AD NUMBER

ADB013005

LIMITATION CHANGES

TO:

Approved for public release; distribution is unlimited.

FROM:

Distribution authorized to U.S. Gov't. agencies only; Test and Evaluation; DEC 1975. Other requests shall be referred to Air Force Avionics Laboratory, AFAL/WRP, Wright-Patterson AFB, OH 45433.

AUTHORITY

afal ltr, 23 mar 1979

AFAL-TR-75-219

ADVANCED RADAR REFLECTOR STUDIES

The ElectroScience Laboratory
The Ohio State University

December 1975

TECHNICAL REPORT AFAL-TR-75-219

AIR FORCE AVIONICS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This technical report has been reviewed and is approved for publication.

Vittal P. Pyati

VITTAL P. PYATI, Ph.D.
Program Monitor

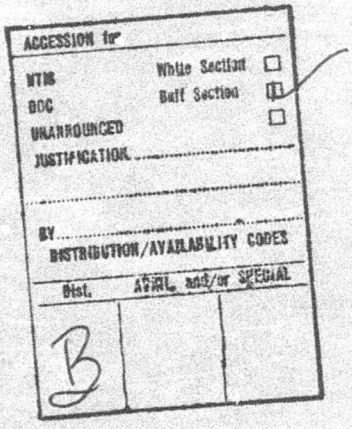
William F. Bahret

WILLIAM F. BAHRET
Actg Chief, Passive ECM Branch
Electronic Warfare Division

FOR THE DIRECTOR

Ollie H. Edwards

OLLIE H. EDWARDS, Colonel, USAF
Chief, Electronic Warfare Division



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

(19) REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: (18) AFAL-TR-75-219
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER

4. TITLE *(and Subtitle)*: (6) ADVANCED RADAR REFLECTOR STUDIES.

5. TYPE OF REPORT & PERIOD COVERED: Final Summary Report 4/12/72-6/30/75

6. PERFORMING ORG. REPORT NUMBER: (14) ESL-3401-3

7. AUTHOR(s): (10) R. J. Garbacz, V. Cable, R. Wickliff, R. Caldecott, J. Buk, D. Lam, K. Demarest, A. Yee

8. CONTRACT OR GRANT NUMBER(s): F33615-72-C-1435 (15)

9. PERFORMING ORGANIZATION NAME AND ADDRESS: The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering, Columbus, Ohio 43212

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 62204F; (17) 76331383; (16) AF-7633

11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force Avionics Laboratory, Air Force Wright Aeronautical Laboratories, Air Force Systems Command, WPAFB, Ohio 45433

12. REPORT DATE: (11) Dec [redacted] 75

13. NUMBER OF PAGES: 345

14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*: (12) 368 P.

15. SECURITY CLASS. *(of this report)*: Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Distribution limited to U.S. Government agencies only (test and evaluation); December 1975. Other requests for this document must be referred to Air Force Avionics Laboratory (AFAL/WRP), WPAFB, Ohio 45433.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

(9) Final summary rept. 12 Apr 72 - 30 Jun 75,

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Chaff
Random arrays
Dipoles
Scattering
Extinction
Sparse matrix
Iteration

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

The purpose of this effort was to develop user-oriented computer programs for the investigation of the electromagnetic scattering and extinction characteristics of clouds of linear metallic resonant wires for chaff applications, including the effects of mutual coupling between wires. Programs based upon Crout-like algorithms working on matrices generated by the method of moments were developed to accommodate up to 200 wires; sparse matrix and iterative algorithms were

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED 402 251

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

20.

programmed for larger numbers of wires. All programs are documented in this report.

Data generated during the course of the effort are also presented in this report, including curves of the reduction (due to coupling) in average scattering cross section as a function of number density of chaff elements.

FOREWORD

This report was prepared and submitted August, 1975 by The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering, 2015 Neil Avenue, Columbus, Ohio 43210 under Contract F33615-72-C-1435, Project No. 62204F, Task No. 76331333, extending from April 1972 to June 1975. Dr. V. P. Pyati, AFAL/WRP, was the Program Monitor.

The authors of this report wish to acknowledge the many conversations they had with the immediate sponsors, Dr. V. Pyati, Mr. R. Puskar, and Ms. M. Gauvey, during the entire course of the contract. They also express thanks to Professor J. Richmond who showed a lively interest in the work and who helped in many of its phases.

REPORTS AND PUBLICATIONS UNDER CONTRACT F 33615-72-C-1435

3401-1 Annual Summary Report, June 1974

3401-2 "Application of Linear Iteration to Electromagnetic Scattering by Random Arrays of Wires," V. P. Cable, August 1975

3401-3 Final Summary Report, "Advanced Radar Reflector Studies," R.J. Garbacz, V. Cable, R. Wickliff, R. Caldecott, J. Buk, D. Lam, K. Demarest, A. Yee

Wickliff, R.G. and Garbacz, R.J., "The Average Backscattering Cross Section of Clouds of Randomized Resonant Dipoles," IEEE Trans. on Antennas and Propagation, Vol. AP-22, No. 3, pp. 503-505.

Cable, V.P., "Application of Linear Iteration to Electromagnetic Scattering by Random Arrays of Wires," Ph.D. Dissertation, The Ohio State University, August 1975.

LIST OF ILLUSTRATIONS

LIST OF TABLES

LIST OF SYMBOLS

N	- number of dipoles in a cloud, Gaussianly distributed in the radial direction
δ	- standard deviation of the Gaussian dipole distribution
d/λ	- average spacing between dipoles
M	- number of clouds in an ensemble of clouds
λ	- wavelength
β	- bistatic angle
$\sigma_m(\theta)$ or $\sigma(\theta)$	- backscattering cross section of mth cloud as a function of angle θ around great circle cut
$\langle\sigma_m\rangle$ or $\langle\sigma\rangle$	- spatial average or mean value of $\sigma_m(\theta)$ over 360° of the angle θ
$\overline{\langle\sigma_m\rangle}$ or $\overline{\langle\sigma\rangle}$	- ensemble average of $\langle\sigma_m\rangle$ values over $1 \leq m \leq M$ clouds forming an ensemble
$\sigma_{m\ 1/5}$ or $\sigma_{1/5}$	- first quintile of σ_m
$\sigma_{m\ 1/2}$ or $\sigma_{1/2}$	- median of σ_m
$\sigma_{m\ 4/5}$ or $\sigma_{4/5}$	- fourth quintile of σ_m
$\bar{\sigma}_{1/5}$	- ensemble average of $\sigma_{m\ 1/5}$
$\bar{\sigma}_{1/2}$	- ensemble average of $\sigma_{m\ 1/2}$
$\bar{\sigma}_{4/5}$	- ensemble average of $\sigma_{4/5}$
$p_m(\sigma_m)$ or $p(\sigma)$	- probability density function of σ_m
$P_m(\sigma_m)$ or $P(\sigma)$	- cumulative probability function of σ_m
s_m	- standard deviation of σ_m
$q_{mean}(\langle\sigma\rangle)$	- (Gaussian) probability density function of the mean, $\langle\sigma\rangle$
s_{mean}	- standard deviation of $\langle\sigma\rangle$
$q_{1/5}(\sigma_{1/5})$	- (Gaussian) probability density function of the first quintile $\sigma_{1/5}$

$s_{1/5}$ - standard deviation of $\sigma_{1/5}$

$q_{1/2}(\sigma_{1/2})$ - (Gaussian) probability density function of the median $\sigma_{1/2}$

$s_{1/2}$ - standard deviation of $\sigma_{1/2}$

$q_{4/5}(\sigma_{4/5})$ - (Gaussian) probability density function of the fourth quintile $\sigma_{4/5}$

$s_{4/5}$ - standard deviation of $\sigma_{1/2}$

$F_m(\omega)$ - spatial frequency spectrum of σ_m

W_m - highest spatial frequency of $F_m(\omega)$

W - average of W_m for all M clouds in an ensemble

W' - average highest spatial frequency as calculated using a two-dipole interference model

$Z \equiv A$ - impedance matrix of cloud

$I \equiv x$ - vector of dipole currents

V, b - plane wave excitation voltage vector

H - iteration or error reducing matrix

H_J - iteration or error reducing matrix for Jacobi

H_{GS} - iteration or error reducing matrix for Gauss-Seidel

H_ω - iteration or error reducing matrix for successive overrelaxation

ω - relaxation factor

$\varepsilon(k)$ - convergence norm

Other symbols are defined in context as necessary.

I. INTRODUCTION

Since World War II, chaff, which is a code name for a collection of thousands of linear resonant dipoles, has been used as an effective passive ECM against pertinent threat radar systems. One generally recognizes at least two significant roles for chaff; first, self-protection as in the case of aircraft against fire control radars, and second, in situations where initially sown dipole corridors saturate radar receivers and the corridors are subsequently utilized as penetration aids. Heretofore, the echoing area or the radar cross section of a chaff cloud has been calculated by multiplying the number of dipoles by the so-called "tumble average radar cross-section" of a single dipole. Estimates based on this simple model have been poor. Experimental measurements are between 2-50% of the theoretical value, depending upon the situation. Furthermore, once certain dipole densities have been reached doubling or even quadrupling the number of dipoles show very little increase in echo area. The significance of these discrepancies is that the simple tumble average model is not satisfactory and it is high time one undertakes a more realistic study of the electromagnetic scattering and attenuation properties of chaff clouds. To fulfill the requirements, the ElectroScience Laboratory under sponsorship of the Air Force Avionics Laboratory has undertaken a comprehensive study of the electromagnetic behavior of chaff clouds. The effort has been conveniently divided into three phases of increasing complexity. These are

1. Scattering behavior of single length, i.e., one frequency, dipoles with moderate mutual coupling between the elements.

2. Same as above but with close coupling, even touching.

3. Clouds of different dipole lengths, i.e., multiple frequency clouds

The work performed under this contract emphasized (1), with some effort devoted to (2) and (3).

The scattering and extinction behavior of large ensembles of particles has long been a subject of study in such diverse disciplines as acoustics, quantum mechanics and electromagnetics [1]. Most work is based upon certain assumptions which make the problem analytically tractable, such as very small particle size, large spacing, no coupling or forward-neighbor coupling only, etc. In their domains of validity, mathematical models based on such

assumptions have indeed been useful in treating particulate media. In the case of a chaff cloud, however, two features complicate the problem: the particles are linear dipoles of lengths $\lambda/2$ (resonant) or greater and therefore cannot be considered small; and during the early history of the cloud, before it fully blooms, these dipoles are closely spaced and strongly coupled electromagnetically. Furthermore, blooming implies non-stationary cloud statistics, and packaging configuration, dispensing technique and atmospheric conditions all influence the electromagnetic behavior of the cloud in time. These and many other problems face the investigator who wishes to answer such questions as, "How many dipoles is optimum for a cloud in a given tactical situation." "Is there a particular shape or density or density distribution of a cloud that is preferred?" "What are the expected scintillation rates?" "Can one make a cloud bloom faster electromagnetically?" These questions cannot be answered until we understand how a medium composed of many strongly resonant scatterers, which may be closely coupled, interacts with a radar wave, that is, until we can answer the basic question, "How does a chaff cloud scatter?"

Many attempts have been made in the past to answer the above question, usually to obtain the spatial average backscatter at resonance for a cloud of dipoles "frozen" in time [2,3]. Extensions were made to include nonresonant dipoles and dipoles with preferred orientations [4] as well as the dynamics of the dipoles [5,6,7,8]. In all instances, however, the effects of coupling among elements were not included in the analysis due to ensuing computational difficulties. Only recently has it become possible to account for coupling, at least on a limited basis, by use of large digital computer techniques [9,10,11]. Although we shall never be able (or ever wish) to account for all interactions among the millions of dipoles in a typical chaff cloud, the present capability of handling 250 resonant dipoles gives hope of accounting for sufficient numbers of interactions to obtain an accurate statistical description of the behavior of any cloud.

The purpose of our work was to bring the computer to bear on the chaff cloud problem in order to investigate the limits of simplifying approximations, to support, refine, or replace simple models, to obtain and interpret statistical data, and, basically, to better understand the scattering mechanism. This final report describes results developed over the three year time span of the contract. Because the effort extended over such a long period, many of the earlier methods for generating scattering data were superceded by improved methods, but the results still remain valid and valuable for the inferences that can be made from them. Thus, many of these early results, reported in Reference 12, are presented here as well to provide a complete and integrated overview of the effort.

The main chapter of this report, entitled, Technical Discussion and Results, is divided into several sections. In Section A we discuss the concept of a frozen cloud as a useful chaff model in the absence of realistic time varying data; in Section B (and Appendix A) are discussed the statistical quantities we have used to describe the radar cross section of a chaff cloud. Section C is a lengthy one which itself is divided into several parts: Introductory Remarks, which is intended to provide a very brief and general discussion of the method of moments (more details appear in Appendix B) by which the integral equation describing the electromagnetic chaff interaction problem is reduced to a set of simultaneous algebraic (matrix) equations suitable for processing by digital computer; Direct Methods, which describes the most commonly applied techniques for solving the above-mentioned matrix equations, such as the method of Crout; Sparse Matrix Methods, which describes special algorithms which are useful if the matrix is large and is sparse, i.e., has many zeros in it; i.e., weak coupling between chaff elements, and Indirect, or Iterative, Methods, which appear to be useful for large matrices, i.e., large numbers of chaff elements, without the assumption of sparsity. Typical results, as derived by each method, are presented in appropriate sections, together with a discussion and conclusions inferred from those results. In some instances verifying experimental data are also given to support the computations. Computer programs used to generate the results, either by the direct, sparse or iterative methods, are documented in Appendices D, E and F, respectively.

The primary emphasis during the contract was the investigation of clouds of resonant (half-wave) dipoles which were not "too closely" spaced. Some effort was expended to better define what "too closely" means in terms of the computer models used in our work, and this is discussed in Section D of Chapter II. Section E is addressed to chaff clouds containing multi-length elements for purposes of broadbanding the chaff echo to meet threats over a range of frequencies. Section F is devoted to experimental results. Although the bulk effort was primarily computational, some experimental data were recorded to verify the computed results and to observe certain scattering and extinction behaviors of moving dipoles in numbers much greater than can be handled by computer ($\sim$8000). These and other experimental efforts are reported in this section.

Section G of Chapter II is on a topic somewhat divorced from that of chaff cloud scattering characteristics. In it we present an initial effort to investigate the aircraft-chaff cloud-tracking missile intercept problem. Many of the parameters of this problem are unknown, such as location and motion of scattering centers from a particular aircraft as a function of its maneuvers, the precise aerodynamic and electromagnetic behavior of chaff clouds spawned by the aircraft, and the range and tracking behavior of

the missile radars under such complex returns. Although these quantities were assumed in this study, it is anticipated that the approaches suggested here will become very useful for computerized simulation studies when more accurate input data become available through diverse research programs. More detail is given in Appendix G.

Chapter III concludes the body of the report with an overall discussion of our findings and suggestions for future effort.

Six appendixes were already alluded to. One additional appendix (C) describes the Gaussianly distributed density of dipoles employed throughout most of the contract. In the late stages of our work uniformly dense clouds were preferred and their generation is briefly described as well.

II. TECHNICAL DISCUSSION AND RESULTS

A. The Frozen Chaff Cloud Model

It is appropriate to discuss the first fundamental assumption upon which all our work, be it by computer or by laboratory experiment, rests. This is the assumption of the "frozen" chaff cloud model.

Scattering by a real chaff cloud is a stochastic process in the independent variable, time. At any given instant not only do we find the dipoles randomly positioned and oriented, but over a short interval of time they move and give rise to random fluctuations in the cross section (be it monostatic, bistatic, or foreward). Moreover, with the passage of time, the cloud evolves from a dense to a tenuous conglomerate of dipoles so that, viewed over a long interval, the stochastic scattering process appears nonstationary, i.e., its statistics change with time.

In order to approximate the lower order statistics associated with a certain instant of time, one might consider an ensemble of similarly evolving clouds and take averages over this ensemble at the time of interest. This viewpoint leads us to the so called ensemble model, in which time is stopped at regular intervals, a "snapshot" taken of each cloud in the ensemble of clouds, and the ensemble average of backscatter calculated for each time sample. As time progresses and the cloud blooms, we assume the ensemble averages from each successive set of "snapshots" change and faithfully characterize the time average's behavior of a random cloud in evolution.

The generation of a large ensemble of clouds and the computation of ensemble average backscatter, for example, as the clouds evolve in time is an expensive process, especially if the clouds contain many dipoles. Thus there arises the proposition, instead of generating many <u>different</u> clouds (requiring the calculation of mutual impedances among dipoles for each new cloud) to form an ensemble over which to average, can we more efficiently obtain an equivalent ensemble average by viewing the <u>same</u> cloud (requiring the calculation of mutual impedances among dipoles only once) at many <u>different</u> aspects, then <u>spatially</u> averaging the back scattering cross section over all these aspect angles? As will be seen, the answer appears to be a qualified affirmative in that the spatial average backscattering cross sections for similar (i.e., same number of dipoles with same average spacing) but different clouds do differ in general, so that it is not sufficient to spatially average only <u>one</u> cloud return and accept that as a good equivalent ensemble average. One must generate an ensemble of clouds, obtain a spatial average backscattering cross section for each and then obtain an ensemble average of these spatial

averages. The point being that this latter ensemble is smaller than the former, thereby demanding fewer calculations of mutual impedances, etc. with resulting enhanced efficiency of computation (at least for large clouds). In all our work we obtain ensemble averages using this modified ensemble model, which we call the frozen model.

Going one step further in the search for computational efficiency, there arises the proposition, can we illuminate one or a few similar clouds from one aspect (requiring the calculation of induced currents only once for each cloud generated) and average the bistatic scattering cross section over a range of bistatic angles and expect this average to be simply related to the ensemble average of backscattering cross section? Or further, can one relate the average of total scattering cross section to the ensemble average of backscattering cross section? The answer to both these propositions appears to be negative, or at least the relationships are not clear to us from the data we have generated.

B. Representative Cloud Characteristics

In the previous section, we discussed the frozen model of a chaff cloud as a substitute for the more complex time-varying model, under the assumption that the scattering characteristics derived from each model agree. The characteristics which we have in mind are, of course, statistical in nature and should be discussed more fully so that the reader understands the results presented later.

Viewed in time, the monostatic or bistatic echo from a cloud consists of an average return plus a scintillation term. The average is expected to change as the cloud blooms - a symptom of non-stationarity - but if its rate of change is slow with respect to the scintillation rate, the scattering process might be considered stationary over small time intervals. With each such time interval, therefore, are associated a mean value, i.e., the time average radar cross section, a variance, i.e., the mean square of the time-varying component of the radar cross section, and a frequency spectrum of the cross section. The totality of all such sets of quantities taken during selected time intervals constitute a partial statistical description of the cloud behavior.

By assuming a frozen model, appropriate to one of the above-mentioned intervals of time (i.e., with average dipole spacing appropriate to the time interval in the evolution of a blooming cloud), we substitute viewing angle for time as the independent variable and obtain a spatial average radar cross section. As mentioned earlier, it turns out that this spatial average radar cross section differs from cloud-to-cloud, so in the frozen cloud

model we assume an ensemble of clouds and obtain a distribution of spatial average radar cross sections. The ensemble average of this distribution of spatial averages is assumed to be equivalent to the time average radar cross section for the time interval of interest. From this distribution we also obtain a variance of the spatial average, a quantity which has no obvious meaning in the time-averaging process, but is useful for estimating a confidence level for the ensemble average cross section obtained from the frozen model. It may be that the variance of the spatial average is simply related to the variance of the random time process, but at present we have no supporting evidence since no time-varying clouds have been generated.

The frequency spectrum of the frozen model is not expected to equal that of the time-varying cloud; it is useful, however, for estimating the minimum number of aspect angles at which to view the clouds in the frozen model, since a number smaller than this causes obvious aliasing of the spectrum.

A more quantitative discussion of the statistical notions and notation employed in later sections of this report are presented in Appendix A.

C. Computer-Generation of Scattering Data

1. Introductory Remarks

The second fundamental assumption underlying this work is that the generation of volumes of scattering data necessary for a statistical study of frozen models ultimately is more efficient, convenient and inexpensive by means of a computer than by laboratory experiment. Experimental data were considered essential to the contract, but primarily as verification of corresponding computed data. We leave discussion of the experimental aspects to a later section and here elaborate on the computer-generation of scattering data.

The computer-solution of scattering by a cloud of coupled resonant dipoles is based on the reaction matching technique of Richmond [9]. This is a moment method of the Galerkin type, i.e., in which the testing functions and basis functions are identical. It assumes that each dipole is divided into P segments (P = 2 has been found to be satisfactory for the configurations discussed in this report), and a piecewise sinusoidal current of unknown amplitude and phase is assumed to flow on each segment. The coupling (i.e., mutual impedance) between each such segment of current and any other segment (or itself) can be expressed in the form of a reaction integral (i.e., an inner product integral) from which the method takes its name. The significant fact which makes the reaction matching technique particularly attractive is that all

these reaction integrals may be evaluated in closed form, thereby permitting the rapid determination of all the elements of a N x N impedance matrix [Z] (representing all self-and-mutual impedances among the N dipoles in a cloud) whose inversion yields the desired dipole currents (I) induced by a plane wave (E) incident from any angle. This technique is well established and has been used to obtain scattering data for many wire obstacles. A more detailed description of the reaction matching technique is given in Appendix B.

With the assurance that the computer-generated scattering data are within the tolerance of experimental data, we turn our attention to the simulation of chaff clouds by the frozen model. Early in the program the N dipoles in a typical cloud were assumed to be resonant in free space, randomly oriented according to a spherical probability density function (i.e., all orientations equally likely) and randomly located according to a Gaussian radial density with average spacing d/λ between dipoles. This average spacing was obtained by considering 76% of the N dipoles to be located within a sphere of radius 2.05δ, where δ is the standard deviation of the aforementioned Gaussian radial distribution. The volume of this sphere is equated to the volume of a cube which itself is subdivided into 0.76N equal cubes, each of which is size d/λ on an edge and is considered to contain one dipole, yielding $d/\lambda = 3.62\, N^{-1/3}\, \delta/\lambda$. Appendix C contains the details of this inhomogeneous cloud generation.

The aforegoing choice of a cloud tapering from a dense central region to tenuous edge blending with free space seemed logical in the beginning. An actual chaff cloud might be expected to display such an inhomogeneity; furthermore, a uniformly dense cloud, for high densities, might be expected to exhibit a coherent scatter from the abrupt free space-cloud interface as well as an incoherent part. Our choice of a tapered density reduces the coherent part, which is desirable since this part would be dependent upon the exact shape of the cloud, which in the actual case is unknown and changing with time. At the same time, however, the tapered density suffers drawbacks. The parameter which we used to describe the tightness of the dipoles, d/λ, or "average spacing", is an average over a substantial part of the cloud. The average spacings are much smaller than this number near the cloud center and much larger closer to its edge. As the program progressed, it became clear that it would be better to assume clouds with uniform densities so that trends in the various methods, such as the sparse matrix and the iterative, could be correlated with respect to a more uniquely defined average spacing (or density) parameter. The details of the homogeneous cloud generation are contained in Appendix C.

We state here once and for all that, except where noted, all results appearing in this report are based on the Gaussian radial distribution for the cloud. The reader will find uniformly dense clouds assumed only in the section describing indirect methods.

2. Direct Methods

(a) Theoretical Considerations

As discussed above, and in more detail in Appendix II, the electromagnetic scattering problem can be transformed via the method of moments into an N x N matrix equation of the form

$$ZI = V \tag{1}$$

where the right hand vector V is known from the direction, polarization, and strength of the known incident plane wave and the elements of the Z matrix can all be calculated using reaction matching. The problem is to determine the current vector I, each component of which is the current I_n induced on the nth chaff dipole.

A direct solution for nonsingular Z can be expressed in terms of the inverse matrix Z^{-1}; i.e.,

$$I = Z^{-1}V \tag{2}$$

However, the solution process may or may not include actual computation of the inverse. Practical examples of solutions expressible in the form of Eq. (2) are Gaussian elimination and LU decomposition. Both of these methods are based on triangularization of Z; Gaussian elimination yields one solution per triangularization whereas, LU decomposition yields any number of solutions for different right hand side vectors. LU decomposition represents a class of compact methods including the Crout, Doolittle and Choleskey methods [37] which do not require storage of intermediate matrices during triangularization as does Gaussian elimination. Final elements of the triangular form are obtained by accumulation and when done in double precision arithimetic and rounded to single precision before storage, solutions by any of these methods will contain a minimum of roundoff error. Solutions to certain electromagnetic problems require repeated responses to variety of excitations. LU decomposition methods are well suited to this requirement and are probably the most widely used in electromagnetic computations.

Successful decomposition or factorization of a matrix is based on the LU theorem. The theorem is stated as follows: Let Z_k represent the kth principal submatrix of Z, formed by eliminating n-k rows and columns from Z. If

$$\det Z_k \neq 0, \quad k = 1,2,\ldots n-1, \tag{3}$$

then there exist two unique triangular matrices $L = [\ell_{ij}]$ and $U = [u_{ij}]$, with L the unit lower triangular (i.e., ones on the main diagonal and zeros above the diagonal) such that

$$Z = LU \tag{4}$$

and

$$\det Z = \prod_{i}^{n} u_{ii} \quad . \tag{5}$$

The U matrix in this case is the same upper triangular matrix obtained by performing Gaussian elimination and L is related to the sequence of matrices M_k, k=1,2,...,n-1, which accomplished this triangularization. Details of computing elements of L and U are left to Appendix I of Reference 38. Equation (1) can now be restated in factored form as

$$LU\ I = V \tag{6}$$

and the solution is computed by setting

$$UI = I \tag{7}$$

in Eq. (6) and solving the resulting triangular system for I by forward substitution. This solution is then substituted back into Eq. (7) and the final triangular system is solved by backward substitution. These forward and backward substitutions are the only calculations needed for other solutions to the same system with different E (excitation) vectors. The factored form of Z defined by Eqs. (4) and (5) is referred to by Westlake [39] as Doolittle decomposition. The familiar Crout decomposition as described by Westlake performs lower triangularization on Z and

U becomes unit upper triangular. Choleskey's method, or the square-root method, requires Z to be at least symmetric. Factorization in this case leads to the form

$$Z = GG^T \tag{8}$$

(T denotes transpose) with the determinant given by

$$\det Z = \prod_{i=1}^{N} (g_{ii})^2 \quad . \tag{9}$$

Gaussian elimination along with the Crout and Doolittle methods generally gives better results when a column reordering strategy is used on $Z^{(k)}$ to position the element of largest absolute magnitude in the kth row in the pivotal position (diagonal) at the kth step of the triangularization process. Choleskey's method, on the other hand, does not require this repositioning strategy when applied to positive definite matrices. The EM problems treated in this study result in complex symmetric (nonhermitian) matrices and in general this partial positioning process should be included. Experience has shown, however, that for most EM problems solved in this manner, sufficient accuracy is obtained without pivoting in spite of the indefiniteness of the coefficient matrix. Elements along the main diagonal generally are larger in magnitude than the off diagonal elements which no doubt contributes to this characteristic.

The size of a particular computer's fast access memory along with growth of roundoff accumulation are inherent limitations of these methods. The size problem can be overcome to a certain extent. However, unless precision is also improved, roundoff must eventually obscure acceptable solutions. One method for studying conditions which affect solution errors is to compute a relative error bound for the solution algorithm being used. Relative error is expressed in the form

$$\text{Relative Error} = \frac{\|I-\bar{I}\|}{\|I\|} \quad , \tag{10}$$

where I and $\bar{I}$ represent the exact and computed solutions, respectively, to Eq. (1) and $\|\cdot\|$ signifies an appropriate vector norm. Definitions of useful vector and matrix norms are given in Appendix C of Ref. [38].

Error bounds naturally tend to be conservative and are often considered useless for this reason. Nevertheless, bounds considered in proper perspective can yield information otherwise unavailable to the user. Computation of a bound based on the number of unknowns (N), the algorithm, and the precision, may reveal trends which can bring confidence or a note of caution into play and is justified if only to indicate such a trend is possible when pushing the limits of a particular machine's size and accuracy. More discussion of condition numbers and error bounds appears in Ref. [38].

(b) Calculated Results for Chaff Clouds

Using the computer routines based upon the method of Crout and documented in Appendix II of Reference 12, clouds with N = 10, 15, 20, 25, 30, 50, 100, 150, 200 dipoles were considered for average spacings, d/λ = 0.5, 1.0, 1.5, 2.0. Not all combinations of (N, d/λ) were investigated equally intensively since computations for larger N values are time-consuming and certain trends could be discerned without them. Most work concentrated on clouds with $N \leq 30$, and on the backscattering cross section. Figures 1-4 show the average backscattering cross section $\langle\sigma_m\rangle$ of the mth cloud in an ensemble of M = 29 clouds in the frozen model, where $1 \leq m \leq M$. These figures give data for clouds containing up to N = 30 dipoles and average spacings d/λ = 0.5, 1.0, 1.5, 2.0. As expected, the values of $\langle\sigma_m\rangle$ distribute themselves over a range (note that where the density of dots in Figs. 1-4 is high, they are plotted aside one another), so it is appropriate to present an average value of the $\langle\sigma_m\rangle$, which we denote by $\langle\overline{\sigma}\rangle$. This has been done in Figs. 5-8, where $\langle\overline{\sigma}\rangle$ is represented by a point. For the cases, N = 10,30, which were investigated more extensively, the ranges which enclose 95.45% of all the values of $\langle\overline{\sigma_m}\rangle$ can be represented by a vertical line (extending from $\langle\overline{\sigma}\rangle - 2\,S_{mean}$ to $\langle\overline{\sigma}\rangle + 2S_{mean}$), where S_{mean} is the standard deviation of the distribution of $\langle\sigma_m\rangle$. The details of the distributions of $\langle\sigma_m\rangle$ are discussed more fully in Appendix I; here, it suffices to say that these curves give some idea of the expected cross section from a cloud of chaff with coupling as a function of number of dipoles and average dipole spacing (i.e., dipole density). In Figs. 1-8, each straight line represents the ideal case of no coupling, in which case the average cross section of N dipoles is expected to be simply N times $\langle\sigma_0\rangle$, the average cross section of a single resonant dipole.* If the average cross section of a single resonant dipole is defined to be the cross section of that dipole averaged over all possible tumble angles, equally weighted (spherical

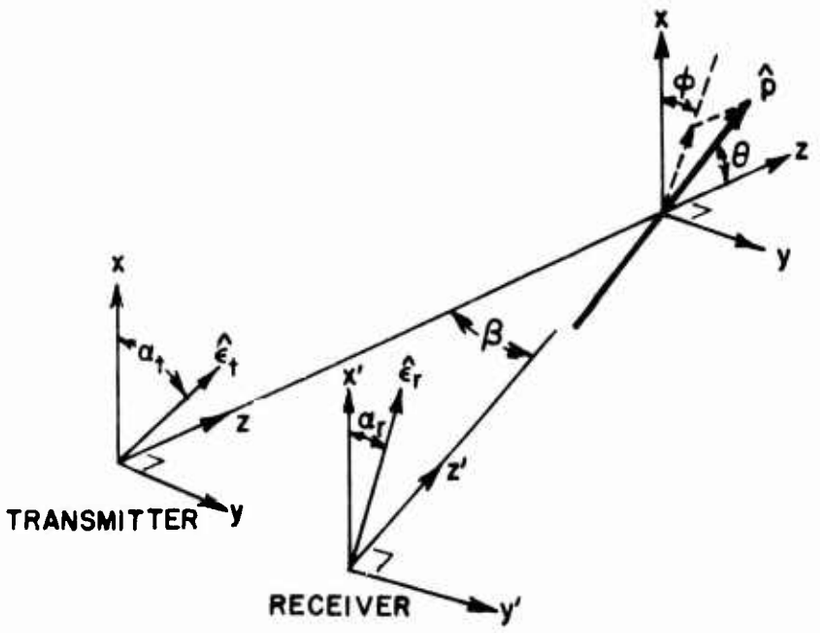
*Actually, this straight line is an approximation strictly valid for uniform density clouds. However, for the non-uniform clouds considered here, it is an extremely good approximation.

probability density function for orientation), then $\langle\sigma_0\rangle$ is equal to about 1/5 times the maximum cross section of the dipole, or $\langle\sigma_0\rangle \approx 0.15\lambda^2$. From these curves it is evident that with an average spacing of $d/\lambda = 2.0$, the curve $N\langle\sigma_0\rangle$ fairly well predicts the values of $\langle\overline{\sigma}\rangle$, implying that coupling effects are weak and decoupled theory may as well be applied. But as d/λ decreases below 2.0 the values of $\langle\overline{\sigma}\rangle$ drops below those predicted by the curve $N\langle\sigma_0\rangle$ for the decoupled dipoles. Although fewer clouds were investigated for N > 30, the same trends persist, as indicated by Figs. 9 and 10.

Although most data generated were of backscattering cross section, some bistatic scattering cross sections were investigated as well. Figures 11-14 present results for rather dense clouds ($d/\lambda \approx 0.59$) and bistatic angles $\beta = 0°$ (monostatic), 45°, 90°, 135° for vertical-to-vertical and vertical-to-horizontal polarizations. Computed data appear as circles and measured data appear as solid dots. (The methods used to obtain the experimental data are described below). Again, the straight lines $N\langle\sigma_0(\beta)\rangle$ represent the ideal case of uncoupled elements, where $\langle\sigma_0(\beta)\rangle$ is the tumble average bistatic cross section a single resonant dipole, calculated according to the formula,

$$(11)\quad \langle\sigma_0(\beta)\rangle = 0.05\lambda^2[1+2\ (\cos\alpha_t \cos\alpha_r + \cos\beta \sin\alpha_t \sin\alpha_r)^2]$$

where α_t and α_r are the angles of the polarization vectors as shown in the accompanying sketch. In every case, we observe the same phenomenon - coupling effects a decrease in average cross section for both polarization combinations and all bistatic angles.



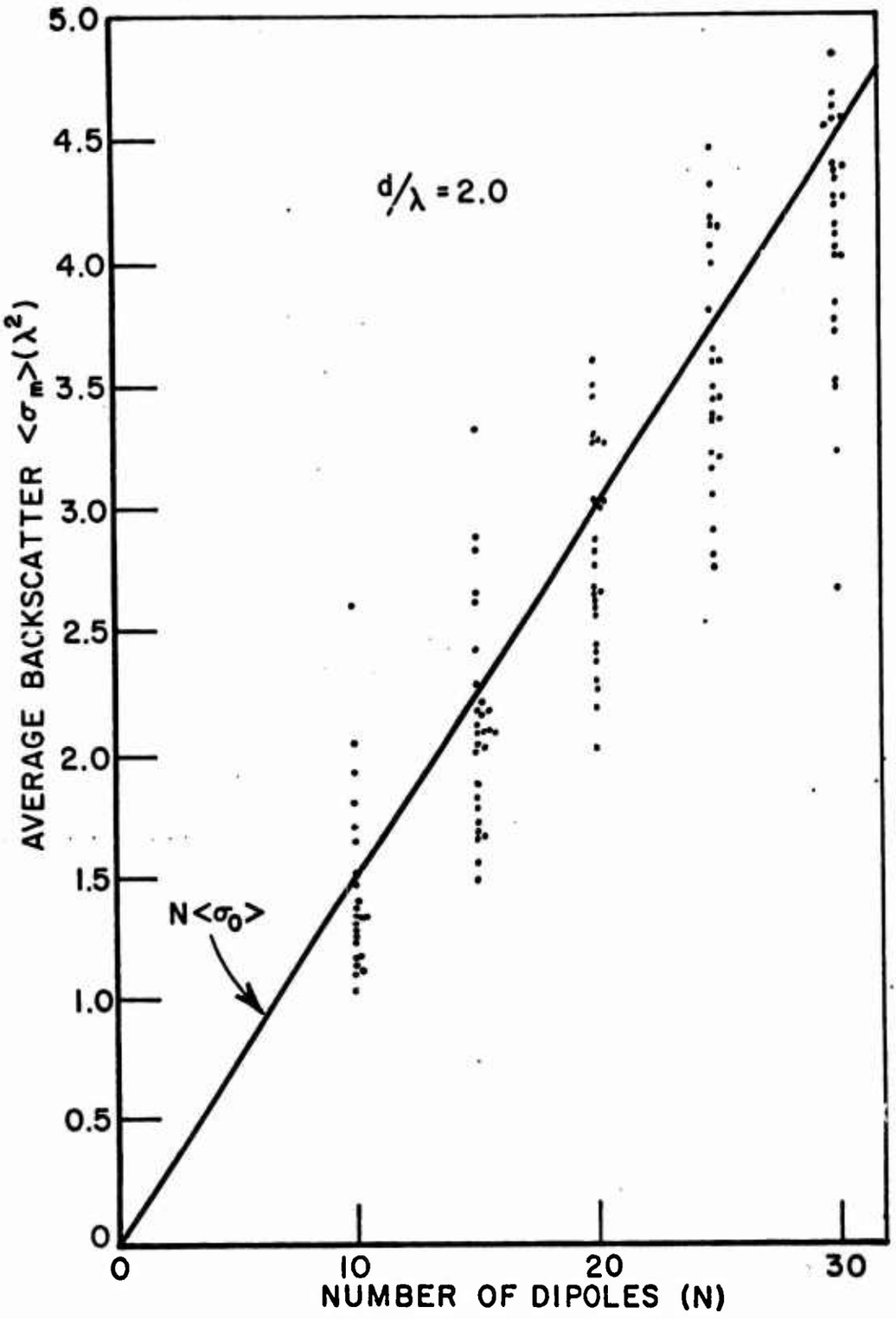


Figure 1. Calculated average backscattering cross sections for ensembles of clouds containing $N \leq 30$ dipoles with an average spacing $d/\lambda = 2$. Straight line represents decoupled dipoles.

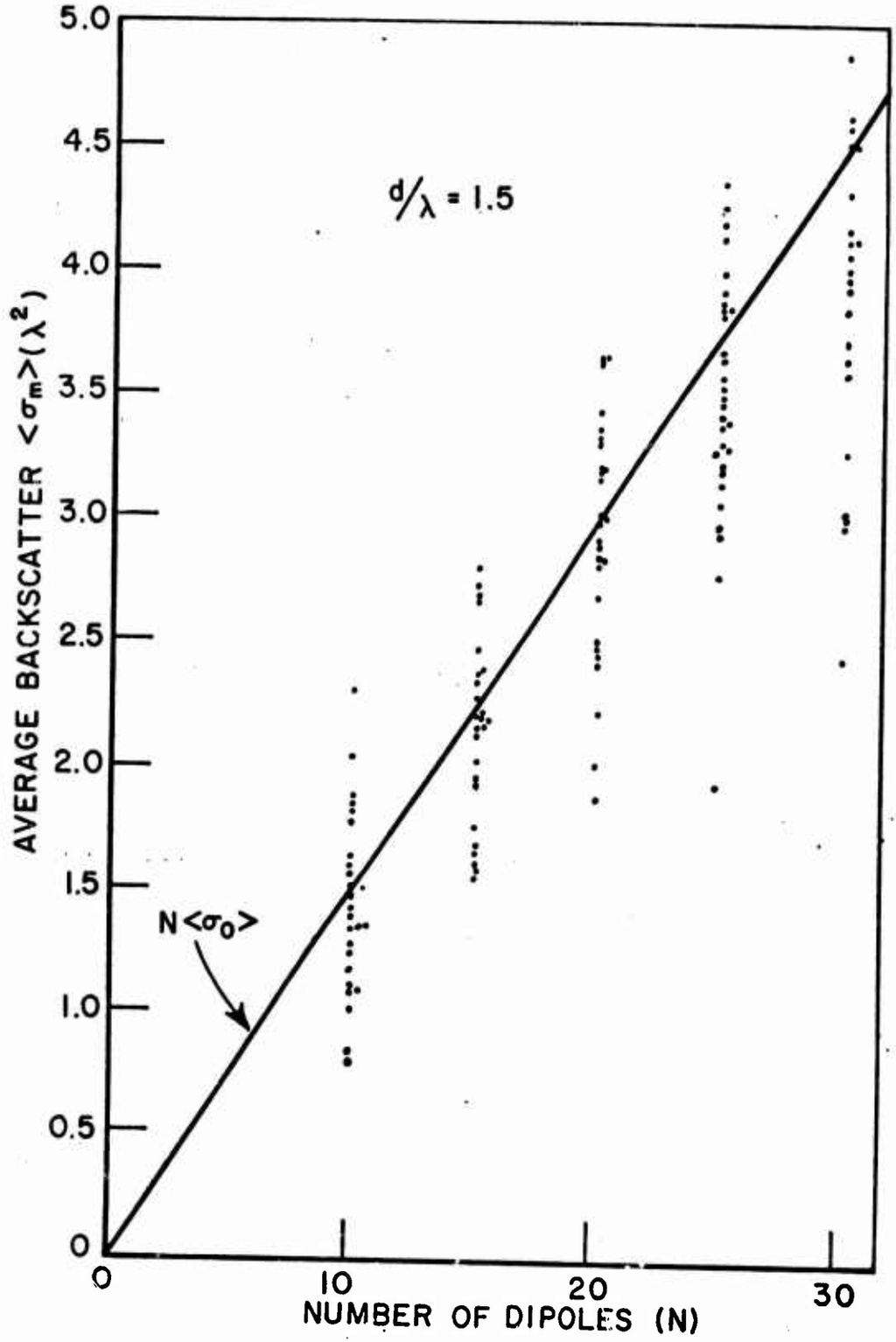


Figure 2. Calculated average backscattering cross sections for ensembles of clouds containing $N \leq 30$ dipoles with an average spacing $d/\lambda = 1.5$. Straight line represents decoupled dipoles.

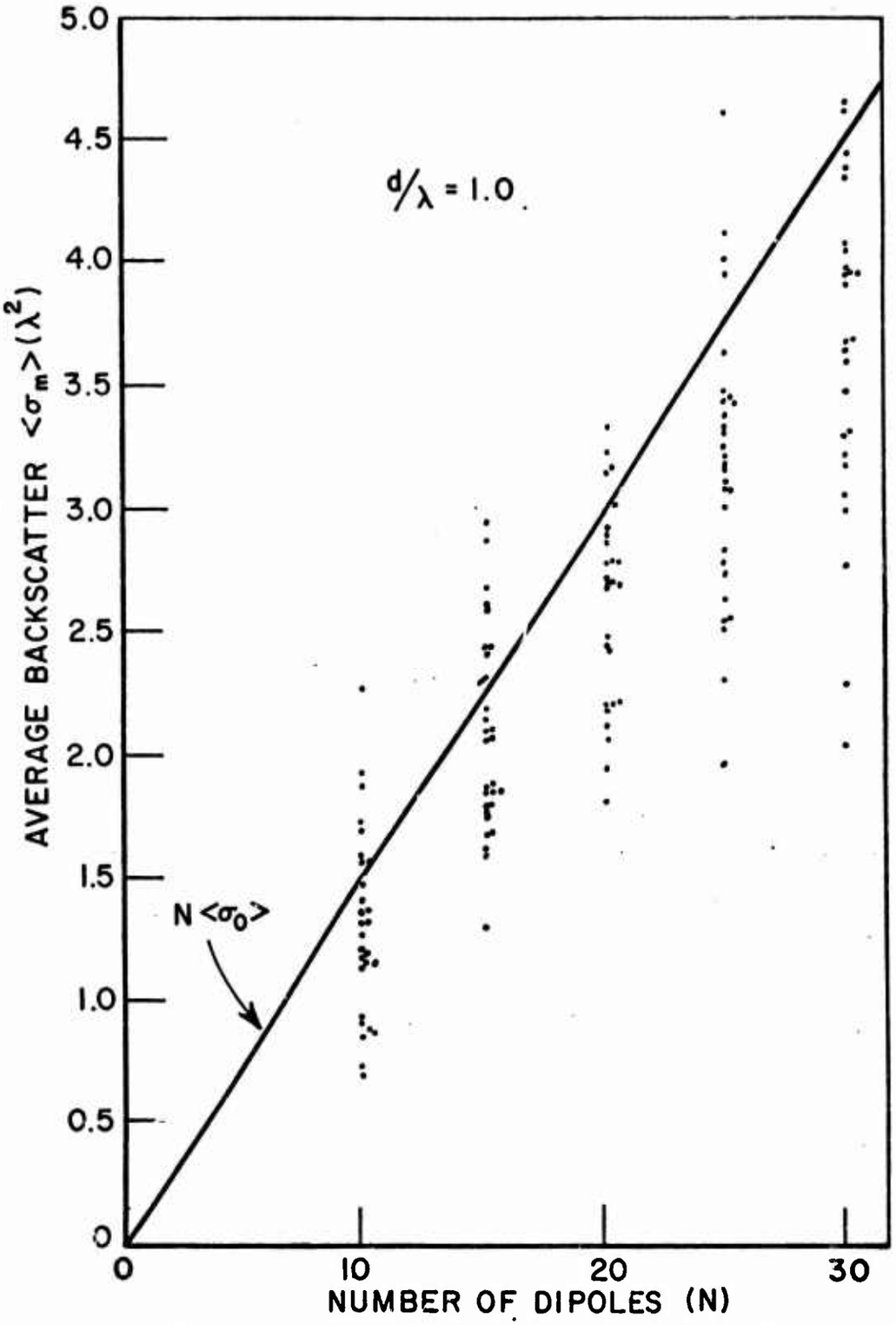


Figure 3. Calculated average backscattering cross sections for ensembles of clouds containing $N \leq 30$ dipoles with an average spacing $d/\lambda = 1.0$. Straight line represents decoupled dipoles.

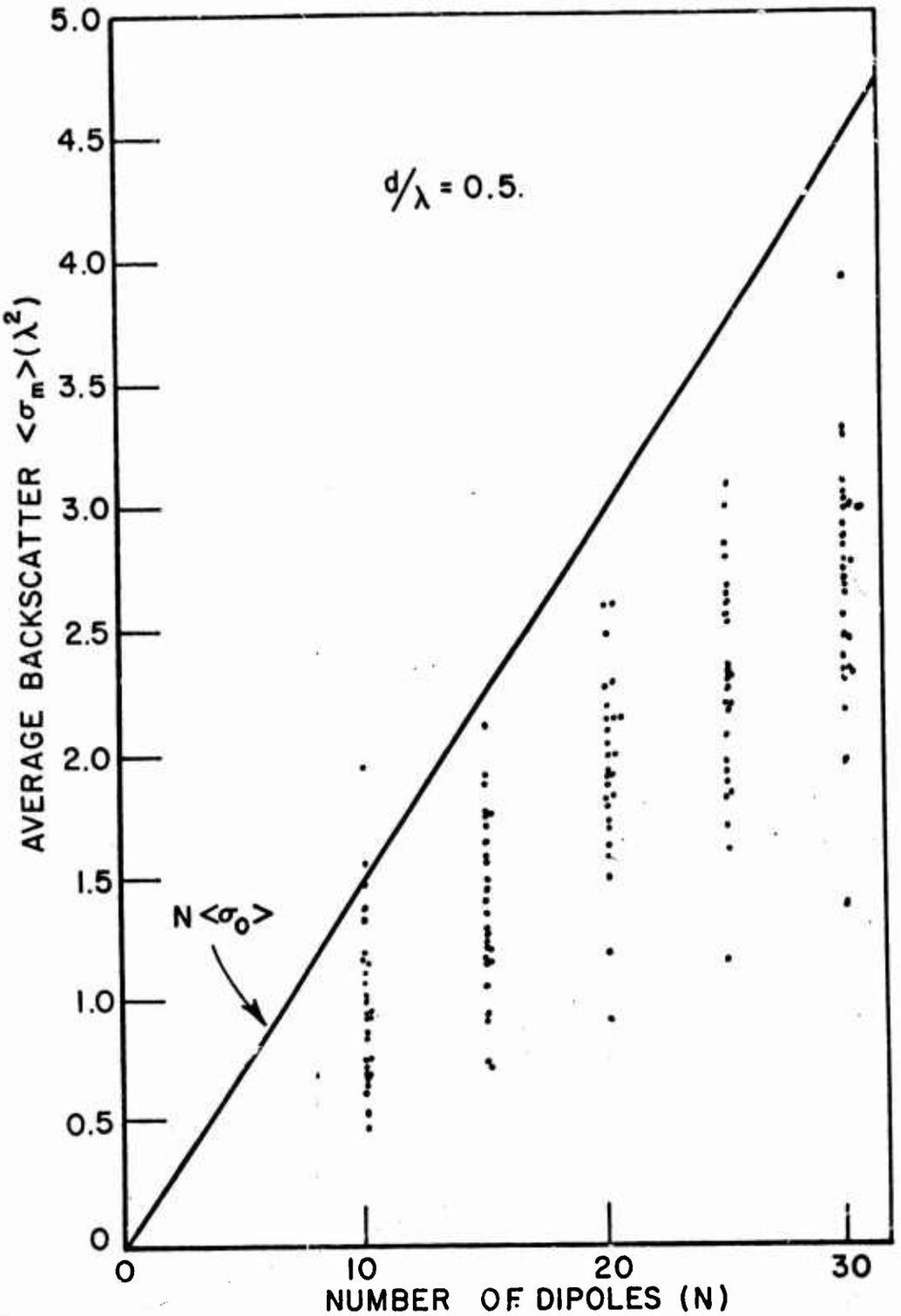


Figure 4. Calculated average backscattering cross sections for ensembles of clouds containing $N \leq 30$ dipoles with an average spacing $d/\lambda = 0.5$. Straight line represents decoupled dipoles.

