYPE OF REPORT

Technical

7. PERIOD COVERED *(Inclusive Dates)*

8. PERFORMING ORGANIZATION – NAME AND ADDRESS *(If NRC, provide Division, Office or Region, U.S. Nuclear Regulatory Commission, and mailing address; if contractor, provide name and mailing address.)*

Oak Ridge National Laboratory
Oak Ridge, TN 37831-6285

9. SPONSORING ORGANIZATION – NAME AND ADDRESS *(If NRC, type "Same as above"; if contractor, provide NRC Division, Office or Region, U.S. Nuclear Regulatory Commission, and mailing address.)*

Division of Engineering Technology
Office of Nuclear Regulatory Research
U.S. Nuclear Regulatory Commission
Washington, DC 20555-0001

10. SUPPLEMENTARY NOTES

J. Muscara, NRC Project Manager

11. ABSTRACT *(200 words or less)*

The objective of the Improved Eddy-Current ISI for Steam Generator Tubing program is to upgrade and validate eddy-current inspections, including probes, instrumentation, and data processing techniques for inservice inspection of new, used, and repaired steam generator tubes; to improve defect detection, classification and characterization as affected by diameter and thickness variations, denting, probe wobble, tube sheet, tube supports, copper and sludge deposits, even when defect types and other variables occur in combination; to transfer this advanced technology to NRC's mobile NDE laboratory and staff. This report descripes the design of specialized high-speed 16-coil eddy-current array probes. Both pancake and relection coils are considered. Test results from inspections using the probes in working steam generators are given. Computer programs developed for probe calculations are also supplied.

12. KEY WORDS/DESCRIPTORS *(List words or phrases that will assist researchers in locating the report.)*

Array Probe
Computer Programs
Neural Networks
Steam Generators
Tubing Inspection

13. AVAILABILITY STATEMENT

Unlimited

14. SECURITY CLASSIFICATION

(This Page)
Unclassified

(This Report)
Unclassified

15. NUMBER OF PAGES

16. PRICE