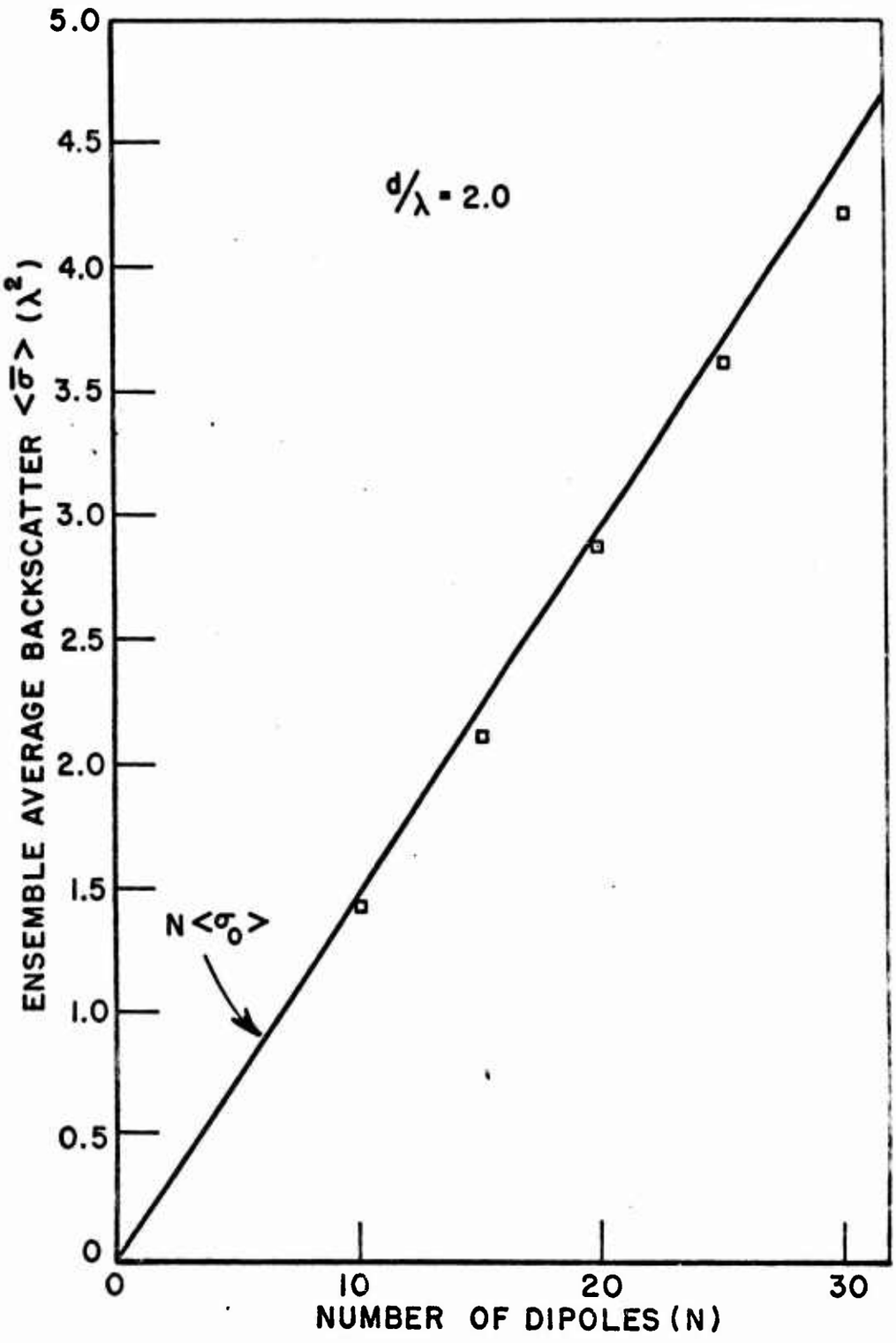


Figure 5. Calculated ensemble averages of the spatial averages shown in Fig. 1. Straight line represents decoupled dipoles.

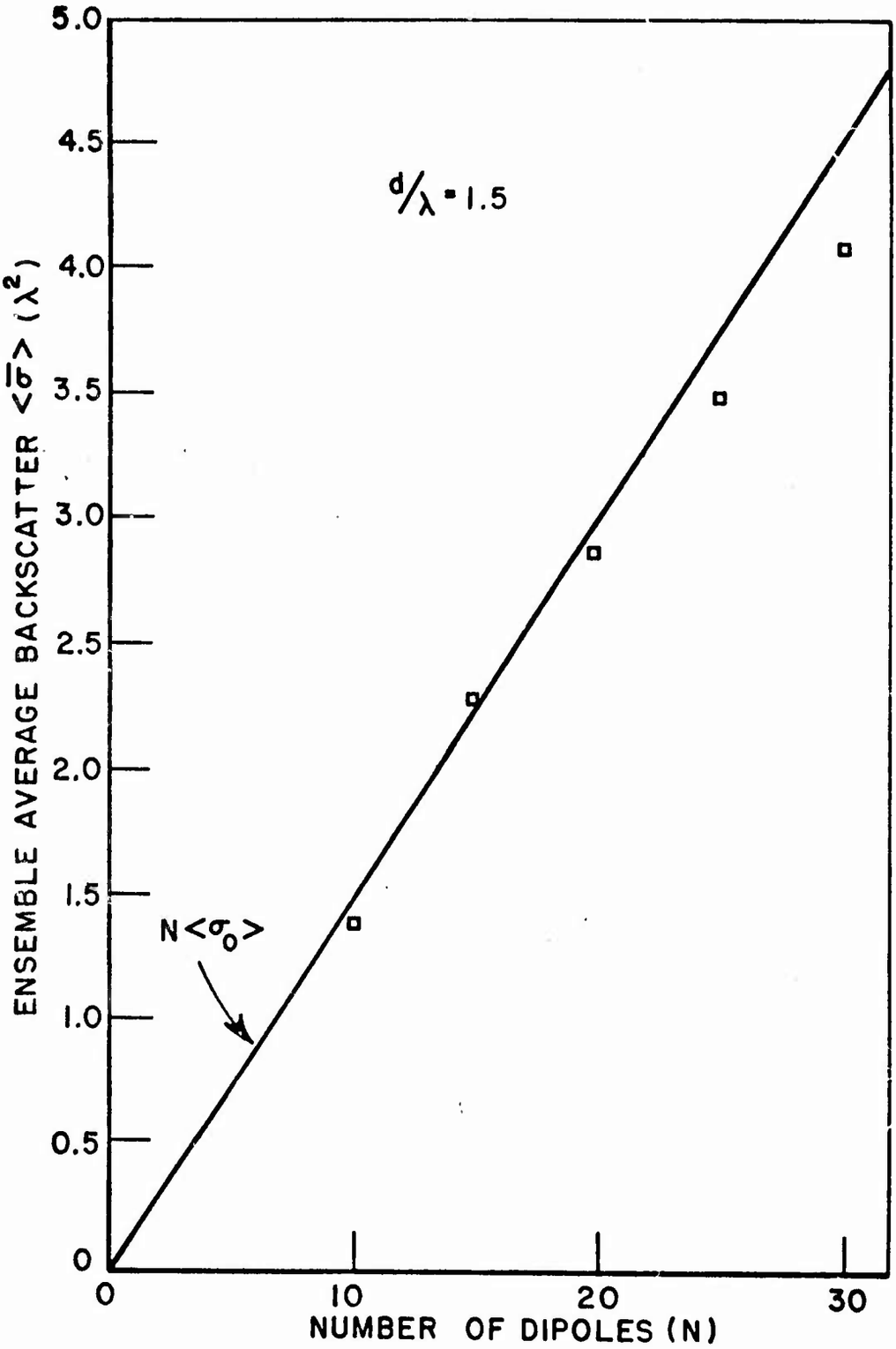


Figure 6. Calculated ensemble averages of the spatial averages shown in Fig. 2. Straight line represents decoupled dipoles.

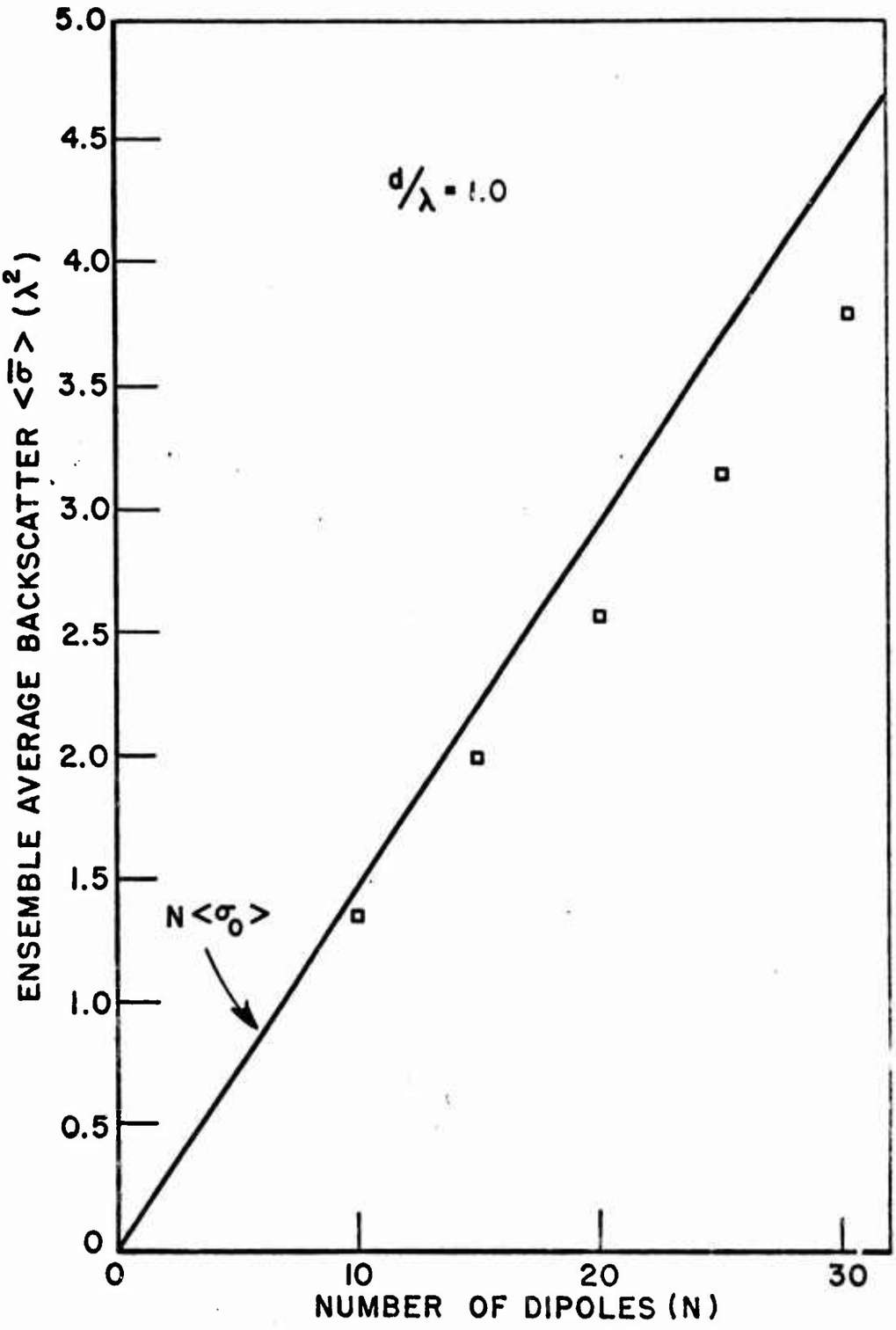


Figure 7. Calculated ensemble averages of the spatial averages shown in Fig. 3. Straight line represents decoupled dipoles.

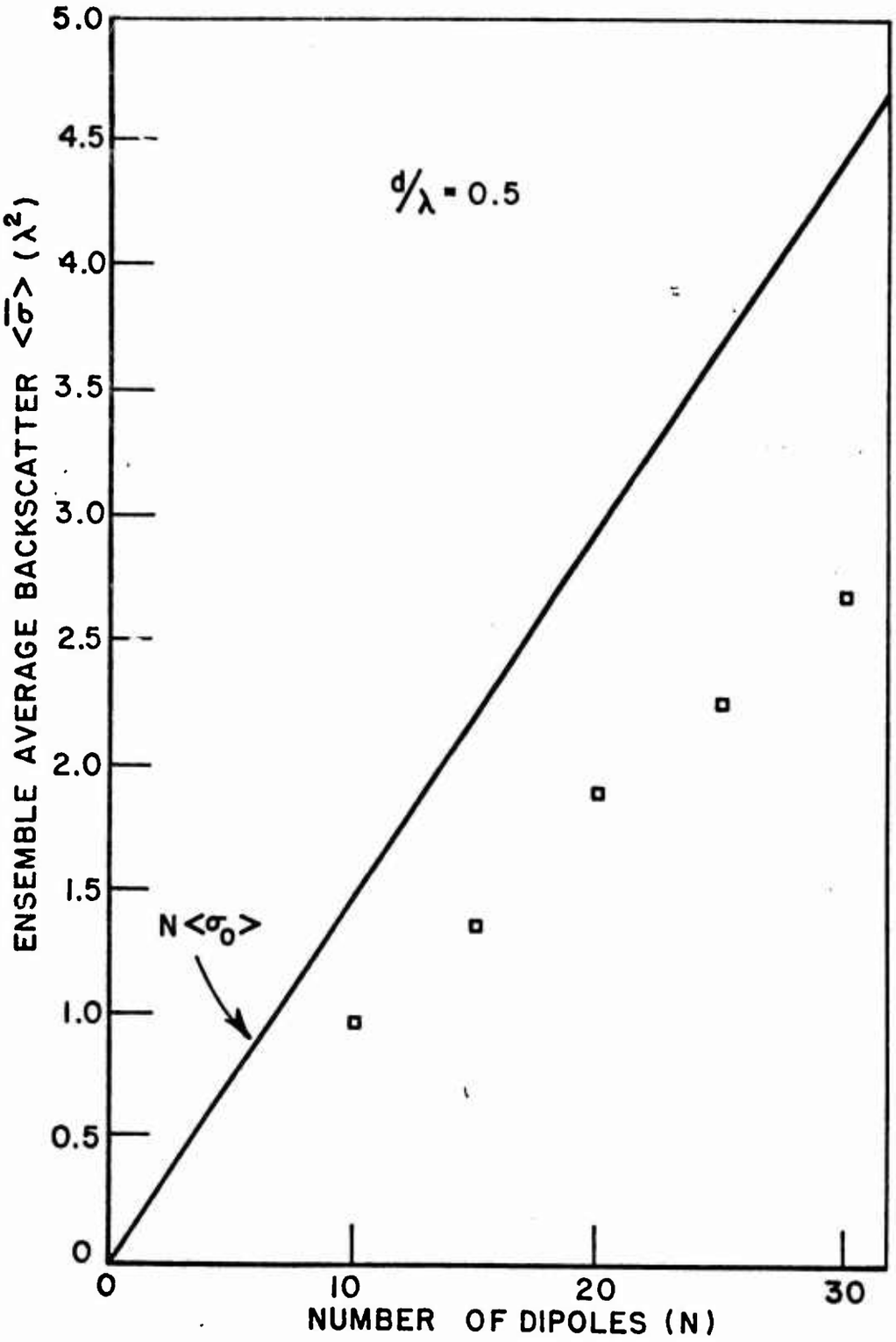


Figure 8. Calculated ensemble averages of the spatial averages shown in Fig. 4. Straight line represents decoupled dipoles.

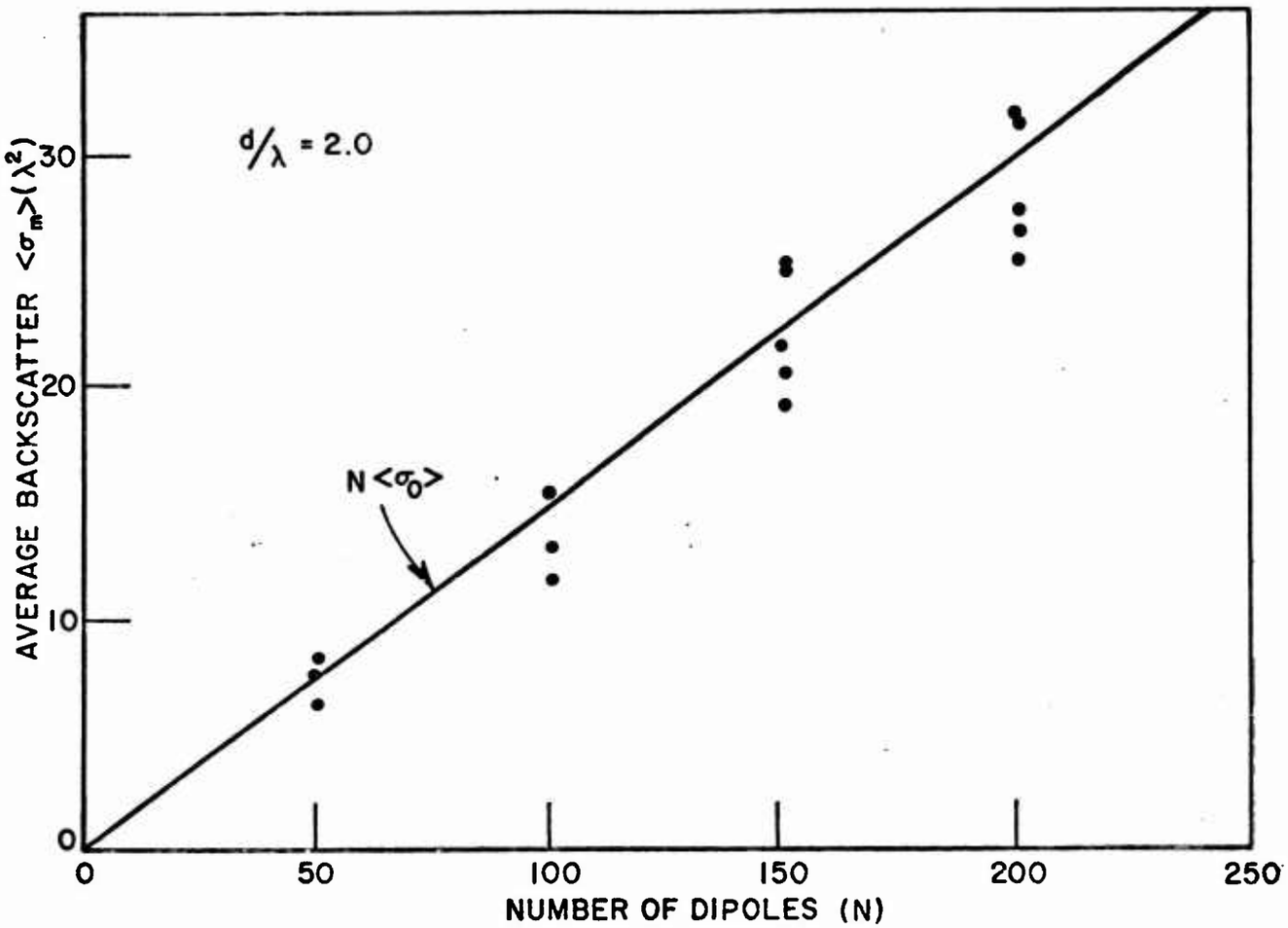


Figure 9. Calculated spatial average backscattering cross sections for ensembles of clouds containing $50 < N < 200$ dipoles with an average spacing $d/\lambda = 2$. Straight line represents decoupled dipoles.

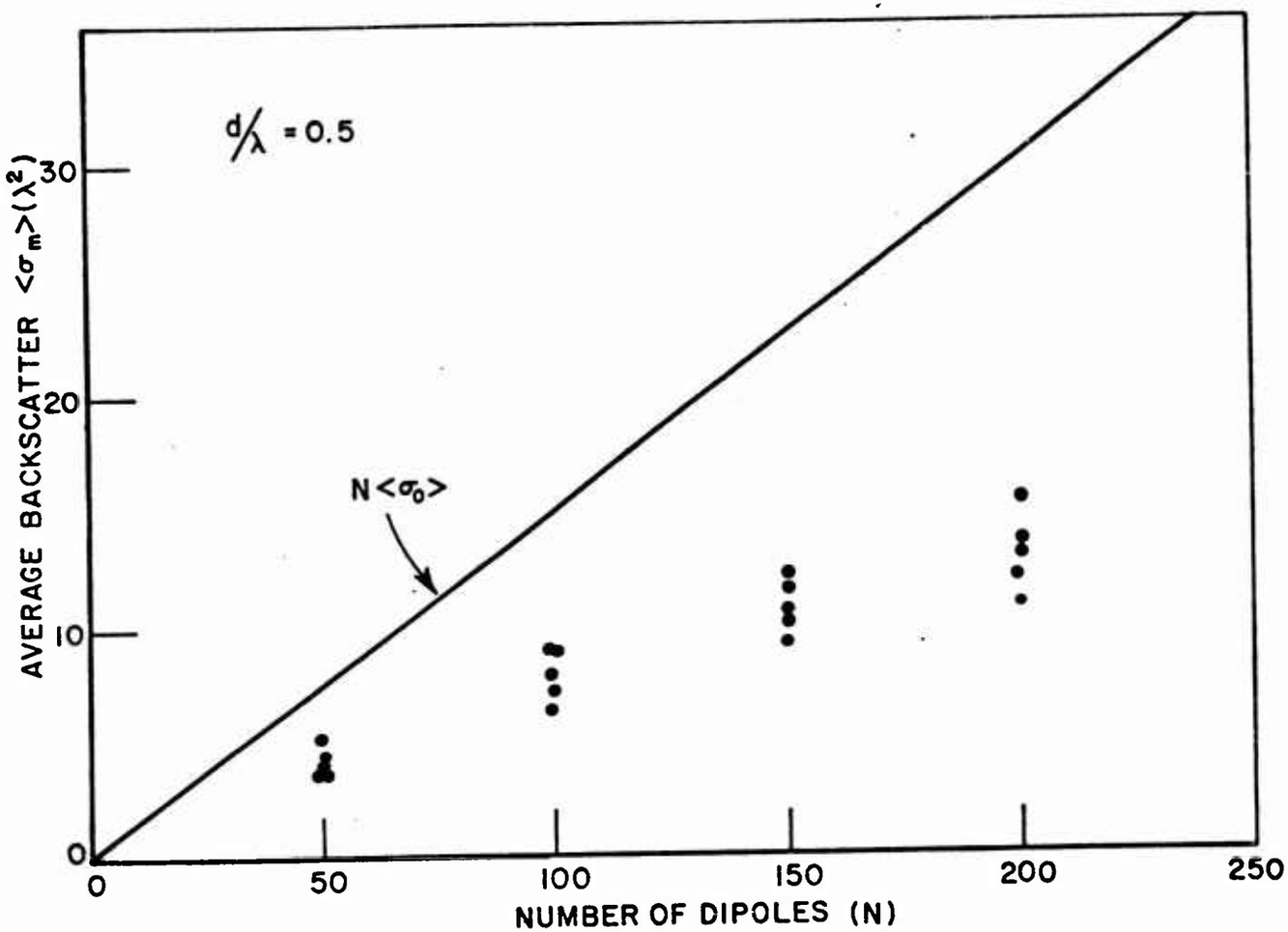


Figure 10. Calculated spatial average backscattering cross sections for ensembles of clouds containing 50 < N < 200 dipoles with an average spacing $d/\lambda = 0.5$. Straight line represents decoupled dipoles.

To obtain the bistatic scattering data of Figs. 11-14 up to 800 polyfoam spheres, each containing a dipole, were enclosed in a polyethelene bag which was rotated by means of strings. Horizontal polarization was transmitted to minimize string reflections and as the bag was rotated a cross section pattern was recorded and automatically averaged. Between runs, the bag was jostled to form a new cloud so that a variance could be observed for the average return.

Figure 15 shows the calculated spatial average backscatter as a function of frequency of four particular random clouds of N = 30 dipoles each. In this figure, vertical-to-vertical polarization is assumed and ℓ/λ is the electrical length of each dipole which is varied through the resonance region. The curves marked $N\langle\sigma_0\rangle$ is for the ideally decoupled case and the other curves are for average spacings for each cloud of $d/\lambda = 2.0$ and 0.5. As expected, the closer spacing reduced the backscatter, but it does not significantly change the frequency of resonance. This result leads us to conclude that it is fruitless to seek a chaff cloud which blooms to a higher value of radar cross section than expected early in its evolutionary history by cutting the dipoles to any length other than the free space resonant length.

3. Sparse Matrix Methods

(a) Theoretical Considerations

In addition to the gathering of computed and measured data to obtain averages of backscattering cross sections, some effort has been directed at alternative methods for solving large matrix equations. The reaction method of Richmond leads to kernel matrices of the order N x N which effectively must be inverted by one method or another. Using Crout-type methods just discussed and a large scale computer limit N to about 250; if more dipoles than this are of interest other methods must be sought to overcome the storage and time problems. In this and the following section we discuss two methods which we investigated - sparse matrix and iterative techniques.

Before launching into a discussion of these techniques, it is appropriate to enquire why one is interested in larger numbers of dipoles, especially since informaton concerning far scattered data are more easily derived from smaller clouds. The answer lies in the intent to characterize a chaff cloud by more than its average cross section, in particular, to calculate the fields inside a cloud as a function of depth of penetration and obtain some insight to the extinction and phase shift incurred. In order to obtain a substantial depth, it may be necessary to account for more than 250 dipoles, in which case new computer methods are

necessary. Such information would be useful for estimating the thickness of a layer of dipoles beyond which additional dipoles add very little to the average backscatter.

The solution to the problem of scattering from a cloud of N dipoles involves a system of N equations in N unknowns. Each of these equations contains N terms. Since all N^2 terms must be stored, even large computer systems run out of fast-access memory for relatively few ($N < 300$) dipoles. In order to study larger clouds, some means of reducing the number of stored elements is required.

The terms in the equations relate to the interaction (mutual impedance) between pairs of dipoles in the cloud. For dipoles that are widely separated or nearly perpendicular to each other, the associated mutual impedance can become quite small. If some threshold level is chosen for the magnitude of the mutual impedance and all mutual impedances below this threshold are ignored (i.e., set to zero), an approximate solution to the scattering problem may be obtained. The often-used assumption of completely independent dipoles is an extreme example of this type of approximation. Systems of linear equations of this type (i.e., where each equation contains only a few terms) may be solved by what are known as sparse matrix methods.

Sparse matrix methods are similar to other techniques (e.g., Crout, Gauss reduction), except that only non-zero terms are stored and only operations involving non-zero terms are performed. Thus they are faster and require less storage when applicable.

In order to determine whether such an approximate solution can be used for studying chaff clouds, a few tests were run using standard solution techniques (i.e., without implementing the time- and storage-saving algorithms) for several values of the threshold mentioned earlier. In this way the applicability of sparse matrix techniques could be determined before effort was expended to develop specialized computer programs.

Setting the threshold to a value equal to 10% of the magnitude of the dipole self-impedance resulted in a satisfactory percentage of zeros (nearly 80%) in the impedance matrix for several test clouds. The bistatic scattering patterns of twenty thirty-dipole clouds (with $d/\lambda = 0.5$) were calculated using both the full impedance matrix and the sparse matrix obtained with the 10% threshold described above. Each pattern was averaged over 360° of bistatic angle and for each cloud the average obtained using the full matrix solution was compared with the average obtained using the sparse matrix solution. The percentage error for each of the twenty clouds is listed in Table I (where a + error means the sparse matrix yielded an average higher than did the full matrix).

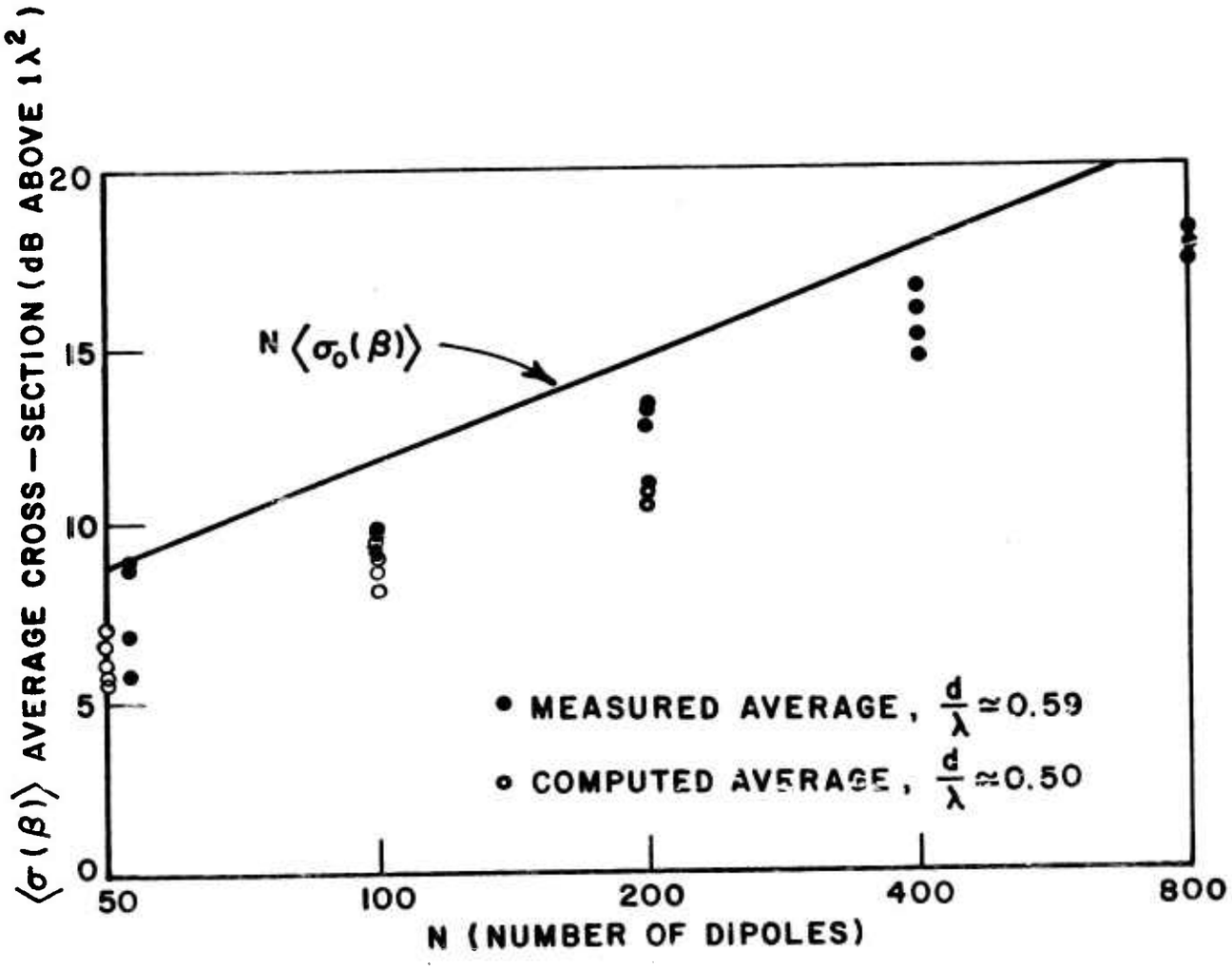


Fig. 11. Measured and calculated values of the spatial average cross sections of ensembles of clouds containing 50 < N < 800 dipoles with average spacings $d/\lambda \simeq 0.5$-0.6. Bistatic angle $\beta = 0°$ (monostatic case). Straight line represents decoupled dipoles.

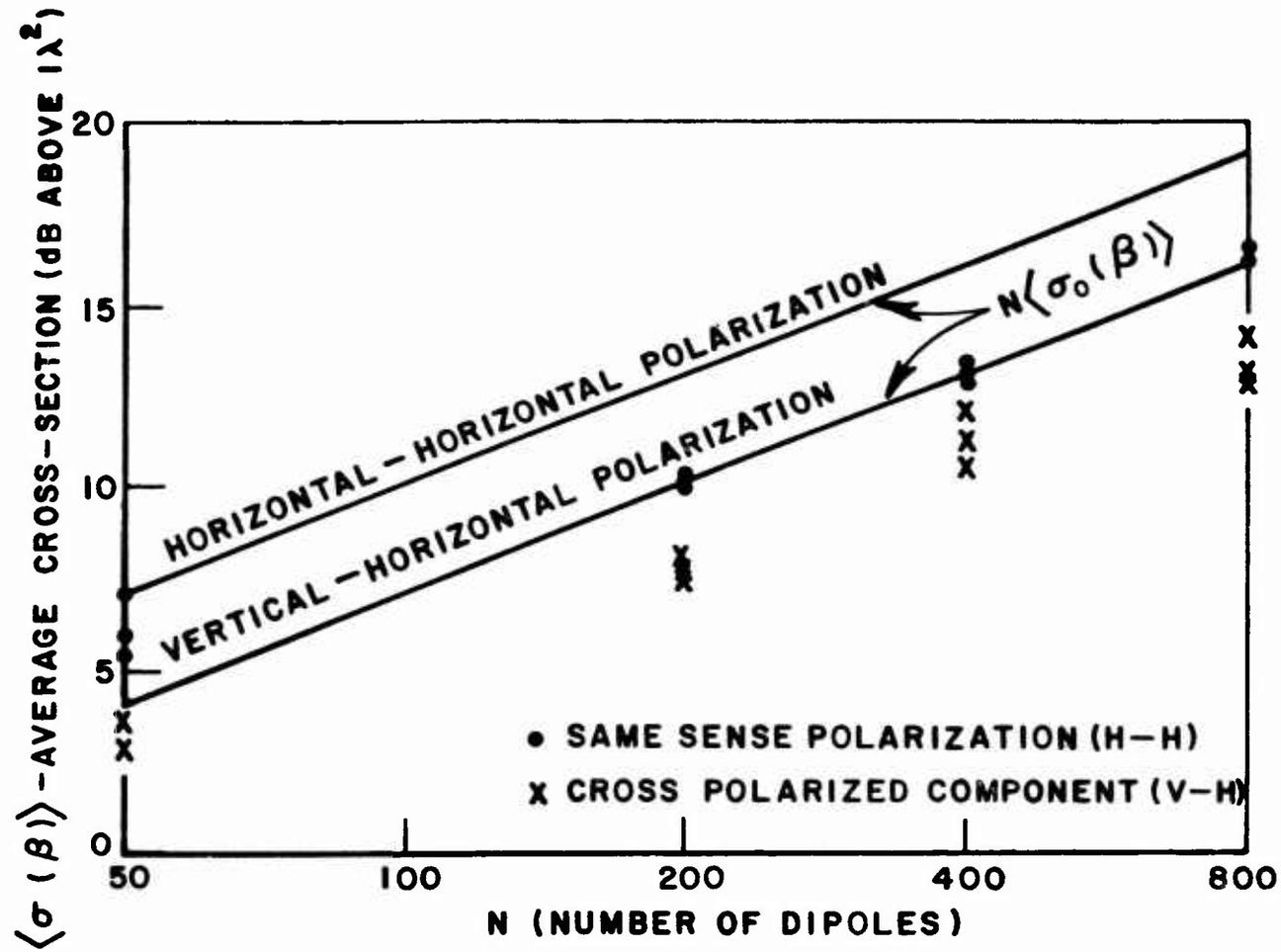


Figure 12. Measured values of the spatial average cross sections of ensembles of clouds containing $50 < N < 800$ dipoles with average spacings $d/\lambda \simeq 0.5\text{-}0.6$. Bistatic angle $\beta=45°$. Straight line represents decoupled dipoles.

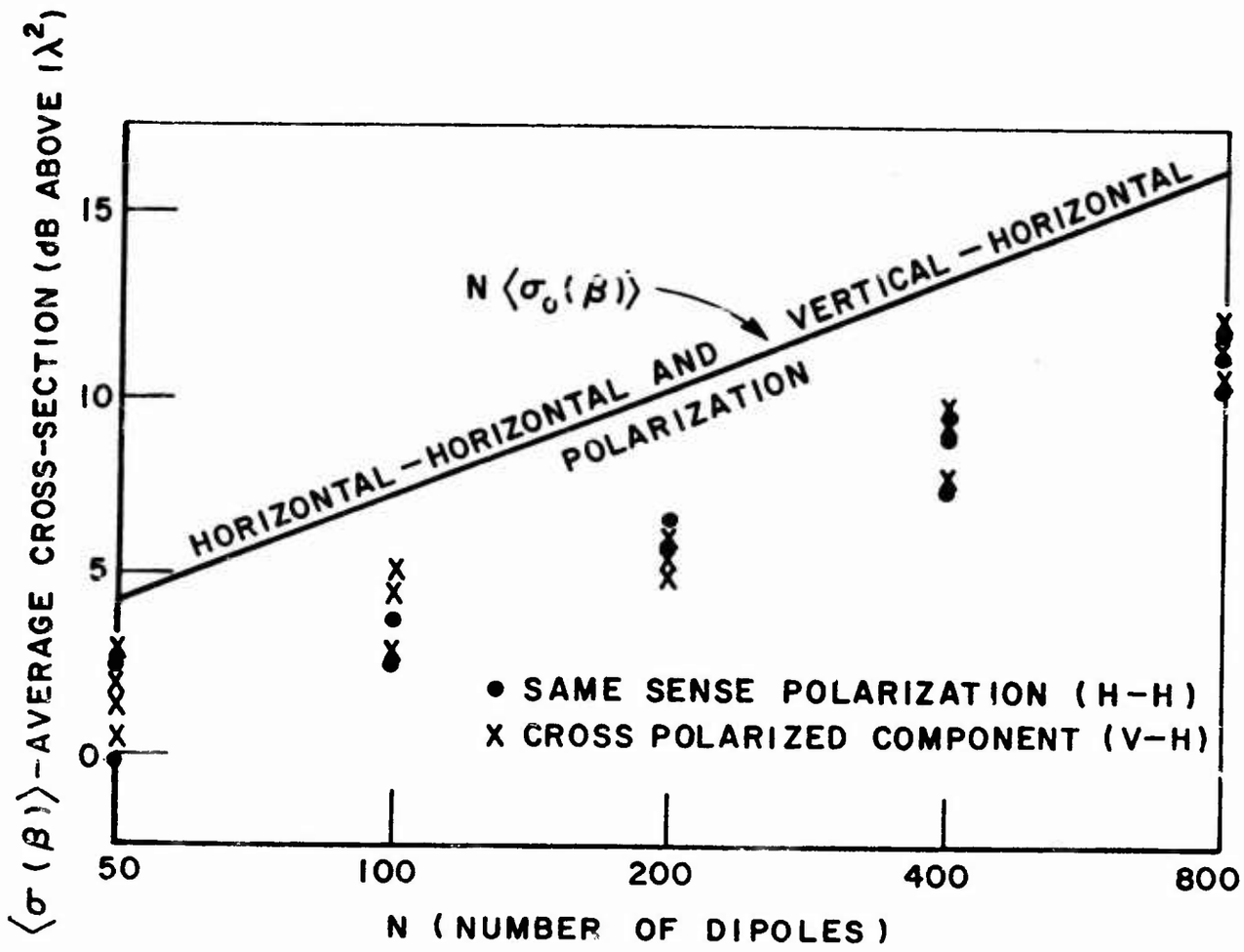


Figure 13. Measured and calculated values of the spatial average cross sections of ensembles of clouds containing $50 \leq N \leq 800$ dipoles with average spacings $d/\lambda \approx 0.5$-0.6. Bistatic angle $\beta=90°$. Straight line represents decoupled dipoles.

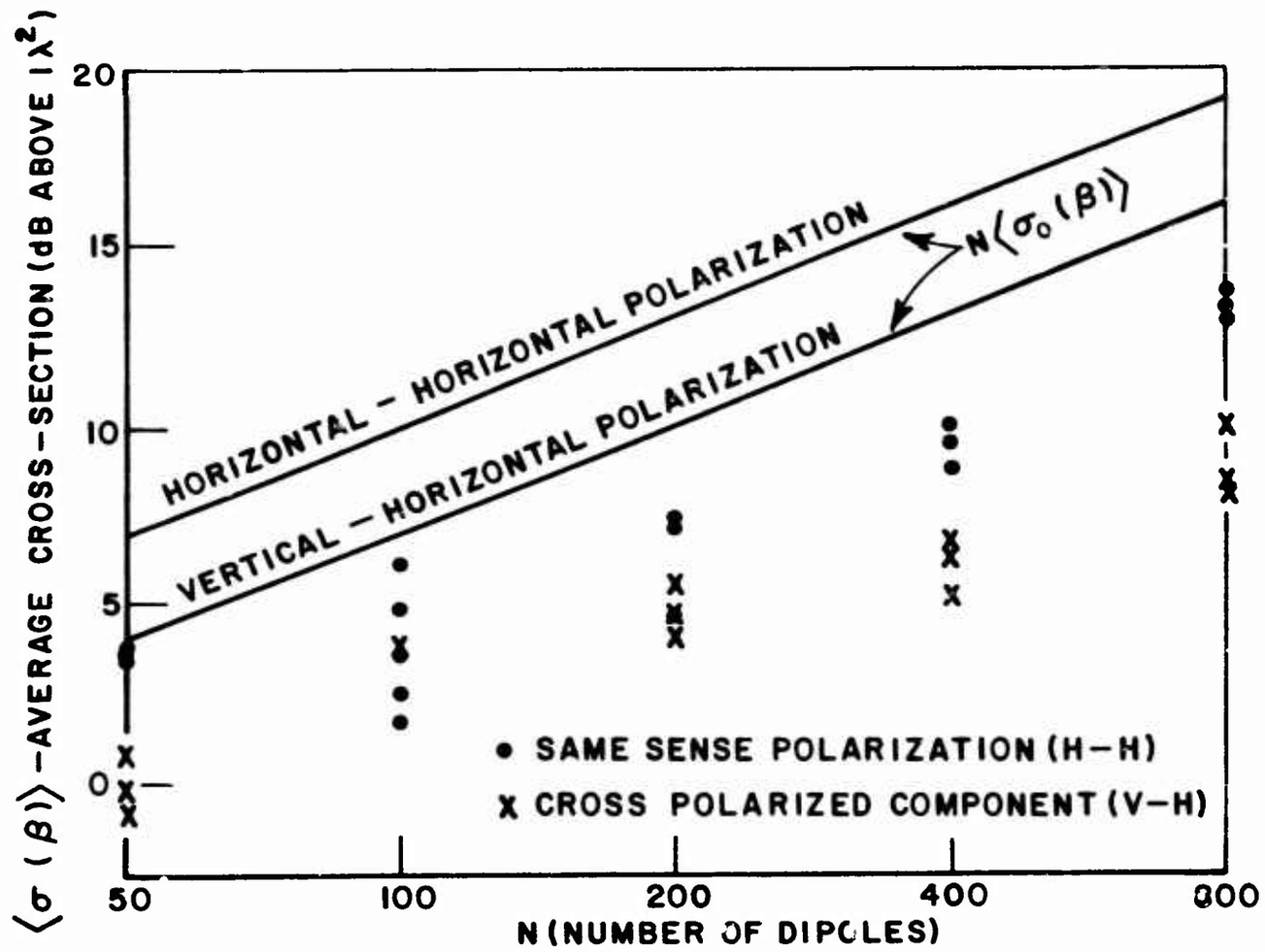


Figure 14. Measured and calculated values of the spatial average cross sections of ensembles of clouds containing $50 < N < 800$ dipoles with average spacings $d/\lambda \simeq 0.5\text{-}0.6$. Bistatic angle $\beta = 135°$. Straight line represents decoupled dipoles.

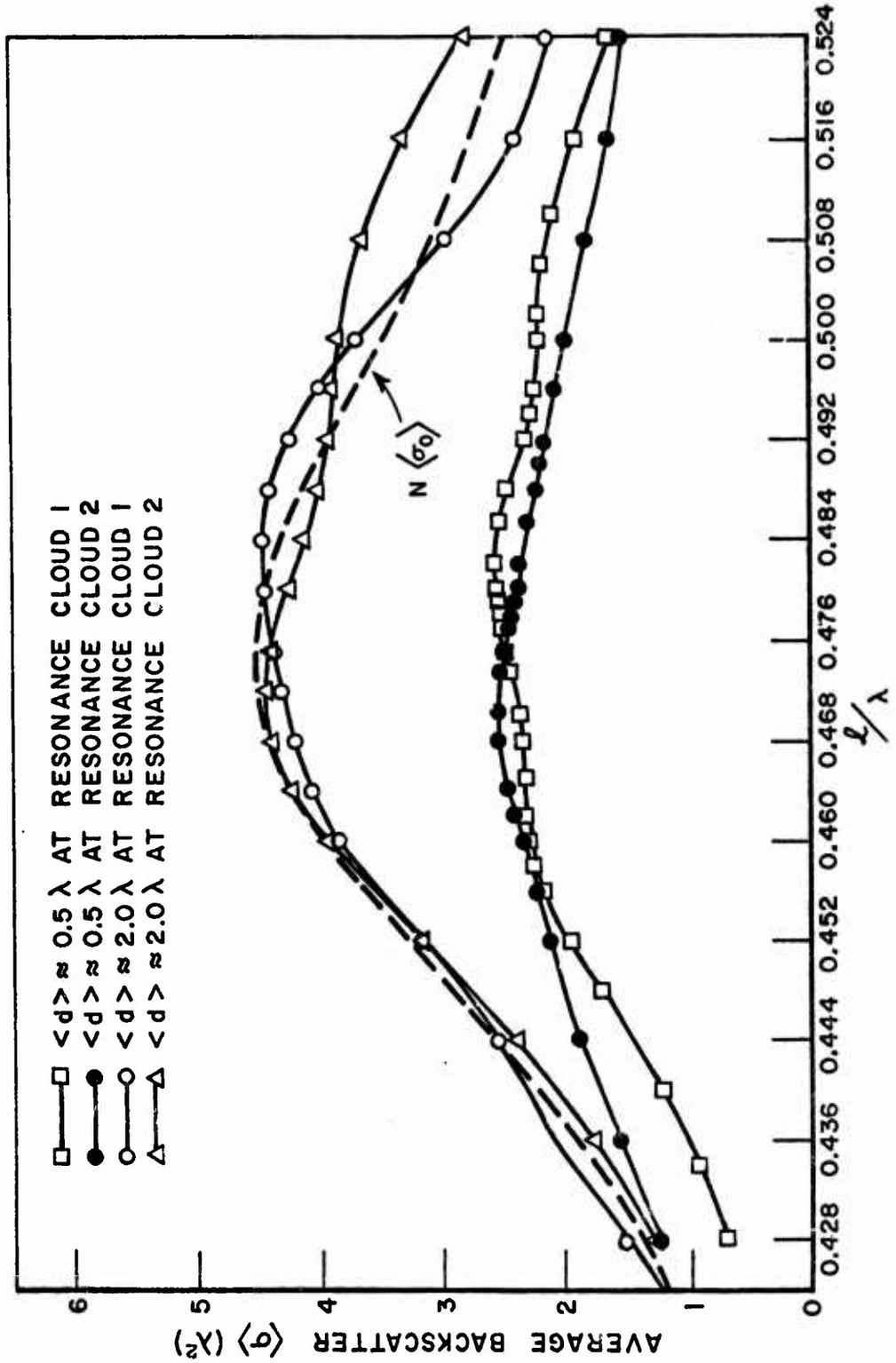


Figure 15. Average backscatter as a function of frequency of four random clouds.

The average error was 5.3% and the average absolute error was 6.8%, well within tolerance levels of practical measurements.

From these calculations it appeared feasible to further develop sparse matrix programs for use on chaff cloud scatter calculations.

TABLE I

PERCENT ERROR IN BISTATIC AVERAGES CAUSED BY SETTING MUTUAL IMPEDANCES BELOW (0.1) (Z_{11}) TO ZERO.

-4.2	+5.3	+ 2.9	+11.9	+8.7	-3.2
-7.7	+6.7	+10.0	+13.3	+13.8	+2.6
+10.2	+5.5	+ 4.8	+ 3.5	+ 4.3	

Sparse matrix methods require that a special scheme be used to index the stored elements of the matrix. Also most direct methods of solving systems of linear equations operate on the matrix to produce a new matrix which in general is not sparse even though the original matrix was sparse. Sparse matrix methods require that this new matrix be sparse as well. These two requirements have been approached and formulated in a variety of ways [40-45].

The approach used here is that given by Berry [44]. The off-diagonal non-zero elements of the upper trianagular portion of the matrix are stored consecutively in linear array U. The diagonal elements (which are all non-zero) are stored in a linear array D. Two pointer arrays II and J are used to index the array U. II(K) contains the starting location of row K in U and J contains the column indices of the elements in the same order as the elements as contained in U. An example given by Berry [44] should help clarify this scheme. For the matrix Y given below, the arrays would be as follows:

$$Y = \begin{bmatrix} y_{11} & 0 & y_{13} & 0 & y_{15} \\ 0 & y_{22} & y_{23} & y_{24} & 0 \\ y_{31} & y_{32} & y_{33} & y_{34} & 0 \\ 0 & y_{42} & y_{43} & y_{44} & 0 \\ y_{51} & 0 & 0 & 0 & y_{55} \end{bmatrix}$$

II(1) = 1	J(1) = 3	U(1) = y_{13}	D(1) = y_{11}
II(2) = 3	J(2) = 5	U(2) = y_{15}	D(2) = y_{22}
II(3) = 5	J(3) = 3	U(3) = y_{23}	D(3) = y_{33}
II(4) = 6	J(4) = 4	U(4) = y_{24}	D(4) = y_{44}
II(5) = 6	J(5) = 4	U(5) = y_{34}	D(5) = y_{55}

A specialized matrix decomposition known as the "square root method" [46] is used to solve the system of equations. This method is similar to those associated with the names Gauss, Crout, Doolittle, Cholesky, Banachiewicz, etc. [47].

Before decomposition, the algorithm given by Barry is used to determine a renumbering of the unknown (pivoting on the diagonal) such that the number of non-zero elements in the auxiliary matrix produced by the decomposition is reduced. The advantage of this renumbering is easily seen in a couple of examples. Figure 16 shows the structure of an 11 by 11 matrix and its auxiliary before renumbering. Crosses represent non-zero elements occuring in both the original matrix and its auxiliary. Zeros represent non-zero elements occuring only in the auxiliary matrix, i.e., non-zero elements that were introduced by the decomposition. Blanks represent zero elements occuring in both the original matrix and its auxiliary. Figure 17 shows the structure of the matrix after renumbering and the structure of the auxiliary of this new matrix in the same way. The renumbering used was as follows:

original unknown no.	1	2	3	4	5	6	7	8	9	10	11
new unknown no.	1	9	7	6	5	2	3	8	4	10	11

The structure of Fig. 17 may be obtained from that of Fig. 16 and the above table. For example: to generate the seventh row of Fig. 17, first note that the seventh unknown in the renumbered system was the third unknown in the original system. This means that the third row of the original matrix is the seventh row of the new matrix. Columns have also been interchanged according to this same renumbering so that $Z_{33} \rightarrow Z_{77}$. To find the other elements in the new seventh row, note in Fig. 16 that the off-diagonal elements in row 3 are Z_{34}, Z_{36}, and Z_{38} and convert both subscripts as given in the table to obtain

	1	2	3	4	5	6	7	8	9	10	11
1	X						X		X		
2		X								X	X
3			X	X		X		X			
4			X	X		X		X			
5					X		X		X	X	
6			X	X		X		0			
7	X				X		X		X	0	
8			X	X		0		X		X	X
9	X				X		X		X	0	
10		X			X		0	X	0	X	0
11		X						X		0	X

Fig. 16. The structure of an 11 x 11 matrix and its auxiliary before renumbering. Crosses are non-zero elements occuring in both matrices; zeros are non-zero elements occuring only in the auxiliary; blanks are zero elements in both matrices.

	1	2	3	4	5	6	7	8	9	10	11
1	X		X	X							
2		X				X	X				
3	X		X	X	X						
4	X		X	X	X						
5			X	X	X					X	
6		X				X	X	X			
7		X				X	X	X			
8						X	X	X		X	X
9									X	X	X
10					X			X	X	X	0
11								X	X	0	X

Fig. 17. The structure of the 11 x 11 renumbered matrix of Fig. 16 and its auxiliary. Symbols are the same as in Fig. 16.

$Z_{34} \rightarrow Z_{76}$, $Z_{36} \rightarrow Z_{72}$, and $Z_{38} \rightarrow Z_{78}$ which is the structure shown in Fig. 17.

Figures 18 and 19 show the structure of a 28 by 28 matrix before and after renumbering in the same way.

The renumbering used in this case was as follows:

original	1	2	3	4	5	6	7	8	9	10	11	12	13	14
new	9	18	24	22	8	20	14	21	27	15	19	16	13	17

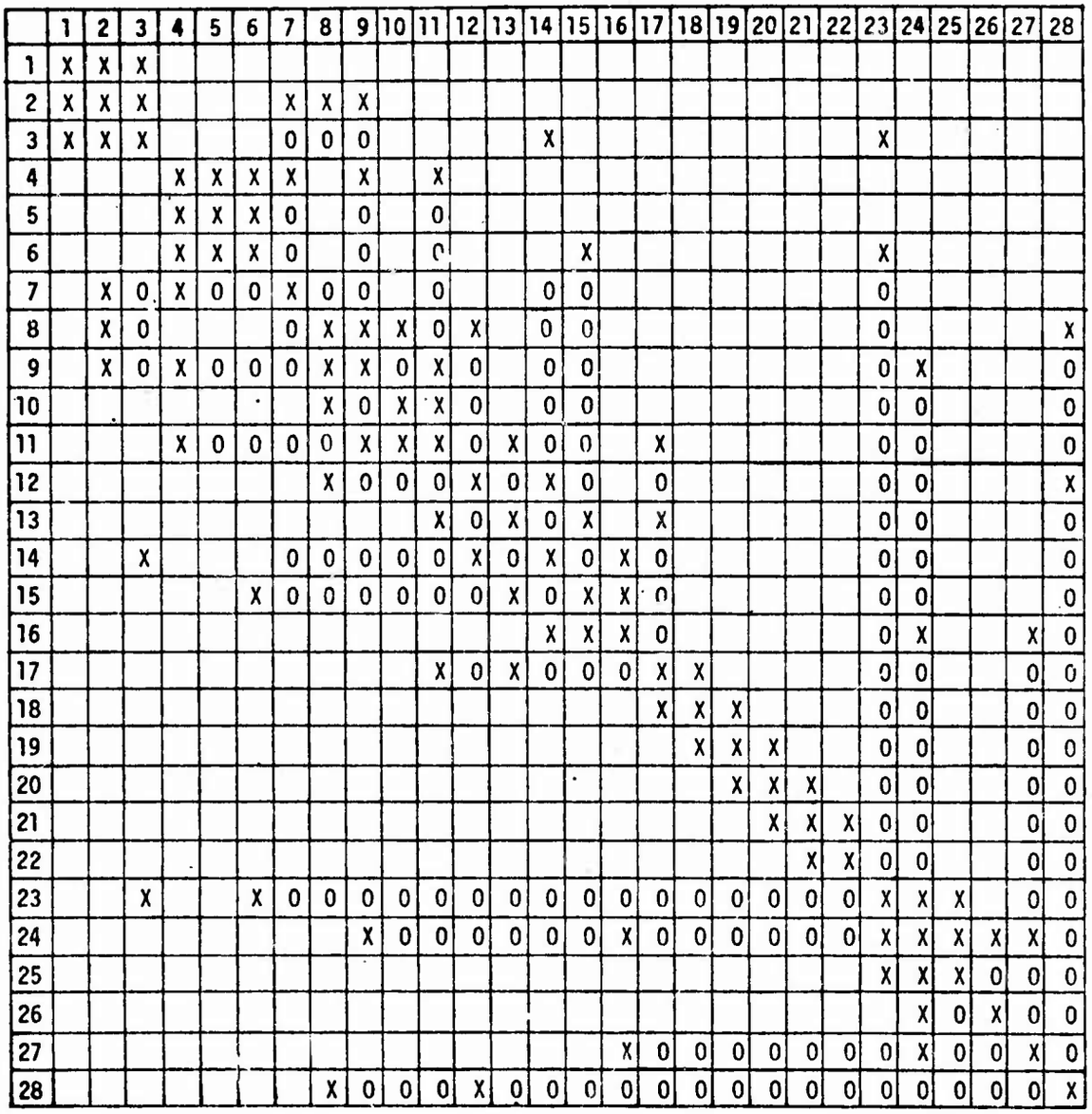
original	15	16	17	18	19	20	21	22	23	24	25	26	27	28
new	25	28	7	5	4	3	2	1	26	23	10	6	11	12

The number of non-zero elements occuring in the auxiliary matrix is substantially reduced by the renumbering as may be seen by comparing the number of zeros in Figs. 16 and 18 with the number of zeros in Figs. 17 and 19.

(b) Calculated results for Chaff Clouds

In order to estimate the savings in time and computer storage requirements resulting from use of the sparse matrix algorithm, a study was made of these parameters using the ElectroScience Laboratory Datacraft 6024 computer and the Wright-Patterson Air Force Base CDC 6600 computer.

In particular, it would be useful to obtain some estimate of the number of non-zero elements which are regarded as significant enough to retain and store. If we regard as zero any elements in the impedance matrix whose magnitude is less than 10% of the magnitude of the self impedance (diagonal) elements, and we calculate the number of non-zero elements remaining in the upper triangle matrix (Table 2), we can obtain the percent of non-zero elements in the upper triangle (Table 3). The numbers presented in these tables are averages of values obtained from 10 randomly generated clouds for each combination of average spacing d/λ and number of dipoles N.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
1	X	X	X																									
2	X	X	X				X	X	X																			
3	X	X	X				0	0	0					X									X					
4				X	X	X	X		X		X																	
5				X	X	X	0		0		0																	
6				X	X	X	0		0		0				X								X					
7		X	0	X	0	0	X	0	0		0			0	0								0					
8		X	0				0	X	X	X	0	X		0	0								0					X
9		X	0	X	0	0	0	X	X	0	X	0		0	0								0	X				0
10								X	0	X	X	0		0	0								0	0				0
11				X	0	0	0	0	X	X	X	0	X	0	0		X						0	0				0
12								X	0	0	0	X	0	X	0		0						0	0				X
13											X	0	X	0	X		X						0	0				0
14			X				0	0	0	0	0	X	0	X	0	X	0						0	0				0
15						X	0	0	0	0	0	0	X	0	X	X	0						0	0				0
16														X	X	X	0						0	X			X	0
17											X	0	X	0	0	0	X	X					0	0			0	0
18																	X	X	X				0	0			0	0
19																		X	X	X			0	0			0	0
20																			X	X	X		0	0			0	0
21																				X	X	X	0	0			0	0
22																					X	X	0	0			0	0
23			X			X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	X	X		0	0
24									X	0	0	0	0	0	0	X	0	0	0	0	0	0	X	X	X	X	X	0
25																							X	X	X	0	0	0
26																								X	0	X	0	0
27																X	0	0	0	0	0	0	0	X	0	0	X	0
28								X	0	0	0	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X

Fig. 18. The structure of a 28 x 28 matrix and its auxiliary. Symbols are the same as in Fig. 16.

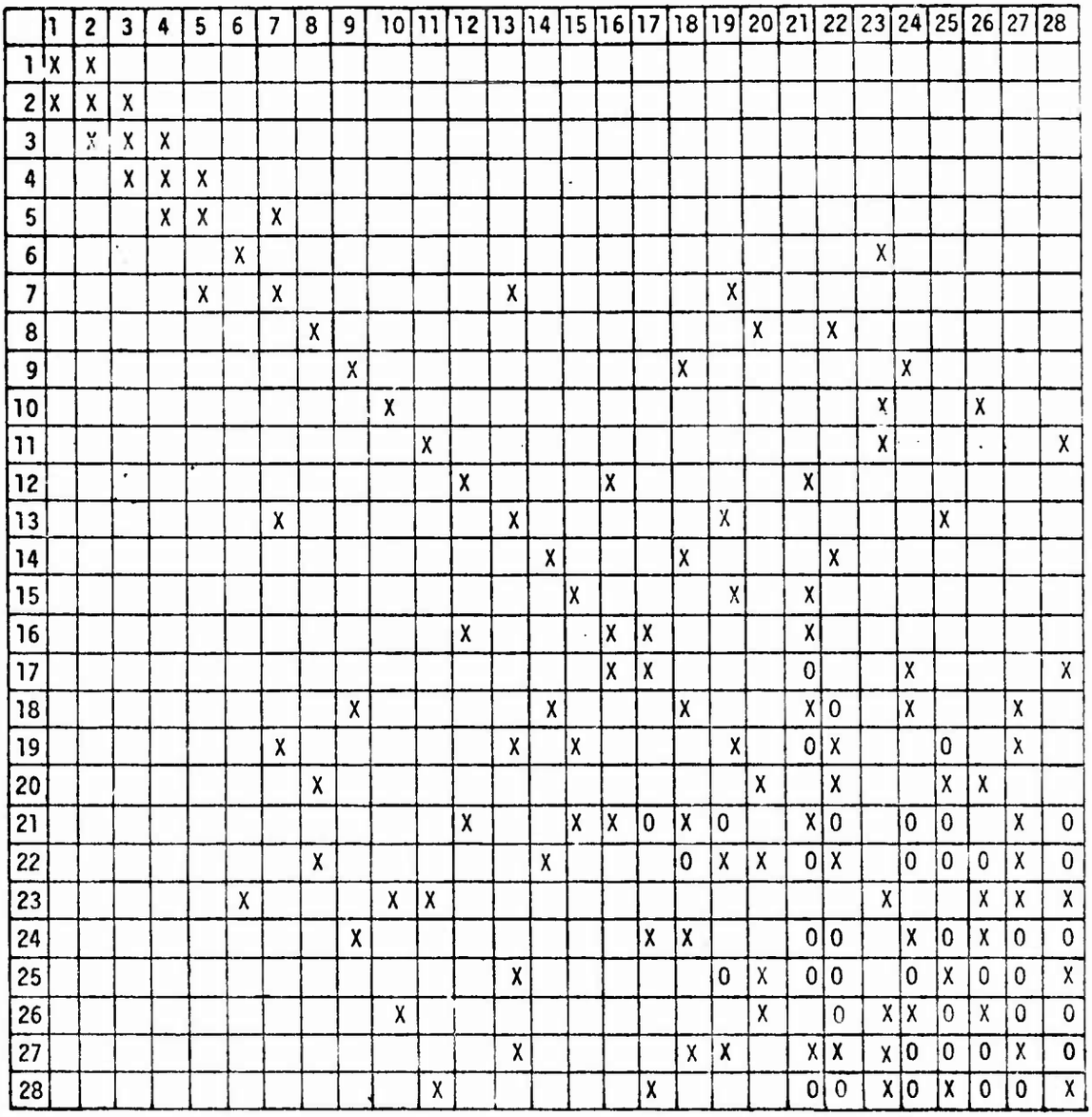


Fig. 19. The structure of the 28 x 28 renumbered matrix of Fig. 18 and its auxiliary. Symbols are the same as in Fig. 16.

The average numbers presented in Table 2 are plotted vs N with d/λ as a parameter in Figs. 20 and 21. They all show a remarkably linear character, indicating that significant coupling (non zero elements) exists between an arbitrary dipole and only its neighbors inside a surrounding "volume of influence". Thus, with d/λ fixed and N increasing, we expect, and do observe, the number of non-zero elements to increase proportionally to N, not N^2. Consequently, the percent of non-zero elements for a fixed d/λ decreases as 1/N with increase in N. Recalling the rule of thumb that this percent should not exceed about 20% if sparse matrix techniques are to be effective, we see that this condition is satisfied for all $d/\lambda \geq 0.5$ for $n \geq 200$, a fortiori for the larger d/λ values. The absolute number of non-zero elements, (Table 2) or course, determines the memory required of the computer. Extrapolating the linear curves of Figs. 20 and 21, it appears that a capability of storing 20,000 non-zero elements (about the number of elements in the upper triangle of the full matrix associated with a cloud of 200 fully coupled elements - an entirely feasible problem of the W-P computer) permits the sparse matrix solution of clouds containing approximately 1100, 5300, 15,300, and 32,000 dipoles if the average spacings d/λ are 0.5, 1.0, 1.5, and 2.0, respectively. Investigations involving time savings, described later, lead us to less optimistic estimates.

The variation of the number of non-zero elements in the upper triangle with d/λ, N fixed, is not as clearly explainable in physical terms as is the variation with N, d/λ fixed. If we consider each dipole to be coupled only to m neighbors within a surrounding "volume of influence", then the number m should be equal to the number of non-zero elements in the upper triangle divided by N. Performing this operation on Table 2, we obtain Table 4, and observe that, except for the smallest spacing $d/\lambda = 0.5$, the values of m (i.e., the number of elements in a "volume of influence") are approximately independent of N, as one would expect. For $d/\lambda = 0.5$, clouds with lesser values of N probably are too small to obtain fair values for m, so we presume those values of m obtained for the largest clouds (N = 200) are most accurate. Accepting these latter numbers, one recognizes, of course, that they are only <u>symbolic</u> of the influence of coupling; they only give some indication of the (integer) number of neighbors which are effectively coupled to a given element in some average sense. We can venture one step further and assume that each "volume of influence" is a "sphere of influence", with volume $V_{d/\lambda} = (d/\lambda)^3 m$ (where the subscript recognizes that the "sphere of influence" has a size which is probably dependent on the cloud density, i.e., d/λ). Doing this for N = 200, the radii in wavelengths $R_{d/\lambda}/\lambda$ of the "spheres of

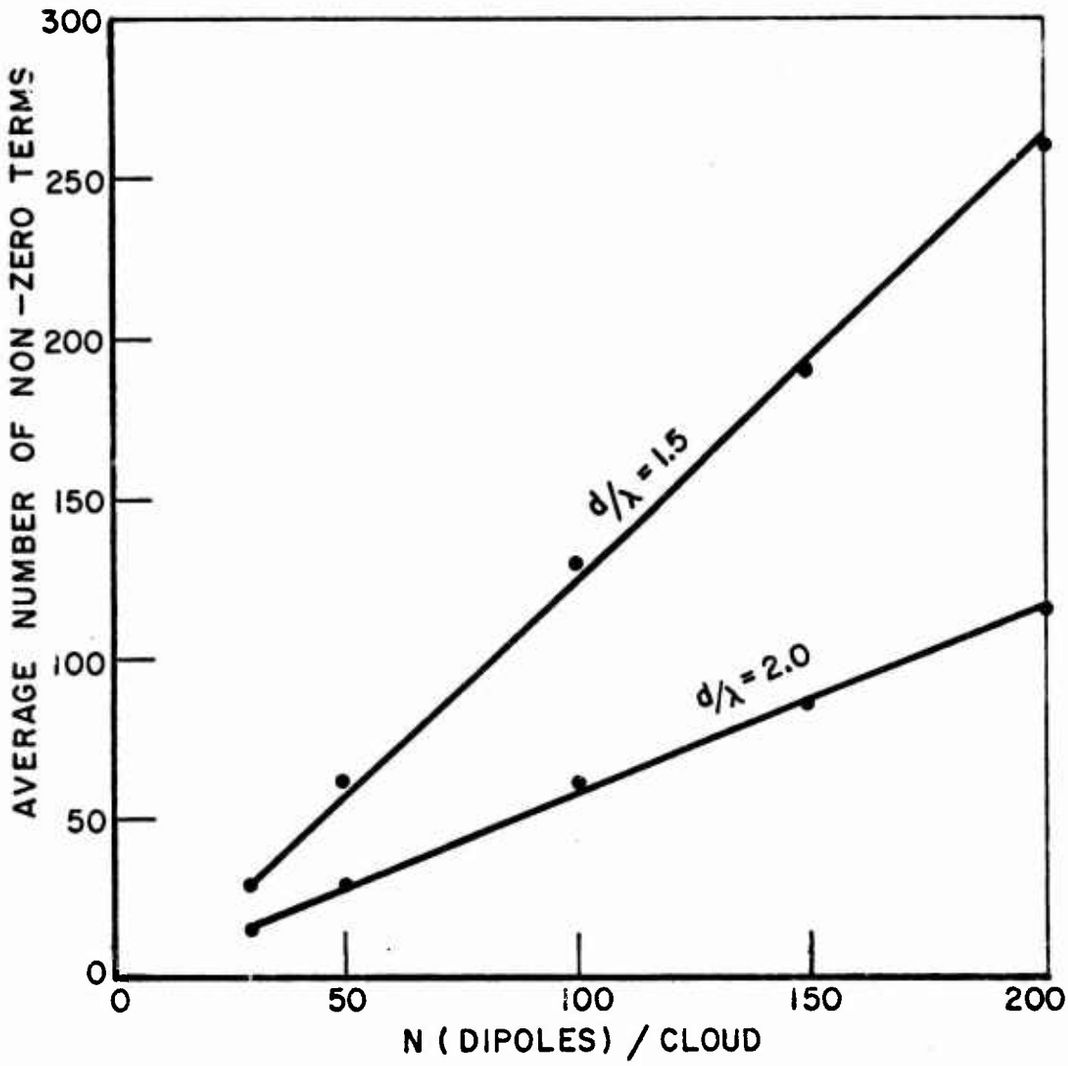


Figure 20. Average number of non-zero terms in the upper triangle of the sparse matrix using 10% rule.

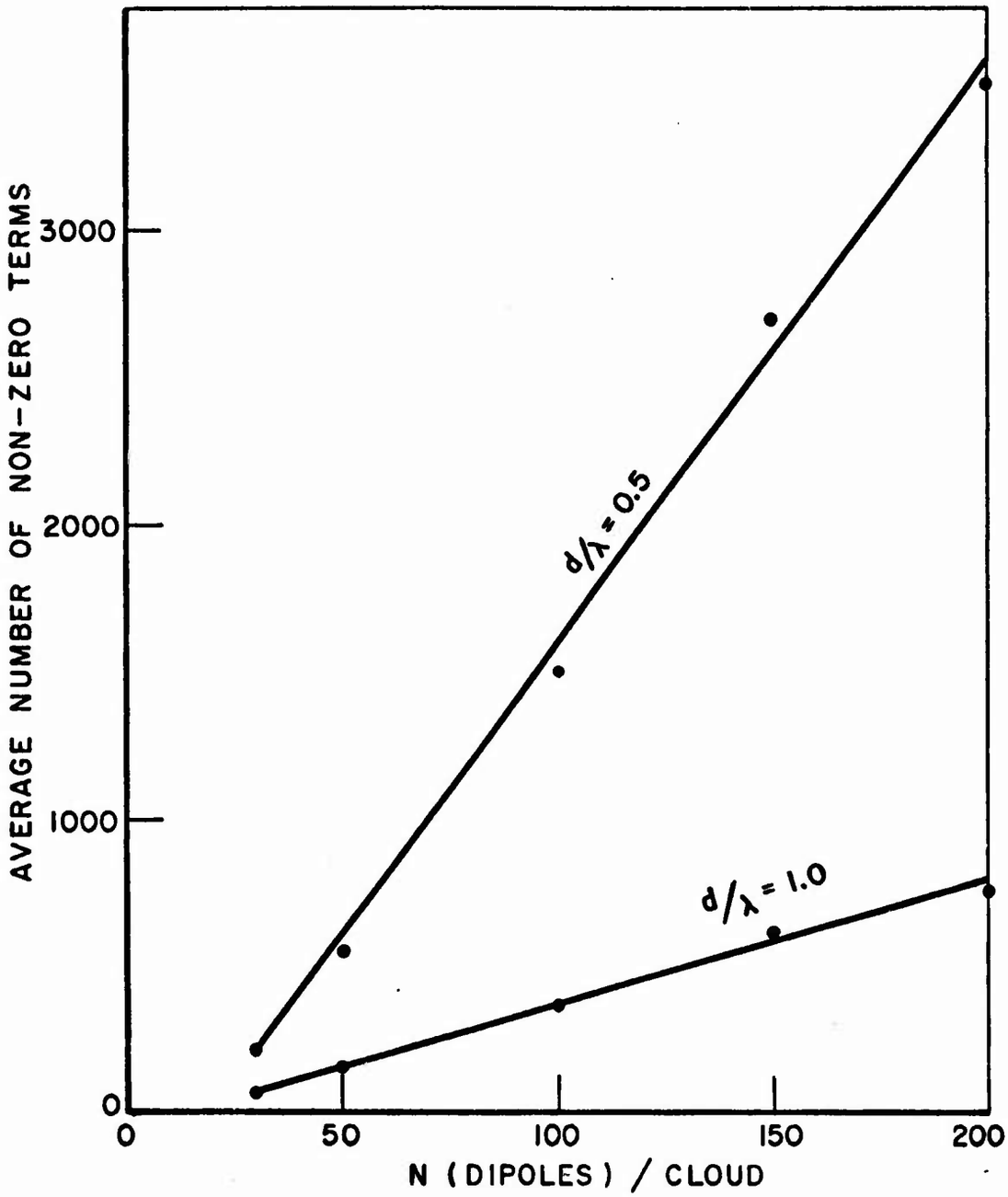


Figure 21. Average number of non-zero terms in the upper triangle of the sparse matrix using 10% rule.

TABLE 2

NUMBER OF NON-ZERO TERMS IN UPPER TRIANGLE

	$\frac{N}{d/\lambda}$	50	100	150	200
Sparse Matrix	2.0	30	62	86	125
	1.5	62	130	190	260
	1.0	160	360	600	750
	0.5	550	1500	2800	3500

TABLE 3

% OF NON-ZERO TERMS IN SPARSE MATRIX UPPER TRIANGLE

$\frac{N}{d/\lambda}$	50	100	150	200
2.0	2.45%	1.25%	0.77%	0.63%
1.5	5.06%	2.62%	1.70%	1.30%
1.0	13.06%	7.27%	5.37%	3.75%
0.5	44.90%	30.30%	25.05%	17.58%

TABLE 4

m, THE NUMBER OF ELEMENTS IN A "SPHERE OF INFLUENCE"

$\frac{N}{d/\lambda}$	50	100	150	200
2.0	0.6	0.62	0.573	0.625
1.5	1.24	1.3	1.26	1.3
1.0	3.2	3.6	4.0	3.75
0.5	11.0	15.0	18.7	17.5

influence" are found to be 2.35, 2.18, 2.07, and 1.73 for spacings d/λ of 2.0, 1.5, 1.0 and 0.5, respectively. Although more data would be necessary to substantiate it, this variation in $R_{d/\lambda}/\lambda$ appears to be a linear increase with d/λ, as shown in Fig. 22. The fact that the "radius of influence", $R_{d/\lambda}/\lambda$, decreases as the cloud becomes more dense, i.e., as d/λ decreases, could be explained by the increased shielding effect of the outermost elements from the center dipole of interest by those elements in-between. And the fact that the values of $R_{d/\lambda}/\lambda$ exceed 2.0 for the larger spacings lends credence to our present analysis because previous data showed the dipoles to be essentially decoupled for these larger spacings.

All the foregoing work is based upon the 10% threshold level below which a matrix element is regarded as zero. The question arises, how severely does this change the scattering cross section and, in particular, the spatial average backscatter from that which would be obtained using the full matrix? To show the effect of sparsing the impedance matrix we present in Figs. 23-26 backscattering patterns (same sense polarizations of transmitter and receiver for clouds containing N = 30 dipoles with two different average spacings, d/λ = 0.5 and 2.0, calculated on the ElectroScience Laboratory computer using the full matrix and the sparse matrix (with 10% sparsing rule). We expect that the sparsed matrices for these clouds contain about 95% zeros when d/λ = 2.0 and about 50% zeros when d/λ = 0.5. Of course, as N increases, these percentages will increase. A similar set of calculations were performed on the Wright-Patterson Air Force Base CDC-6600 computer for three different clouds containing N=200 dipoles, each 0.475 wavelengths long, and with average spacing, d/λ=2.0. Figures 27-29 compare superimposed backscattering patterns (same sense polarizations and cross polarizations of transmitter and receiver) using the full matrix and sparse matrix (with 10% sparsing rule). Figures 30-41 show similar patterns for two other clouds with N = 200, d/λ = 2.0. We expect that the sparsed matrices for these clouds contain about 99.4% zeros (see Table 3).

All these patterns, particularly those for the N = 200 clouds, are interesting because they display three features worth mentioning. First, the patterns show differences in fine structure but are very similar in gross structure in all cases. Second, Figs. 25-41, all for average spacings d/λ = 2.0, show a recognizable repetition of the pattern every 180°, i.e., the backscattering pattern behaves about the same when the cloud is viewed from a selected direction or from the opposite to that direction. Furthermore, the patterns corresponding to the sparsed matrix show this symmetry even more than do those for the full matrix. This behavior is expected because in all these

cases, the clouds are tenuous enough (i.e., dipoles are weakly coupled) and do not contain sufficient numbers of dipoles to exhibit significant extinction of energy from front to back of the clouds. If all dipoles were of resonant length and were perfectly decoupled, we would observe perfect symmetry of the patterns; our dipoles are of resonant length (making each one essentially a single mode structure with a 180° phase shift upon reflection from it, i.e., all diagonal elements of the Z matrix are essentially pure real), but they are not decoupled, upsetting the symmetry somewhat. Sparsing artificially decouples many elements (95%, 99.4% as mentioned earlier), so we expect the sparsed results to closer approach the ideal, i.e., display more symmetric patterns than do the full matrix patterns. Notice that for the denser clouds, Figs. 23 and 24, where $d/\lambda = 0.5$, pattern symmetry disappears for full or sparse matrix solutions. Here, the strong coupling definitely upsets the symmetry and even the artifice of decoupling with a 10% rule does not decouple enough elements (only about 50% as mentioned above) to regain symmetry. A third feature, not directly observable from Figs. 25-41 but derivable from them, is the effect of sparsing upon the spatial average backscatter. Figure 42 presents bar graphs of average backscatter obtained from each of 10 different clouds with $N = 30$, $d/\lambda = 2.0$, each calculated using full matrices and matrices sparsed by the 10% rule. Clearly, the average backscatter, even with the full matrix, varies from cloud to cloud, as expected from results presented earlier, but the error incurred by using the sparse matrix is less than this variance, and results in a value for average backscatter which is slightly too high in most cases by a few percent. That it is too high and not too low is expected because sparsing results in a cloud which closer approaches the ideal decoupled cloud and our results have shown that coupling lowers the average echo below that for the ideal. Another mode of presenting the same effect of sparsing on spatial average backscatter is shown in Figs. 43-45. For the three clouds containing $N = 200$ dipoles, the cumulative probabilities $P(\sigma/\lambda^2)$ of backscattering cross section were calculated. The solid line in each figure is associated with the sparse matrix, the dots with the full matrix, and the crosses with the ideal decoupled case (calculated from $P(\sigma/\lambda^2) = 1 - e^{-\sigma/30}$; see Appendix I). The spatial averages associated with the three algorithms are indicated by the vertical lines. Notice that all three mathematical algorithms infer that the backscattering cross section exceeds the average cross section approximately 40% of the time.

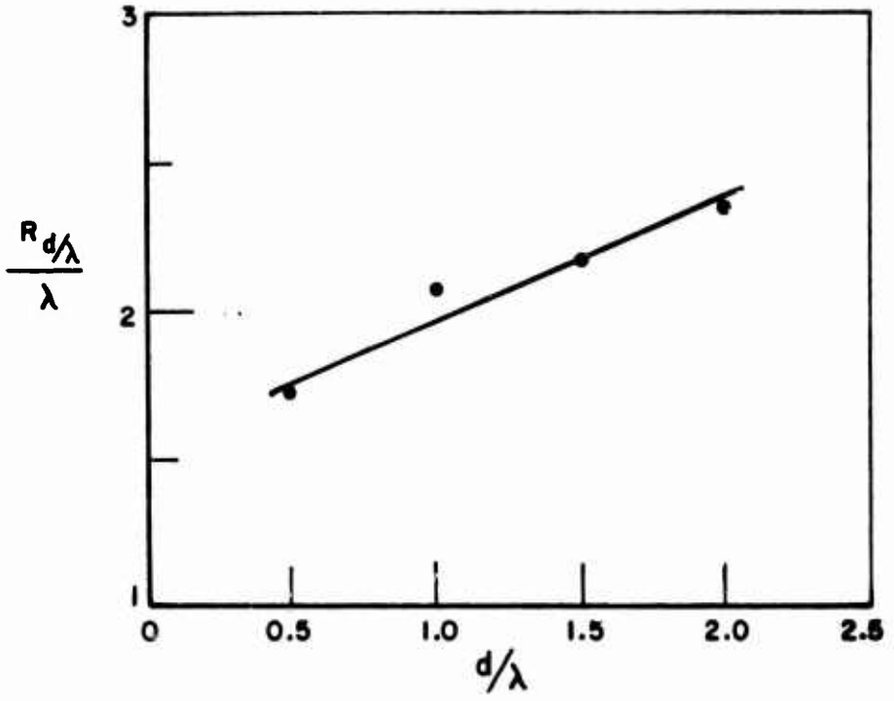


Figure 22. Radius of "sphere of influence" vs average dipole spacing.

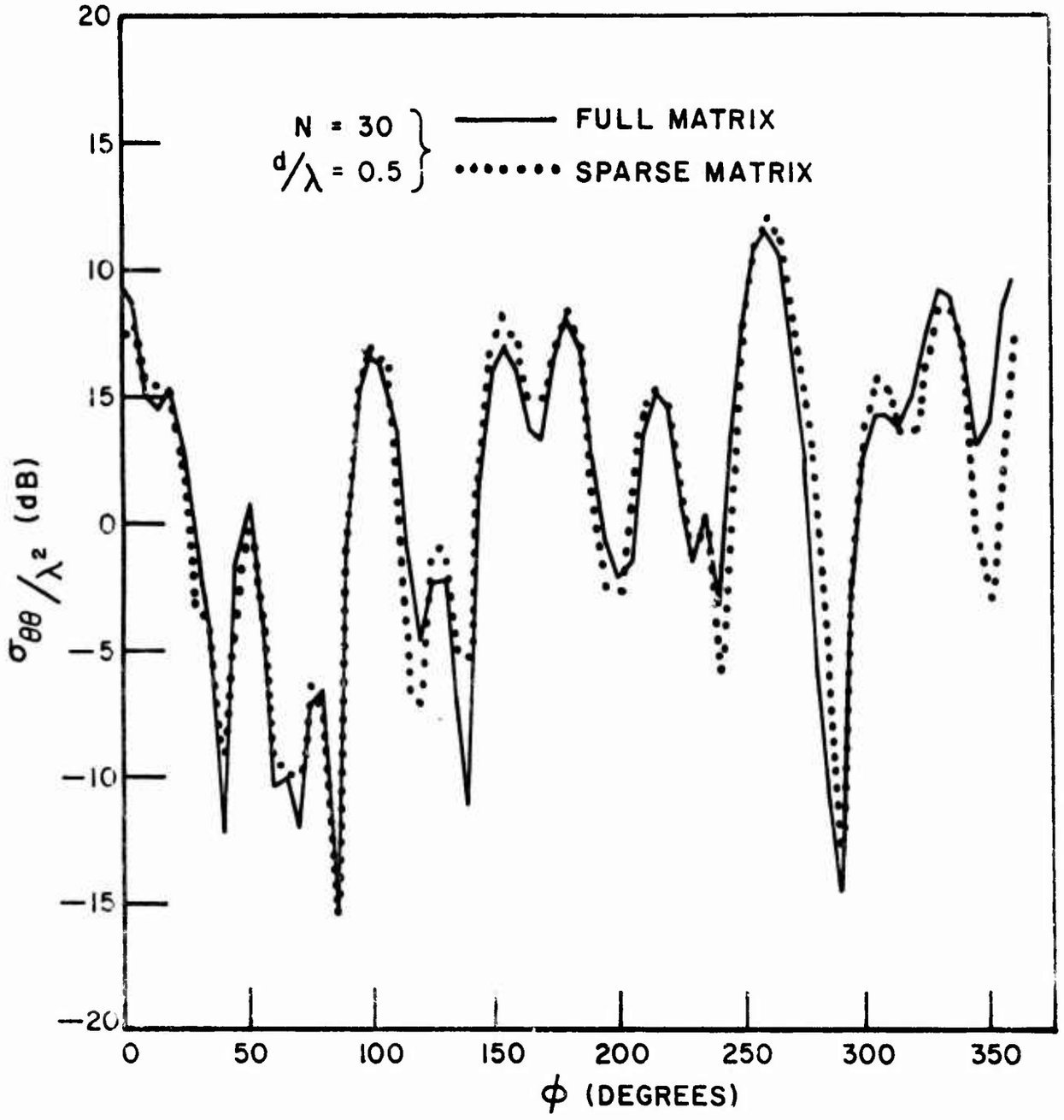


Figure 23. θ -θ backscattering patterns as calculated using the full and sparse matrix, cloud #1, d/λ = 0.5.

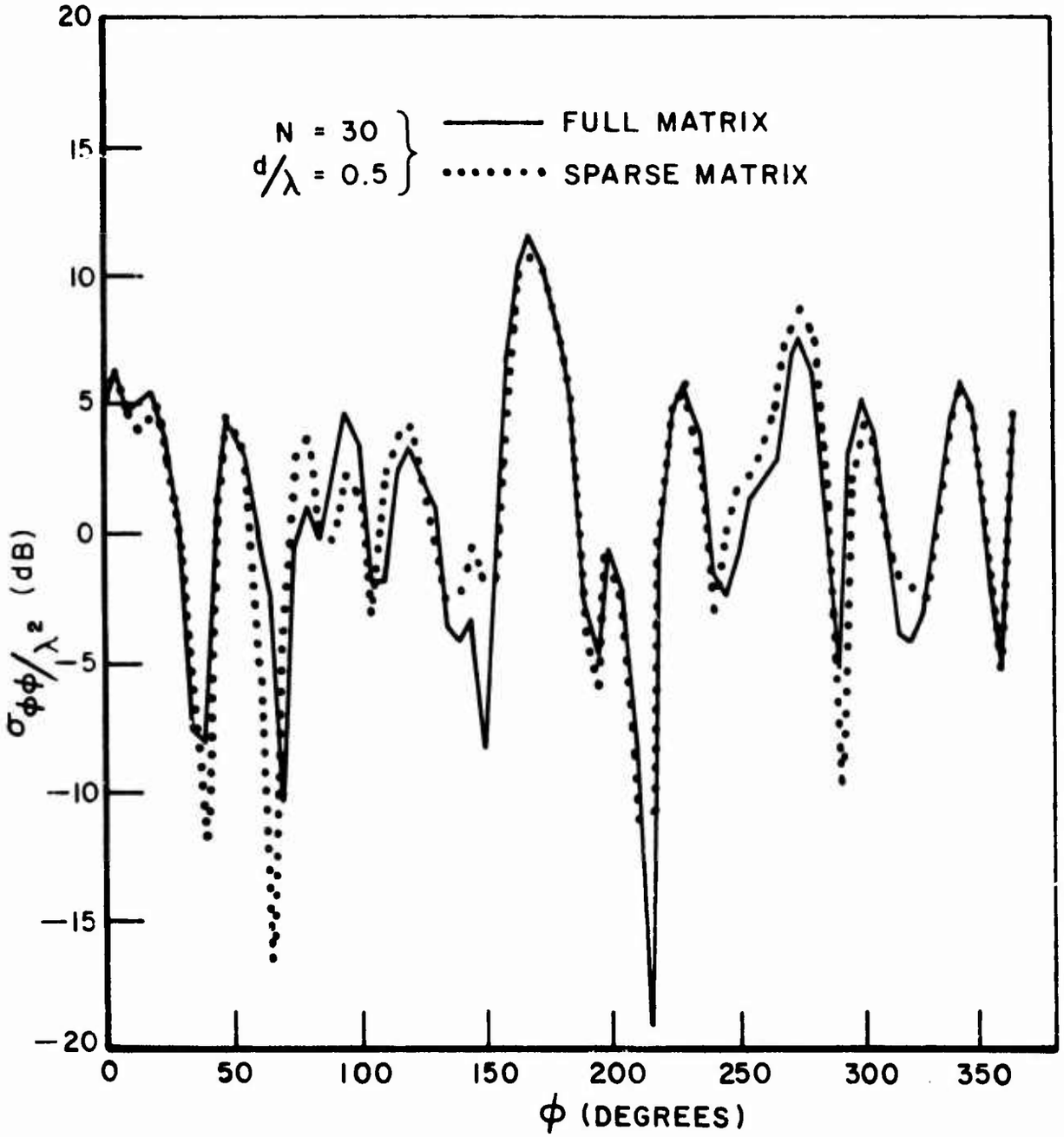


Figure 24. ϕ-ϕ backscattering patterns as calculated using the full and sparse matrix, cloud #1, $d/\lambda = 0.5$.

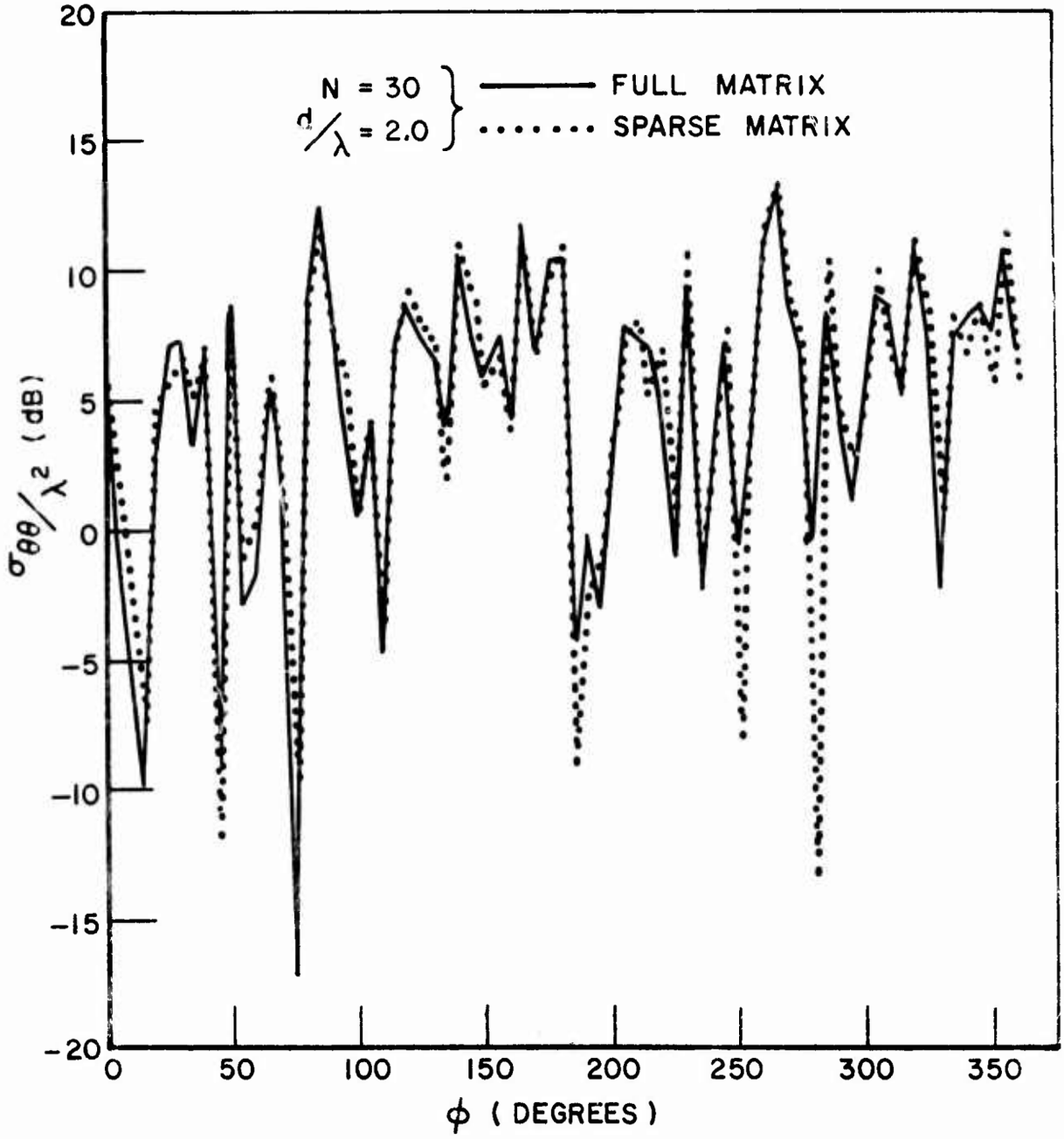


Figure 25. θ-θ backscattering patterns as calculated using the full and sparse matrix, cloud #2, $d/\lambda = 2.0$.

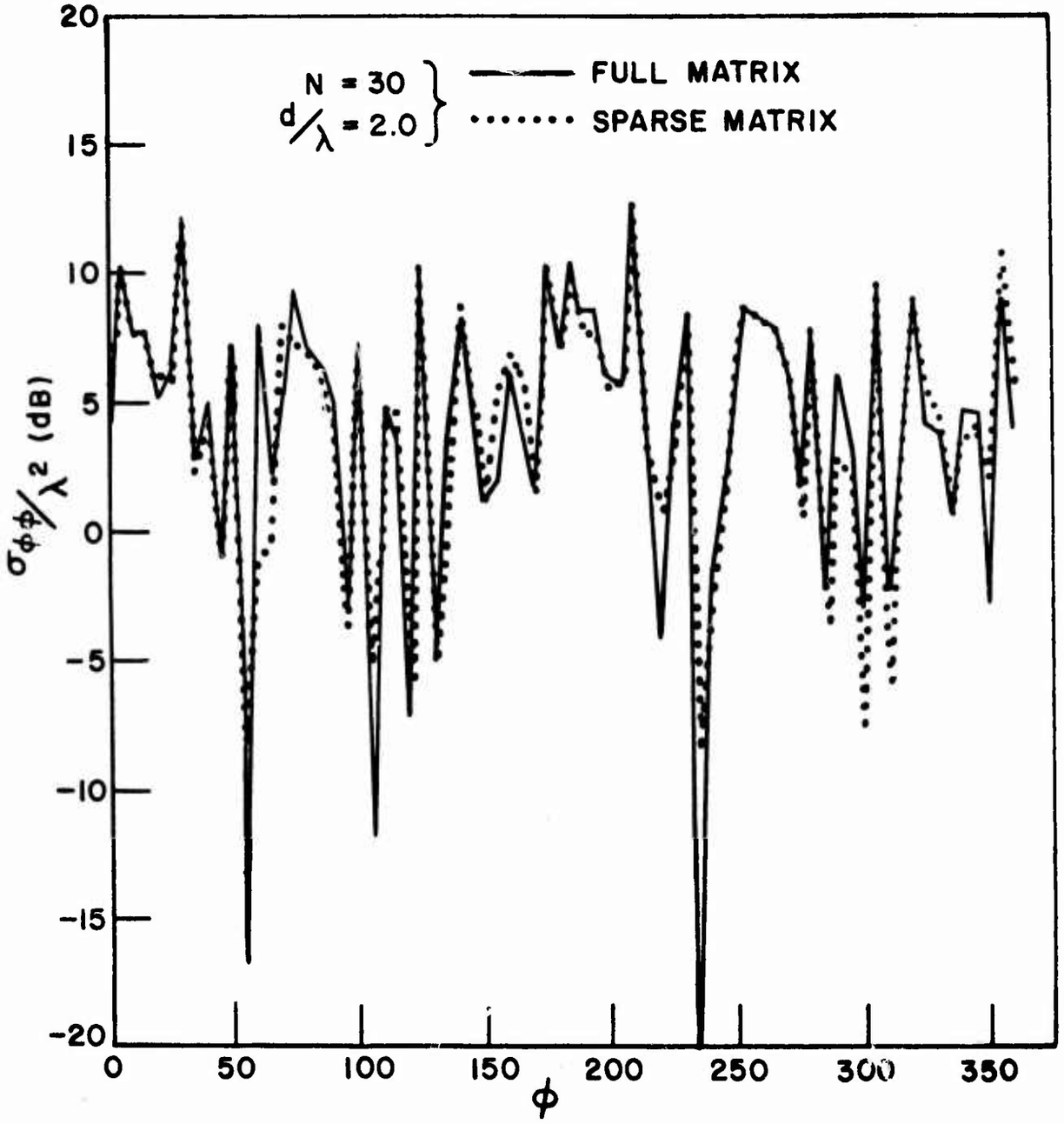


Figure 26. ϕ-ϕ backscattering patterns as calculated using the full and sparse matrix, cloud #2, $d/\lambda = 2.0$.

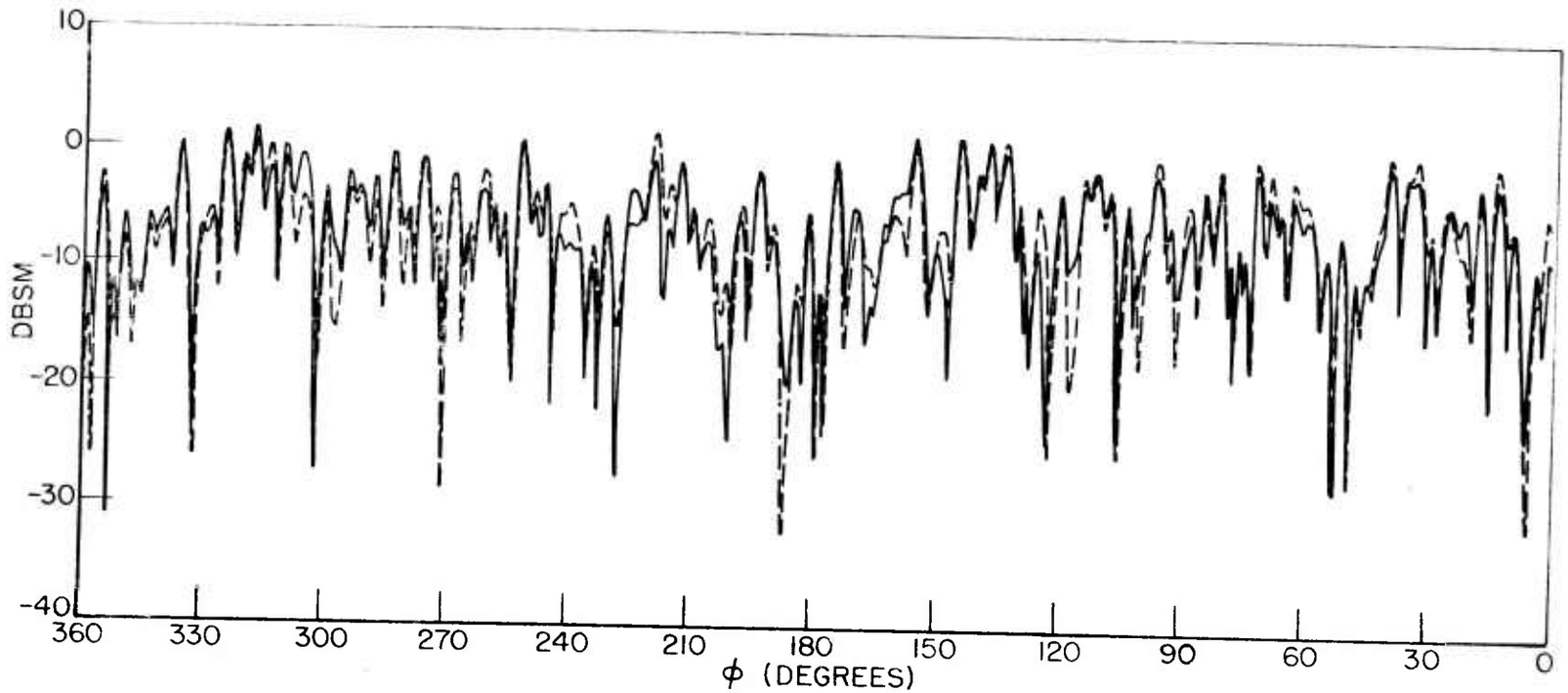


Figure 27. θ-θ backscattering patterns as calculated using the full and sparse matrix, cloud #1, N = 200.

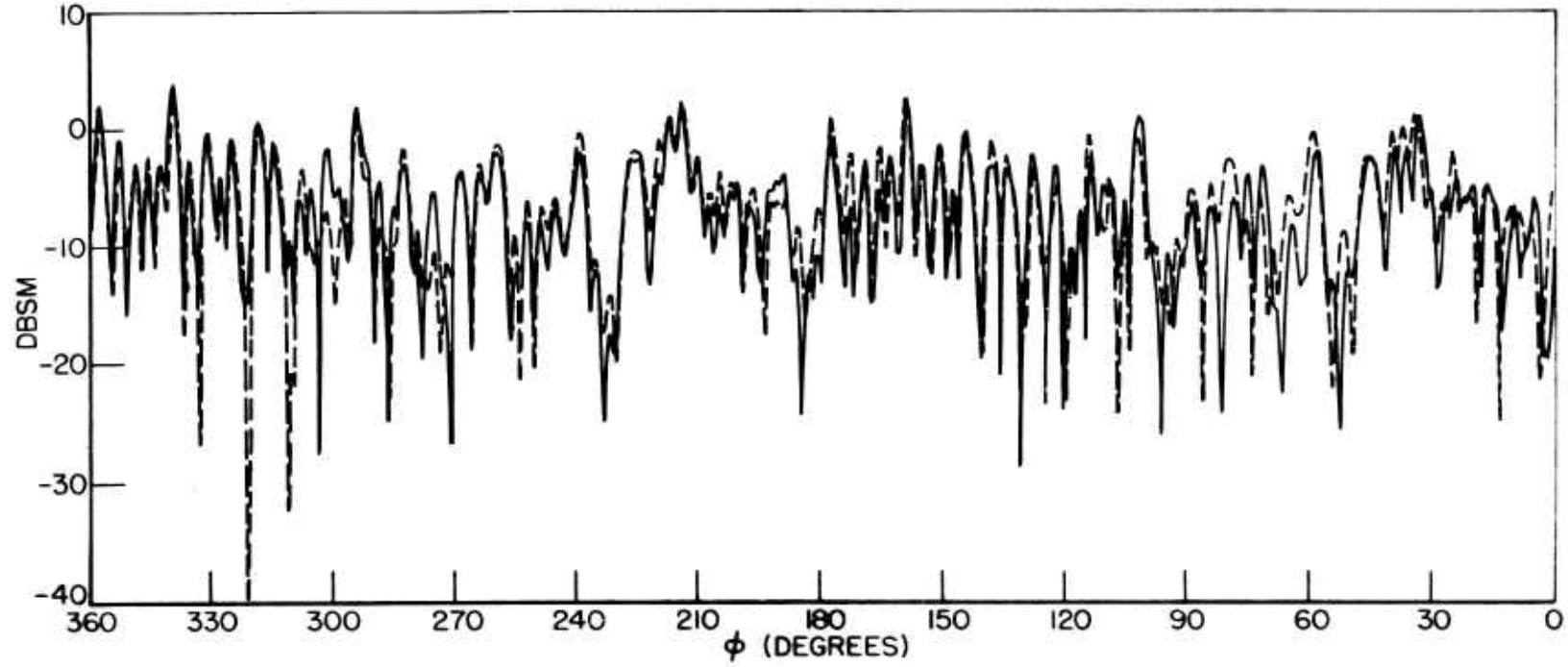


Figure 28. ϕ-ϕ backscattering patterns as calculated using the full and sparse matrix, cloud #1, N = 200.

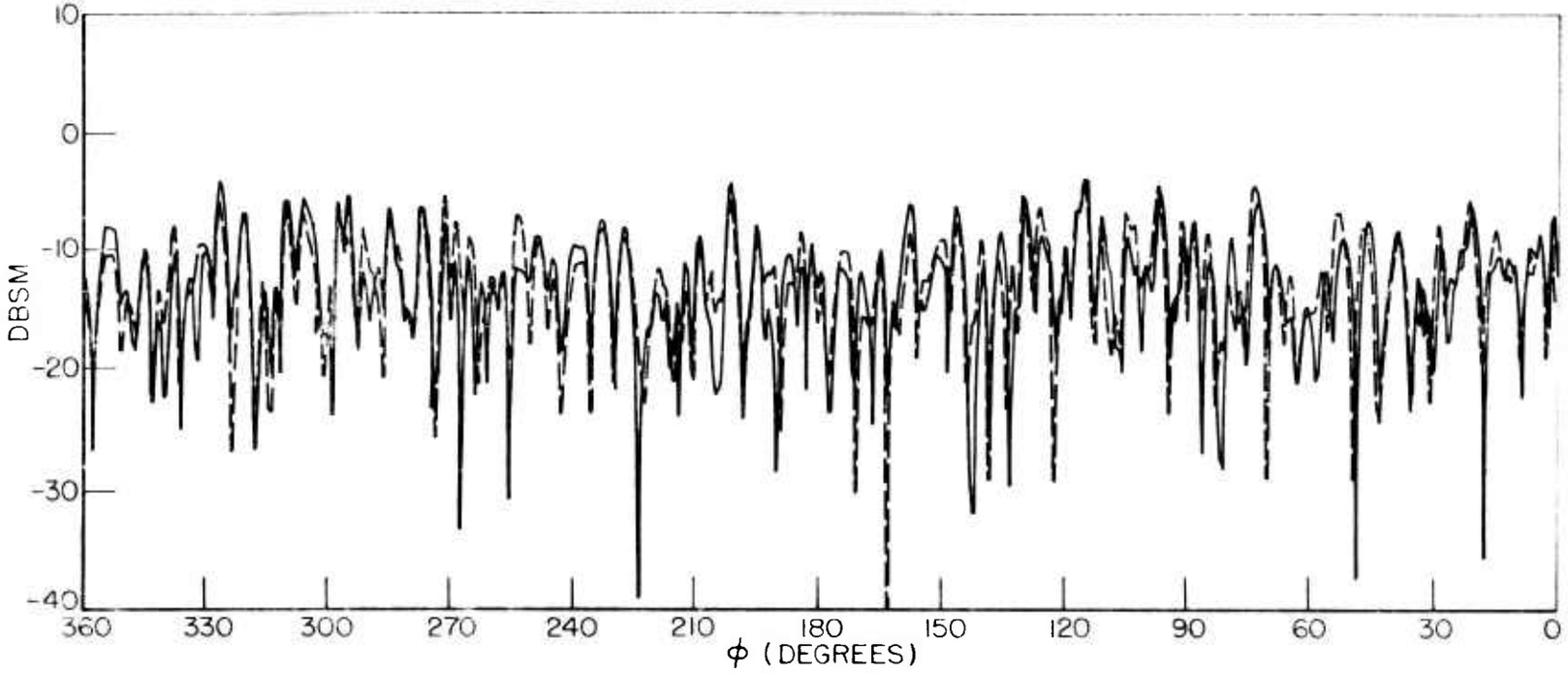


Figure 29. θ-ϕ backscattering patterns as calculated using the full and sparse matrix, cloud #1, N = 200.

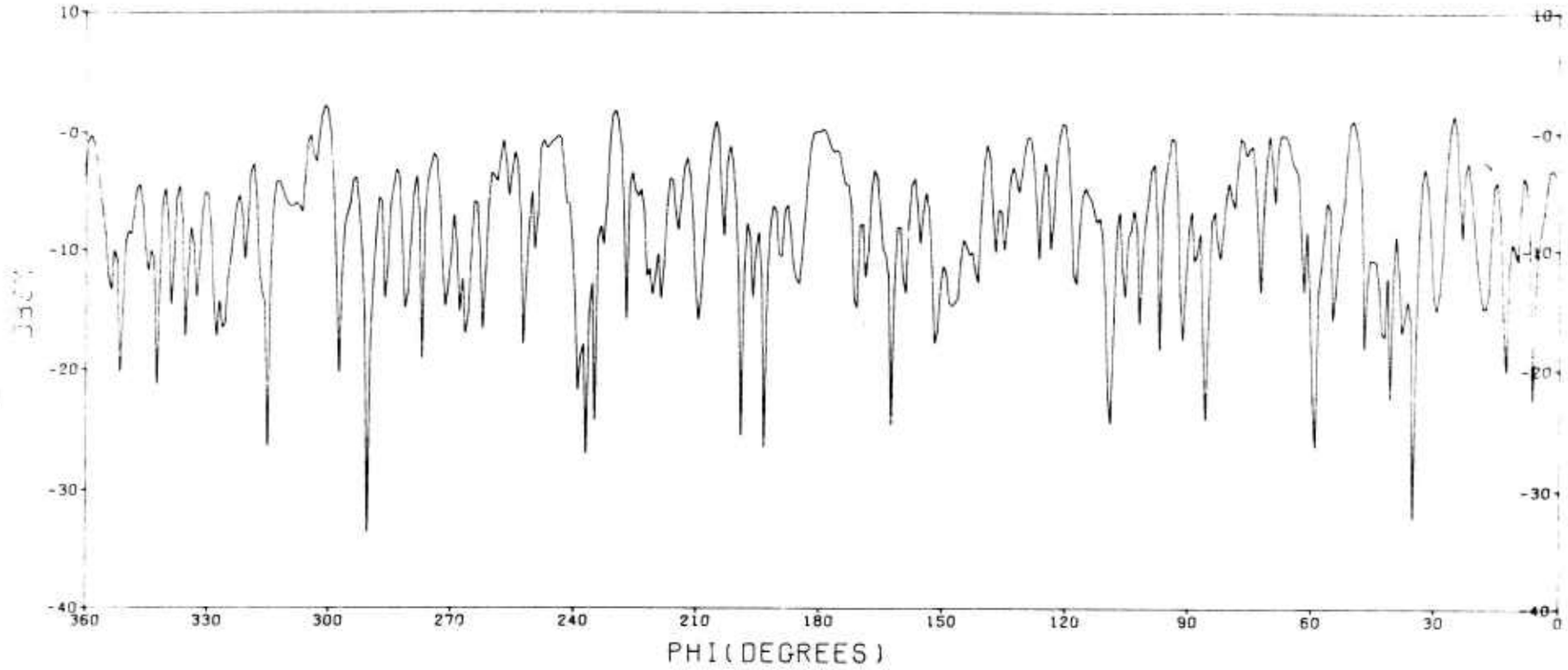


Figure 30. θ-θ backscattering patterns as calculated using tne full matrix, cloud #2, N = 200, $d/\lambda = 2.0$.

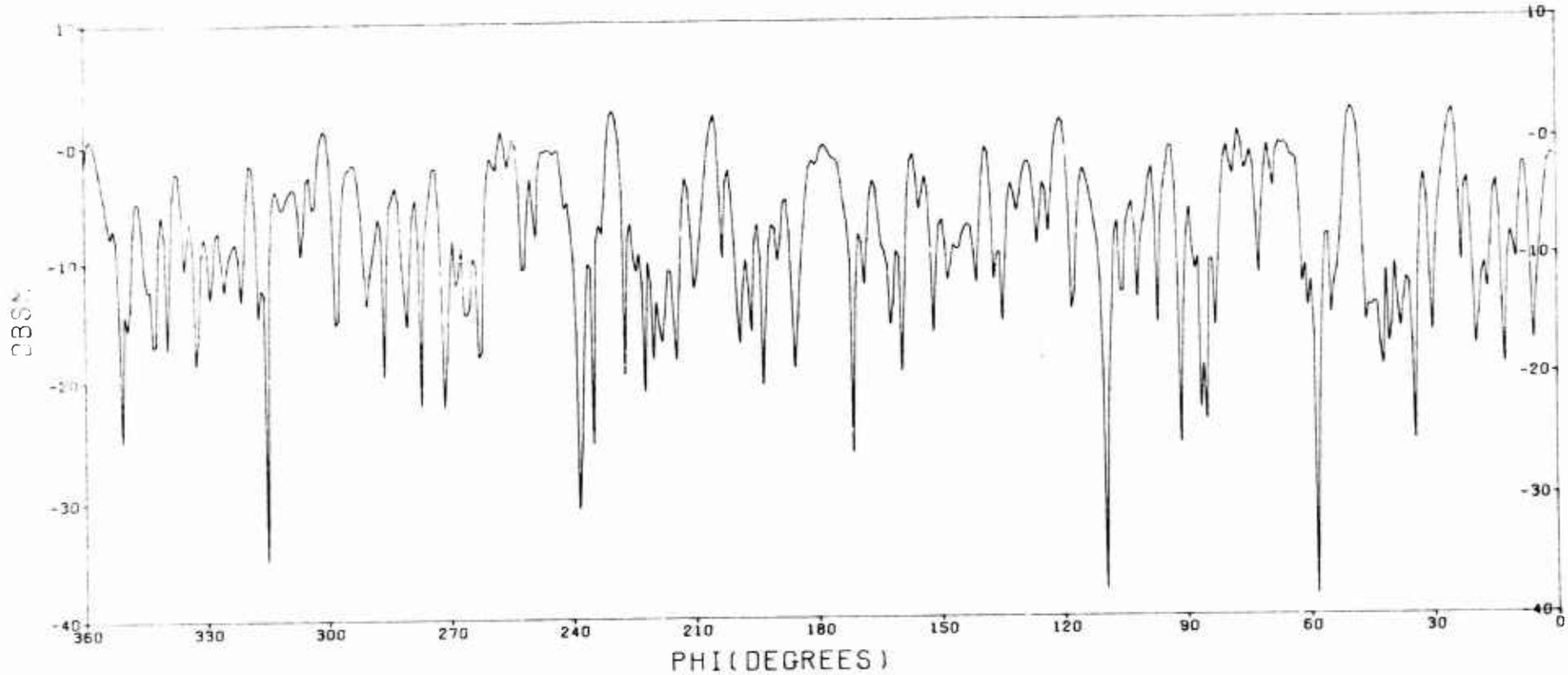


Figure 31. θ-θ backscattering patterns as calculated using the sparse matrix, cloud #2, N = 200, $d/\lambda = 2.0$.

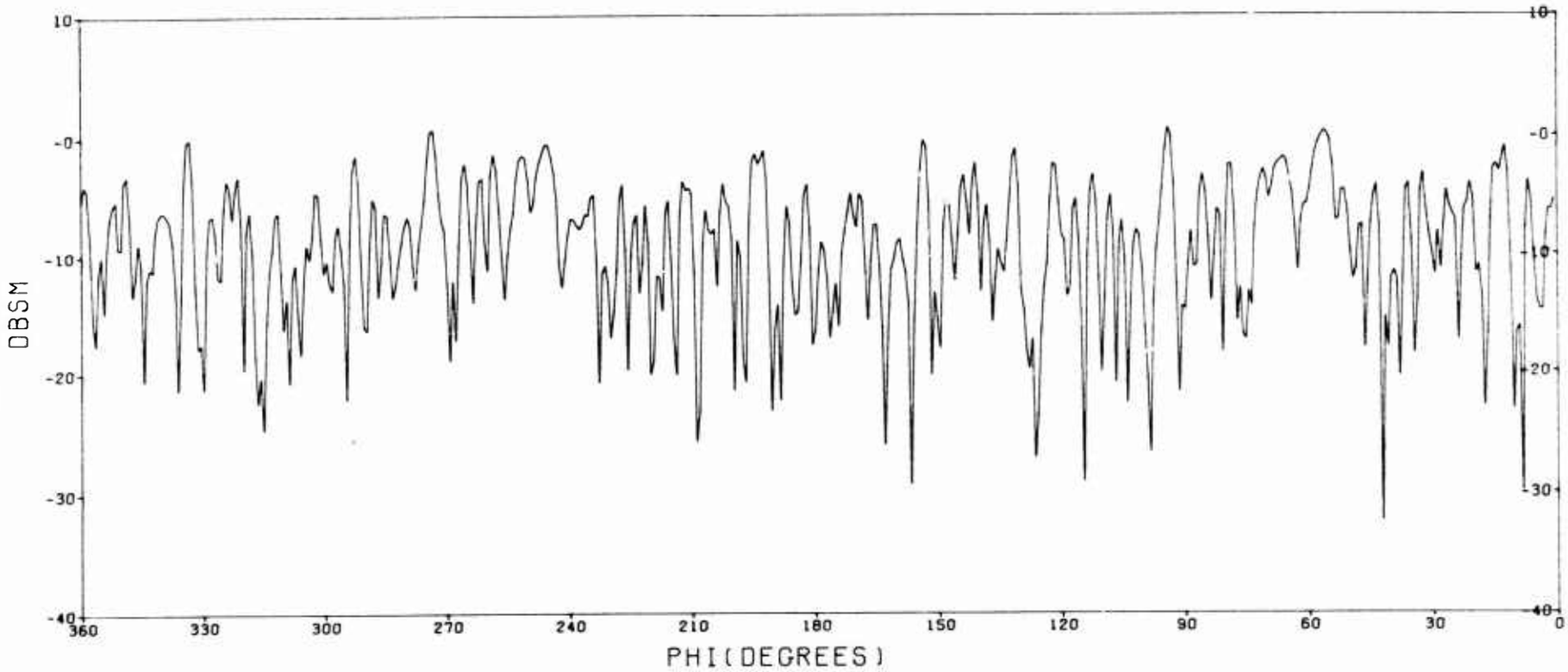


Figure 32. ϕ-ϕ backscattering patterns as calculated using the full matrix, cloud #2, N = 200, c/λ = 2.0.

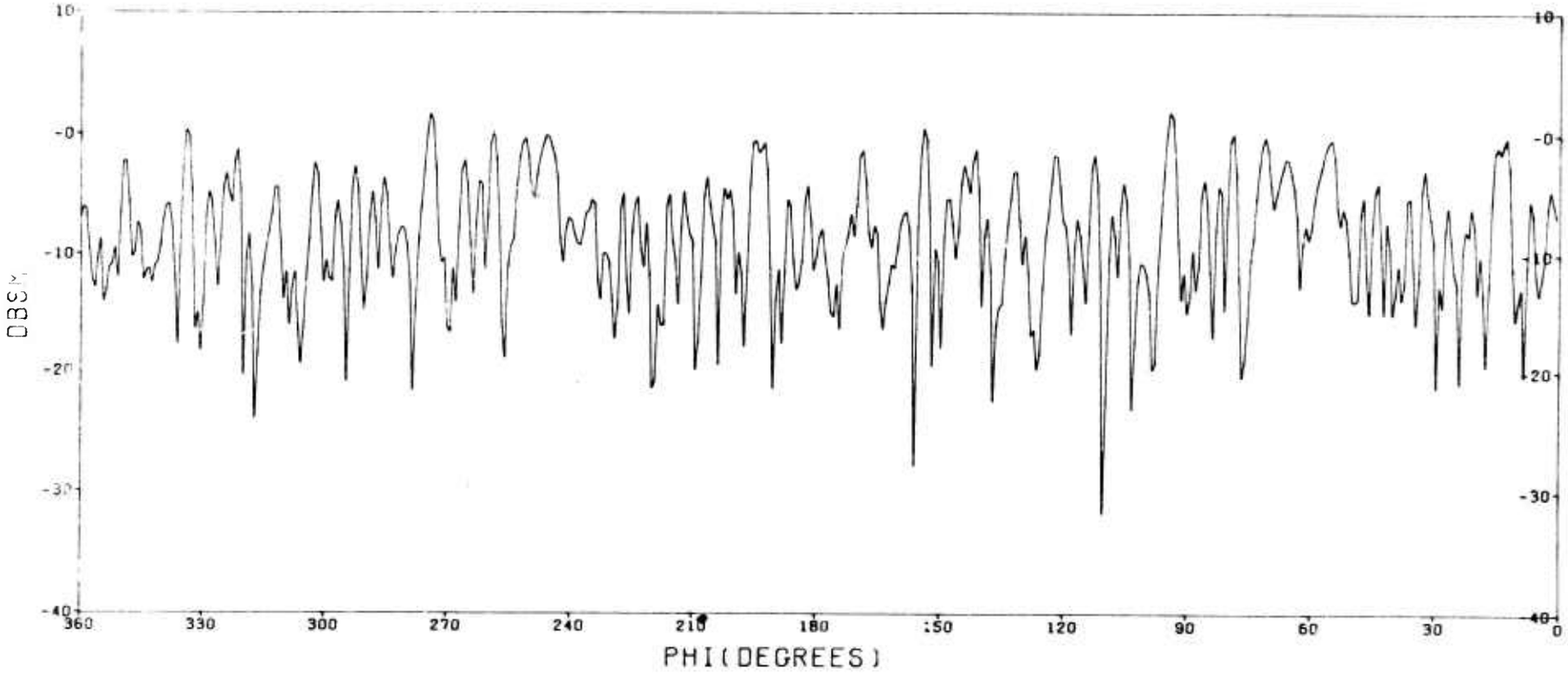


Figure 33. ϕ-ϕ backscattering patterns as calculated using the sparse matrix, cloud #2, N = 200, d/λ = 2.0.

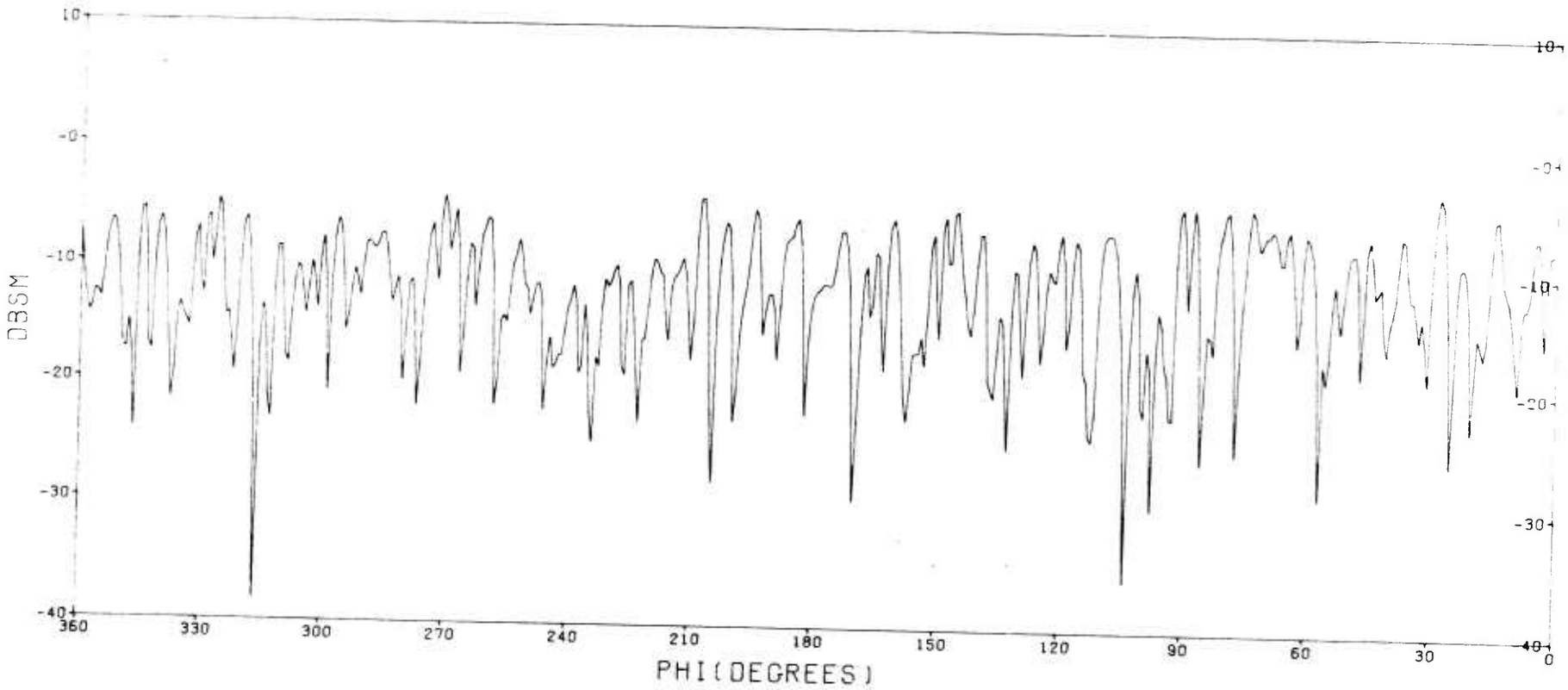


Figure 34. θ-ϕ backscattering patterns as calculated using the full matrix, cloud #2, N = 200, d/λ = 2.0.

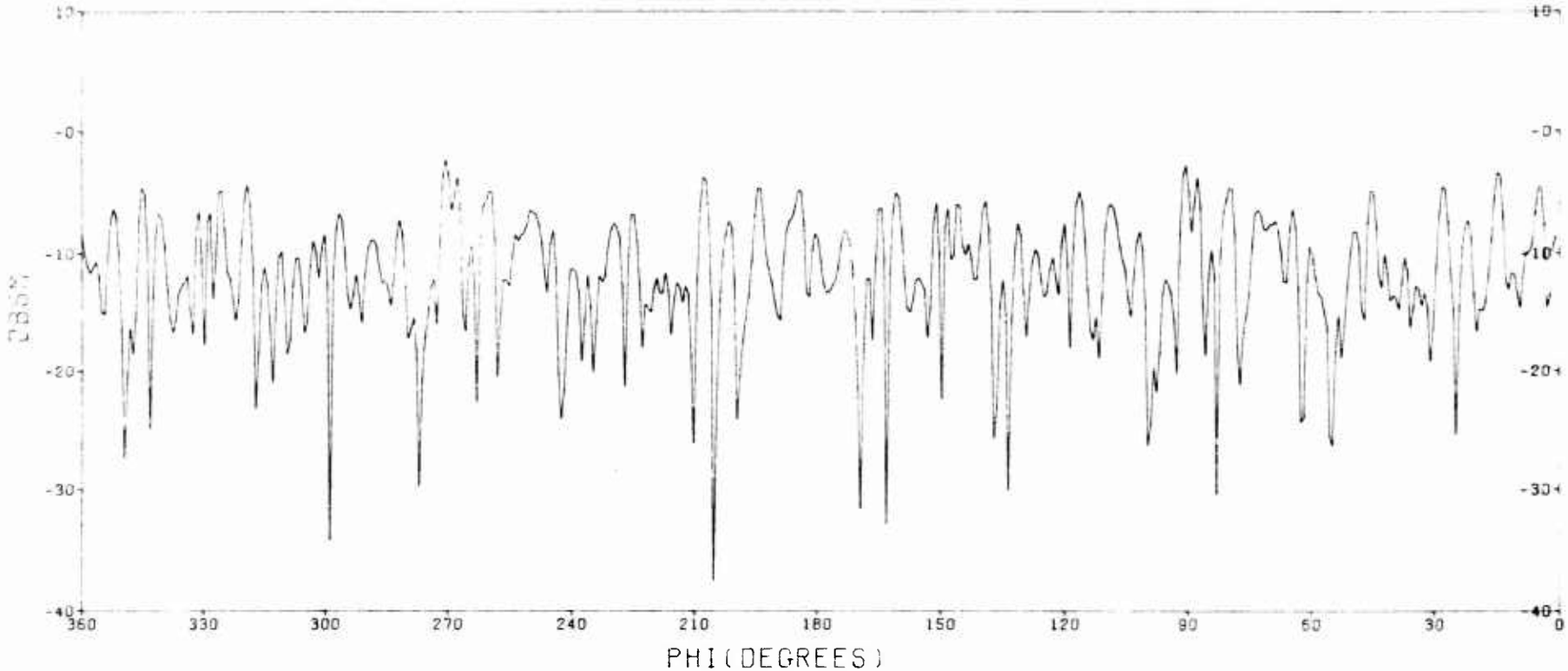


Figure 35. θ-ϕ backscattering patterns as calculated using the sparse matrix, cloud #2, N = 200, d/λ = 2.0.

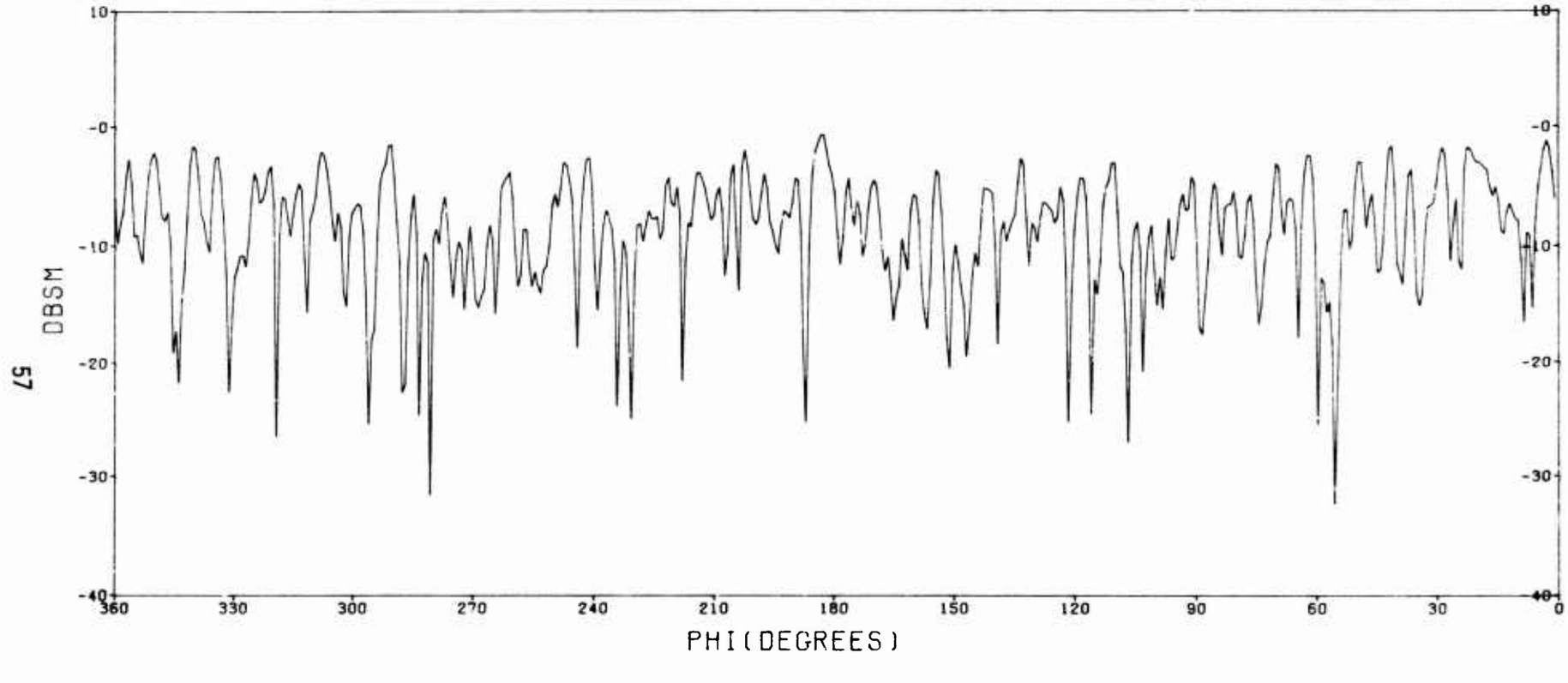


Figure 36. θ-θ backscattering patterns as calculated using the full matrix, cloud #3, N = 200, $d/\lambda = 2.0$.

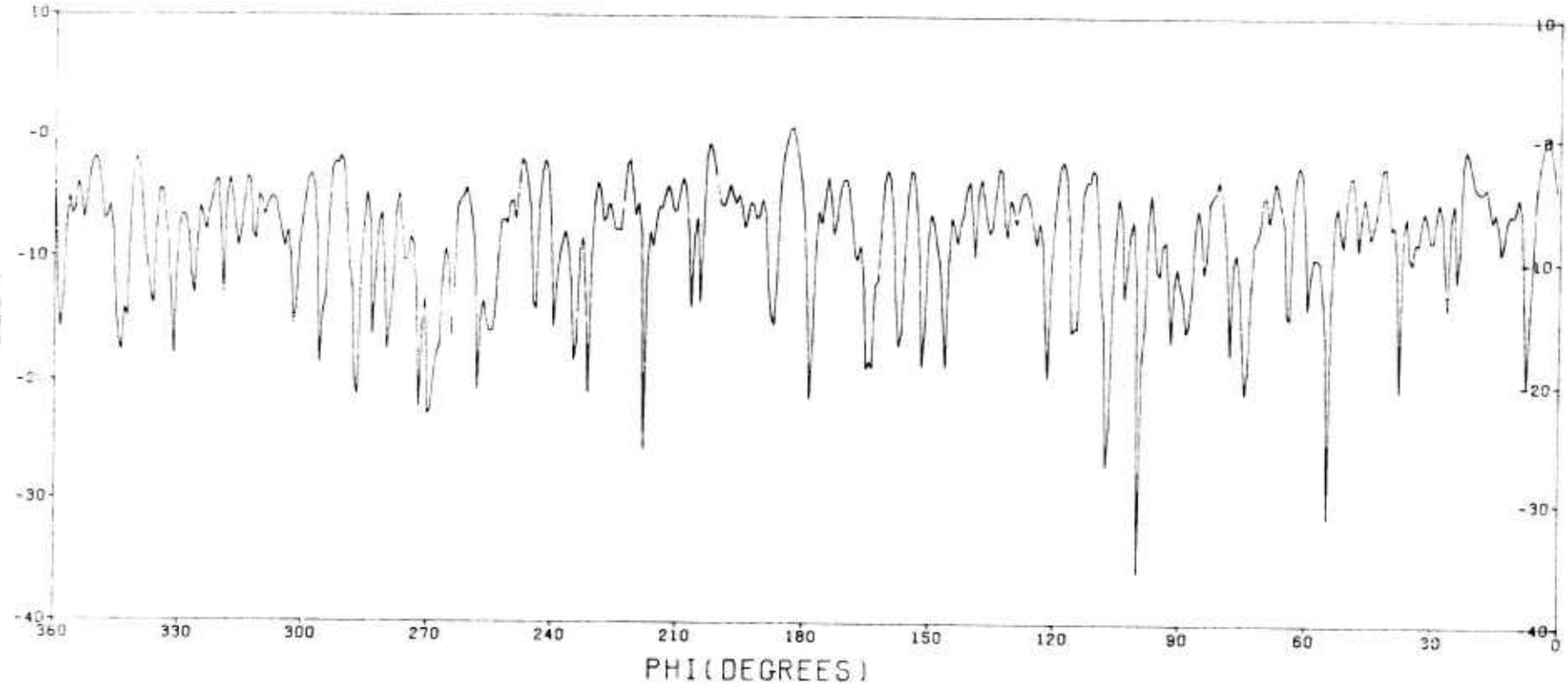


Figure 37. θ-θ backscattering patterns as calculated using the sparse matrix, cloud #3, N = 200, $d/\lambda = 2.0$.

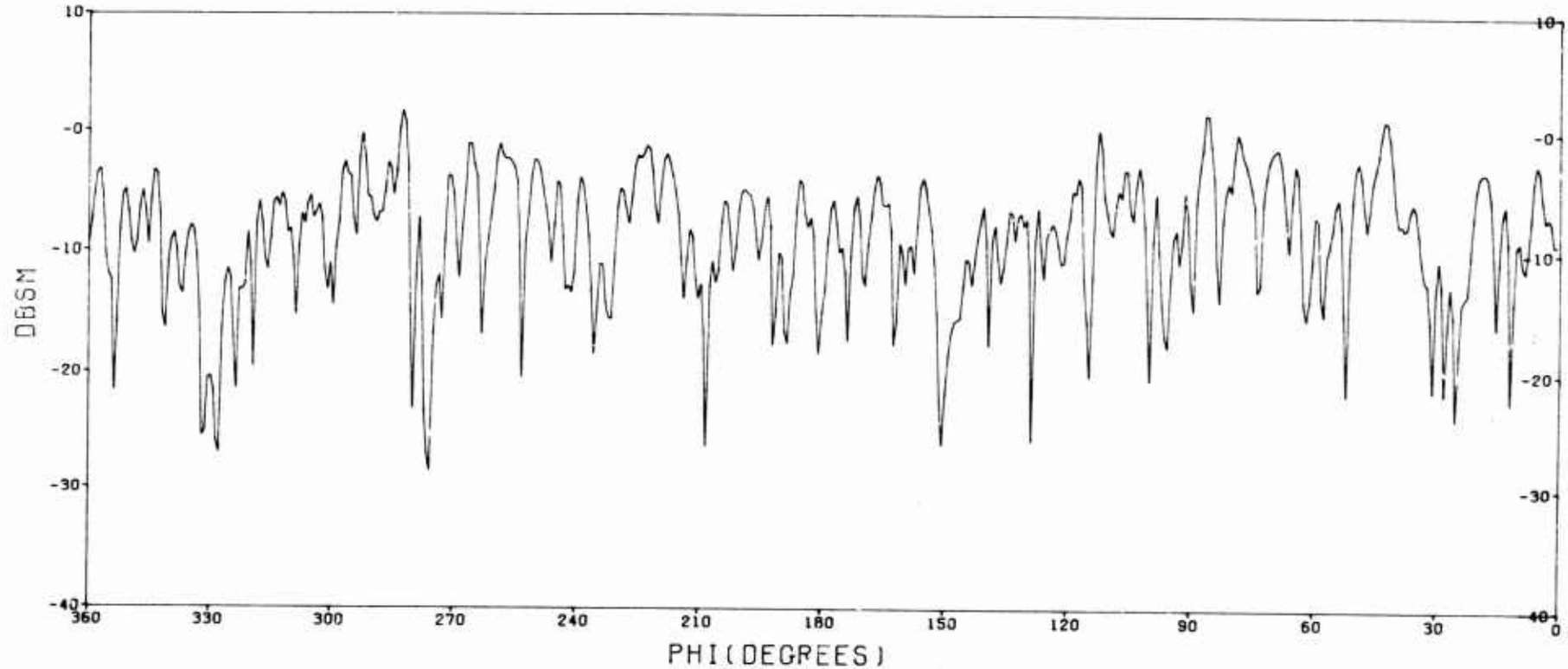


Figure 38. ϕ-ϕ backscattering patterns as calculated using the full matrix, cloud #3, N = 200, d/λ = 2.0.

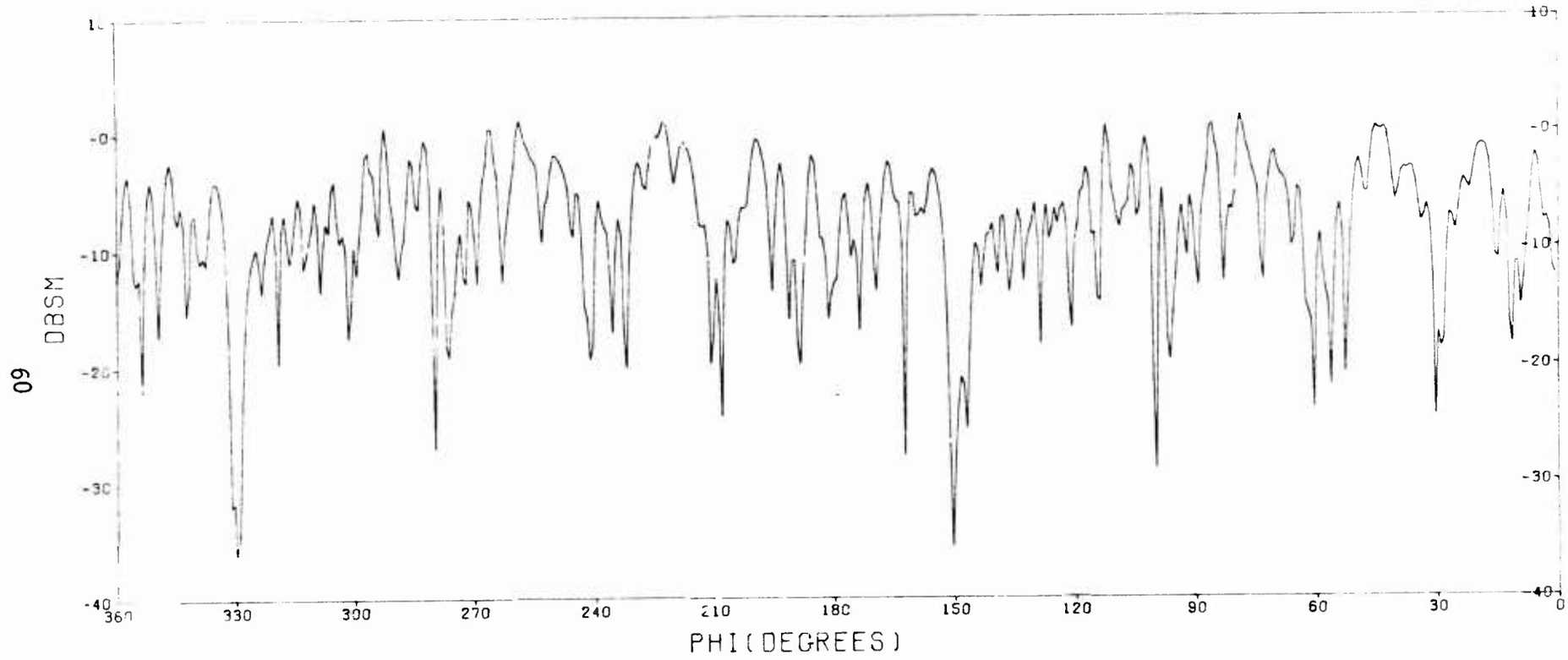


Figure 39. ϕ-ϕ backscattering patterns as calculated using the sparse matrix, cloud #3, N = 200, d/λ = 2.0.

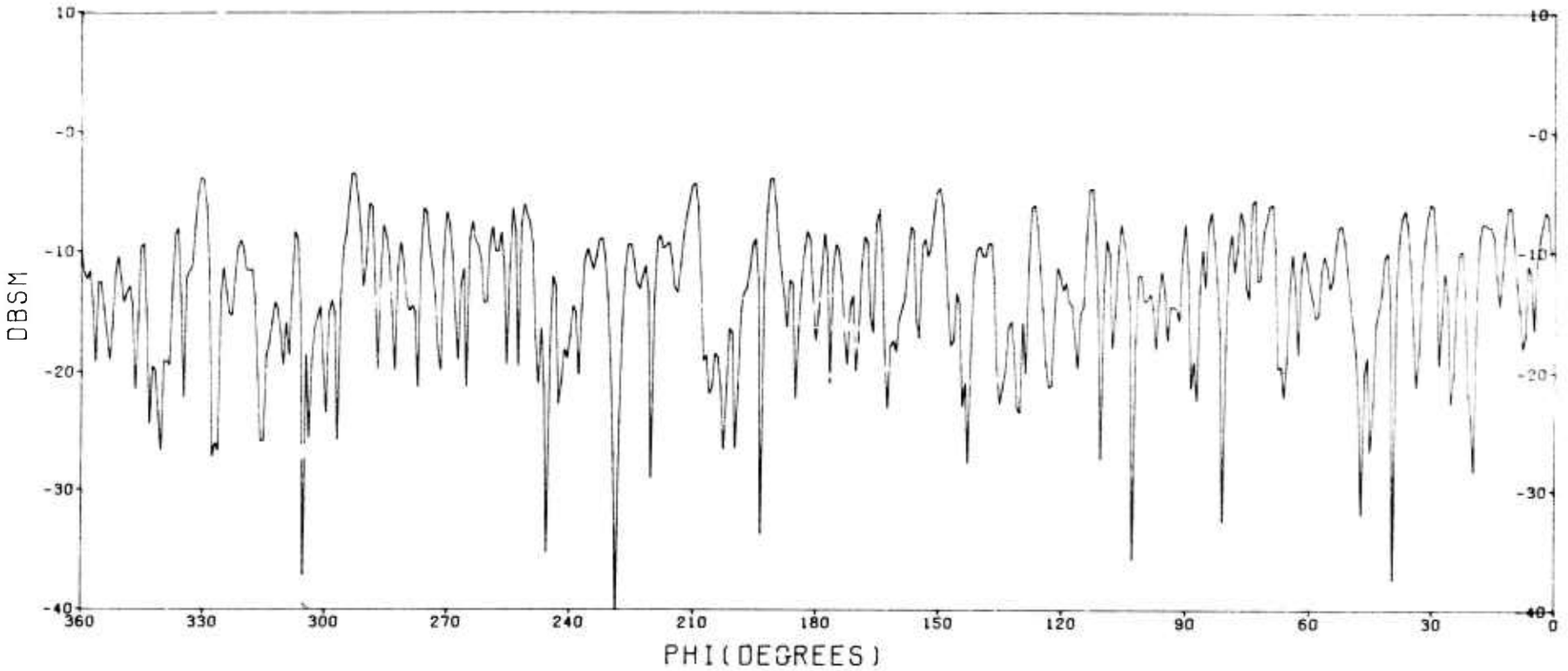


Figure 40. θ-ϕ backscattering patterns as calculated using the full matrix, cloud #3, N = 200, d/λ = 2.0.

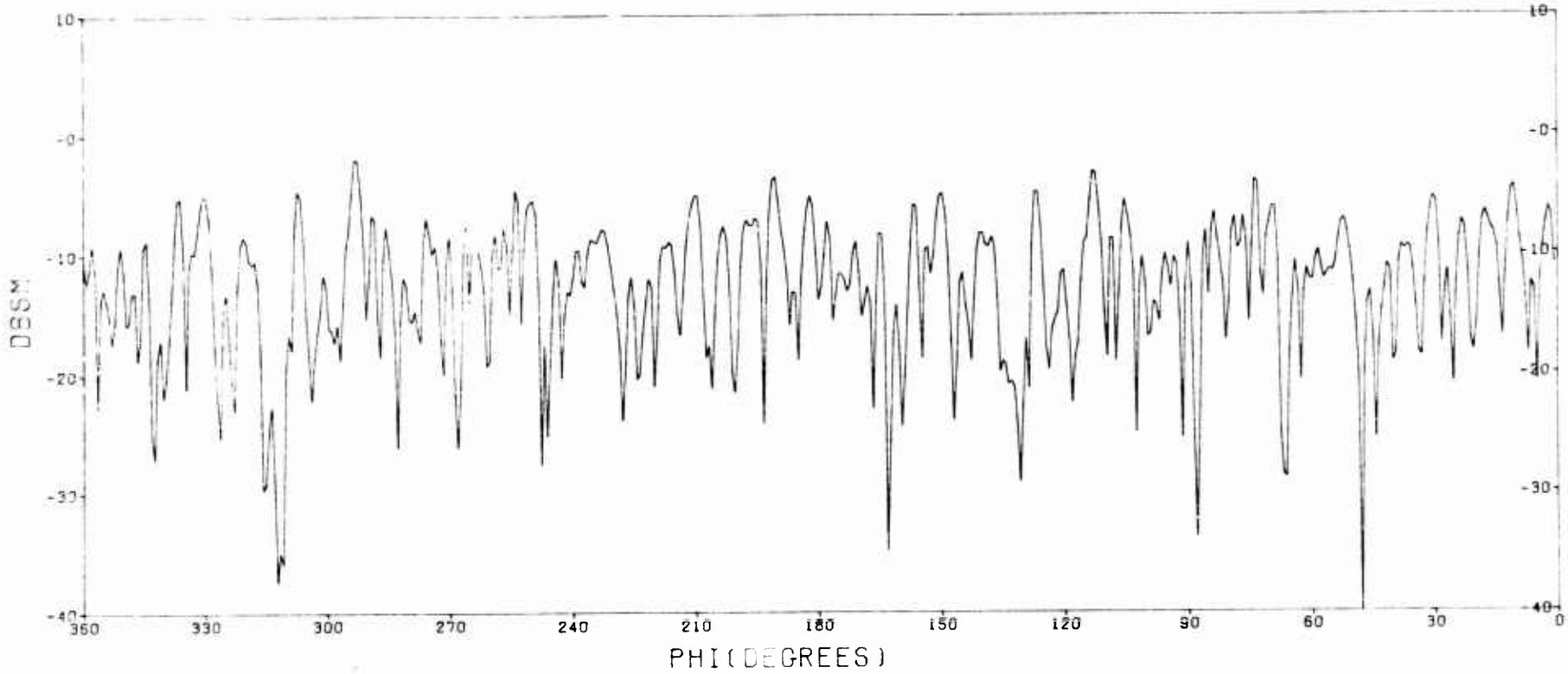


Figure 41. $\sigma_{\phi\phi}$ backscattering patterns as calculated using the sparse matrix, cloud #3, N = 200, d/λ = 2.0.

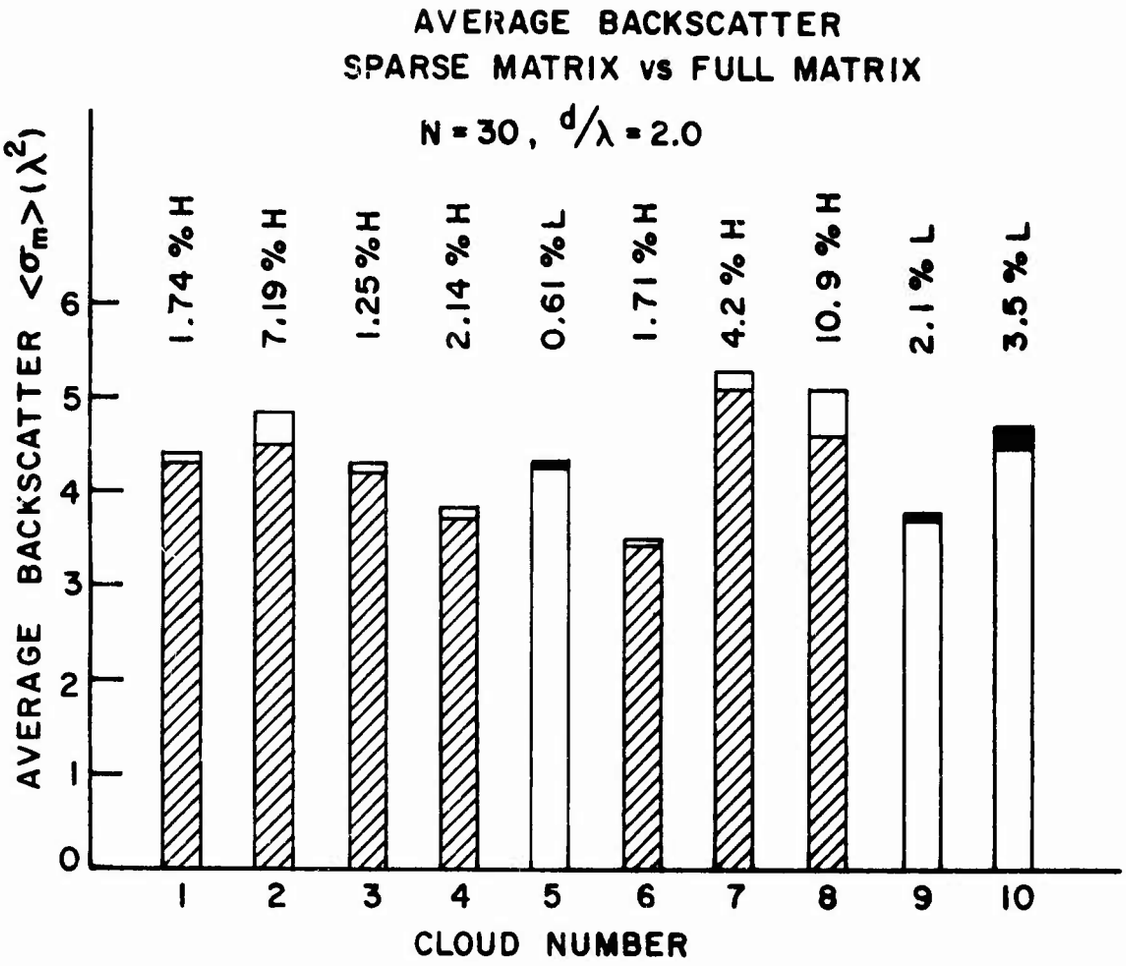


Figure 42. Spatial average backscatter from ten different clouds using full and sparse matrices; the symbolism L or H indicates that the sparse matrix result was lower or higher, respectively, than the full matrix result by the indicated percentage.

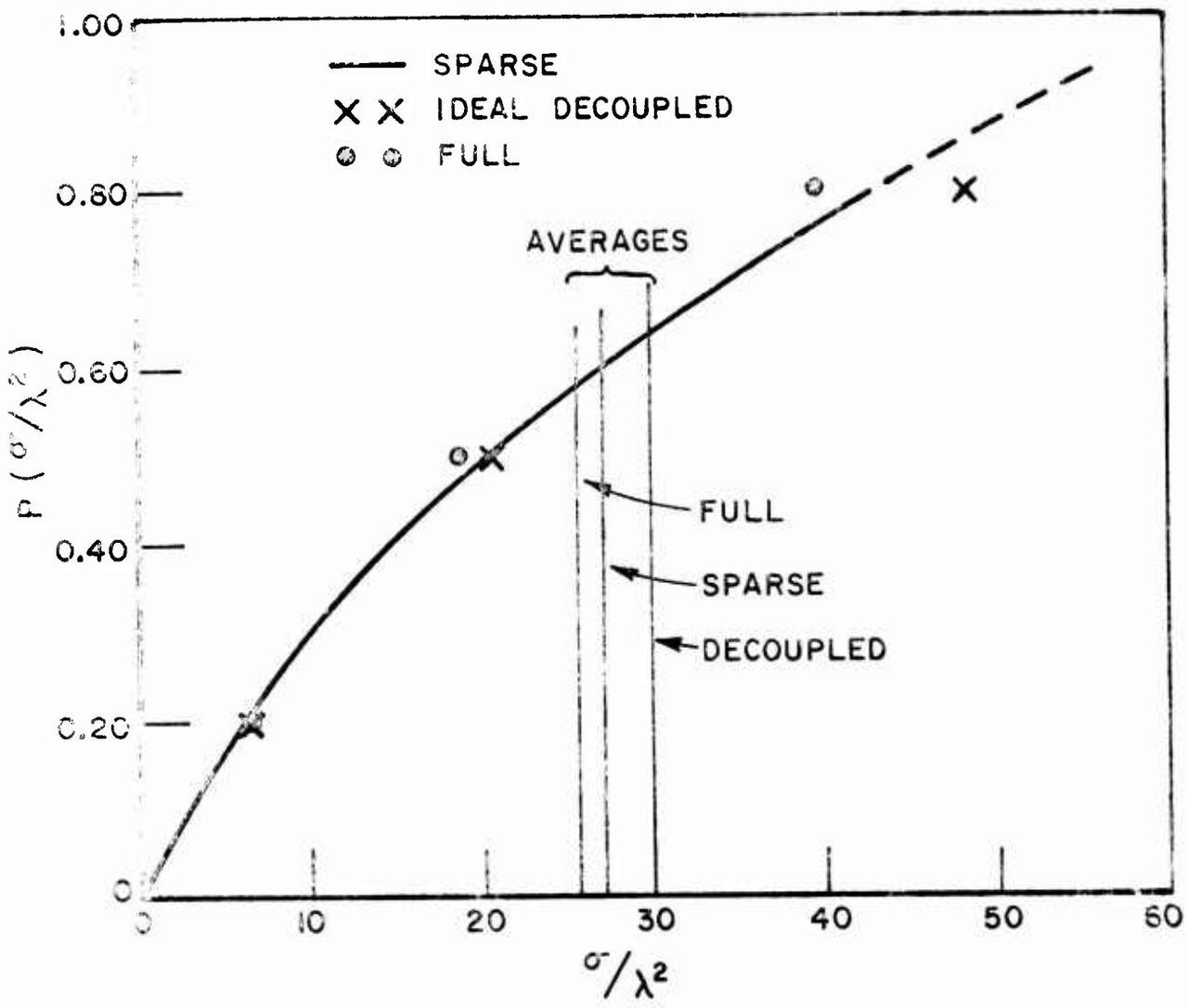


Figure 43. Cumulative probability function of backscattering cross section, cloud #1.

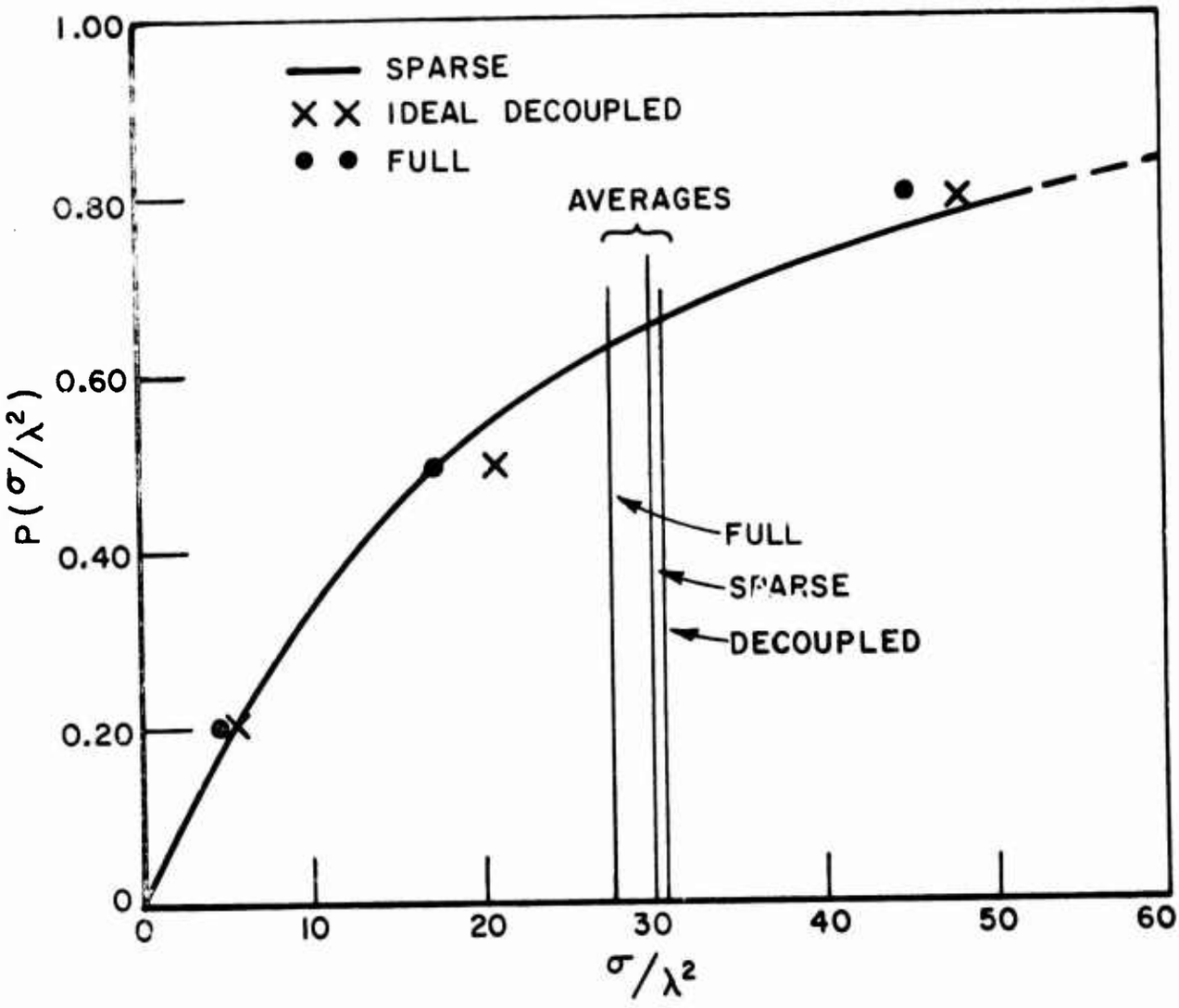


Figure 44. Cumulative probability function of backscattering cross section, cloud #2.

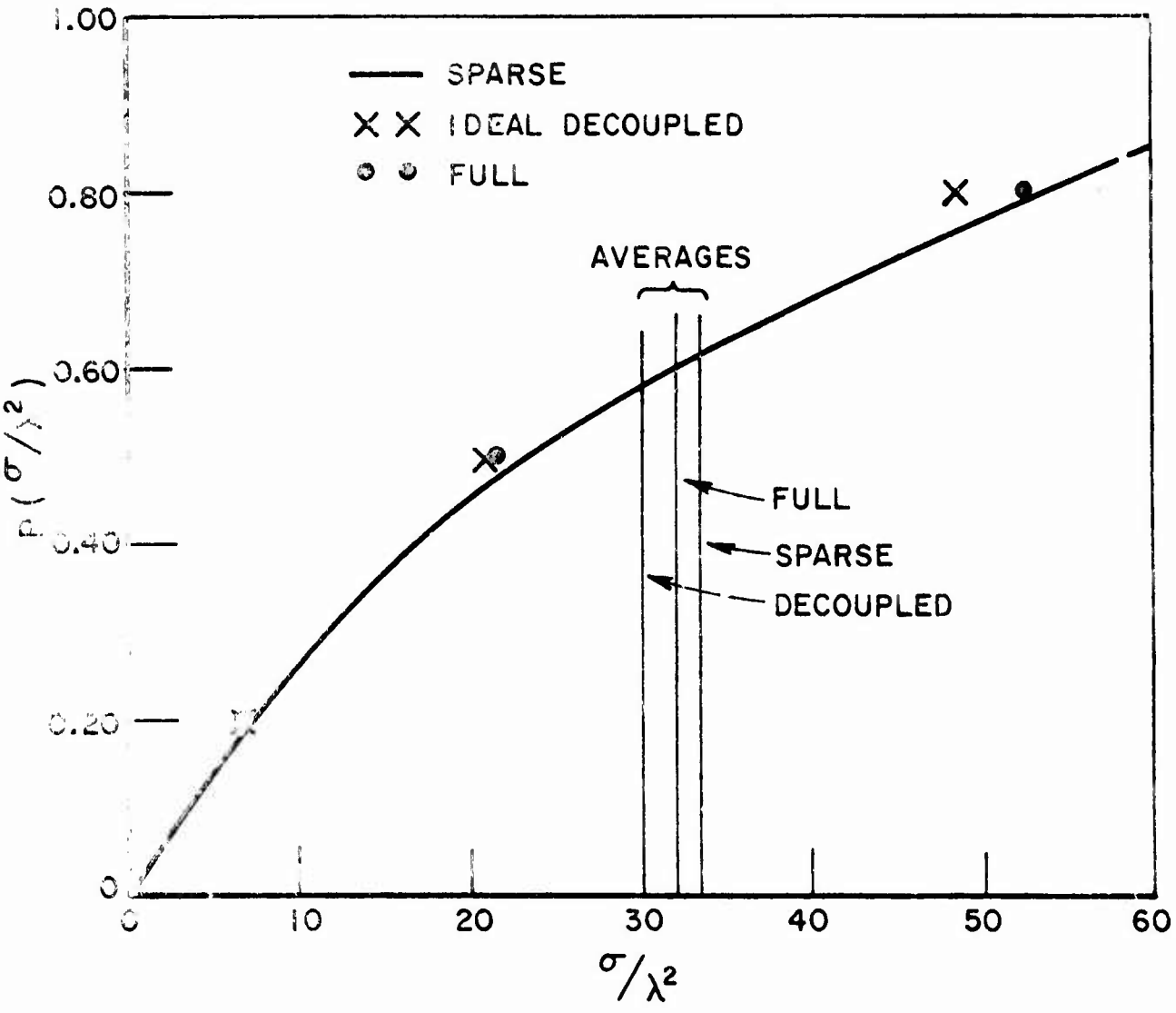


Figure 45. Cumulative probability function of backscattering cross section, cloud #3.

It is clear from what has been said previously that the sparse matrix incurs substantial savings in computer memory. But how about time saved? We may consider the time consumed (on the ElectroScience Laboratory Datacraft computer) by three separate operations: matrix setup time, i.e., the time taken to generate the Z matrix elements; reordering, i.e., the time taken to reorder the Z matrix so that its auxiliary matrix will also be sparse; and backscattering, i.e., the time taken to calculate the backscattering cross section at one look angle.

The number of elements in large matrices rapidly becomes exhorbitant, even taking into consideration the identity of all the diagonal elements and symmetry about the diagonal. Applying the 10% rule permits us to store only a few or less percent of these elements, but to apply the rule, all of them must be calculated. This takes a great deal of time. In order to reduce this matrix setup time, we appealed to the evidence of Fig. 22 to create what we call a "sphere-of-influence" model. In this model we avoid the calculation of the vast majority of the matrix elements by superimposing on the 10% rule, a sphere-of-influence rule, whereby one calculates only those matrix elements representing the coupling of the dipole of interest to its neighbors lying within a specified spherical volume centered at the dipole, all other couplings being assumed zero. Figure 46 shows the computer time saved by applying the sphere of influence rule as well as the 10% rule over the time taken by applying the 10% rule only. It is based upon averages of 20 clouds of 100 dipoles each, and shows the time saved for assumed sphere of influence radii from 2.07λ to 2.5λ. The larger the sphere-of-influence, the smaller the time savings, of course. But the larger the sphere of influence, the more identical become the matrices sparsed by the two different rules. The number of elements which differ in the two matrices so sparsed, are presented in Fig. 46 as the percentage of the N^2 elements in each matrix. Clearly, at about a radius of 2.4 , the two become identical, implying that the sphere-of-influence sparse model should yield backscattering patterns equally as good as those obtained from the 10% sparse model. Note that our average spacing of $d/\lambda = 1.0$ is assumed for the clouds. Denser clouds would exhibit less time saving. Figure 47, also for fixed $d/\lambda = 1.0$, indicates the time saving for a variety of choices of N, using 2.07λ and 2.5λ radii for the sphere of influence. As expected, the time saving rises as N increases.

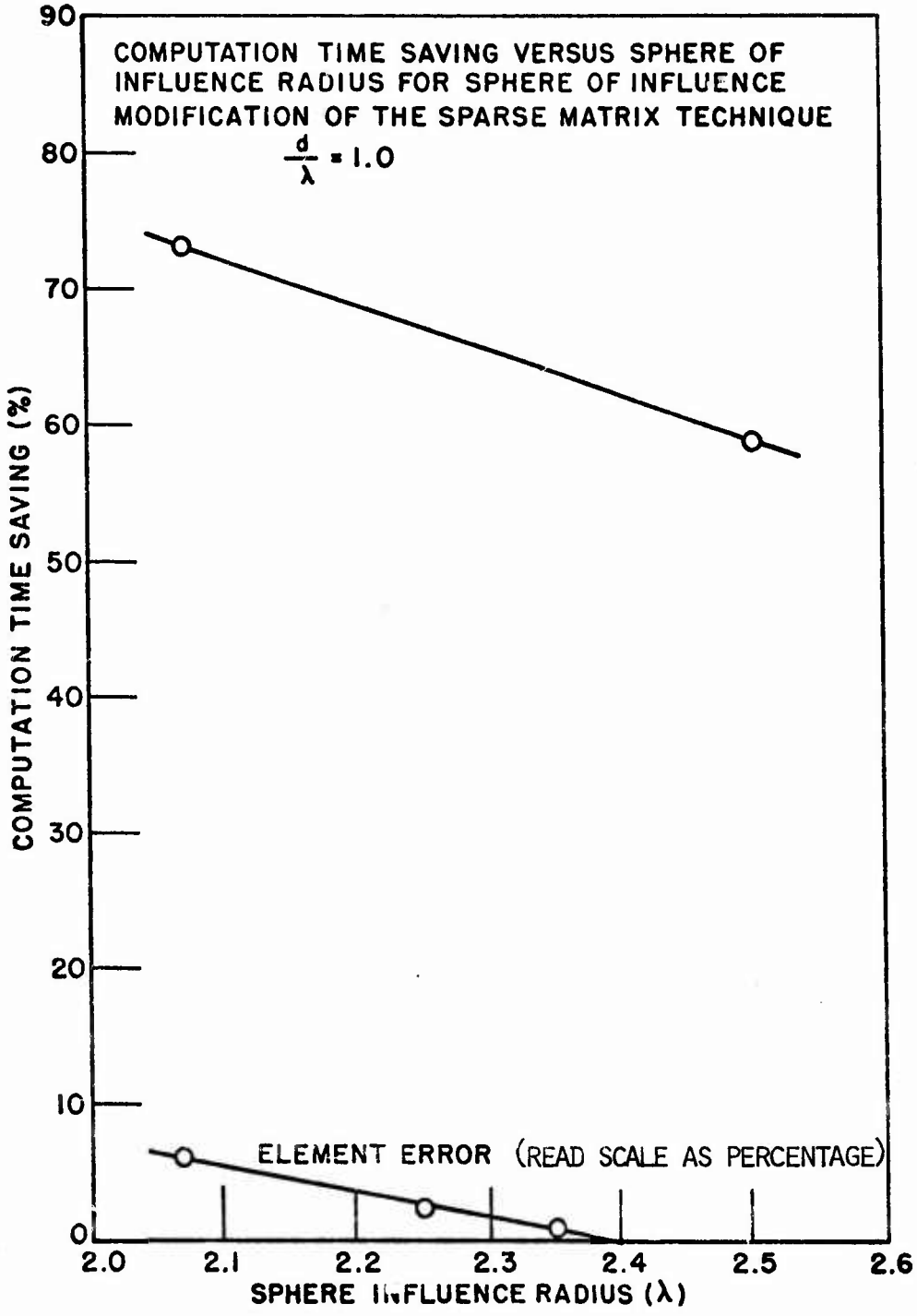


Figure 46. Time saving and element error vs sphere of influence radius using sphere of influence model plus 10% rule over the 10% rule alone

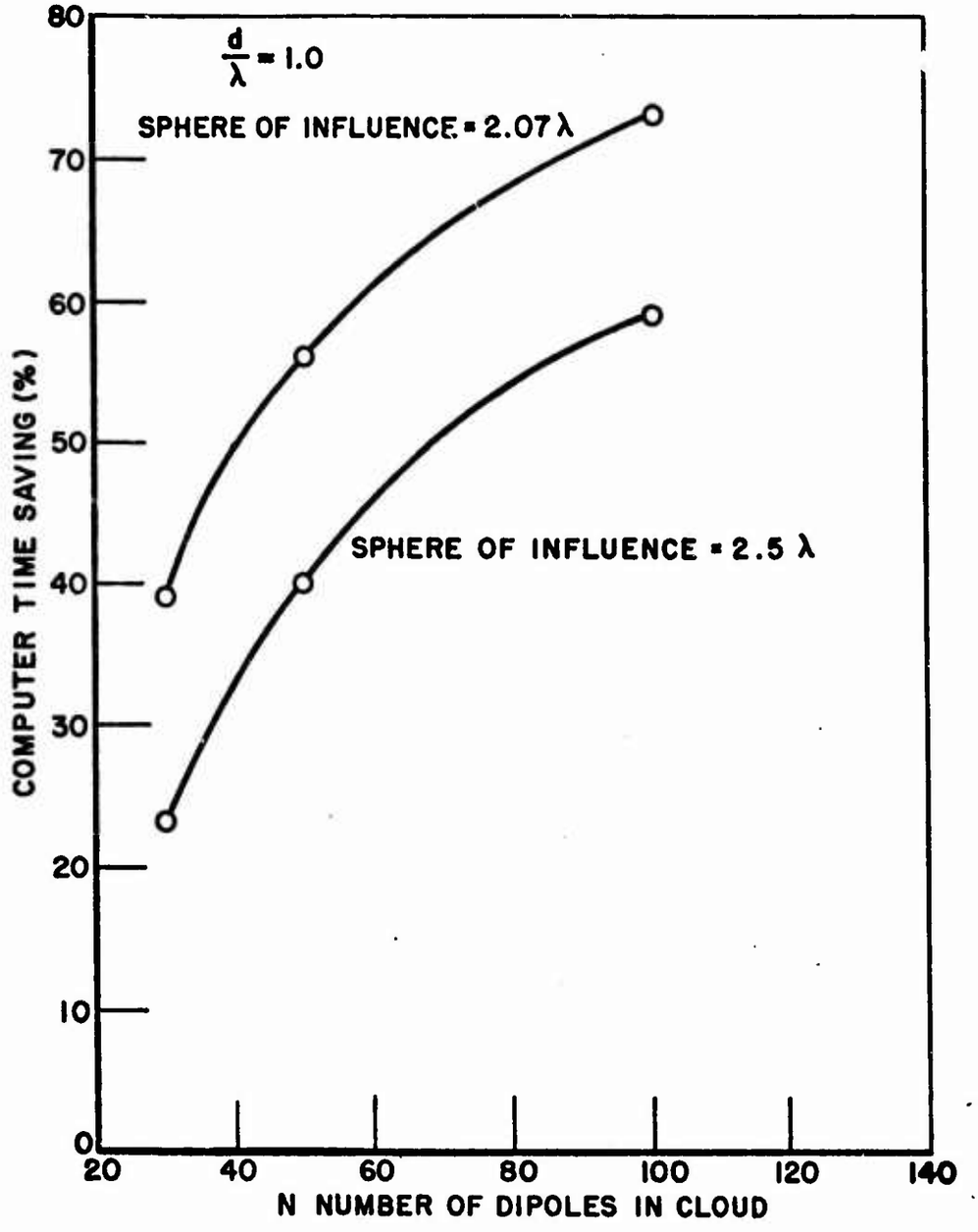


Figure 47. Time saving vs number of dipoles using sphere of influence model plus 10% rule over the 10% rule alone.

The foreging data reflect a very substantial time saving in matrix setup time with little penalty in echo area. Evidence did exist, however, that the sparse matrix algorithm, even with the sphere-of-influence rule built in, was time consuming. This evidence was verified when a computation of an N = 500 dipole cloud with $d/\lambda = 1.0$ the Wright-Patterson Air Force Base CDC 6600 computer overran its alloted time of 5000 seconds. In order to investigate this time consumption more carefully on our own machine, a set of backscatter data were accumulated for one look angle with clouds of 30, 50, and 100 dipoles, each with average spacings of $d/\lambda = 0.5$ and 1.0. Twenty such clouds were considered for each case and average times obtained for the three parts of the sparse matrix program. The sphere-of-influence plus 10% rules were applied to sparse each matrix and the results tabulated in Table E. The numbers do not represent real times but clock times on the ESL machine. Time ratios are of importance here.

TABLE 5
CLOCK TIMES OF THREE PARTS OF SPARSE MATRIX ROUTINE

d/λ	N	No. of non-zero elements	Matrix Setup	Reordering	1 look angle	Total time
1.0	30	76	842	475	17	1,350
1.0	50	178	1,863	1,884	31	3,743
1.0	100	343	5,033	17,410	76	22,534
0.5	30	220	1,117	2,807	30	3,968
0.5	50	500	3,069	21,277	66	24,449
0.5	100	1,462	11,311	414,121	219	425,667

In this table total time is the sum of the previous three operations plus some small amount for inherent operations. The average number of nonzero elements in the upper right triangle of the Z matrix are also given. Clearly, with the sphere-of-influence rule applied, it is the reordering time which is preponderant and causes the sparse matrix algorithm to be so time consuming. In an effort to reduce this reordering, an attempt was made to partially reorder. The results, however, were not encouraging and the effort was terminated.

The sparse matrix computer program used to obtain the foregoing results is documented in Appendix E.

Before turning to another topic we should point out one facet of the sparse matrix approximation as arrived at by the 10% and sphere-of-influence rule. That is, these sparsing techniques are determined by the Z matrix alone; they do not take into consideration the currents induced on the dipoles. For example, the influence of the ith dipole current upon the voltage induced in the jth dipole is proportional to $z_{ij}I_i$, the product of the ijth matrix element and the ith current. Simply setting z_{ij} to zero if it is smaller than 10% of z_{jj} may not be rigorously appropriate if I_i is large. However, the 10% rule appears to do a satisfactory job for obtaining the average backscatter. If, however, one is interested in extinction of current through the cloud, the 10% rule or, even worse, the sphere-of-influence model, cannot be expected to yield good results for, by their nature, these approximations modify the coherent forward scattered wave as it proceeds through the cloud. Since this is an important phenomenon dictating the extinction rate in the first few wavelengths into the cloud, a better model would have to be devised if one is interested in extinction. The indirect methods described below might serve such a purpose.

4. Indirect (Iterative) Methods

a. Theoretical Considerations

Sections 2 and 3 have discussed direct and sparse matrix methods for solving the equation,

$$ZI = V \quad (12a)$$

In this section we discuss indirect methods, of which linear iteration forms a special class and which we will emphasize. In order to avoid ambiguity in notation, in this section we will rewrite Eq. (12a) as

$$Ax = b \quad (12b)$$

and develop all pertinent equations in terms of A, x, and b rather than Z, I, and V.

All indirect methods of solving Eq. (12b) for x can be viewed from the implicit formulation given by

$$x = f(A,b,x) \quad (13)$$

implicitness being characterized by the appearance of unknown vector x on both sides. The symbol f in the above expression represents a function or set of rules (algorithm) with the minimal property that the exact x satisfies Eq. (13) identically. One additional condition on f needed here is that it be able to transform an approximation to x into an improved approximation. It would be too much to ask that one application of Eq. (13) yield the exact solution. However, repeated applications might be expected to give successively better approximations and this is precisely the essence of iteration. Notation can be added to the implicit form of Eq. (13) to give a general formula for iteration, i.e.,

$$x^{(k)} = f^{(k)}(A,b,x^{(k-1)},x^{(k-2)},\ldots,x^{(k-m)}), \tag{14}$$

where $x^{(k)}$ represents the k^{th} iterate or approximation of x. Note, in this form, $x^{(k)}$ is considered to be related to m previous iterates, in which case the iteration is of m^{th} degree. Also note that, in general, the function $f^{(k)}$ can change from step to step. If $f^{(k)}$ remains invariant throughout the iteration process ($k = 1,2,\ldots$), then the iteration is called stationary and if not, it is called non-stationary. The iteration process is referred to as linear for $f^{(k)}$'s which are linear functions of $x^{(k-1)}$, $x^{(k-2)}$, $\ldots,x^{(k-m)}$ and nonlinear otherwise. Iterative methods subdivide still further into point-step and group or block-step methods and these categories depend on the choice of $f^{(k)}$. More specifically, the point-step methods proceed to improve the individual components of solution vector $x^{(k)}$ one-at-a-time, independently of the other elements, while block-step methods normally improve blocks of elements of $x^{(k)}$, independently of other blocks. A rather unique block type iterative method will be introduced later which will allow "overlap" of these blocks based on the physical scattering problem. Disussed in this section are three classical linear stationary methods of first degree; the Point-Jacobi (J) method, the Gauss-Seidal (GS) method and the method of Successive Over-relaxation (SOR) together with their physical interpretation from the scattering viewpoint. Also included is a discussion of convergence criteria for these methods and finally a presentation of results, mostly calculated using SOR.

Linear First Degree Methods (J,GS, SOR)

The basic equation underlying many linear indirect methods is derived from Eq. (12b) by adding x to both sides and rearranging to give

$$x = (I-A)\,x + b, \tag{15}$$

which, in terms of a sequence of iterates can be written as

$$x^{(k)} = Hx^{(k-1)} + b, \tag{16}$$

where

$$H \equiv I - A . \tag{17}$$

H is usually referred to as the iteration or error reducing matrix and is related to the functions $f^{(k)}$ described in the previous section. Iteration via Eq. (16) is linear, stationary and of first degree. This expression yields a number of classical techniques which differ by the "splitting" of matrix A. Consider the splitting defined by

$$A = D - E - F \tag{18}$$

where $D = [a_{ii}]$, $i = 1, \ldots, N$, is a diagonal matrix and $E=[-a_{ij}]$, $i>j$, is strictly lower triangular and $F=[-a_{ij}]$, $i<j$, is strictly upper triangular. The iteration of Eq. (16) then becomes

$$x^{(k)} = D^{-1}(E+F)x^{(k-1)} + D^{-1}b , \tag{19}$$

where the iteration matrix is identified as

$$H_J = D^{-1}(E+F) . \tag{20}$$

Equation (19) describes the well known Point-Jacobi (J) method [48] or method of "simultaneous displacements" [49]. Here, new components of $x^{(k)}$ are computed as functions of components of $x^{(k-1)}$ as follows:

$$x_i^{(k)} = -\frac{1}{a_{ii}} \sum_{\substack{j=1 \\ j\neq i}}^{N} a_{ij}x_j^{(k-1)} + \frac{1}{a_{ii}} b_i \tag{21}$$

Note however, that by carefully considering the ordering of improvements in $x^{(k)}$ Eq. (21) can be modified to incorporate the latest improvements in $x^{(k)}$ at intermediate steps; i.e.,

$$(22) \qquad x_i^{(k)} = -\frac{1}{a_{ii}} \left(\sum_{j=1}^{i-1} a_{ij} x_j^{(k)} + \sum_{j=i+1}^{N} a_{ij} x_j^{(k-1)} \right) + \frac{1}{a_{ii}} \; b_i$$

or, in matrix notation,

$$(23) \qquad x^{(k)} = (D-E)^{-1} F \, x^{(k)} + (D-E)^{-1} b \quad .$$

Here, the iteration matrix is given by

$$(24) \qquad H_{GS} = (D-E)^{-1} F \quad .$$

Equation (23) is the familiar Gauss-Seidel (GS) method [50], also known as the method of "successive displacements".

Both the J and the GS techniques can be considered special cases of a larger class of computer oriented "relaxation" methods often referred to as Over-relaxation (OR) methods [51]. A basic equation governing these methods is given by

$$(25) \qquad \bar{x}^{(k)} = \bar{x}^{(k-1)} + \omega \, (x^{(k)} - \bar{x}^{(k-1)})$$

where the "relaxation factor" is usually chosen to be a real constant in the range $0<\omega<2$ and $x^{(k)}$ is computed by either the J or the GS method [52]. The technique for computing $x^{(k)}$ is clearly not restricted to the above two methods; here, however, only the GS method will be assumed. This assumpiton leads to the defining equation for the familiar Successive Over-relaxation (SOR) method [53]; namely,

$$(26) \qquad x^{(k)} = (I-\omega D^{-1}E)^{-1}[(1-\omega)I-\omega D^{-1}F]x^{(k-1)}+(I-\omega D^{-1}E)^{-1}\omega D^{-1} b,$$

where the iteration or error reducing matrix H is given by

$$(27) \qquad H_{\omega} = (I-\omega D^{-1}E)^{-1}[(1-\omega)I-\omega D^{-1}F].$$

The computational procedure for SOR is given by Eqs. (22) and (25) and therefore, for $\omega=1$, SOR reduces identically to the GS

method. Incidentally, when the solution $x^{(k)}$ in Eq. (25) is computed by the J method, the resulting technique is called the method of "simultaneous over-relaxation" (JOR) [54] and reduces directly to the J method for $\omega=1$.

The SOR method of Eq. (26) is obviously a stationary linear method of first degree. Nevertheless, SOR can be made non-stationary by not restricting ω to being a constant for all iterations. It is not, on the other hand, clear how ω should be varied to improve the speed of the iteration procedure (convergence) for the general case. More will be said of this subject in a later discussion on convergence.

An alternative form for these same iteration procedures can be derived in terms of an approximate or psuedo inverse to matrix A. Let $\tilde{A}^{-1}$ represent an approximation to the inverse A^{-1} of Eq. (13). Then, the kth iterate x(k) can be written as $x^{(k-1)}$ plus a correction term $d^{(k-1)}$ given by

$$d^{(k-1)} = \tilde{A}^{-1} r^{(k-1)}, \tag{28}$$

where the residual vector $r^{(k-1)}$ is defined by

$$r^{(k-1)} = b - Ax^{(k-1)}. \tag{29}$$

Equations (28) and (29) can be combined to give

$$d^{(k-1)} = -\tilde{A}^{-1}A\, x^{(k-1)} + \tilde{A}^{-1}b, \tag{30}$$

whereupon, the kth iterate may be written as

$$x^{(k)} = (I-\tilde{A}^{-1}A)x^{(k-1)} + \tilde{A}^{-1}b\ . \tag{31}$$

The H matrix here has the form

$$H = (I-\tilde{A}^{-1}A) \tag{32}$$

and it is easily shown that the following choices for A lead to the previously derived H matrices; i.e.,

(33) $\tilde{A} = D \rightarrow$ Eq. 20 (J)

(34) $\tilde{A} = D\text{-}E \rightarrow$ Eq. 24 (GS)

(35) $\tilde{A} = \frac{1}{\omega} D\text{-}E \rightarrow$ Eq. 27 (SOR).

An additional point to be noted in this latest discussion is that A need <u>not</u> be identified with a rigorous matrix form such as those given in Eqs. (33) to (35). A can merely be representative of a special algorithm for computing the approximations to x. Equation (31) in this case will no longer represent a rigorous matrix equation. This is in line with the previous comment that f in Eq. (13) may in fact represent only a set of rules or algorithm for computation. More will be said later concerning a less-than-rigorous notation.

Convergence Criteria

Success or failure of any iterative method is measured in terms of the limit of the sequence $\langle x^{(k)} \rangle$ as $k \rightarrow \infty$; i.e., if $x^{(k)}$ reaches the exact solution x in the limit, then the method is obviously successful and if not, the method fails. Although seeemingly straightforward, certain questions remain unanswered. Namely, is information available to indicate, a priori, when a particular method will converge and, if so, what quantitative measures can be counted on to indicate sufficient convergence since the exact solution is never known? The first question is answered rather easily which the following paragraphs will show. The second question however turns out to be the more practical yet difficult question to answer. Reasons for this will be made clearer in the final portions of this section.

The normed vector space defined in Appendix C of Ref. 38 can be reintroduced here in terms of the limit of the sequence $\langle x^{(k)} \rangle$ in the following way,

(36) $$\lim_{k \rightarrow \infty} ||x - x^{(k)}|| = 0,$$

where x is the exact solution satisfying Eq. (16) identically; i.e.,

(37) $$x = Hx + b \quad .$$

The following result is obtained by considering the difference between Eqs. (37) and (16),

$$(x-x^{(k)}) = H(x-x^{(k-1)}) \tag{38}$$

and can be taken recursively to yield

$$(x-x^{(k)}) = H^k(x-x^{(0)}). \tag{39}$$

Note here that $x^{(0)}$ is the initial "guess" corresponding to k=0, hence, $(x-x^{(0)})$ is a constant vector. Compatible norms (see Appendix C of Ref. 38) are needed on both sides of Eq. (39) to give

$$||x-x^{(k)}|| \leq ||H||^k \, ||x-x^{(0)}|| \, , \tag{40}$$

where the inequality $||H^k|| \leq ||H||^k$ has been included in bringing this expression to the form of Eq. (40). Recall, Eq. (36) defines the unique condition for convergence of $\langle x^{(k)} \rangle$ in the established normed space and by applying this condition to Eq. (40), the necessary and sufficient condition for convergence of Eq. (16) becomes

$$\lim_{k\to\infty} ||H||^k = 0, \tag{41}$$

and this condition can only be satisfied if

$$||H|| < 1 \tag{42}$$

Hence, the properties of H determine convergence characteristics of Eq. (16) for any starting vector $x^{(0)}$. The natural norm of Eq. (42) remains as yet unspecified but has a lower bound (see Appendix F of Ref. 38) in the spectral radius given by

$$||H|| \geq \rho\{H\} \tag{43}$$

where the spectral radius of H, $\rho\{H\}$, is defined by

$$\rho\{H\} \equiv \max_i |\lambda_i| \tag{44}$$

and the λ_i's are solutions to the determinantal eigenequation,

$$\det(H-\lambda I) = 0. \tag{45}$$

Therefore, the necessary and sufficient condition for convergence of Eq. (16) to the solution x (see Appendix G of Ref. 38 for this proof) is given by

$$\rho\{H\} < 1 , \tag{46}$$

Convergence properties for the iterative methods outlined earlier can be predetermined as the above procedure indicates; however, for certain special cases, calculation of eigenvalues can be avoided. This would certainly be an advantage, especially for those cases when the order of matrix H is large (e.g., N>100). These special cases can be recognized in terms of the following properties [55] of the original matrix A and the splitting of A defined in Eq. (18):

1. If

   $E + F \geq 0$,

   $D > 0$,

   and

   $\rho\{D^{-1}(E + F)\} < 1$

   then A is an M-matrix.

2. If

   $||D^{-1}(E+F)||_\infty < 1$

   then A is strictly diagonally dominant.

3. If no NxN permutation matrix P which permutes rows and columns of an NxN matrix exists such that

$$PAP^T = \begin{bmatrix} D_1 & G \\ 0 & D_2 \end{bmatrix} \quad , \text{(T denotes transpose)}$$

   where D_1, D_2 are square matrices and

$$||D^{-1}(E + F)||_{\infty} \leq 1 \ ,$$

then A is irreducibly diagonally dominant.

4. A has the following properties;
   A is hermitian (A = A*) and (* denotes complex conjugate transpose)

   A is positive definite (eigenvalues of A are

   λ_i, i = 1,2,···,N and satisfy $\lambda_i > 0$, for all i.

The convergence of the J and GS methods is assured for any matrices satisfying 1, 2 or 3 above and the SOR method necessarily converges for $0<\omega<2$ when condition 4 is met. Proofs of these sufficient conditions for convergence are given in Varga [56]. If, in addition to condition 4, A has "property A" as originally defined by Young [57], then an optimum relaxation factor ω_{opt} can be computed for the SOR method. This optimum factor is given by

$$\omega_{opt} = \frac{2}{1 + \sqrt{1-\nu^2}} \ , \tag{47}$$

where

$$\nu = \rho\{H_J\} \tag{48}$$

and H_J is computed from Eq. (20). If, on the other hand, A does not satisfy "property A", then ω_{opt} can only be determined empirically.

The discussion of convergence, so far, has centered on finding the spectral radii of appropriate iteration matrices or on the special properties of the original matrix A. Consider, however, the more general matrices which appear in the EM problems studied here. The A matrices in these cases are complex symmetric (nonhermitian) and not diagonally dominant in all but the most trivial cases. They are positive definite, or at least positive semi-definite, in the sense that

$$\mathrm{Re}\{x^*Ax\} \geq 0 \quad (*\text{denotes complex conjugate transpose}), \tag{49}$$

where this quantity is related to real power dissipated (radiated) by the system represented by impedance matrix A. These basic characteristics of the EM problem eliminate any possibility of satisfying conditions 1-4 above. Therefore, the only rigorous technique is to compute the appropriate spectral radius, but some difficulty in computing $\rho\{H\}$ is likely to be encountered for many practical EM problems due to the size of N. General subroutines are available [58] for calculating complex eigenvalues of complex matrices; however, when N becomes large (>250), these routines will require more fast-access memory than available on most computing machines. Even if these computations are possible, the authors suggest that the time and effort used in searching for a "largest" eigenvalue would better be used trying the iterative technique.

A suitable measure of convergence characteristics usually must be determined empirically. One natural choice is a measure based on the vector of residuals defined by Eq. (29), or

$$r^{(k)} = b - Ax^{(k)} \tag{50}$$

This expression can be misleading since it states that if $r^{(k)}=0$, the $x^{(k)}$ is the exact solution and this is correct. However, to assume that $x^{(k)}$ is near the exact solution when $r^{(k)}$ is small (but not zero) may be a gross overassumption. A hint of this specious behavior is given by the following bound on the relative error in $x^{(k)}$,

$$\frac{||x-x^{(k)}||}{||x||} \leq \frac{||r^{(k)}||}{||b||} \; \text{Cond}\{A\} \quad . \tag{51}$$

Clearly, the ratio $||r^{(k)}||/||b||$ must be considered in light of the condition number of A and the possible effects it may have on the upper bound of Eq. (51). It is also important to point out that all norms of residuals defined by Eq. (50) do not necessarily decrease monotonically when the iteration process is convergent; i.e., they sometimes oscillate or increase. Even then, Eq. (51) implies that if a monotonically decreasing norm is found, it may still be mere speculation to assume $x^{(k)}$ is in some sense approaching the correct solution. Still another measure of convergence is to consider a norm of the change in $x^{(k)}$, from one iteration to the next. Consider the following normed difference,

$$(52) \qquad ||\delta x^{(k)}|| \equiv ||x^{(k)} - x^{(k-1)}||$$

and ask the following question:

> Does there exist a value of k, say k_∞, and some $\varepsilon > 0$ such that for $k > k_\infty$, $||\delta x^{(k)}|| < \varepsilon$?

If so, the process can be said to converge. The particular choice of ε used to indicate sufficient convergence, however, is critical since the normed difference given by Eq. (52) is not necessarily a monotonically decreasing measure, even if the solution is convergent.

A last comment is in order before proceeding. Certainly, the most reassuring indication of convergence would be to compare solutions obtained by different techniques and possibly even by a physical measurement and find that they agree. This type of comparison should obviously be sought wherever possible and this was indeed the case in this study. In a following section we present certain confirmed iterative results and these results are used to justify the choice of error measure used for reliably indicating convergence.

Physical Interpretation of the Jacobi and Gauss-Seidel Methods

A physical interpretation of the J and GS methods is presented here with the aid of Fig. 48. The A matrix of previous equations here represents the 5 x 5 impedance matrix corresponding to the 5 dipoles shown in the figure. Consider the initial excitation on each dipole to be the incident field and the initial current vector to be $x^{(0)} = 0$.

The J method in general computes $x^{(k)}$ by considering the incident field and the scattered fields produced by $x^{(k-1)}$. The latter contribution is zero for k = 1, hence the J method calculates $x^{(1)}$ corresponding to the "uncoupled" array. The J method improves the solution for $k > 1$ by accounting for the incident field and the scattered fields at each dipole where the scattered fields are produced by "old" currents. This interpretation of the J method in terms of multiple scattering within the array was first described by Tai [59]. All elements of the solution vector are updated simultaneously at the end of each iteration, hence the name "method of simultaneous displacements".

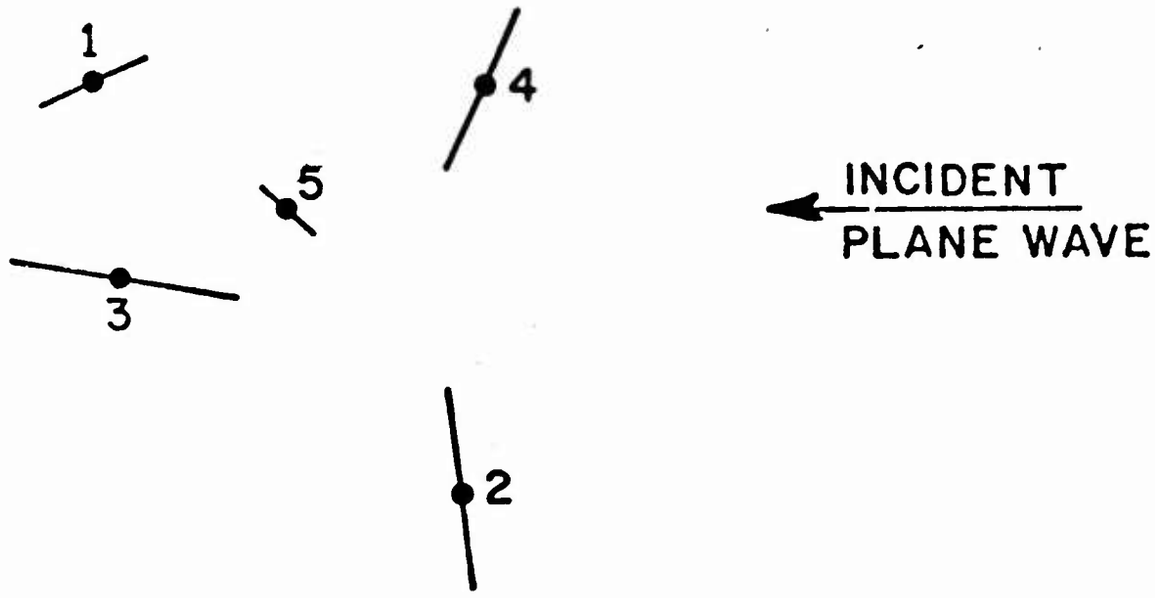


Figure 48. Sample random array for Jacobi and Gauss-Seidel iteration methods.

The GS method uses the "latest" currents whenever possible, i.e., the initial current on element #1 due to the incident field is

$$x_1^{(0)} = \frac{b_1}{a_{11}} \quad , \tag{53}$$

the initial current on element #2 due to the incident field plus the scattered field from element #1 is

$$x_2^{(0)} = (b_2 + a_{21}x_1^{(0)})/a_{22}, \tag{54}$$

the initial current on element #3 is due to the incident field plus the scattered fields from elements #1 and #2

$$x_3^{(0)} = (b_3 + a_{31}x_1^{(0)} + a_{32}x_2^{(0)})/a_{33}, \tag{55}$$

etc. The name "method of successive displacements" clearly applies to the above description and, as we might expect, the GS method has superior convergence properties since it accounts for

$$\sum_{i=1}^{n} i$$

interactions per iteration whereas the J method only accounts for n interactions per iteration.

The SOR method operates on the GS iterates by "relaxing" the latest correction through a weighted averaging process. Note that, even though SOR degenerates to GS for $\omega=1$, convergence of SOR ($\omega\neq1$) can be relatively good while GS may not converge at all. Physical interpretation of SOR in terms of scattering is more difficult than for J or GS. Weighted averaging of currents seems to be a purely mathematical concept. However, by assuming the array to be immersed in a medium which modifies the multiply scattered fields either by introducing "loss" or "gain", this would cause corresponding reductions or increases in interactions between dipoles. The application of an iterative procedure (e.g., GS) under these "relaxation" conditions could also be termed a form of SOR. The loss or gain in this case could either be reduced as the iteration converged or left in if the convergence required it. The solution to a "lossy" problem might be of considerable value in certain cases, especially if the "lossless" case could be deduced from such a solution.

Sphere of Influence (SOI) Method

The SOI technique is an empirically derived concept based on the electromagnetic scattering viewpoint. The approach stems directly from the array problem where the overall scatterer is so large and intricately detailed that it produces a matrix problem too large to handle by direct methods. Hence, the larger problem is broken up into a reasonable number of smaller problems each of which can be solved directly. The heart of the method lies in the hope that the solution to the large case can be obtained by interacting these smaller solutions with one another through an iterative process. The idea of "influence" manifests itself as a mutual impedance or coupling criterion between dipoles as in the case of the random array. Distance between dipoles provides a natural means for determining gross effects between dipoles and relative orientation is another. When these criteria fail to give a precise decision rule, a comparison of the mutual impedance to a preset level can be made. The level or threshold used here is defined to be a prescribed fraction of the diagonal or self impedance term. This criterion is also similar to that used in the

sparse matrix approximation for the scattering problem. Recall, the sparse matrix approach attempts to "thin" the matrix by deciding which elements are less important (i.e., below a certain magnitude) and a special algorithm is used to solve the thinned matrix problem exactly. This, however, is not the solution to the original problem and it is for this reason that iteration may provide the only means for finding the exact solution to the original problem for these large cases.

The basic SOI method computes groups of closest coupled neighbors and uses these "overlapping" groups to form a sequence of N reduced iteration submatrices. Closeness is measured by the relative influence between dipoles using the a priori criterion mentioned above. The N iteration submatrices will in general be distinct and the jth submatrix will be used to compute only the current on the jth dipole (point-step). The N subsystems formed by these submatrices are each solved by a direct technique and the scattered tangential electric fields are computed after each iteration and compared to the incident tangential electric fields as a check on the zero tangential electric field boundary condition along each dipole. The same residual mode voltage column $r^{(k)}$ of Eq. (50) is proportional to the total tangential electric fields and is used as the excitation column for the next iteration if boundary conditions are not sufficiently met. The process is continued until $||r^{(k)}||$ is reduced to an acceptable level.

One possible formulation for SOI is given in the following equations with the understanding that the overall technique cannot be simply described by a single matrix equation as with the other methods mentioned thus far. Let $A(m_j|m_j)$ represent the $m_j \times m_j$ iteration submatrix containing self and mutual impedances for the jth dipole and its m_j-1 most closely coupled neighbors. The members of this jth subsystem (submatrix) are obtained by applying the following condition to the jth row of A,

$$c|a_{jj}| < |a_{jp}| \quad , \quad \begin{matrix} p = 1,2,\ldots,N \\ p \neq j \end{matrix} \tag{56}$$

where c is a prescribed (empirical) real constant in the range $0 < c < 1$. The jth subsystem at the kth step of the iteration process is then given by

$$A(m_j|m_j)\, d^{(k)}(m_j) = r^{(k-1)}(m_j) \tag{57}$$

when $r^{(k-1)}(m_j)$ is the $m_j \times 1$ "subvector" of residuals on the jth group of dipoles and $d^{(k)}(m_j)$ is a $m_j \times 1$ subvector which includes the kth correction to the current on dipole j, i.e.,

$$x_j^{(k)} = x_j^{(k-1)} + d_j^{(k)} \quad . \tag{58}$$

The kth iteration is complete after N subsystems of the form Eq. (57) (j=1,2,...,N) have been solved and all corrections (j=1,2,...,N) of the form E. (58) have been made. A new residual is obtained again by including the original A matrix and b vector in Eq. (50).

Consider a simple application of SOI to the 5 dipole array illustrated in Fig. 49. The region ("sphere') of influence around dipole #1 (j=1) is shown figuratively as a circle about dipole #1. Recall, this circle actually represents the region of influence for which Eq. (56) is satisfied for j=1 for the given value for c. The matrix equation for this subset will be of order $m_1=3$ and for the kth iteration this equation takes the form

$$\begin{bmatrix} a_{11} & a_{13} & a_{15} \\ a_{31} & a_{33} & a_{35} \\ a_{51} & a_{53} & a_{55} \end{bmatrix} \begin{bmatrix} d_1^{(k)} \\ d_3^{(k)} \\ d_5^{(k)} \end{bmatrix} = \begin{bmatrix} r_1^{(k-1)} \\ r_3^{(k-1)} \\ r_5^{(k-1)} \end{bmatrix} \tag{59}$$

Direct solution of this subsystem yields subvector $d_3^{(k)}$ from which the kth correction to $x_1^{(k-1)}$ is obtained, i.e.,

$$x_1^{(k)} = x_1^{(k-1)} + d_1^{(k)} \quad . \tag{60}$$

Some experience is necessary in choosing constant c in order that the maximum of m_j defined by

$$\max_j \; m_j = M \tag{61}$$

remains within the capacity of the machine and yet still yields a convergent solution. The two extreme choices for c are c = 0 and c = 1. All submatrices corresponding to the choice c=0 are identically equal to the original A matrix and the first subsystem therefore yields a total solution for x with one application of the direct method, assuming of course the computer can do this.

The c=1 choice causes SOI to degenerate to the J method since only the diagonal terms are inverted in this case.

A potentially important modification to the SOI method is the inclusion of a "forward scatter" (FS) model. Consider the dipoles which are located on the far side of a very large and dense array. These dipoles are very likely to be shadowed by those located on the directly illuminated side of the array. Hence, an improved "region of influence" for dipoles deep inside the array (or on the back side) could be obtained by taking into account the well known coherent forward scatter phenomenon which occurs along the line-of-sight. The reasoning here is that as the incident wave passes over these resonant dipoles (up front), the rescattered fields in the forward direction are nearly of opposite phase to the progressing incident wave and as this incident wave moves farther into the array, these coherently rescattered fields begin to "buck out" the incident wave. This eventually produces a shadowing effect on dipoles in the deep interior and far side regions of the array.

The above concepts of FS are rather simple to grasp; however, implementation of FS into the SOI algorithm is relatively messy. The FS process entails checking all $a_{ij}x_j^{(k)}$ products which occur on or near the line-of-sight aspect through the array to the ith dipole. The "up stream" jth dipoles with scatter products which satisfy

$$\pi/2 < \arg\{a_{ij}x_j^{(k)}\} < 3\pi/2 \qquad (62)$$

are then chosen to be included in the next (k+1) subsystem (submatrix) for calculating the current on dipole i. The newly modified SOI-FS method is nonstationary since the N submatrices will no longer be constants for the whole process. They will of course become more constant as $x^{(k)}$ nears a constant solution; however, in general, these submatrices will be quite changeable in the early stages of the iteration. Also note that, the resulting subsystems will be larger than for SOI alone for a given constant c and hence, implementing FS into SOI will generally require different values for c in order to maintain M in the viable range for direct solutions. The addition of FS should, however, improve convergence of SOI and thereby allow an increase in c in order to make room for the new dipoles added in by FS.

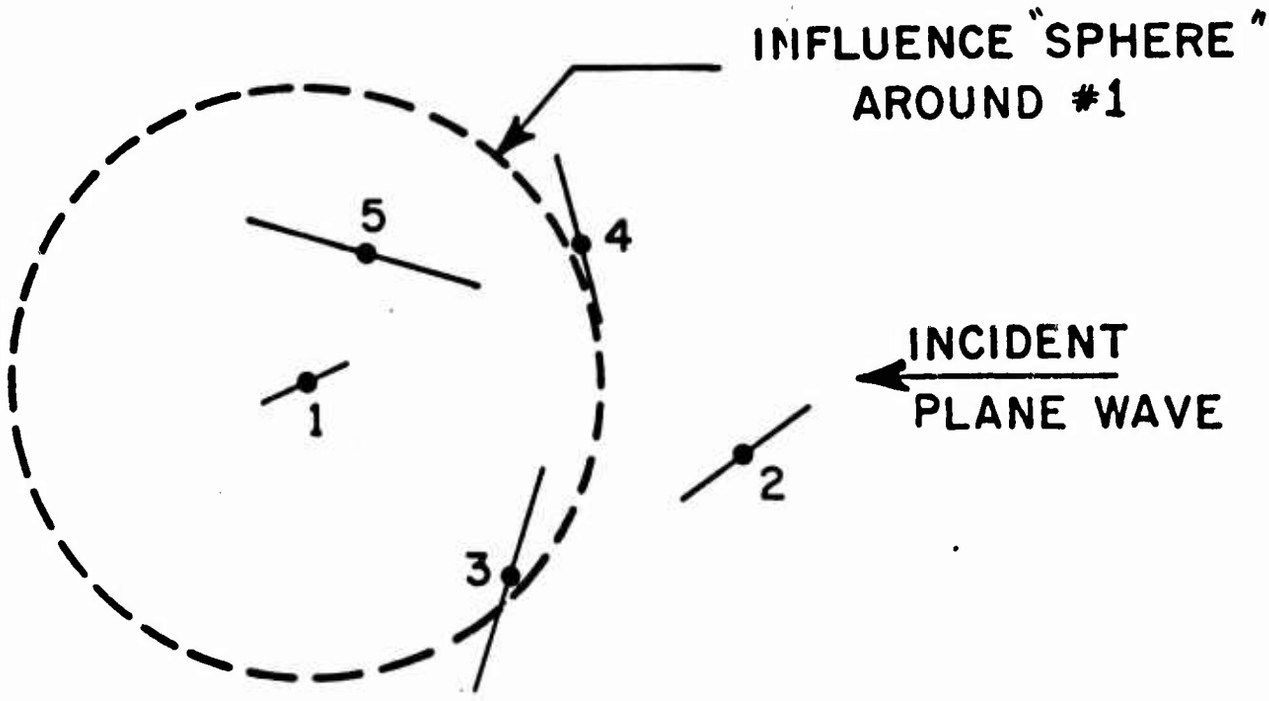


Figure 49. Sample random array for Sphere of Influence iteration method.

(b) Calculated results for chaff clouds

In Chapter VI of Ref. [38] there appear a set of curves of error bounds and condition norms for a few typical impedance matrices arising from chaff clouds. In general, these bounds rise with increase in dipole density and numbers, a trend which eventually must be reckoned with if direct solutions to larger order systems are sought. In light of this the iterative schemes are attractive and are used here to solve for the scattering from clouds of up to 1000 dipoles.

Numerical results presented in this section are divided into four areas: a check case; applications of SOR iteration to the solution of electromagnetic scattering by large clouds of thin resonant dipoles; application of SOI iteration to the solution of electromagnetic scattering by a small cloud of thin resonant dipoles; comments on applications of SOR to surface patch and wire-grid models. The appropriate equations from the preceding sections

have been translated into FORTRAN and documented listings of these programs appear in Appendix VI. All calculations were performed with 11 digit precision on a Datacraft Model 6024 computer having approximately 32k of real fast-access memory and 32k of virtual (disk) memory. Cycle time for this computer is approximately 1 microsecond. Certain special programming techniques, unique to this machine, are incorporated in the FORTRAN programs to allow psuedo-random access to approximately 6-1/2 million (24 bit) additional words of disk memory. Three and one-fourth million complex numbers can be computed, then stored in a special truncated form (6 digits) and retrieved using this technique. Also, a special subprogram is included which computes mutual impedances between "distant" dipoles; description and verification of this subprogram are also given in Appendix VI. This subprogram uses a special simplified calculation of the mutual impedance when dipoles are spaced greater than 1λ and inclusion of this simplified calculation resulted in a computation time for the approximately one-tenth that of the original estimate for the N = 1000 case; estimate $\sim$ 10 - 12 hours, actual time $\sim$1 hour.

It is important to note that all the results up to this point have assumed radially inhomogeneous densities for the clouds; in this section, however, all the results assume randomized clouds of uniform density.

A Check Case

Because some of the clouds treated here by iterative methods are so large, it is difficult to verify that the methods are actually giving correct values for echo, since no other reliable independent methods exist for comparison checks. Yet such checks are imperative if one is to have some confidence in the results. To this end we chose as a check case the planar array sketched in Fig. 50. It contains 841 resonant dipoles interlaced into a periodic structure with average spacings between nearest neighbors of approximately 0.57λ. By the technique developed by Munk [60] scattering from such an array can be readily obtained under the assumption of no edge effects, i.e., the array is considered to be a section of an infinite array. Using Munk's technique and SOR (with $\omega = 0.4$) we have calculated the bistatic cross section at the specular angle ($\theta_r = 180° - \theta_i$) for three different incidence angles ($\theta_i = 90°$, 60°, 30°) in the y-z plane. The resulting values of the cross section σ vs iteration order are shown for the three angles, respectively, in Figs. 51-53. In all cases these values obtained by SOR agree very well with Munk's results, the greatest discrepancy (-0.45 dB) appearing at the $\theta_i = 30°$ incidence angle. This disagreement is thought to be inherent in the Munk solution for angles close to grazing.

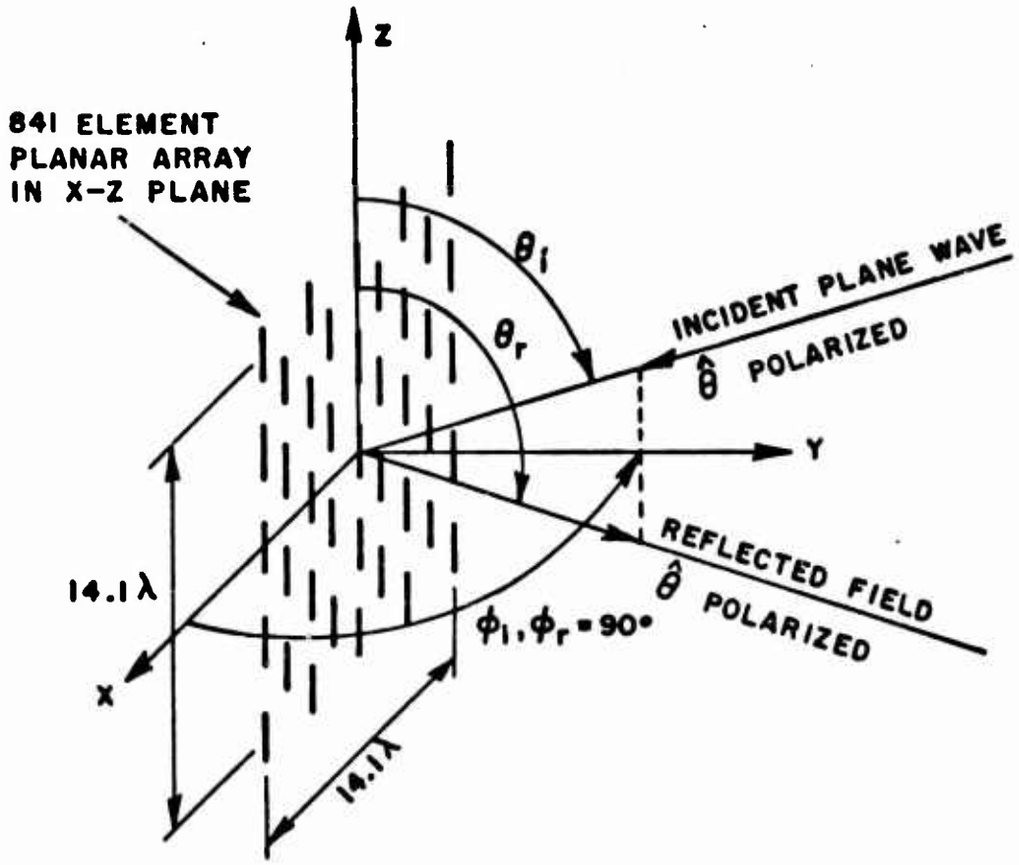


Figure 50. A planar array of resonant wires used as a check case.

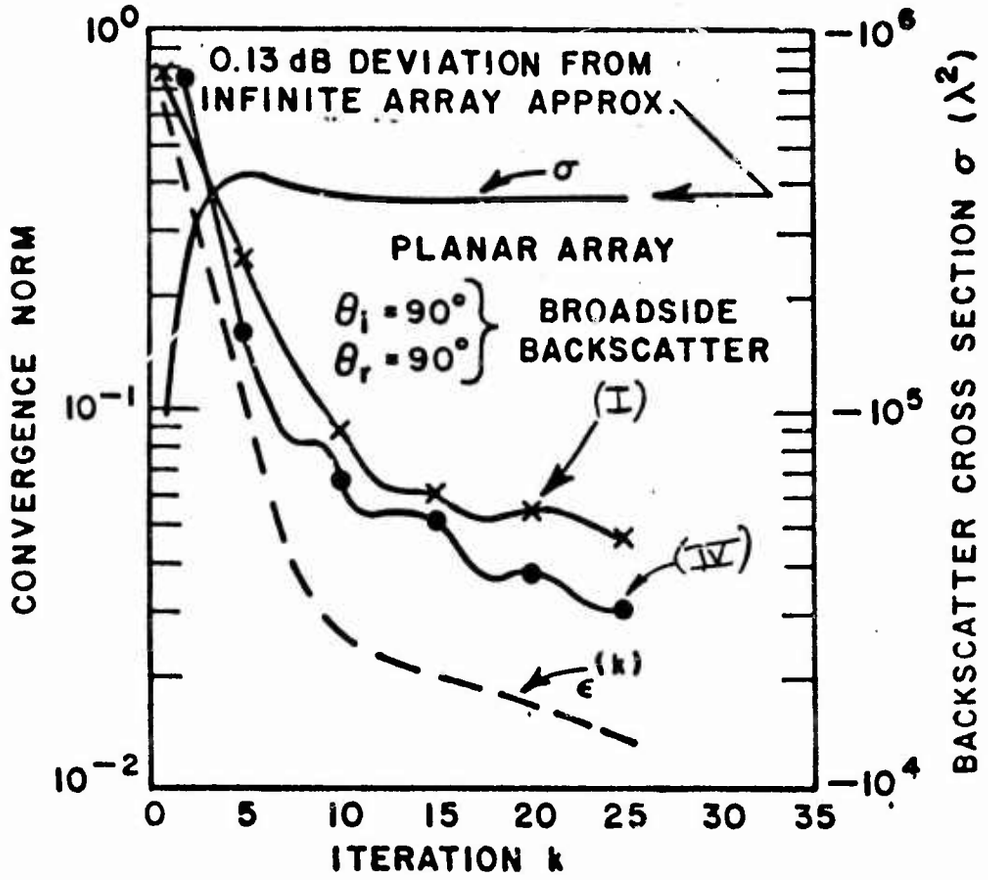


Figure 51. Broadside backscatter and comparison of convergence norms (I), $\varepsilon^{(k)}$ and (IV) versus iteration k for the periodic array of Fig. 50 using SOR with $\omega = 0.4$.

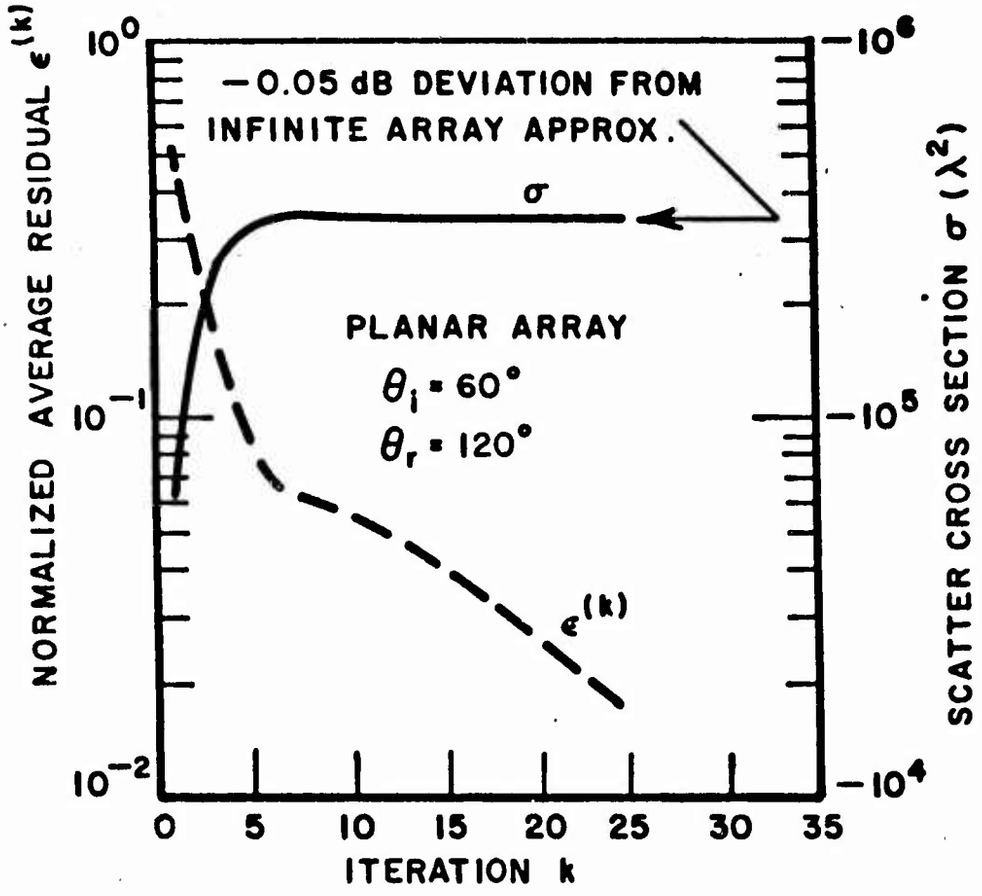


Figure 52. Specular bistatic cross section ($\beta = 60°$) and $\epsilon^{(k)}$ versus iteration k for periodic array of Fig. 50 using SOR with $\omega = 0.4$.

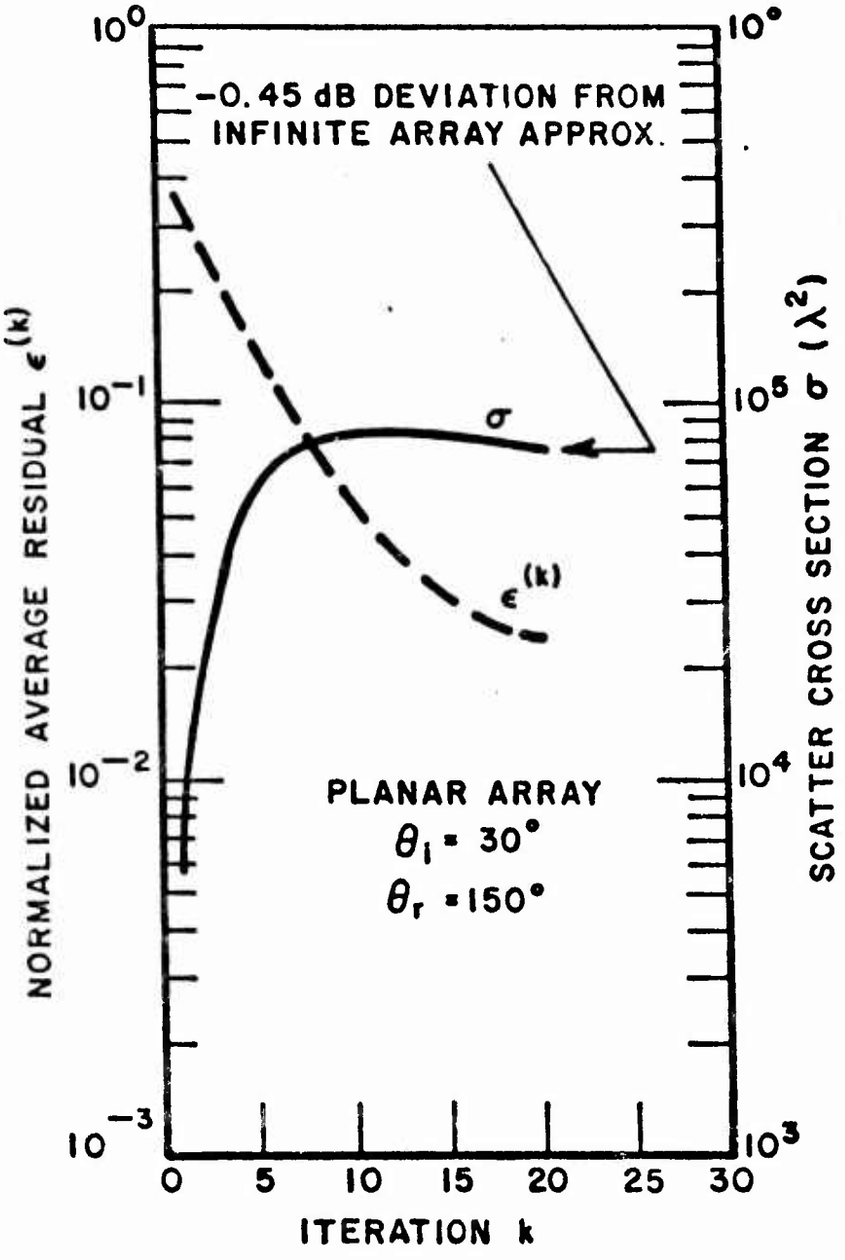


Figure 53. Specular bistatic cross section (β = 120°) and $\epsilon^{(k)}$ versus iteration k for periodic array of Fig. 50 using SOR with ω = 0.4.

In Figs. 51-53 certain "convergence norms" are also computed for each iterate k and displayed for comparison as quantitive measures of convergence. Four different norms appear in various figures of this section; three are based on the residuals $r^{(k)}$ defined by Eq. (50) and one is based on normed changes in solution $x^{(k)}$ similar to that defined by Eq. (52). A summary of these convergence norms is presented in the following table.

TABLE 6
CONVERGENCE NORM DEFINITIONS

Norms* based on $r^{(x)}$			Norm based on $x^{(k)}$
(I)	(II)	(III)	(IV)
$\frac{\|\|r^{(k)}\|\|_\infty}{\|\|b\|\|_\infty}$	$\underset{i}{\text{Max}} \frac{\|r_i^{(k)}\|}{\|b_i\|}$	$\frac{\|\|r^{(k)}\|\|_1}{N\|\|b\|\|_\infty} \equiv \varepsilon^{(k)}$	$\underset{i}{\text{Max}} \frac{\|x_i^{(k)} - x_i^{(k-1)}\|}{\min\{\|x_i^{(k)}\|, \|x_i^{(k-1)}\|\}}$

*See Appendix C of Ref. 38 for definitions of vector norms used.

The (I) and (IV) norms in this table were chosen strictly as representatives of the quantities appearing in Eqs. (51) and (52) while the (II) and (III) norms were defined with the physical problem in mind, i.e., (II) is a normalized measure of the residual indicating the boundary condition ($E_{TAN} = 0$) mismatch on one dipole in the array and (III) is a normalized average of all residuals for the whole array. The (III) norm will be denoted by $\varepsilon^{(k)}$ in all data presented in this section. Two points should be made here. One is that the $\varepsilon^{(k)}$ norm appears in all cases we have calculated to be the best balanced and most trustworthy; the other is that the behavior of $\varepsilon^{(k)}$ appears no different for random arrays of dipoles than it does for the periodic array. Since the results in cross section were very satisfactory for the periodic array, we infer that the similar behavior of $\varepsilon^{(k)}$ implies satisfactory results in cross section for the random arrays.

Very little information is found in the literature on suitable choices for SOR convergence measures for large complex system of equations such as those treated here. The convergence norm calculations are presented for the purpose of empirically determining just such a measure for these types of problems, e.g., one which might eventually be included in the computer programs to indicate a reliable stopping point in the iterative process. The normalized average residual $\varepsilon^{(k)}$ appears to possess

the uniform characteristics needed for this job. It also has the interpretation of being a measure of the "average" boundary condition $E_{TAN} = 0$ over the whole array. Other norms considered do not appear to indicate this same overall condition of the iterated solution but, tend to pin-point specific residuals or changes in the solution which, to a great extent, do not seriously affect the array scattering properties in the far field. Other "averaging" norms might do as well or better than $\varepsilon^{(k)}$; however, this study has concentrated on isolating only this one case which seems to be well suited for these types of problems.

No attempts were made in this study to determine optimum relaxation factors for SOR. An initial choice of ω was made at the outset of each new problem and if convergence was indicated, no changes were made; the exception is Fig. 67, where changes were made during the same iteration run with little observable effect.

SOR Solutions for Scattering by Large Clouds of Chaff Elements

The SOR iteration technique is used to solve Eq. (12b) for the currents induced in arrays of dipoles by plane wave fields of Eqs. (II-2) and (II-3). The $\hat{\theta}$ polarized backscatter σ and bistatic cross section for certain bistatic angles ($\beta = \pm 10°$ range) are calculated from these currents at each step k of the iteration.

Figure 54 considers an initial case of 100 dipoles in the random array configuration. The SOR technique ($\omega = 0.6$) can be compared to solution by a direct method (Cholesky); resulting solutions from both methods agree quite well (< 0.1 dB). This figure also includes calculated values for the four norms appearing in Table 6. The (II) and (IV) norms vary erratically, although both show overall decreases over the range of k. The (I) norm and $\varepsilon^{(k)}$ both show a consistent decrease, but only $\varepsilon^{(k)}$ is "monotonic" over the whole range.

Convergence characteristics of σ for a 500 dipole random array are indicated in Fig. 55 for SOR iteration using two values of relaxation factor, ω=0.5 and ω=0.4. Only $\epsilon^{(k)}$ was calculated in this case. The ω=0.5 case appears to converge faster (steeper slope on $\epsilon^{(k)}$) in the early stages (k = 1 to k $\approx$ 20), however better overall convergence was obtained for ω=0.4. Figure 56 shows a sample of the bistatic cross section pattern for k = 10, 20 and 36. This figure indicates the degree of convergence obtained in this $\beta = \pm 10°$ sector at the corresponding stage in the iteration. The convergence of the

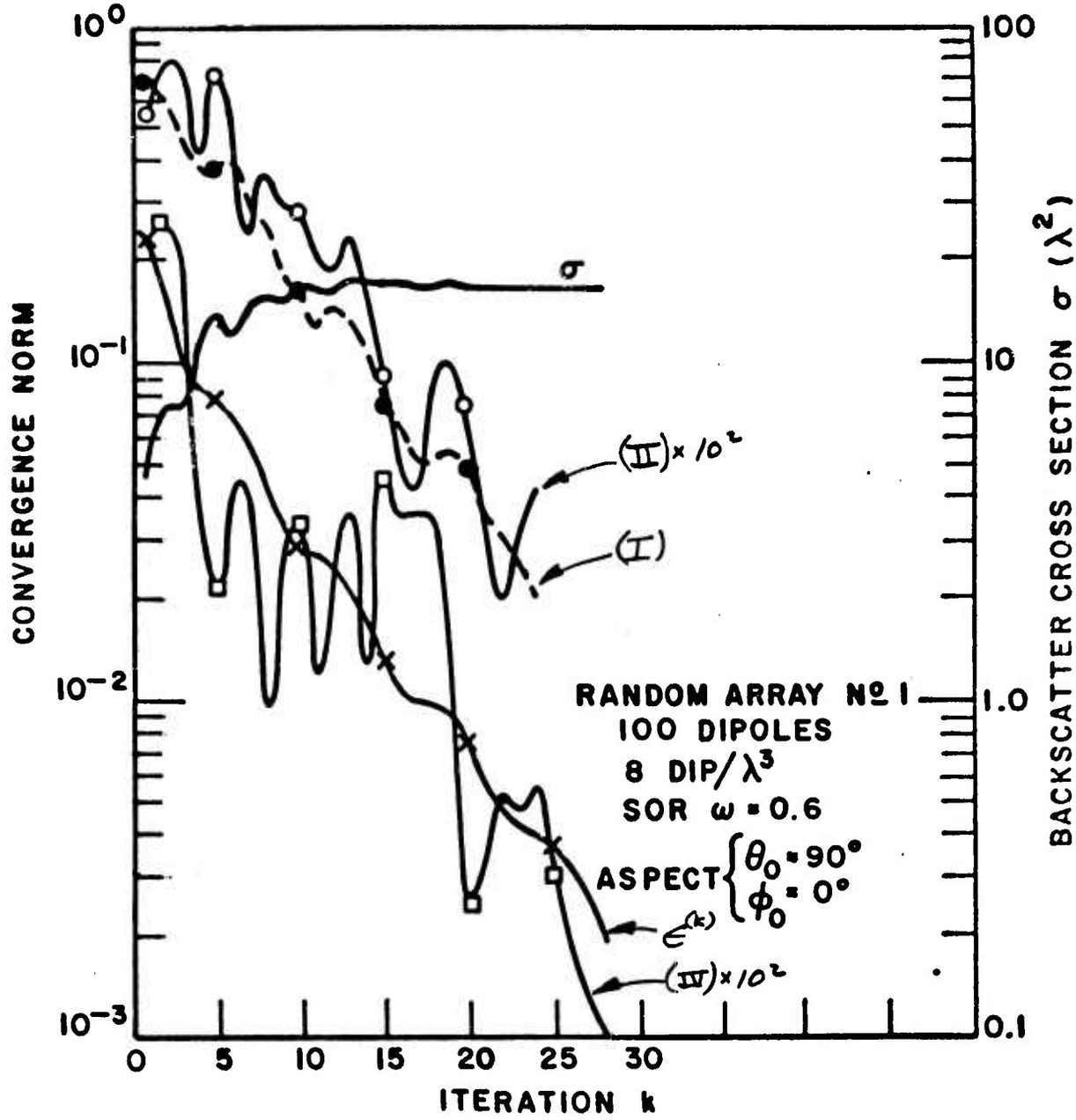


Figure 54. Backscatter cross section and convergence norms (I), (II), $\varepsilon^{(k)}$ and (IV) versus iteration k for 100 dipole random array using SOR with $\omega = 0.6$.

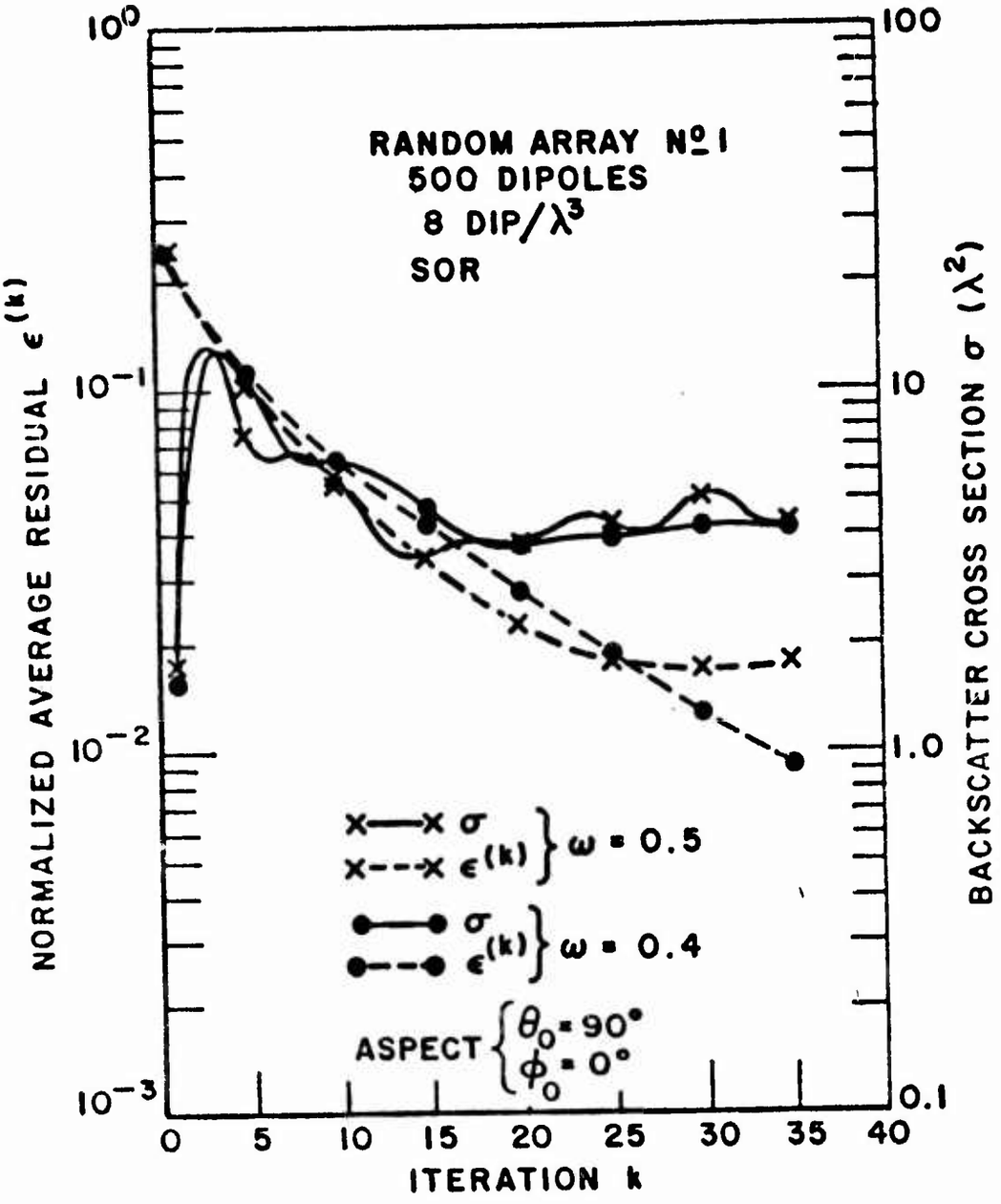


Figure 55. Backscatter cross section and $\varepsilon^{(k)}$ versus iteration k for 500 dipole random array (8 dip/λ^3) using SOR with $\omega = 0.4$ and $\omega = 0.5$.

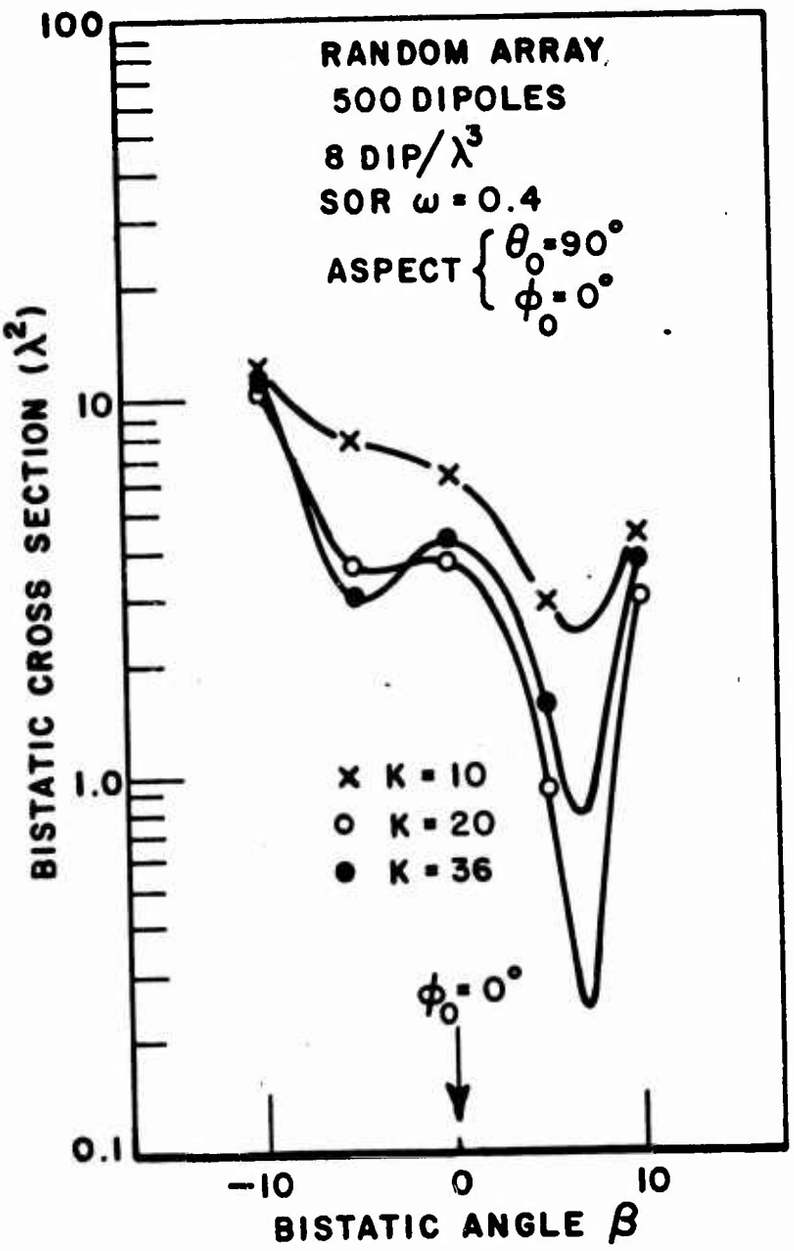


Figure 56. Bistatic cross section pattern for 500 dipole random array for k = 10, 20 and 36 using SOR with $\omega = 0.4$.

bistatic pattern seems to be best in the larger amplitudes and for k > 20, major changes occur only in the null regions.

Figure 57 is the first of a series of 10 figures showing five SOR iterated solutions (ω=0.4) for a single 1000 dipole random array (#1). These figures alternately show σ backscatter and bistatic cross section for five aspect angles of the incident wave. Each of these cases corresponds to a new "b" vector for the right hand side of Eq. (12b).

Figure 57 indicates convergence of σ backscatter and shows a comparison of the (I) norm and $\varepsilon^{(k)}$ for $\theta_0 = 90°$, $\phi_0 = 0°$. The (I) norm in this case has lost all resemblance of being a monotonically decreasing norm while $\varepsilon^{(k)}$ continues to show a smooth decrease with increasing k. The curve for σ backscatter in this case converges smoothly to the value $\sigma \approx 90\lambda^2$, a rather high value for these random arrays. Figure 58 displays a portion of the bistatic pattern ($\beta=\pm 10°$). Here, the amplitude changes on the peak are less than 1 dB for k > 5, while the null depth changes are more than 10 dB over this same interval.

Figure 59 considers a new aspect angle (θ_0=90°, ϕ_0 = 10°) for the same random array. Here, σ backscatter shows somewhat irregular convergence as compared to the previous aspect; however, the same smooth decrease in $\varepsilon^{(k)}$ is omnipresent. The bistatic patterns for k = 6, 15, 30 and 42 of Fig. 60 indicate considerable change is taking place over this range. The largest changes, however, occur in the null regions and peak amplitude regions show the lesser changes.

Figures 61 and 62 indicate σ and bistatic patterns for $\theta_0 = 90°$, $\phi_0 = 20°$. The $\varepsilon^{(k)}$ norm in Fig. 61 again shows monotonic improvement in average residuals and Fig. 62 indicates essentially converged bistatic patterns for k > 15 with changes less than 2 dB in peak smplitude and less than 3 dB in the null region. Oscillations of σ in Fig. 61 are less than plus or minus 1 dB and decreasing for k > 25.

Figures 63 and 64 show σ, $\varepsilon^{(k)}$ and bistatic cross section for $\theta_0 = 90°$, $\phi_0 = 30°$. Fluctuations in σ for k > 16 are less than 2 dB and $\varepsilon^{(k)}$ is again smoothly decreasing. Bistatic patterns appear to change very little for k > 30.

Figures 65 and 66 are the last figures showing data for large random array #1 ($\theta_0 = 90°$, $\phi_0 = 40°$). Convergence norms (I) and (IV) are included in Fig. 65 with $\varepsilon^{(k)}$. Although, norms (I) and (IV) do not have the smooth decrease shown by $\varepsilon^{(k)}$, it appears that an average curve of (IV) over this range of k would repeat the trend indicated by $\varepsilon^{(k)}$. The oscillatory nature of σ backscatter is confirmed in the bistatic pattern curves of

Fig. 66. The final bistatic curve (k = 45) is bounded by the k = 25 and k = 35 patterns and again, largest changes occur in the null region.

Data in the following four figures (Figs. 67-70) were calculated for a second large random array (#2) with the same average density (8 dip/λ^3) and number of dipoles (N = 1000) as in the previous case. The new array was generated with a new initialization of the random positioning programs. The two cases considered for this new array correspond to aspect angles θ_o = 90°, ϕ_o = 0° and 10°.

Figure 67 shows σ and $\varepsilon^{(k)}$ data calculated for θ_o = 90°, ϕ_o = 0° case and Fig. 68 presents the corresponding bistatic patterns. Four values of relaxation factor (ω = 0.4, 0.35, 0.3 and 0.25) were used in this case with the initial iteration performed with ω = 0.4. The results for ω = 0.4 are indicated in Fig. 67 by the marginally convergent curve. The iteration was then restarted (k = 1) with ω = 0.35 and continued through k = 12; at which time, ω was changed and the iteration carried out to k = 30 for ω = 0.3; then ω was again changed this time to ω = 0.25 and the process carried out to the final iteration k = 61. The reason for changing ω during the same iteration run was an attempt to isolate variations, if any, in $\varepsilon^{(k)}$ which might correspond to different values of ω. No recognizable changes were noted; in fact, the iteration appeared to be converged for all k > 30 (ω = 0.3, 0.25) and the bistatic patterns in Fig. 68 confirm this to a great extent.

A second aspect angle (θ_o = 90°, ϕ_o = 10°) is considered in Figs. 69 and 70. Here, SOR was restarted three times for random array #2 with ω = 0.3, 0.25 and 0.2. The two cases ω = 0.3 and 0.25 were not convergent as Fig. 69 shows and ω had to be reduced to ω = 0.2 to obtain the one convergent case indicated in the figure. Figure 70 shows bistatic patterns for k = 20, 30 and 36 for the converging case. The largest changes in these patterns again occur in the null regions.

Three additional figures are included in this section (Figs. 71, 72, and 73) comparing convergence characteristics of σ backscatter, $\langle\sigma\rangle$ (the bistatic cross section average over $\beta = \pm 10°$) and σ_T (total scatter cross section from the forward scattering theorem reviewed in Appendix K). Figure 71 presents σ_T and $\langle\sigma\rangle$ with the σ curve previously calculated in Fig. 57. The bistatic average $\langle\sigma\rangle$ in this case shows little, if any, improvement over the original σ curve; however, σ_T is converged as early as k = 5. The rapid convergence of σ_T indicates that apparently the total power scattered in all directions by the random array is insensitive to the computed currents, compared to either σ or $\langle\sigma\rangle$.

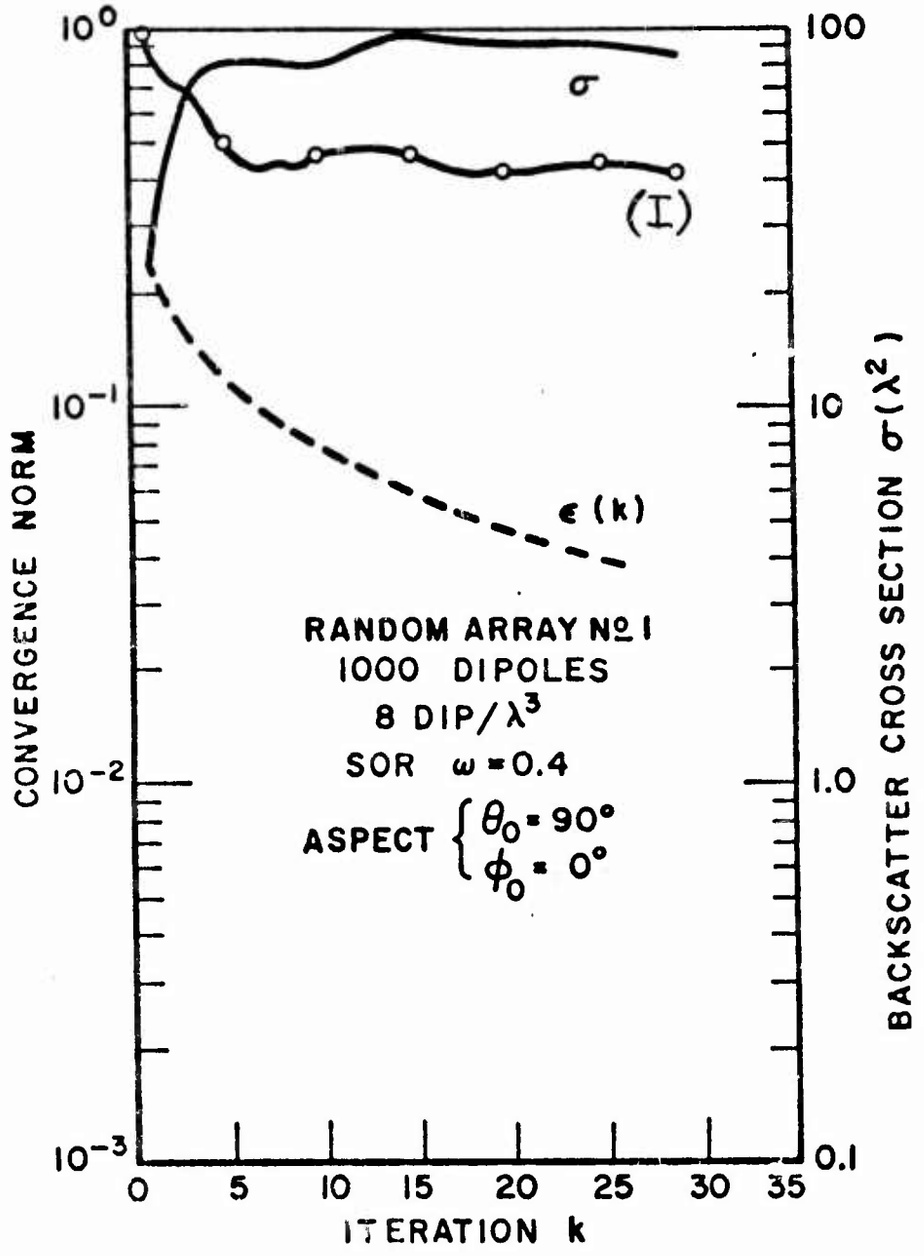


Figure 57. Backscatter cross section and convergence norms (I) and $\epsilon^{(k)}$ versus iteration k for 1000 dipole random array #1 (θ_0=90°, ϕ_0=0°) using SOR with ω = 0.4.

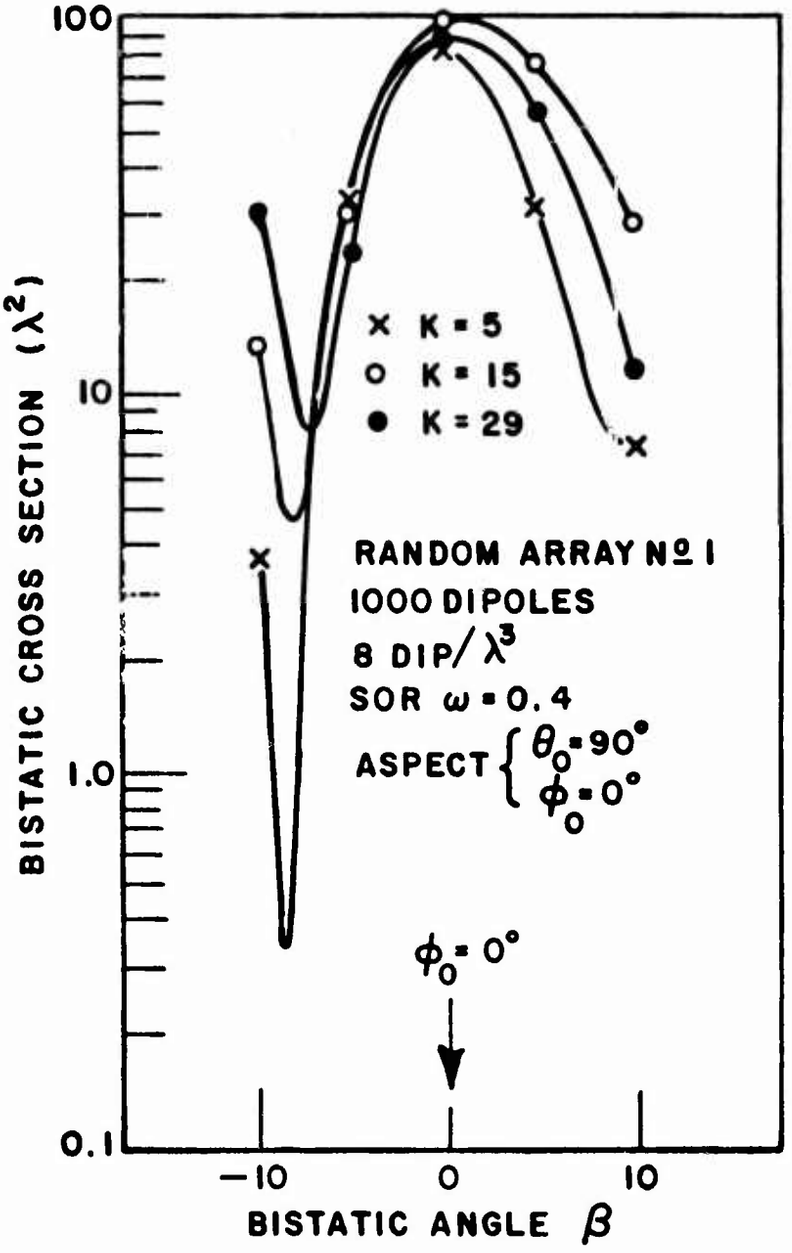


Figure 58. Bistatic cross section pattern for 1000 dipole random array #1 (θ_0 = 90°, ϕ_0 = 0°) at k = 5, 15 and 29 using SOR with ω = 0.4.

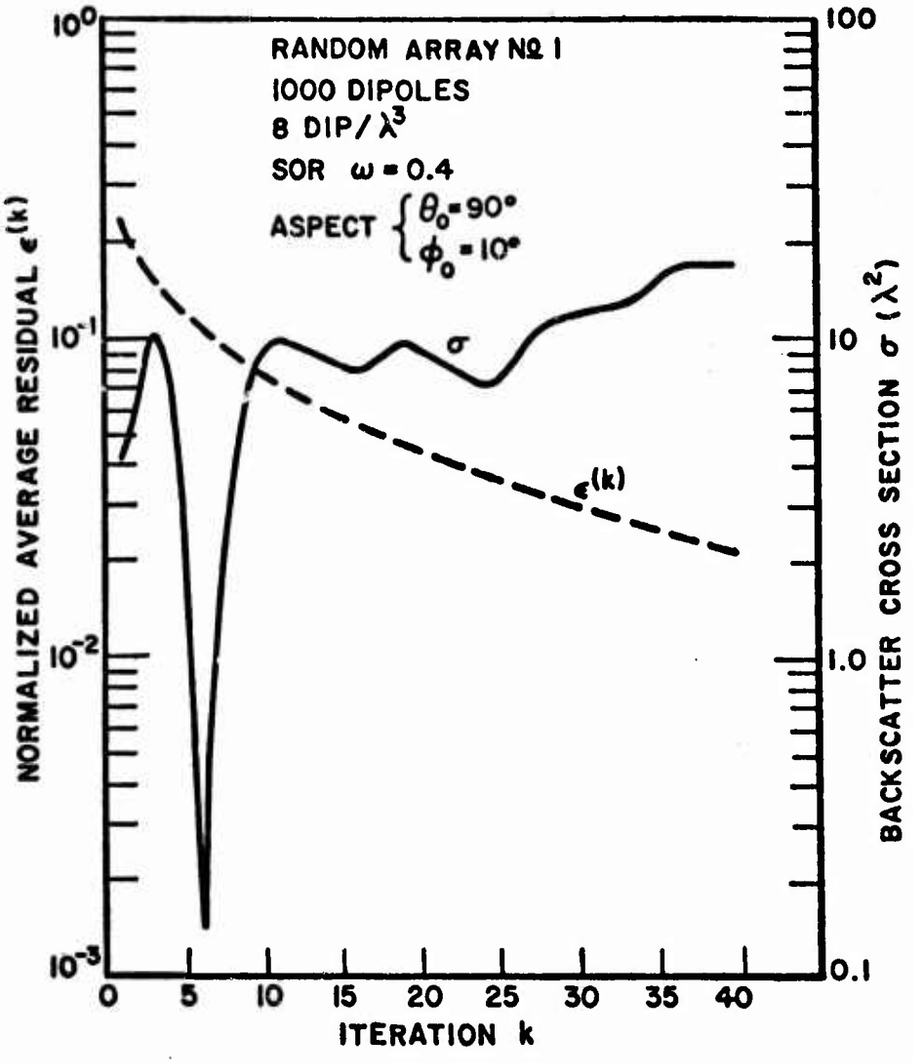


Figure 59, Backscatter cross section and $\epsilon^{(k)}$ versus iteration k for 1000 dipole random array #1 ($\theta_0 = 90°$, $\phi_0 = 10°$) using SOR with $\omega = 0.4$.

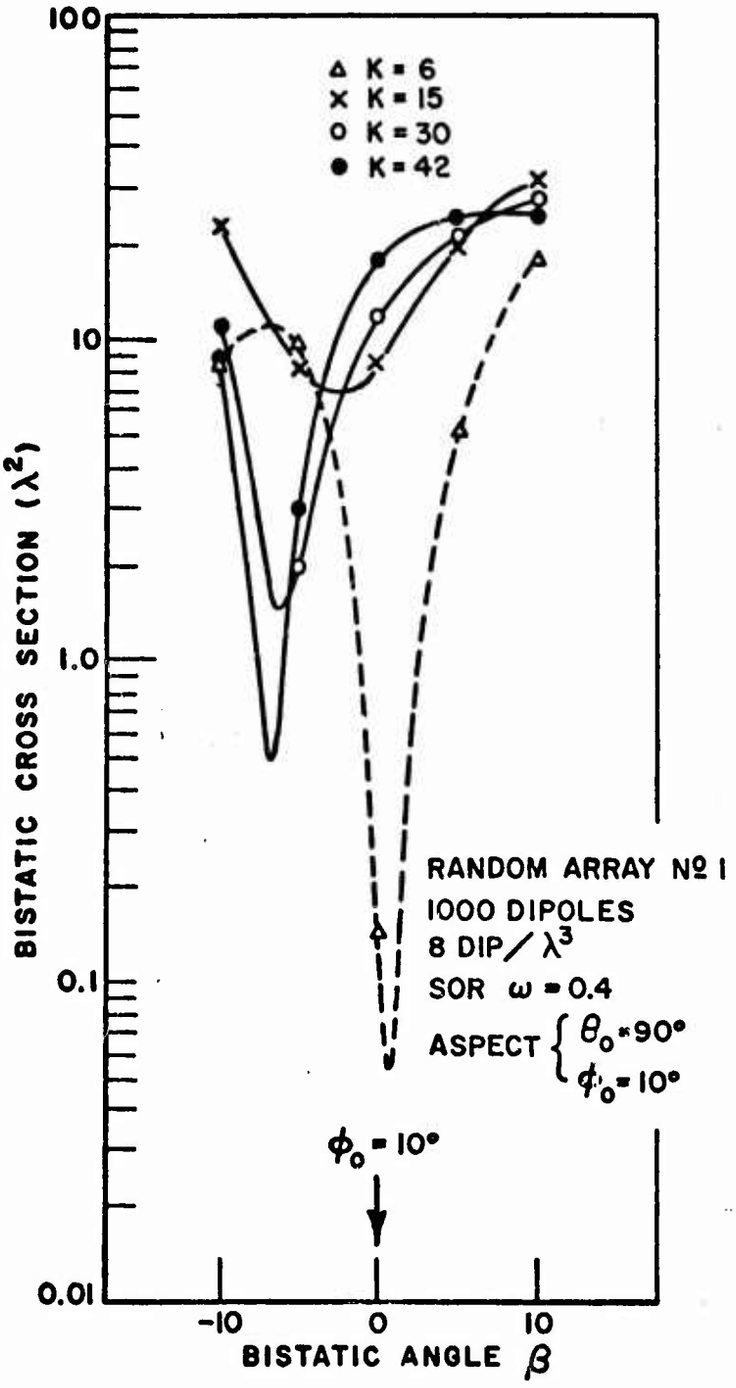


Figure 60. Bistatic cross section pattern for 1000 dipole random array #1 (θ_0 = 90°, ϕ_0 = 10°) at k = 6, 15, 30 and 42 using SOR with ω = 0.4.

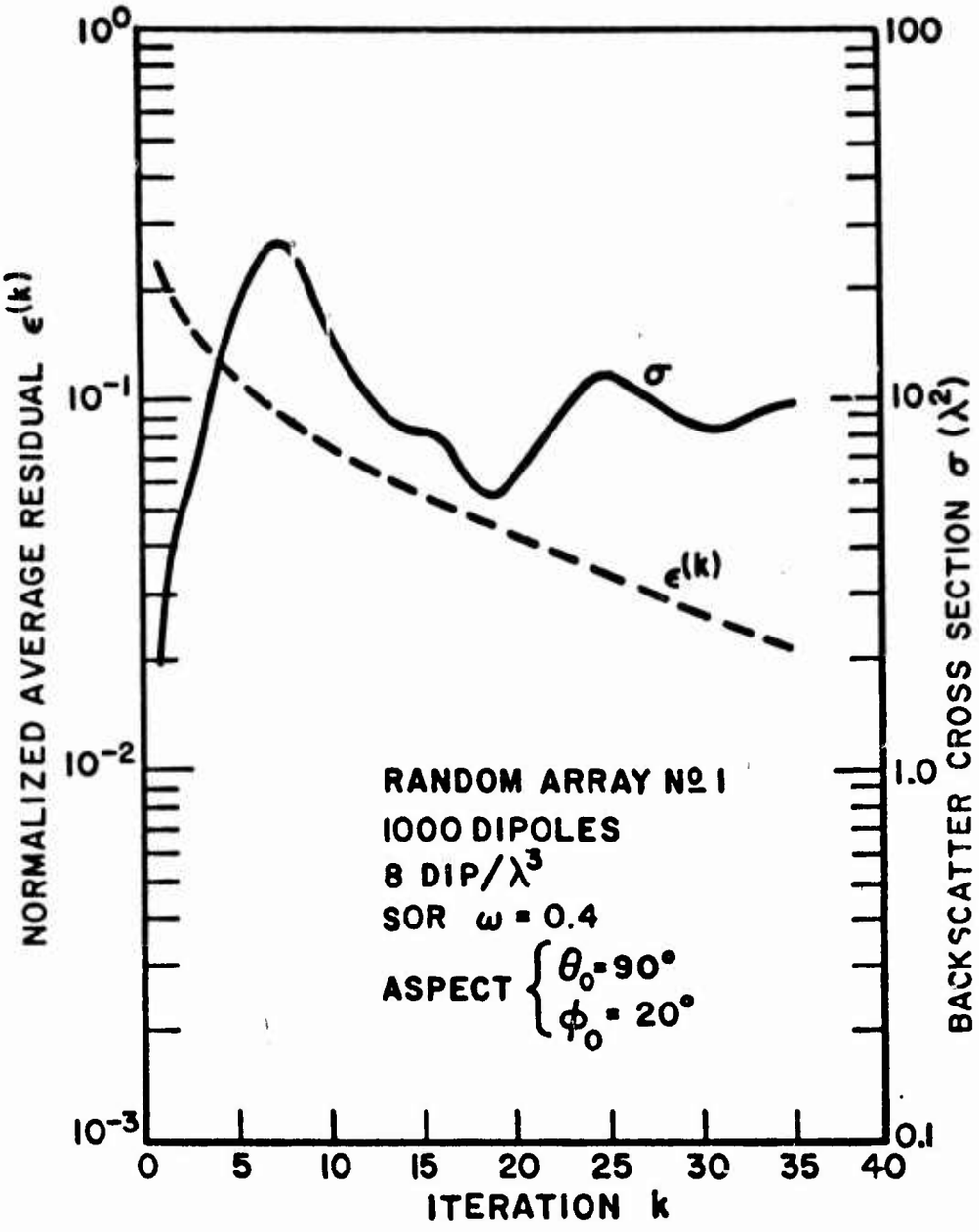


Figure 61. Backscatter cross section and $\epsilon^{(k)}$ versus iteration k for 1000 dipole random array #1 (θ_0 = 90°, ϕ_0 = 20°) using SOR with ω = 0.4.

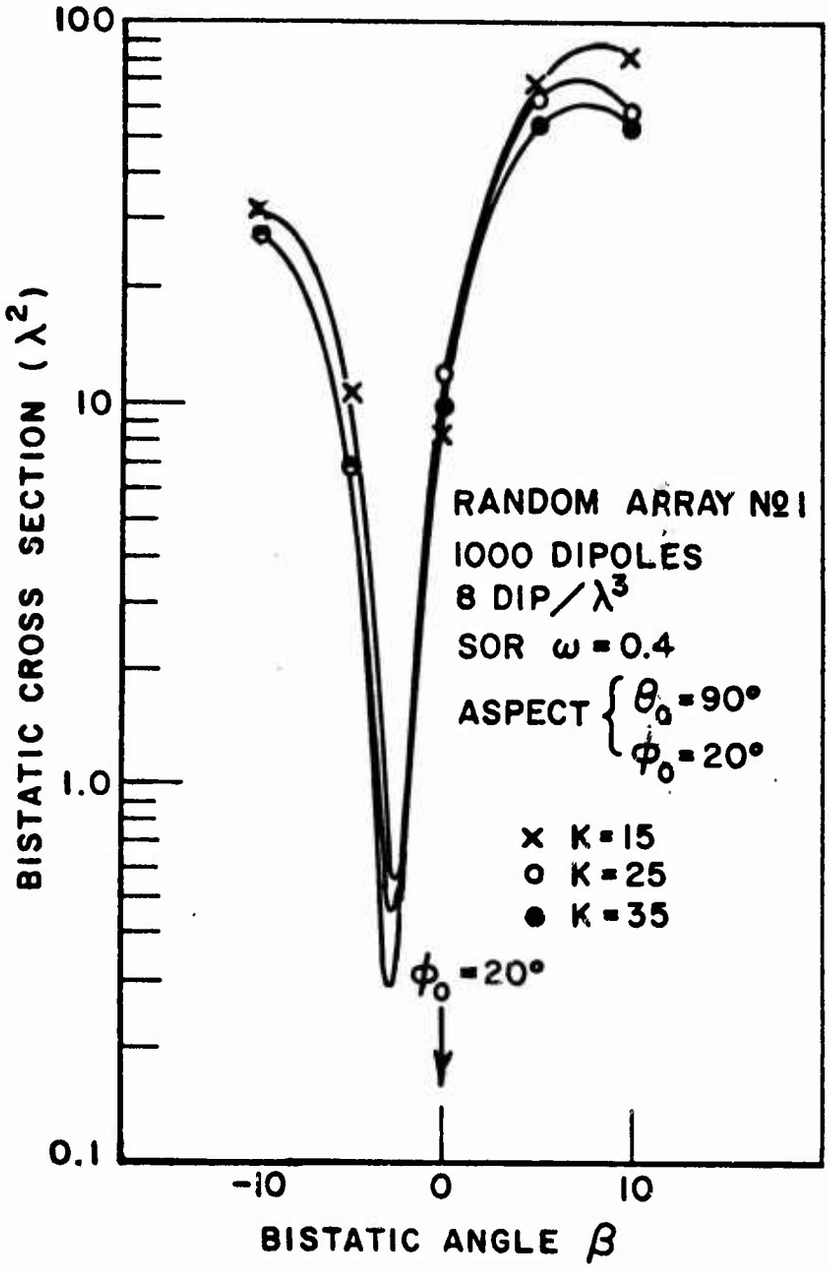


Figure 62. Bistatic cross section pattern for 1000 dipole random array #1 ($\theta_0 = 90°$, $\phi_0 = 20°$) at k = 15, 25 and 35 using SOR with $\omega = 0.4$.

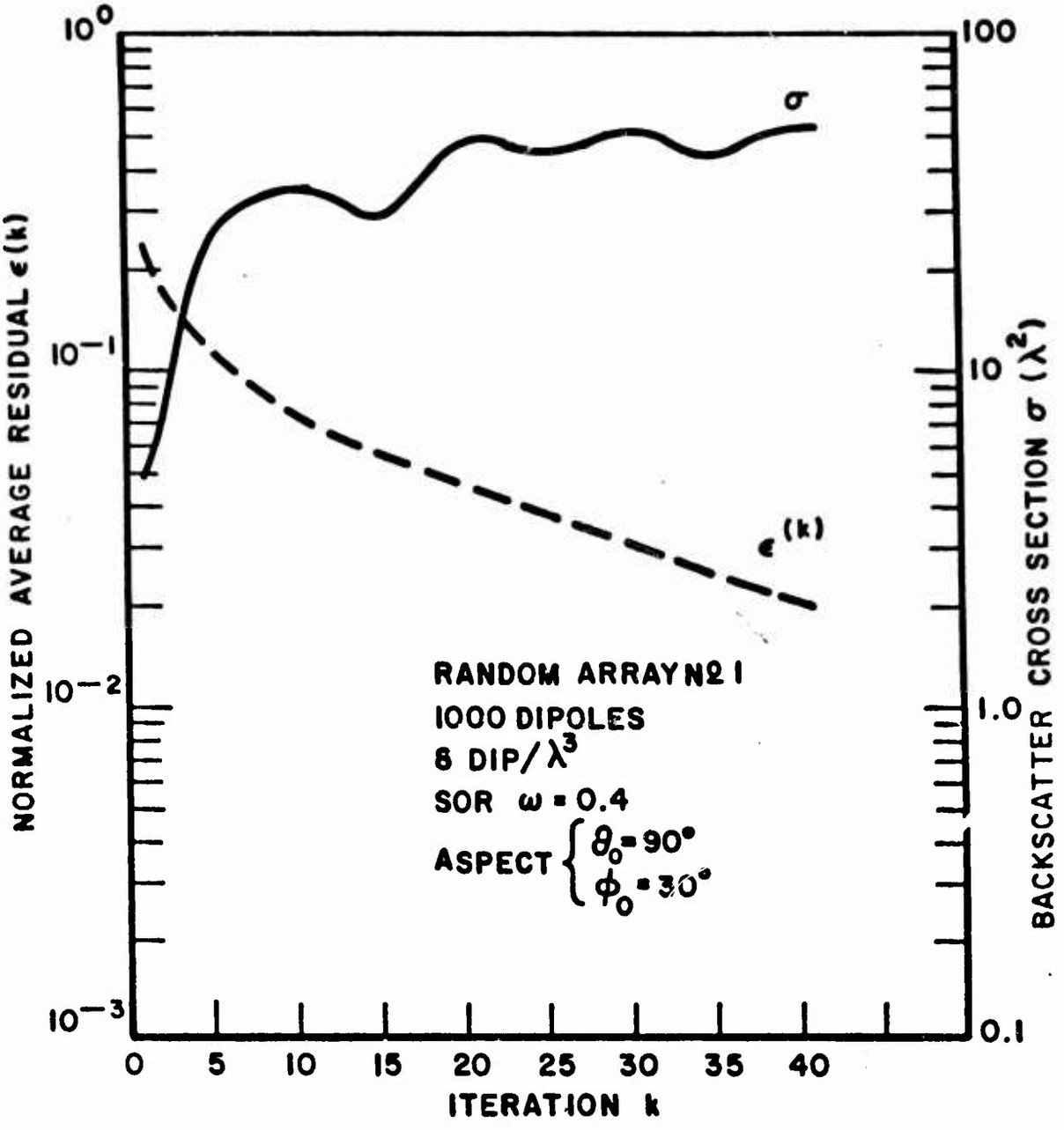


Figure 63. Backscatter cross section and $\varepsilon^{(k)}$ versus iteration k for 1000 dipole random array #1 (θ_o = 90°, ϕ_o = 30°) using SOR with ω = 0.4.

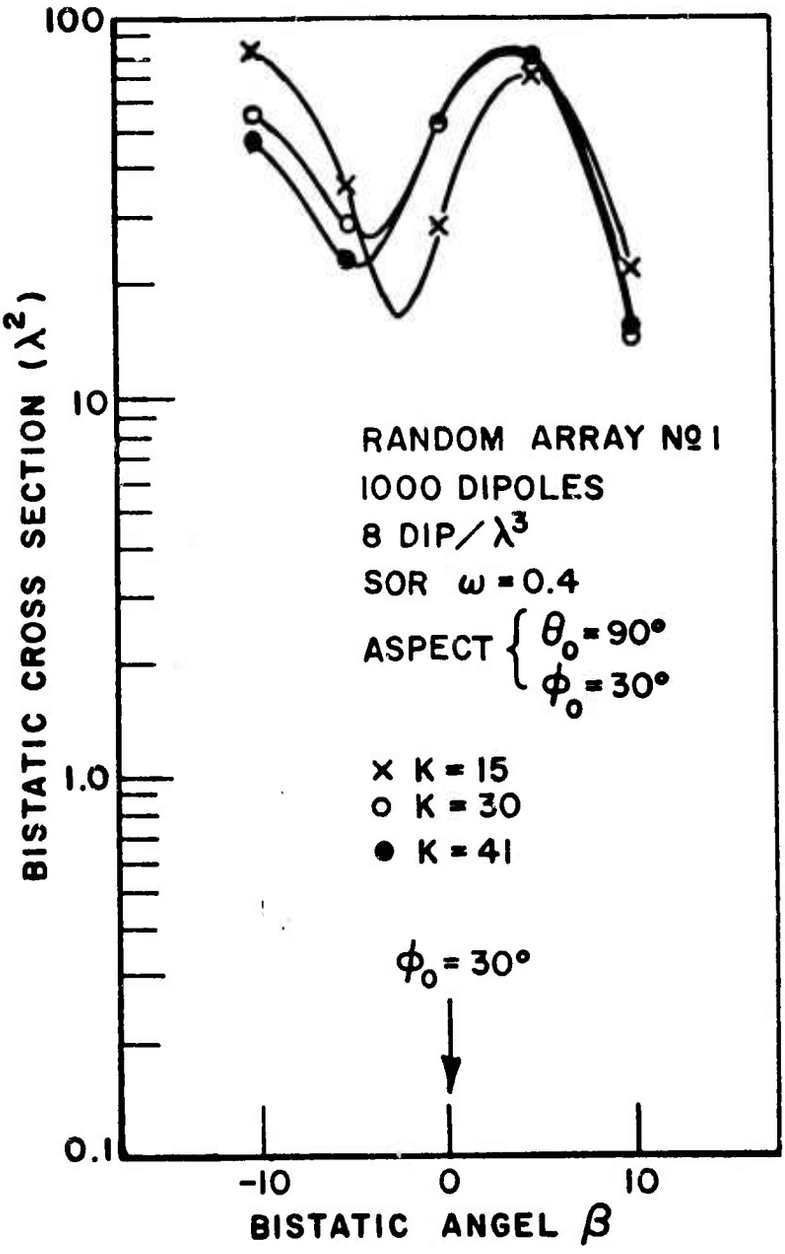


Figure 64. Bistatic cross section pattern for 1000 dipole random array #1 ($\theta_0 = 90°$, $\phi_0 = 30°$) at k = 15, 30 and 41 using SOR with $\omega = 0.4$.

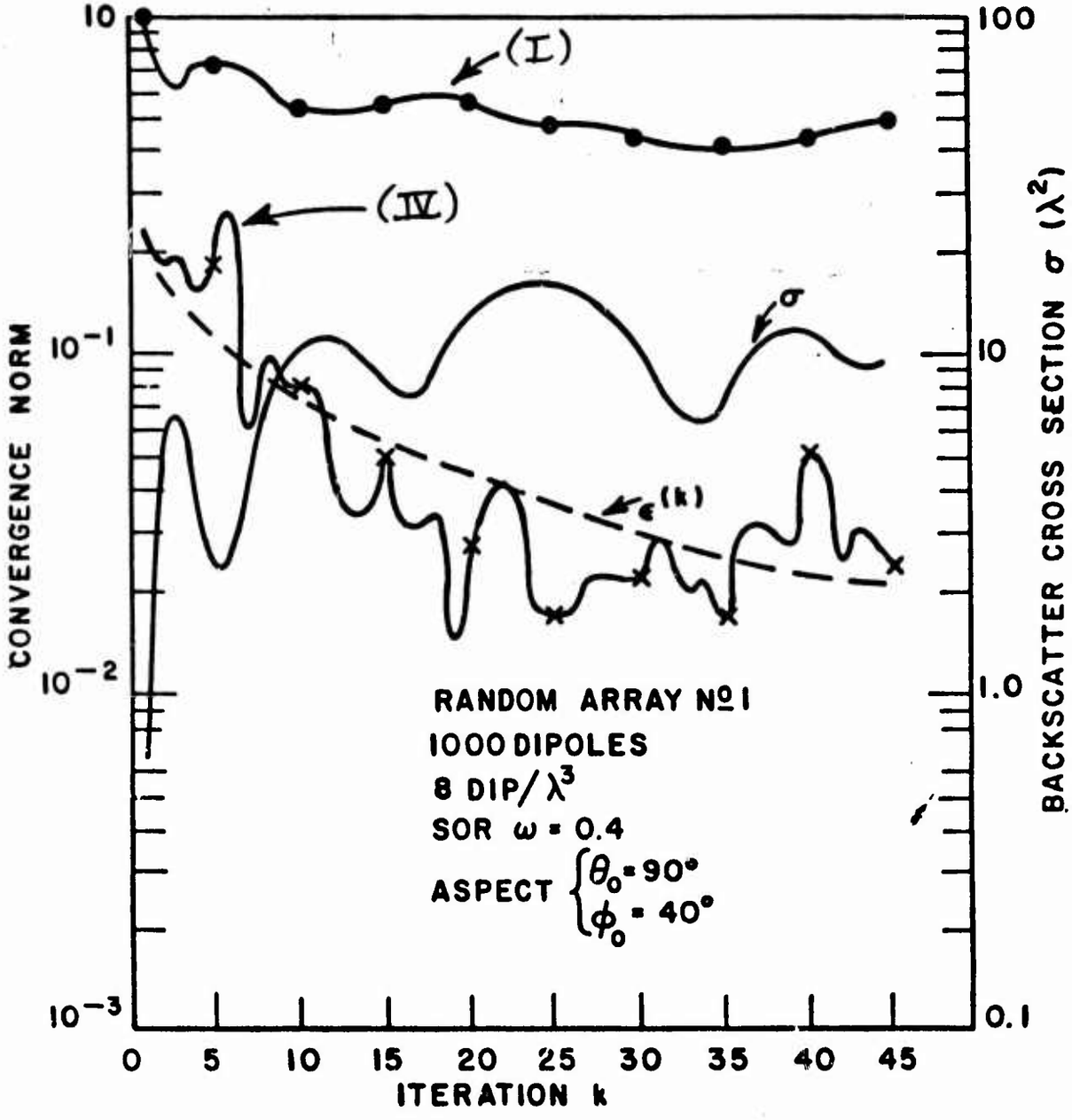


Figure 65. Backscatter cross section and convergence norms (I), $\varepsilon^{(k)}$ and (IV) versus iteration k for 1000 dipole random array #1 (θ_0=90°, ϕ_0=40°) using SOR with ω=0.4.

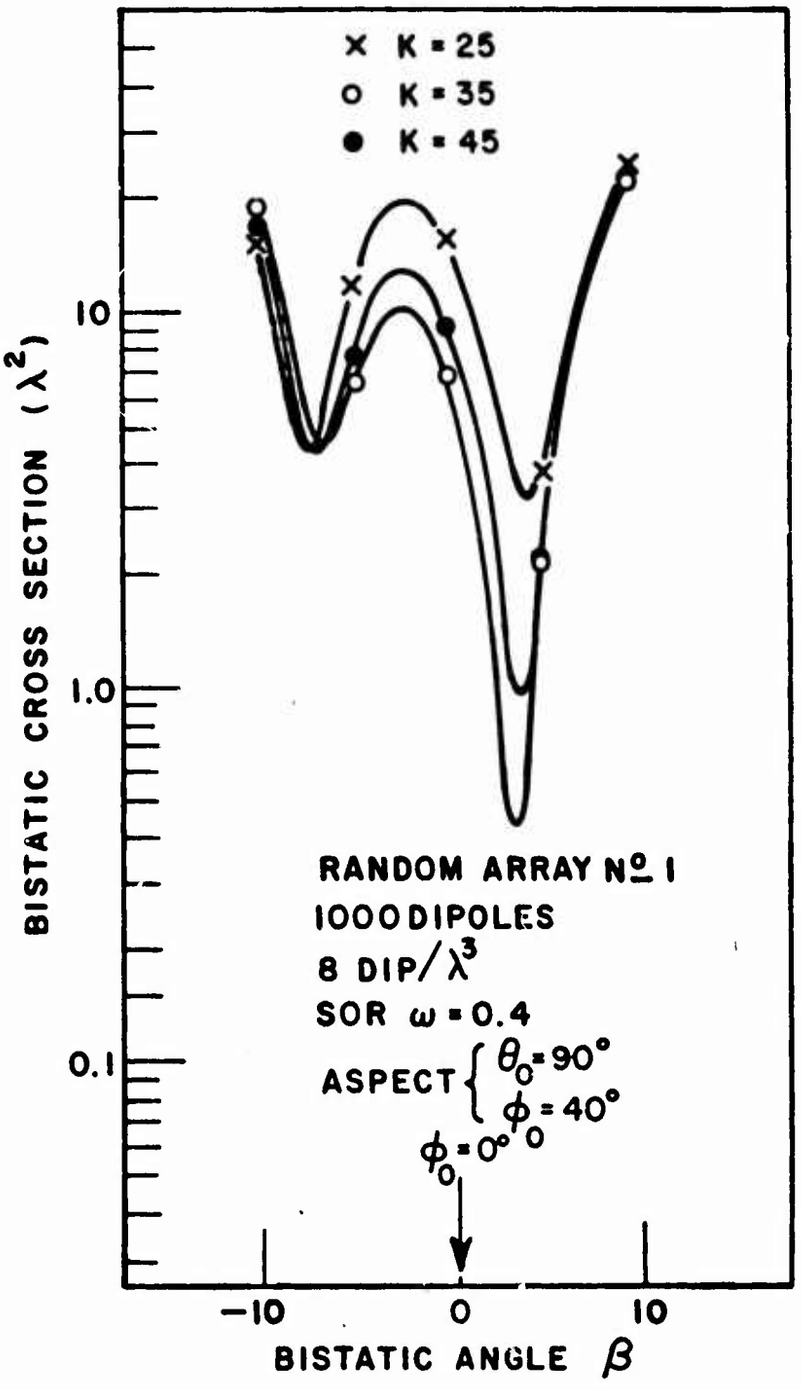


Figure 66. Bistatic cross section pattern for 1000 dipole random array #1 (θ_0 = 90°, ϕ_0 = 40°) at k = 25, 35 and 45 using SOR with ω = 0.4.

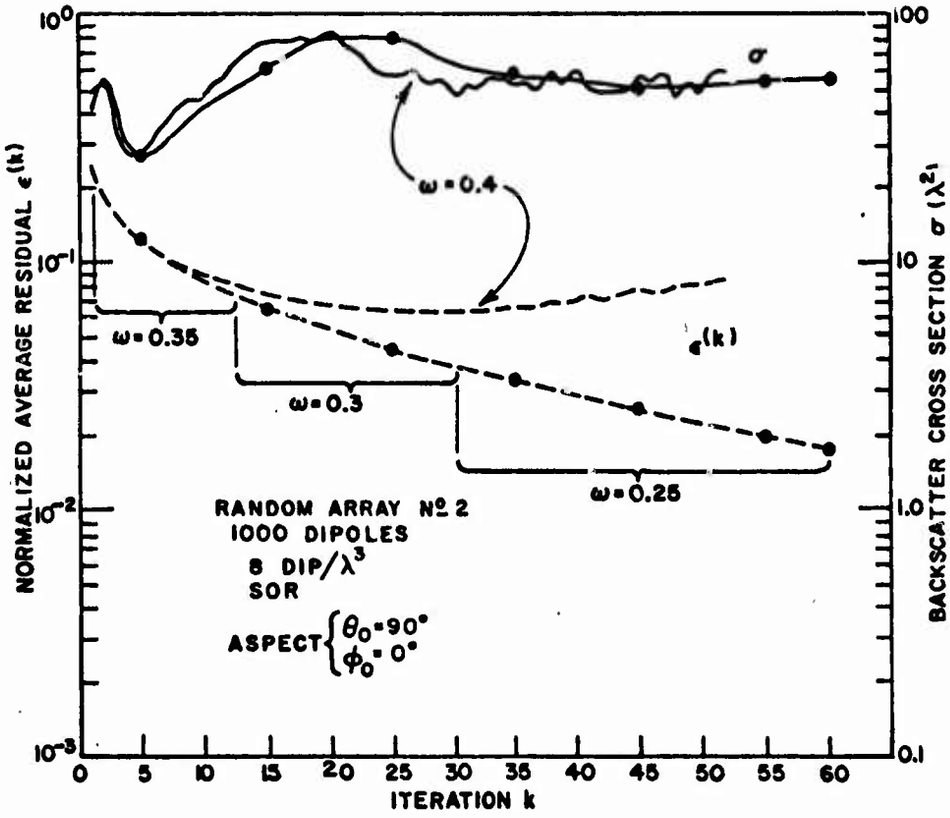


Figure 67. Backscatter cross section and $\epsilon^{(k)}$ versus iteration k for 1000 dipole random array #2 (θ_0 = 90°, ϕ_0 = 0°) using SOR with ω = 0.25, 0.3, 0.35 and 0.4.

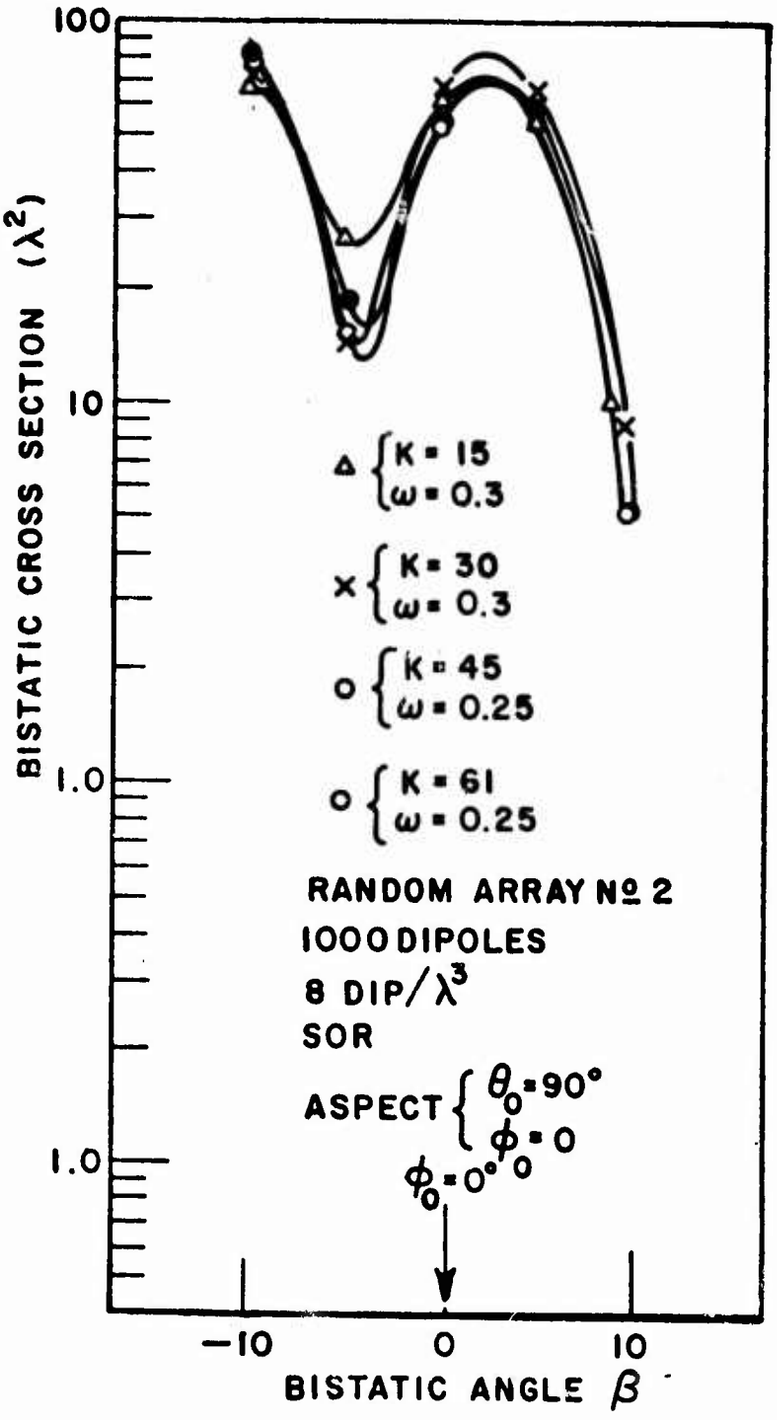


Figure 68. Bistatic cross section pattern for 1000 dipole random array #2 (θ_o = 90°, ϕ_o = 0°) at k = 15, 30, 45 and 61 using SOR with ω = 0.3 and 0.25.

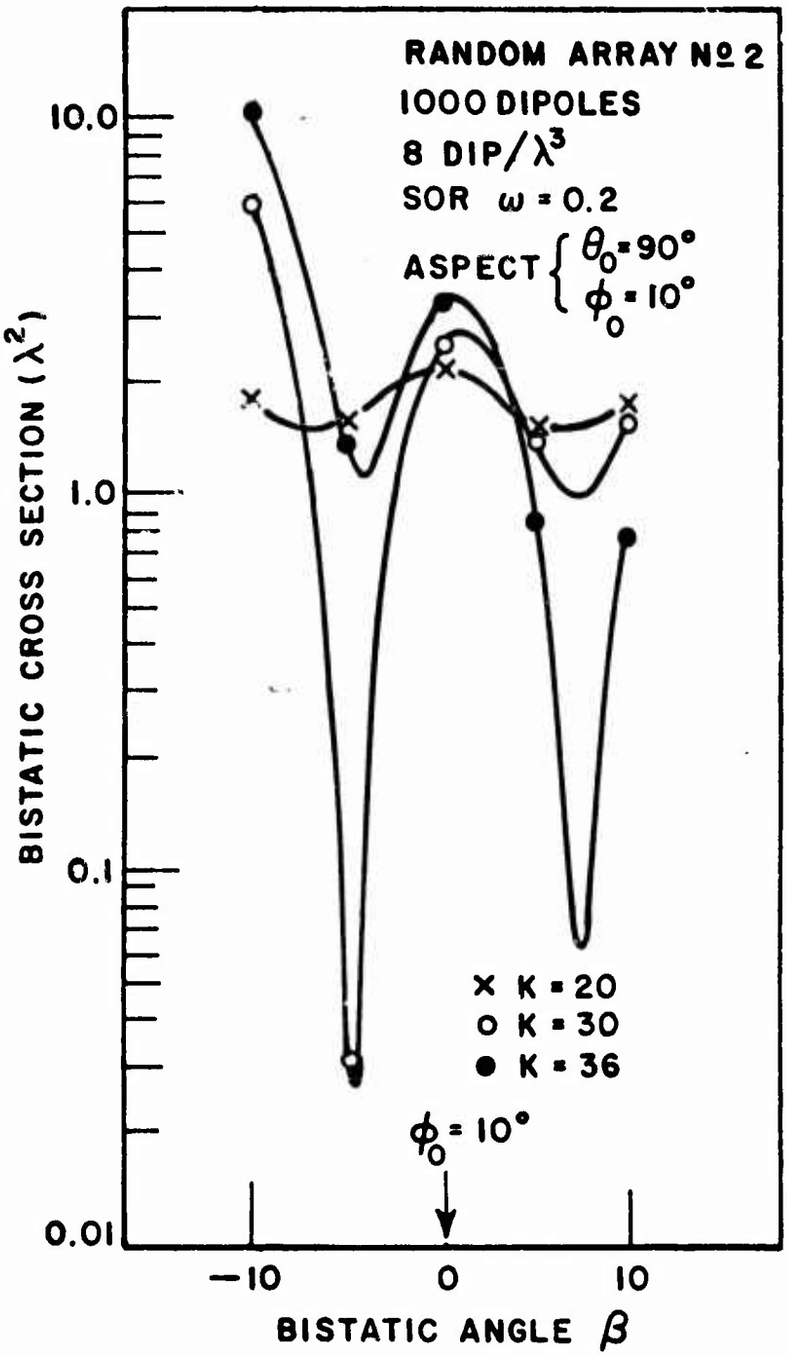


Figure 70. Bistatic cross section pattern for 1000 dipole random array #2 ($\theta_0 = 90°$, $\phi_0 = 10°$) at k = 20, 30 and 36 using SOR with $\omega = 0.2$.

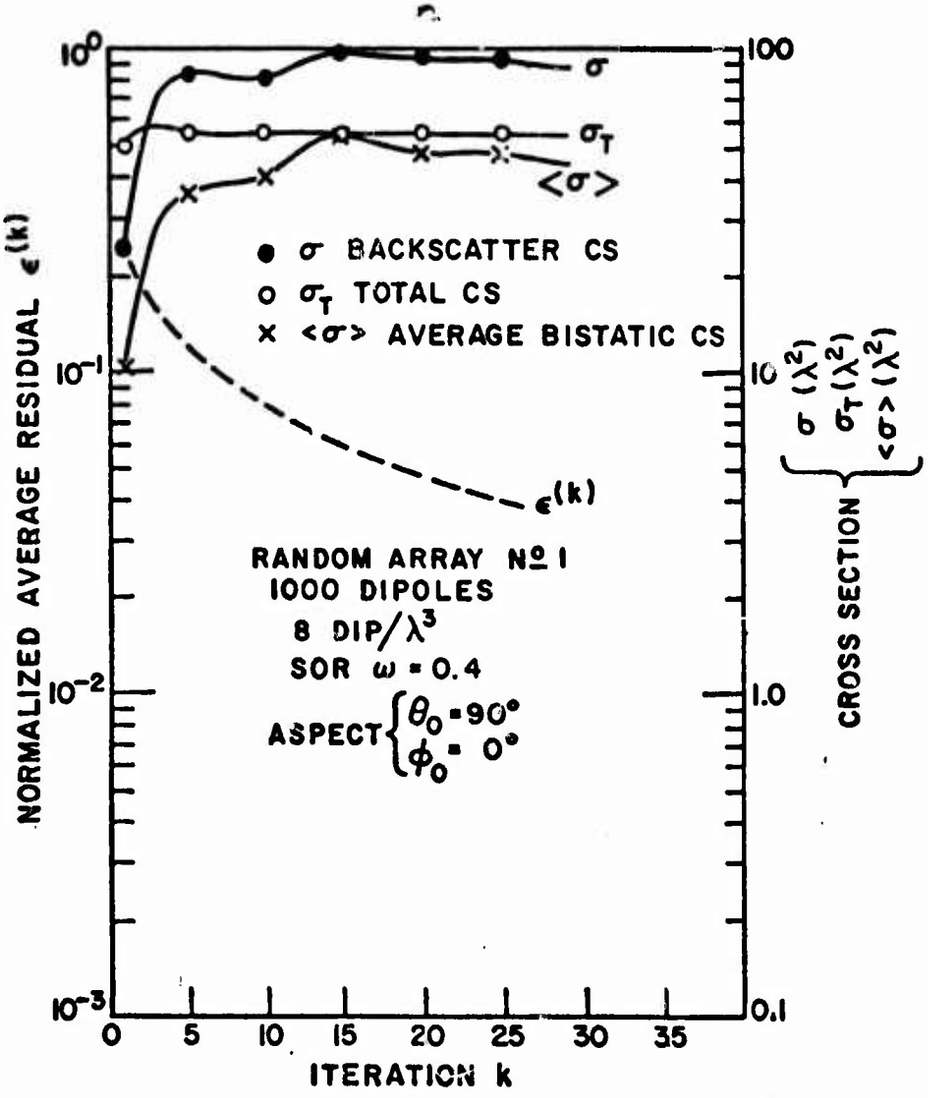


Figure 71. Backscatter, total and average bistatic cross sections versus iteration k for random array #1 (Fig. 57).

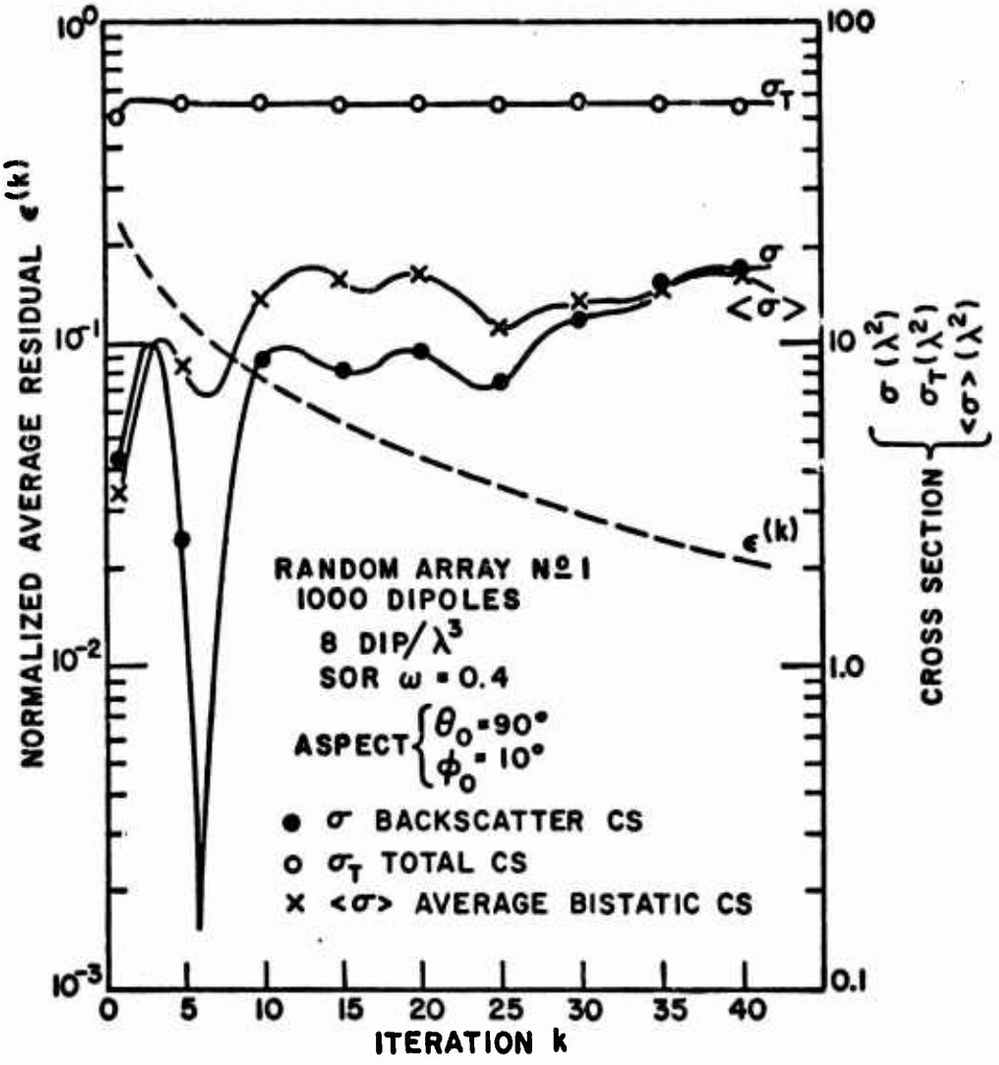


Figure 72. Backscatter, total and average bistatic cross sections versus iteration k for random array #1 (Fig. 59).

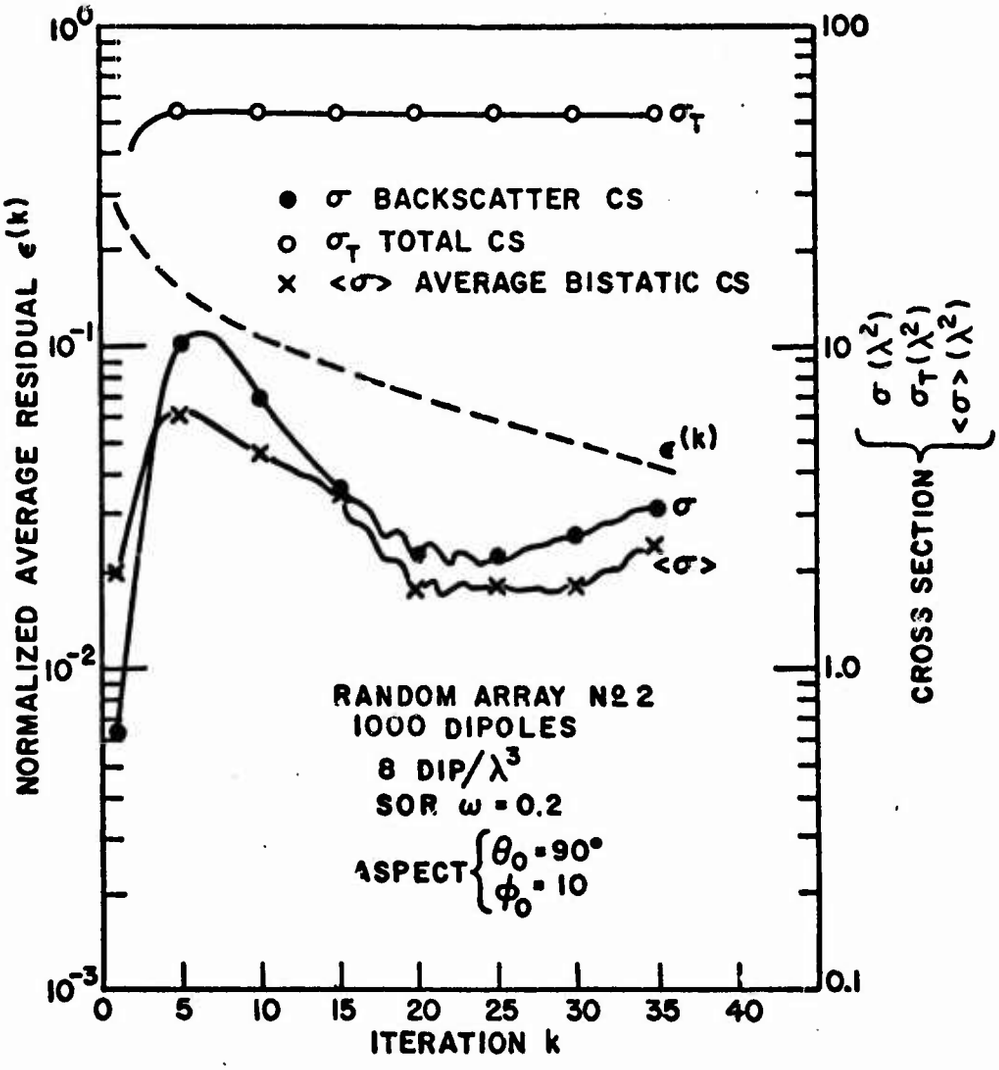


Figure 73. Backscatter, total and average bistatic cross sections versus iteration k for random array #2 (Fig. 69).

Figure 72 illustrates σ_T, $\langle\sigma\rangle$, and the previously computed σ data from Fig. 59. The average $\langle\sigma\rangle$ in this case does smooth out the large dip in the vicinity of k = 6, however, its overall convergence characteristics are no improvement over σ (unaveraged). Total cross section σ_T for k > 5 has converged to very nearly the same final σ_T value in the previous figure (same array).

Values of σ_T and $\langle\sigma\rangle$ are compared in Fig. 73 with σ from Fig. 69 for random array #2. Both σ and $\langle\sigma\rangle$ in this case have similar characteristics, however, neither one shows significant improvement in convergence rate over the other. Note, the converged σ_T for this new case (array #2) is essentially the same as that obtained for random array #1).

An important result brought out by all these data is that convergence rates for many cases appear to be functions of excitation; i.e., given matrix A (e.g., random array #1), ω_{opt} will vary with "b". This is even more apparent for random array #2 where one value of ω gave convergence for the first angle (ω = 0.25), but was not sufficient to give convergence at the second aspect angle. This particular characteristic of SOR solutions to these EM problems merit further investigation.

Much of the σ backscatter data presented in these figures indicates a rather wide range of convergence rates for σ; yet, many of these same cases have very similar characteristics in $\varepsilon^{(k)}$. These same cases often have apparently well converged bistatic patterns with most readjustments occuring in the "null" amplitudes beyond certain values for k. However, the σ backscatter curves sometimes still exhibit considerable instability in spite of the above signs. A probable cause for this wide range in convergence rates for σ is the slope of the σ backscatter pattern at the desired aspect angle; e.g., if the aspect corresponds to a relatively flat amplitude portion of the σ pattern, then convergence of σ will more than likely appear in fewer iterations. (A major exception to this viewpoint is the rapid convergence of σ for the large periodic array. Here, the reason for fast convergence is probably not due so much to the flatness of the pattern as to the generally reduced magnitudes of the off-diagonal elements of matrix A. Convergence rates of σ for random arrays having lesser volume densities of dipoles would certainly be faster for this same reason.) The chosen aspect angle for random arrays can often unknowingly correspond to a steep skirt or be near a null (cusp) in the σ backscatter pattern and the slightest changes in calculated currents will cause pronounced changes in the iterated σ curves. If, however, these same σ curves are accompanied by smooth monotonically decreasing $\varepsilon^{(k)}$'s, then these iterated solutions can still in some average sense be assumed to be nearing

the true solution. This implies that averages of σ, over many seemingly converged cases, might actually be good approximations of the true averages if σ were known exactly. A great deal more data is obviously needed to confirm or deny this relationship. However, if this should be the case, many of the statistics of for these large rather dense random arrays could be calculated without requiring rigorous convergence of the iterative technique to the exact solution.

There are certain distinct characteristics which keep reappearing in these iterated solutions for the 1000 dipole random arrays; namely, rapid convergence of σ_T and the relative stability of angular positions of peaks and nulls in the bistatic patterns. A sample calculation of the half power beam width for a uniformly excited circular aperture with the same projected area as the 1000 dipole array ($\sim 120\lambda^2$) results in an approximate 9° beam width. The half power beam widths of peaks appearing in the bistatic patterns interestingly enough consistently fall in the 6° - 10° range. These characteristics are undoubtedly related to the fundamental size and density of these arrays. Further investigations of these relationships and of overall σ backscatter statistics appears to be warranted.

SOI Iteration Solution for Scattering by a Small Cloud of Chaff Elements

The newly derived SOI technique introduced previously is used here to solve Eq. (12b) for a 100 dipole (8 dip/λ^3) random array. The results are shown in Fig. 74 where the two sets of curves correspond to two values of influence coefficient C. The direct solution obtained by Cholesky's method is also indicated. Computations corresponding to C = 0.2 required approximately 30 seconds per iteration and used a 14 x 14 maximum submatrix size. Convergence of σ in this case was irregular and $\varepsilon^{(k)}$ increased for k > 16. Computations for C = 0.1 required a maximum 44 x 44 submatrix and 150 sec/iteration and convergence in this case took fewer iterations ($k \sim 7$) and $\varepsilon^{(k)}$ exhibited a pronounced decrease over this same range.

Figure 75 is included here for comparison of SOI with SOR. The SOR iteration is used in this case to solve the same system of equations as for the above SOI method. The σ and $\varepsilon^{(k)}$ data for three relaxation factors are shown; $\omega = 0.7$ was a divergent case, $\omega = 0.6$ converged in the fewest number of steps and $\omega = 0.5$ converged, but required more iterations than $\omega = 0.6$. Iteration time for SOR (N = 100) was approximately nine seconds per iteration - a considerable improvement in time over SOI. The SOI algorithm is extremely inefficient compared to the simple form of SOR and for comparable rates of convergence, SOR is estimated to be approximately 15 times faster.

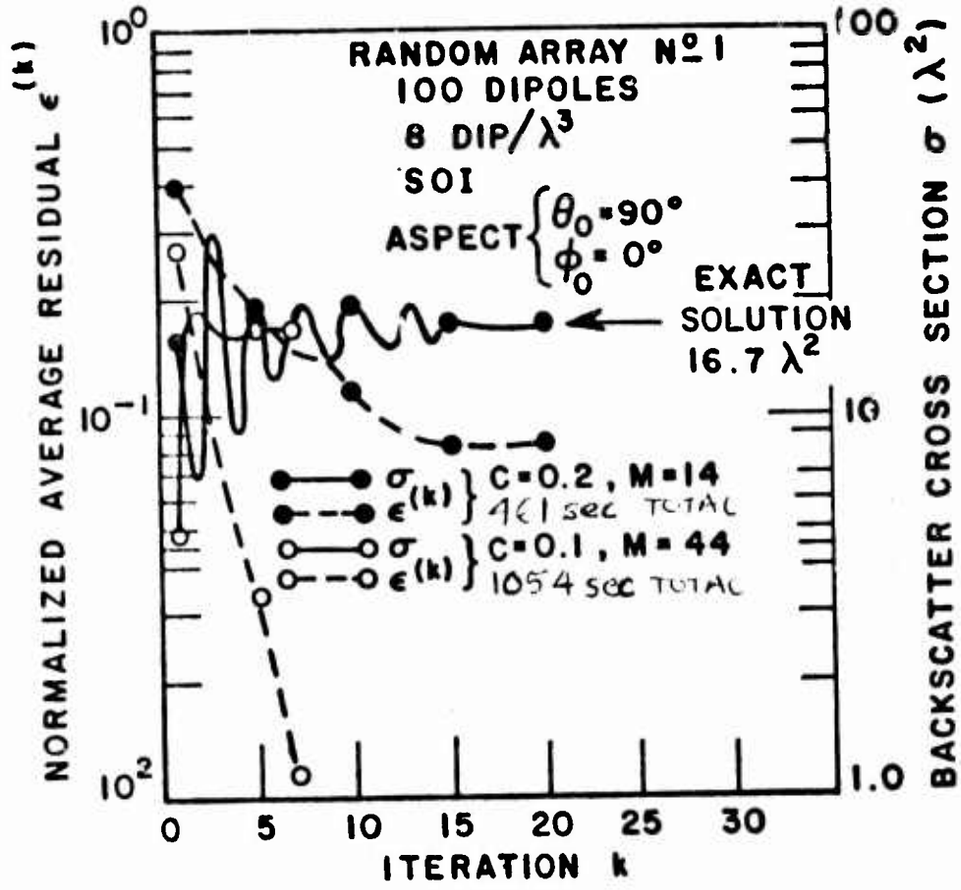


Figure 74. Backscatter cross section and $\epsilon^{(k)}$ versus iteration k for 100 dipole random array using SOI with c = 0.2 (M=14) and c = 0.1 (M=44).

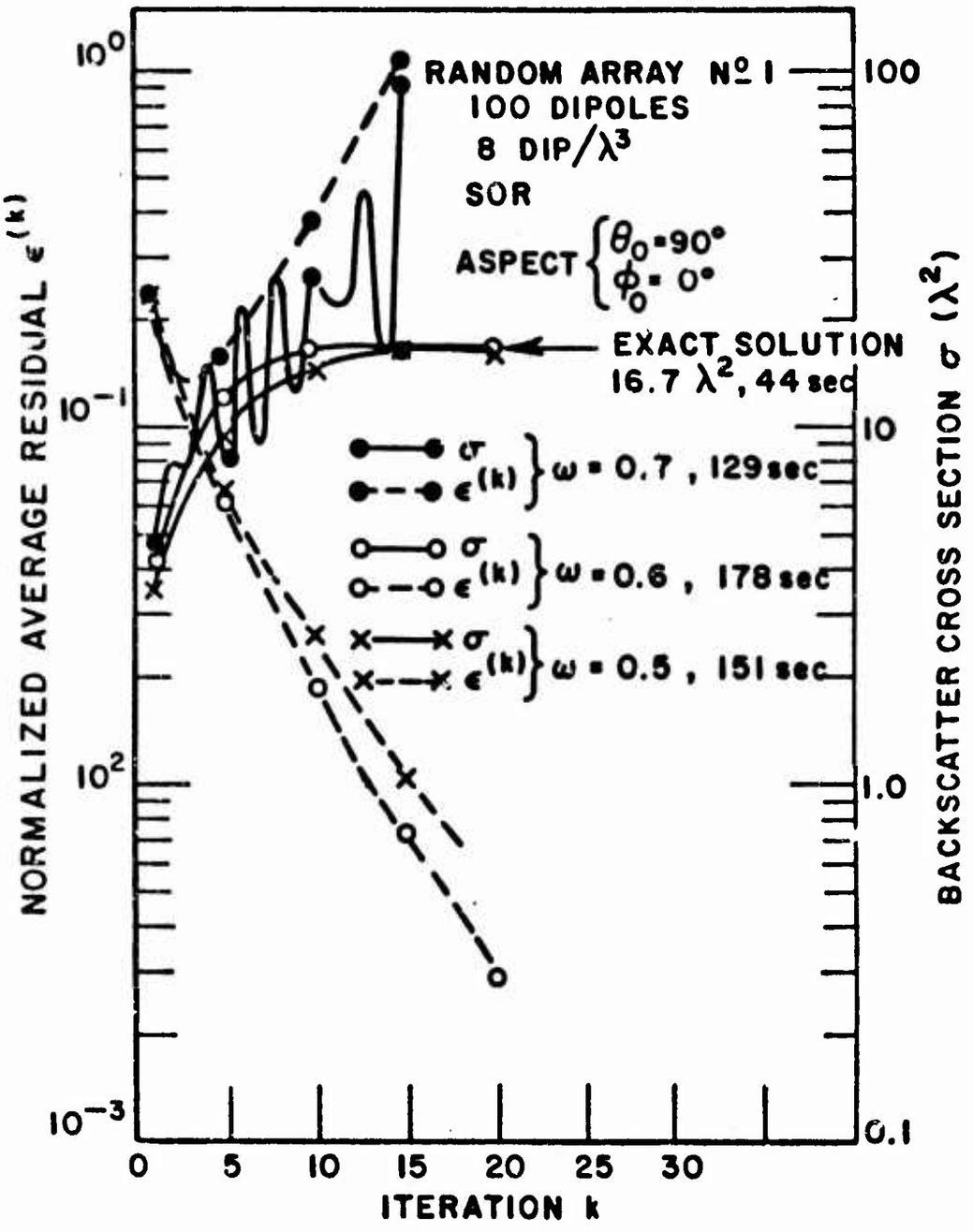


Figure 75. Backscatter cross section and $\varepsilon^{(k)}$ versus iteration k for 100 dipole random array using SOR with ω = 0.7, 0.6 and 0.5.

The direct solution to the above case required approximately 44 seconds while SOR took 90 seconds ($k = 10$, $\omega = 0.6$) to solve the same system. Recall, however, that the number of computations (multiplications) in Cholesky's direct method goes up as $\sim 1/6\ N^3$, while SOR used $\sim N^2$ computations per iteration; therefore, if the number of iterations required to achieve the desired accuracy is $< 1/6\ N$, then the SOR iteration will have a time advantage, even over the direct method.

Comments on the Applications of SOR to Surface Patch and Wire-Grid Models

Calculations using the SOR technique to solve Eq. (12b) for a surface patch-modeled flat plate and wire-grid modeled circular loop (polygon loop) have been unsuccessful, even for trivially small cases using a 12-mode surface patch-modeled square plate ($\lambda \times \lambda$) and a 10-mode wire-grid modeled loop (0.3λ radius). Both types of modeling used the overlapping type modes, cosines for the plate and piecewise sinusoids for the loop. The apparent numerical difficulty arises in the large magnitudes of the overlapping mutual impedances; these mutuals are, in fact, almost as large in magnitude as the self impedances positioned on the main diagonal of A. Hence, it appears that if off-diagonal terms in rows of A are almost as large in magnitude as the self term, then the SOR method fails to converge for all ω. A modified approach which may be worth investigating is a hybrid iteration technique probably combining SOI with SOR. The method would again be based on solving small systems of equations directly (SOI) but then using these current solutions to up-date other currents in the the corresponding "Sphere of Influence". This could be considered another form of "overlapping" block iteration.

D. The Question of Closer Spacings

In all the work described so far, the reader will notice that we have not discussed clouds with average spacings, d/λ, less than 0.5, or in other words, clouds with average dipole densities greater than $8/\lambda^3$. Here we mean "average density" in the sense of Appendix III, which implies that, for the kind of radially inhomogeneous clouds we assumed in the majority of cases, the actual dipole densities in the center of the cloud can be as high as $24/\lambda^3$. (For the uniform clouds discussed in the previous section, of course, the average dipole densities apply throughout the cloud.) Considering that each dipole is almost $\lambda/2$ in length these numbers should convey the impression of a rather tightly packed cloud with many elements very close at their closest points. It was this proximity which led us to be cautious and question the validity of our algorithm for obtaining the currents on dipoles in clouds with $d/\lambda < 0.5$ on the average. In our algorithm we assume that

each wire is divided into two (P=2) equal segments which support one (P-1) piecewise sinusoidal current mode. This assumption forces the effects of coupling from nearby wires to reside only in the complex amplitude of the current mode - coupling cannot change the shape of this single current mode. For two wires which approach each other very closely, except in very special relative orientations, we suspect that the true situation demands a change in the shape of the current distribution as well, meaning that the wires should be divided into more segments (P>2), thereby supporting more than one piecewise sinusoidal mode thereby allowing flexibility in current shape. This is easily done and is provided for in our computer programs; however, doing so has the undesirable effect of reducing the number of wires allowed in a cloud, the impedance matrix size being fixed. We investigated the validity of our two-segment model with increasing cloud densities in the hope that it would hold up for denser clouds than those represented by $d/\lambda = 0.5$. This section presents some of our findings.

In order to investigate the question of closer spacings we calculated spatial average backscattering cross sections using three variants of the Richmond reaction matching technique:

(1) Two-segment model with 12 point numerical integration. This variant is the one used for essentially all the results produced under this contract. In it, each dipole is divided into two segments supporting piecewise-sinusoidal currents whose reaction integrals are performed approximately using a 12 point numerical integration routine.

(2) Two-segment model with exact integration. This variant is similar to (1) but the reaction integrals are expressed analytically in closed form and are evaluated exactly. This method is superior to (1) in precision, is equivalent to (1) in required computer memory, but takes more time (about 60% more time, it turns out).

(3) Four-segment model with exact integration. This variant models each dipole with four segments, thereby allowing a more precise resolution of the induced current on the dipole than is possible with the two-segment model. The currents on each segment are integrated exactly. This method is the most precise of the three, but it requires nine times the computer memory required by the two-segment models and a great deal more computer time. Thus, whereas we can solve for 200 dipole clouds with two segment models we could solve for only 22 dipole clouds using a four-segment model.

We assumed inhomogeneous clouds containing N=10 dipoles and calculated the average backscattering cross section of each (averaged over the usual 512 different aspect angles around a great circle in V-V and H-H polarizations). Twenty clouds were randomly generated for each spacing $d/\lambda = 0.5$ and 0.25 and results for each were calculated using the three variants discussed above. Typical results of these calculations are presented in Table 7.

The conclusions derived from Table 7 may be summarized as follows:

(1) For $d/\lambda = 0.5$, all three methods give results in close agreement. Thus, we have some assurance that the model we have been using heretofore (the two-segment model with numerical integration) is sufficiently accurate.

(2) For $d/\lambda = 0.25$, the two-segment model with exact integration appears to correlate better with the four-segment model, although the model with numerical integration really does not perform badly at all. To be safe, however, we suggest use of the two-segment model with exact integration for average spacings less than 0.5 at the expense of 60% more computation time.

The three reaction matching variants described above were also used to generate (using the Wright-Patterson Air Force Base computer) pattern functions of six inhomogeneous, 50 dipole clouds - three with $d/\lambda = 0.25$ and three with $d/\lambda = 0.125$. The results are plotted in Figs. 76-93. From these patterns it appears that for the larger average spacing, a two segment, exact integration model is adequate to obtain good scattering patterns, but for the smaller average spacing, even the four segment, exact integration model has not clearly converged in its pattern function. We feel that for average spacings less than 0.25λ (i.e., $64/\lambda^2$ density) in the inhomogeneous clouds assumed here, the algorithms presented in this report are not reliable.

One additional study which was made involved the statistics of the echo from 200 clouds, each composed of only two dipoles randomly spaced and oriented in the usual manner. From these clouds we generated histograms of the backscattering cross section at one look-angle and the backscattering cross section averaged over 512 look angles. For the case where the average spacing was $d/\lambda = 1.43$ (Fig. 94 gives the statistical distribution of the spacing), the relative frequencies of the cross sections averaged over 512 look angles, with and without coupling, are given in Figs. 95a,b, respectively. Relative frequencies based on 1 look angle are given in Figs. 96a,b. Note that, although the averages derived in Figs. 95 and 96 are consistent, the distributions are different, the data for 1 look angle being more spread out. For

the 512 look angle case, the appearance of an exponential distribution is clearer. In both cases, the coupled and uncoupled dipoles exhibit similar histograms, as expected with an average spacing as large as1.43λ.

TABLE 7

N = 10 dipoles
d = 0.5λ

Cloud Number	Two-Segment Model Numerical Integration	Two-Segment Model Exact Integration	Four-Segment Model Exact Integration
1	1.080880	1.080899	1.068370
2	1.139636	1.243639	1.277114
3	1.124152	1.131229	1.130523
4	1.789712	1.789627	1.745001
5	0.861137	0.861061	0.893994
6	1.293543	1.444584	1.442656
7	0.489466	0.469868	0.460119
8	1.070567	1.070436	1.070025
9	0.902232	0.901733	0.904238
10	0.649599	0.620836	0.617651

N = 10 dipoles
d = 0.25λ

1	0.657790	0.651528	0.647247
2	0.680643	0.741439	0.745068
3	0.680060	0.686915	0.669449
4	0.575100	0.53224	0.536879
5	0.468432	0.476924	0.477237
6	0.835393	0.989235	1.001033
7	0.337658	0.298744	0.294067
8	0.566090	0.565526	0.537847
9	0.681039	0.872147	0.851511
10	0.306615	0.314593	0.311840

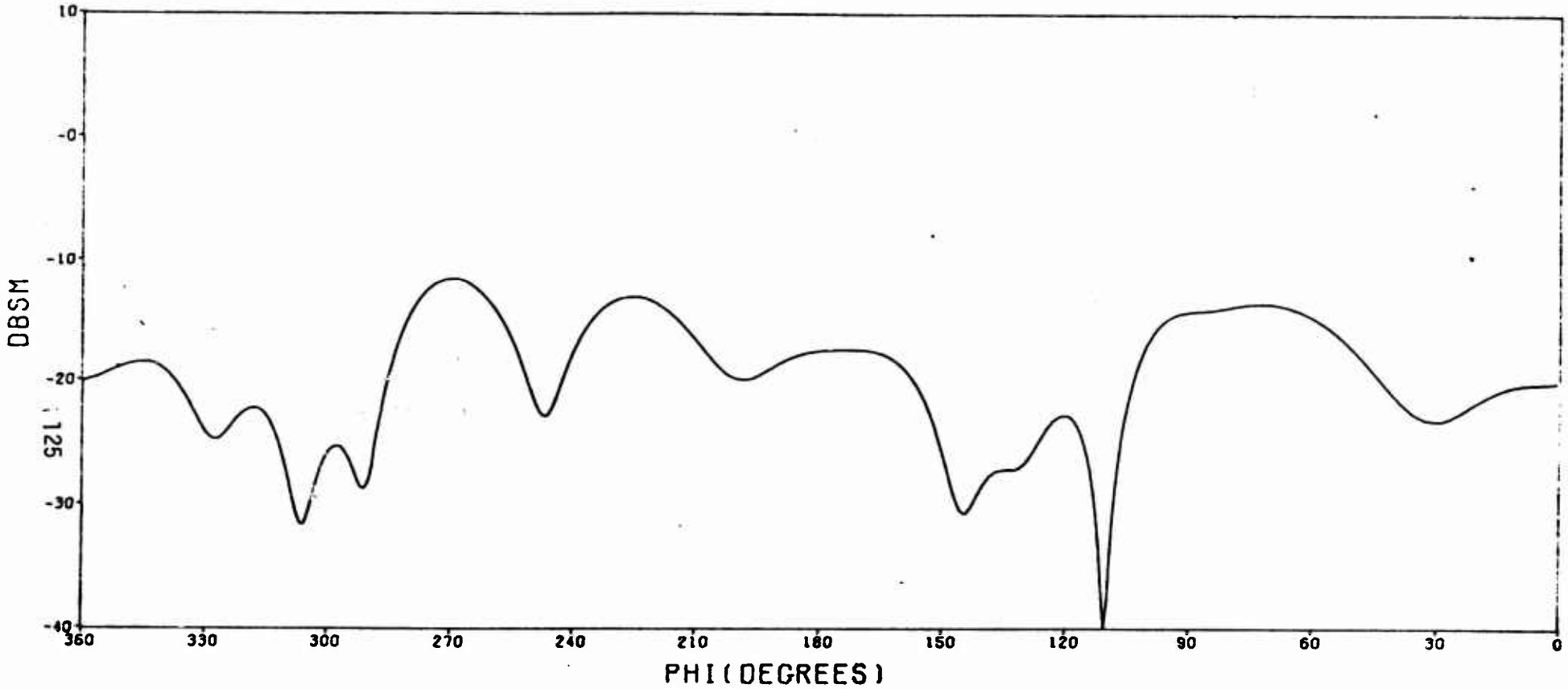


Figure 76. θ-θ backscattering pattern, N=50 dipoles, d/λ=0.25, P=2 segments, 12-point integration.

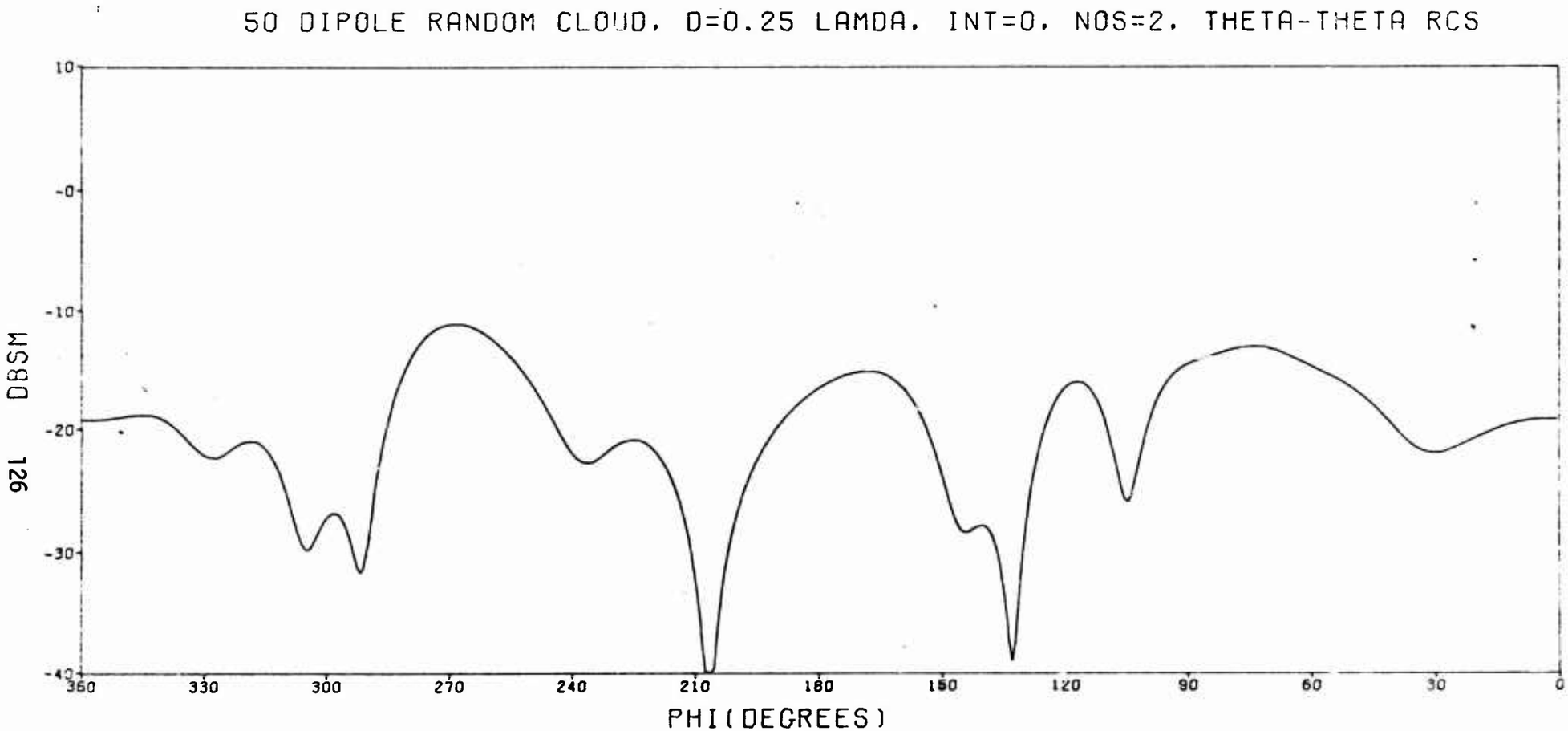


Figure 77. θ-θ backscattering pattern, N=50 dipoles, P=2 segments, $d/\lambda=0.25$, exact integration.

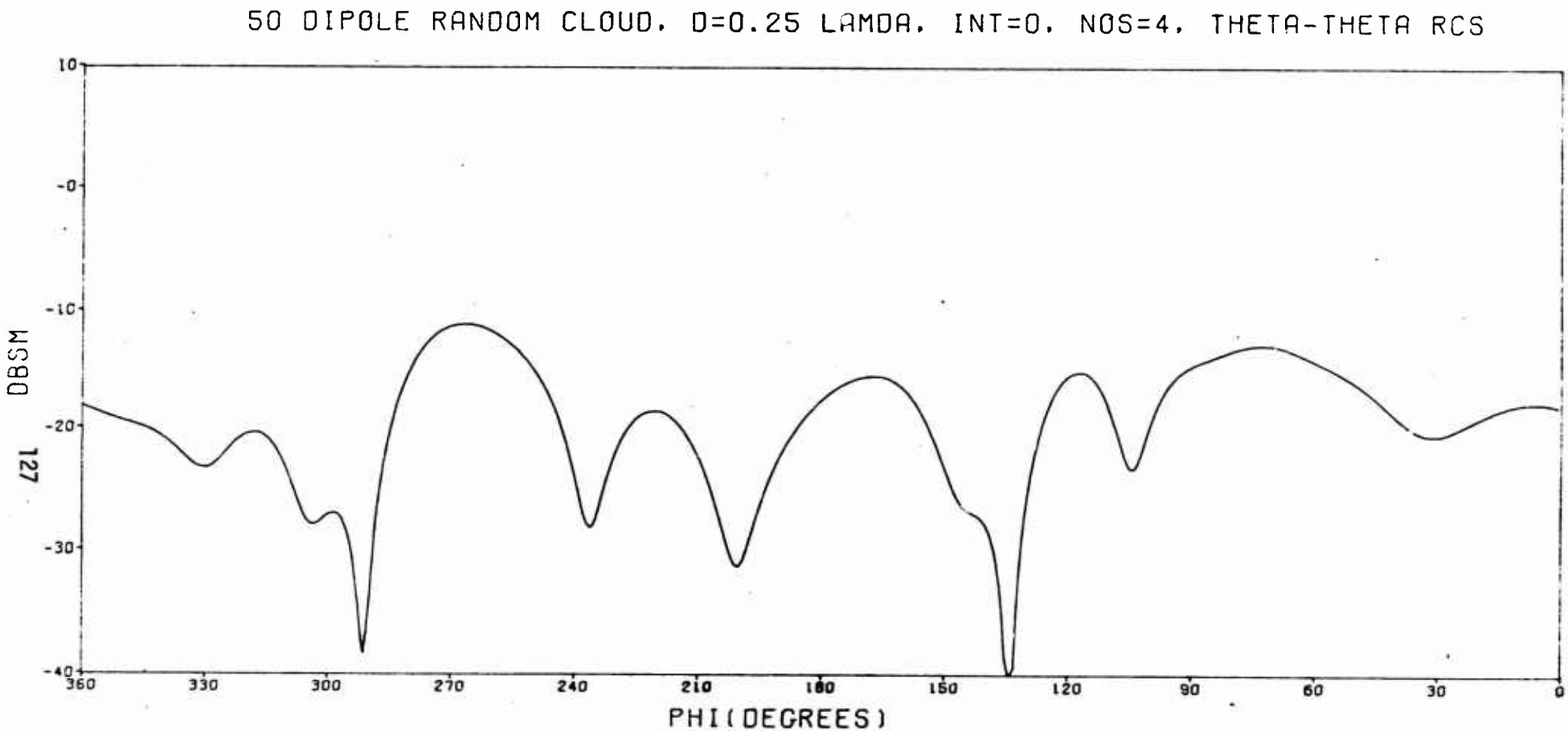


Figure 78. θ-θ backscattering pattern, N=50 dipoles, P=4 segments, $d/\lambda=0.25$, exact integration.

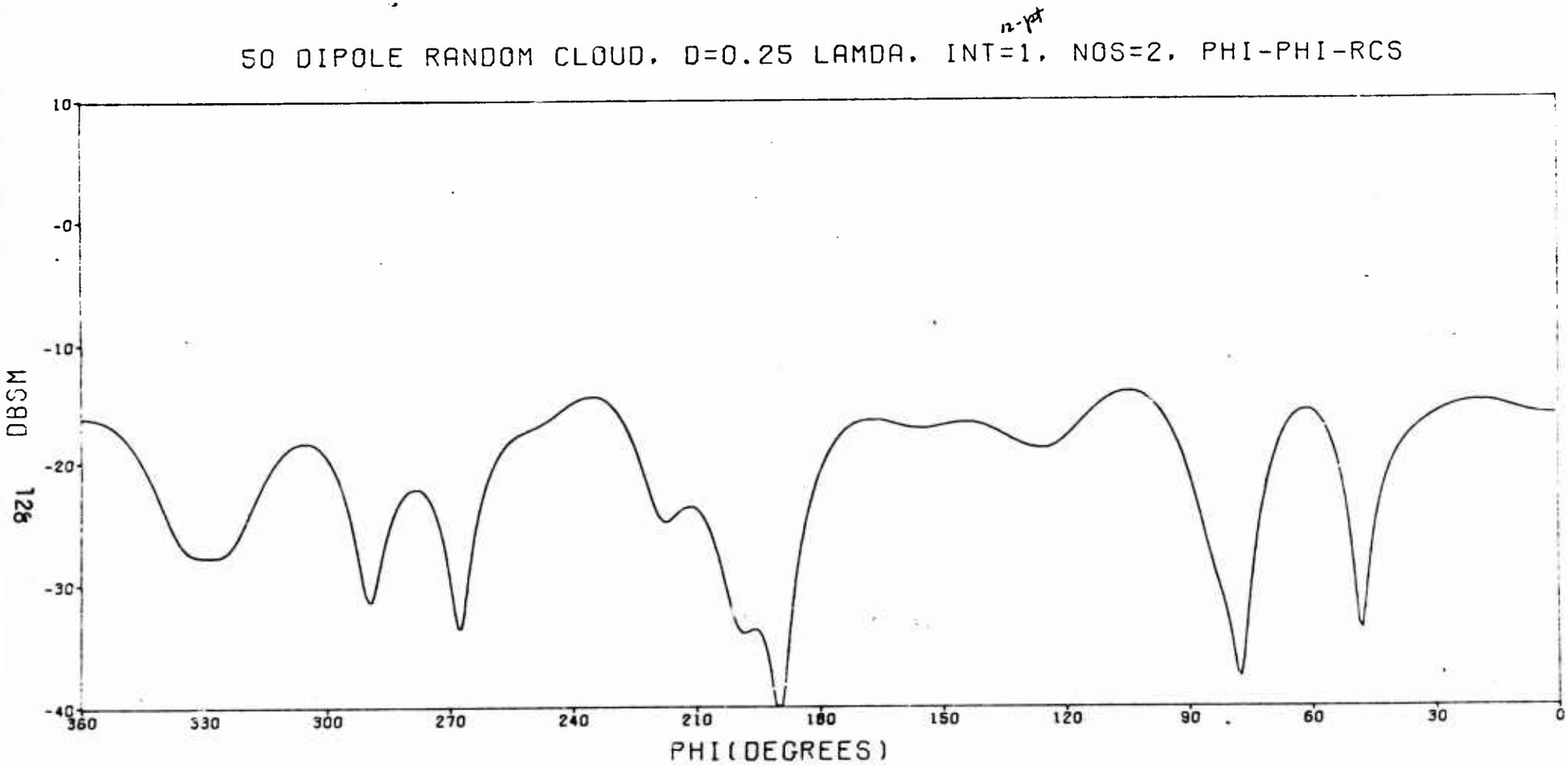


Figure 79. ϕ-ϕ backscattering pattern, N=50 dipoles, P=2 segments, d/λ=0.25, 12 point integration.

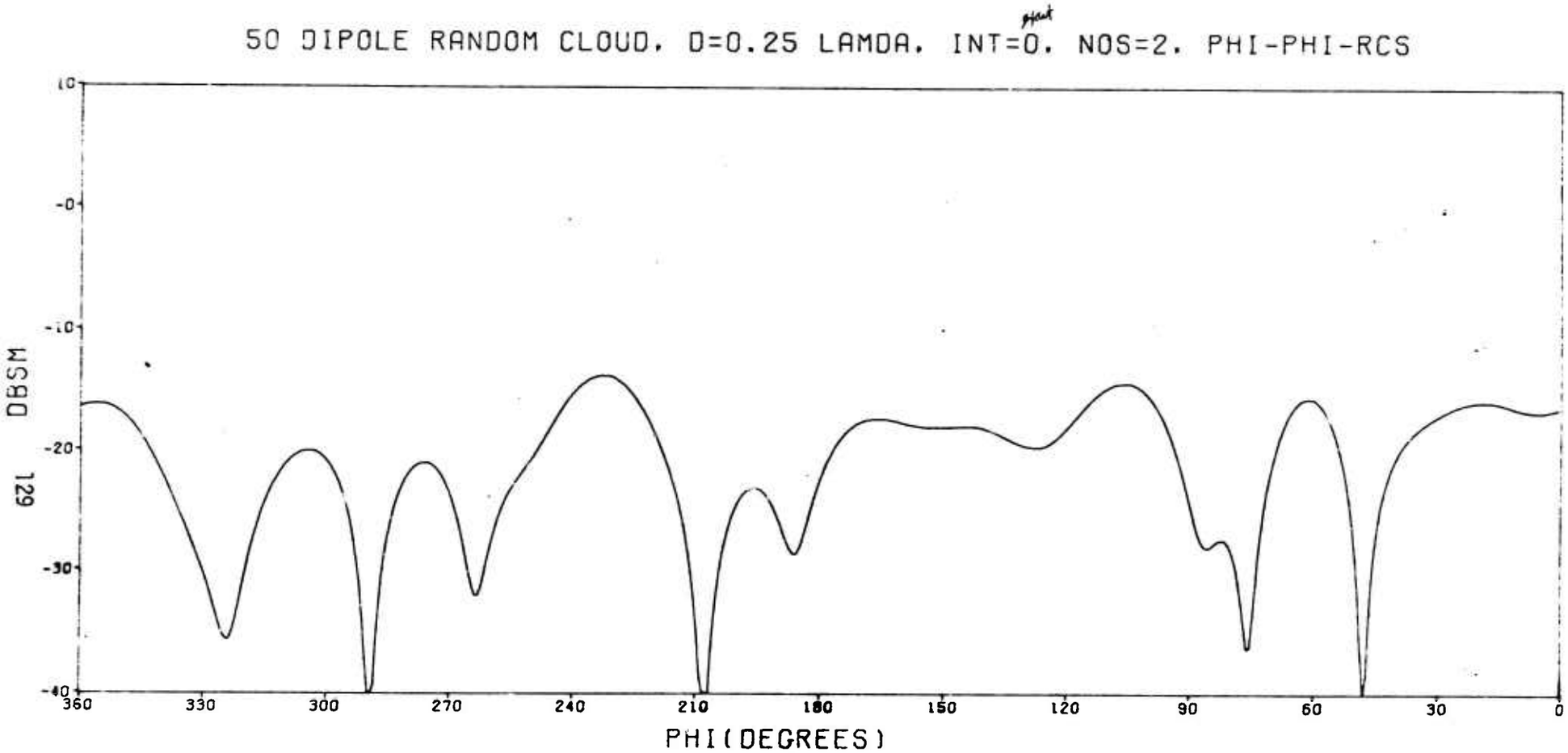


Figure 80. ϕ-ϕ backscattering pattern, N=50 dipoles, P=2 segments, d/λ=0.25, exact integration.

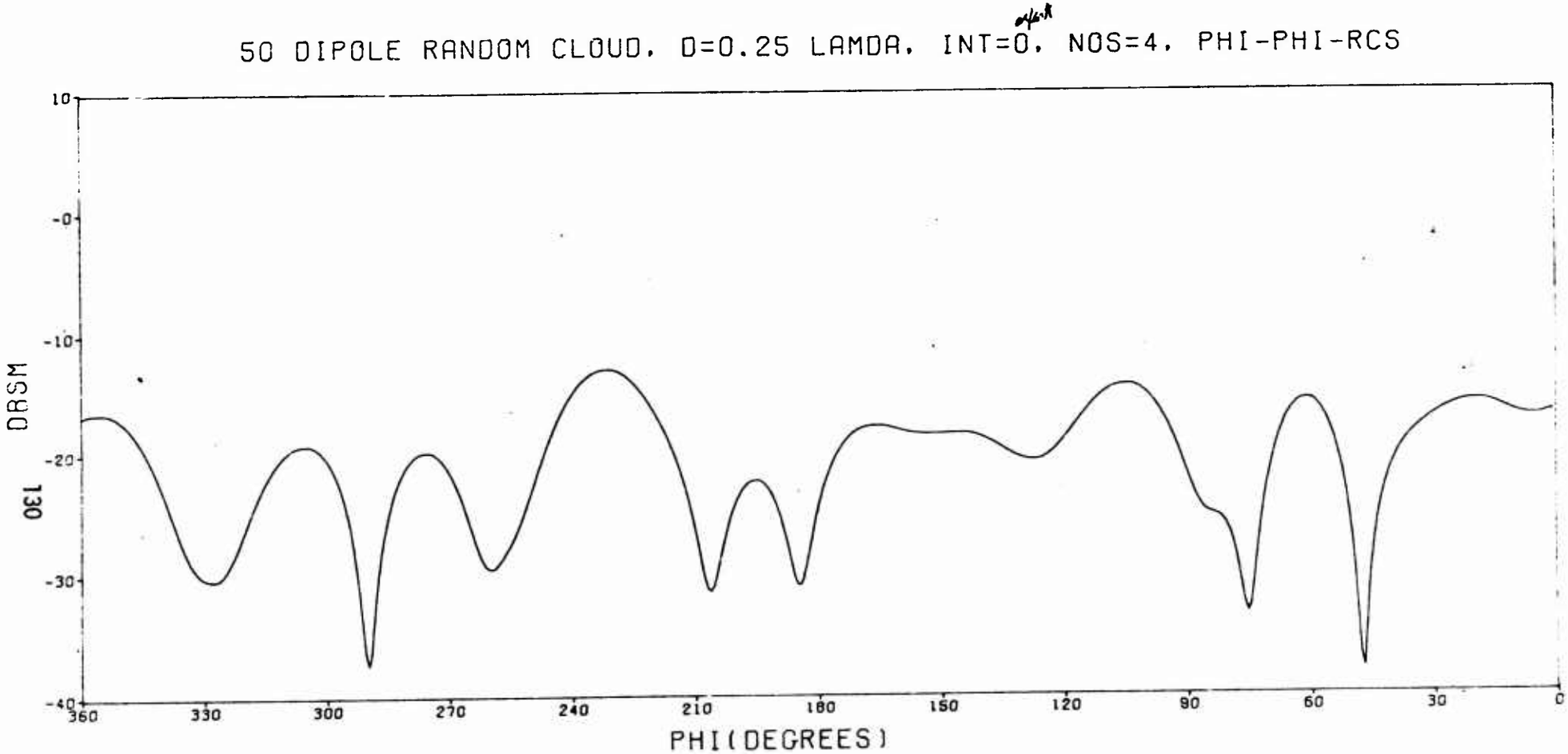


Figure 81. ϕ-ϕ backscattering pattern, N=50 dipoles, P=4 segments, d/λ=0.25, exact integration.

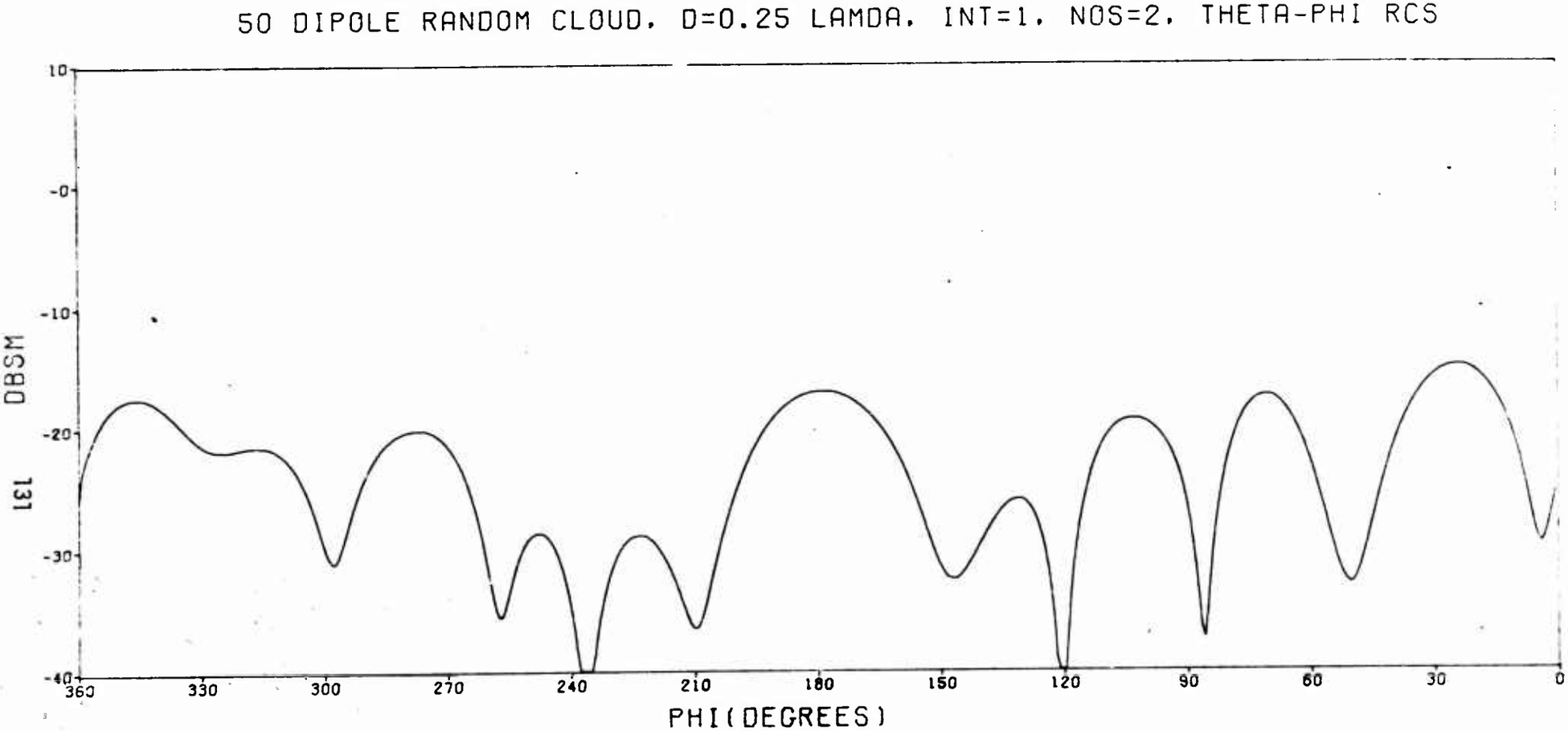


Figure 82. θ-ϕ backscattering pattern, N=50 dipoles, P=2 segments, d/λ=0.25, 12-point integration.

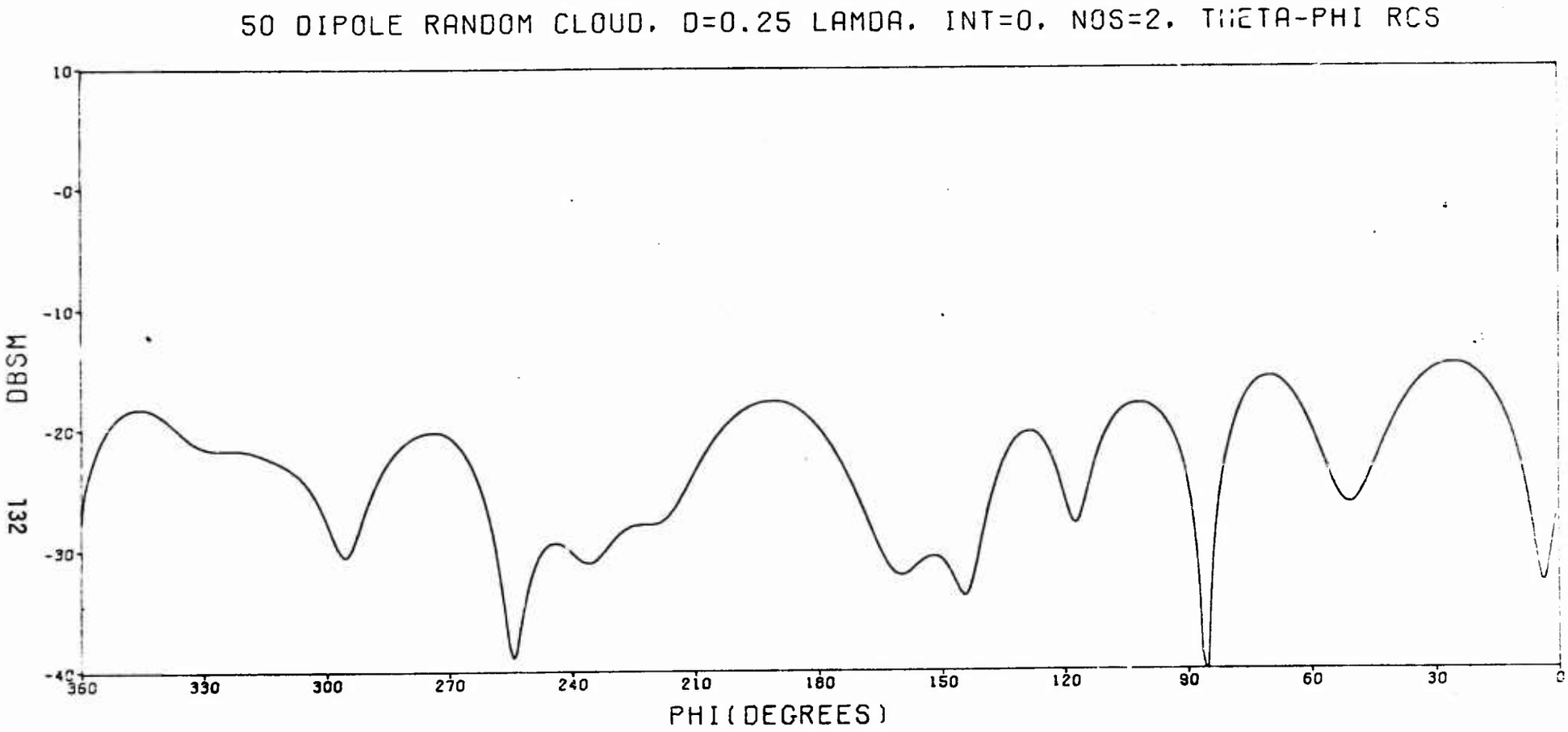


Figure 83. θ-ϕ backscattering pattern, N=50 dipoles, P=2 segments, d/λ=0.25, exact integration.

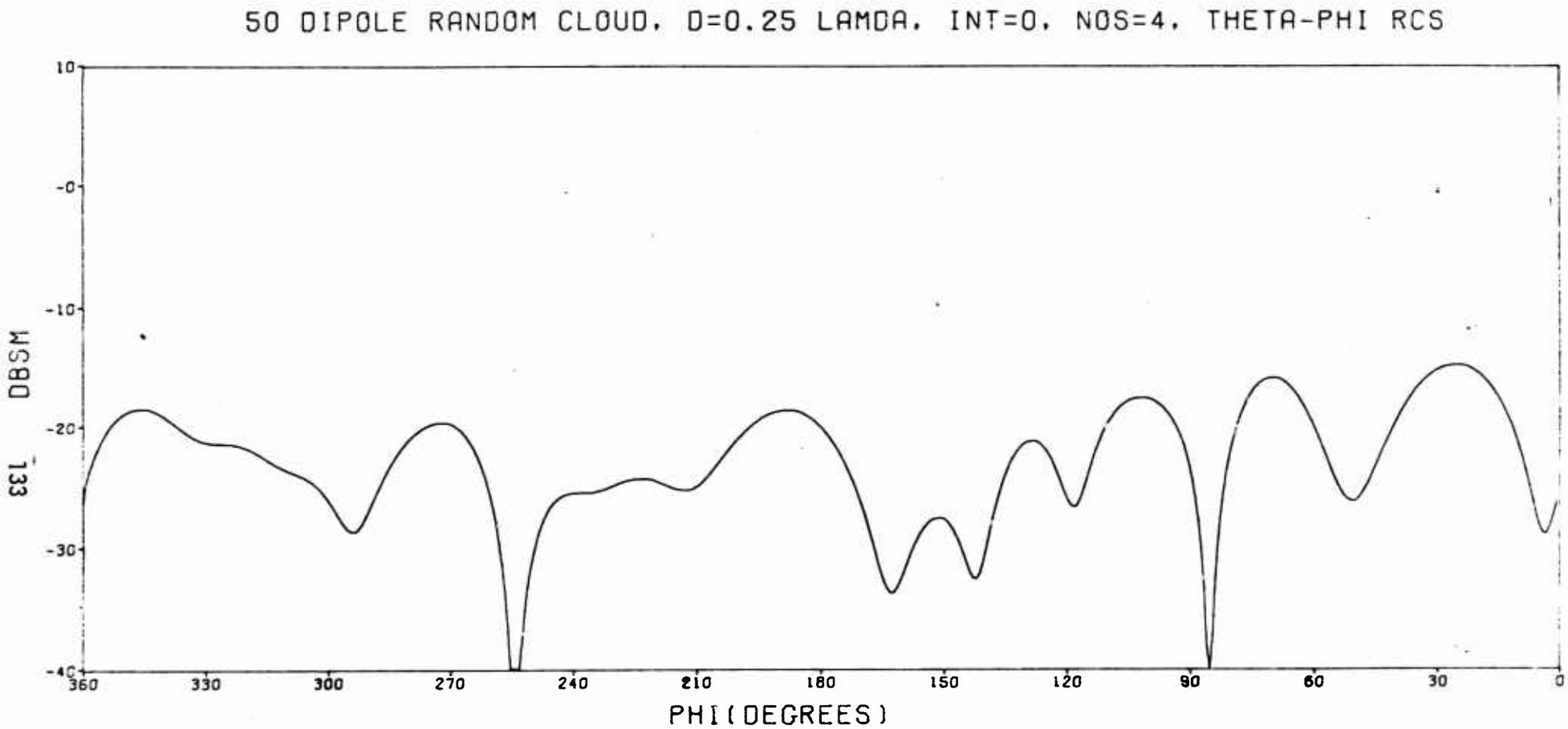


Figure 84. θ-ϕ backscattering pattern, N=50 dipoles, P=4 segments, d/λ=0.25, exact integration.

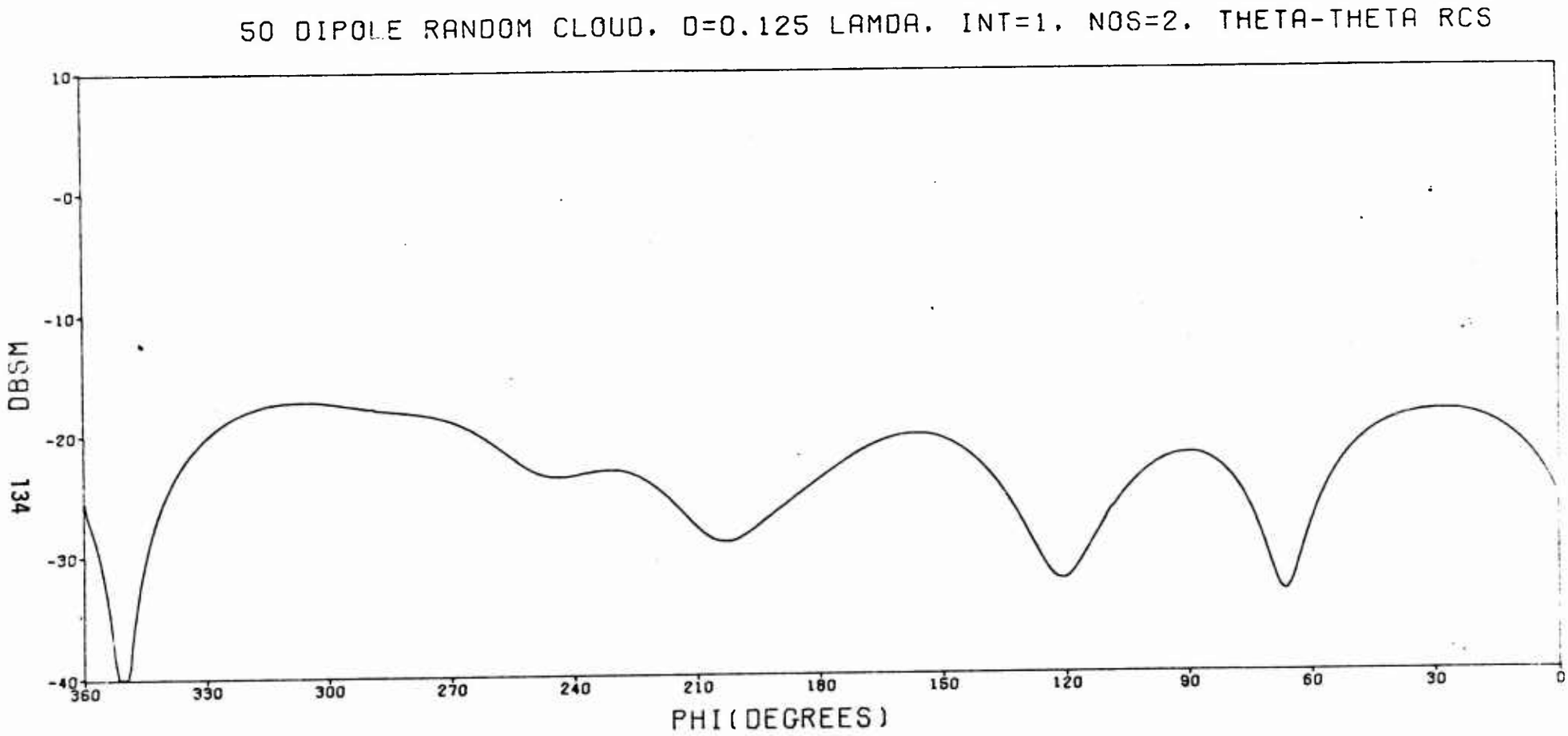


Figure 85. θ-θ backscattering pattern, N=50 dipoles, P=2 segments, $d/\lambda=0.125$, 12-point integration.

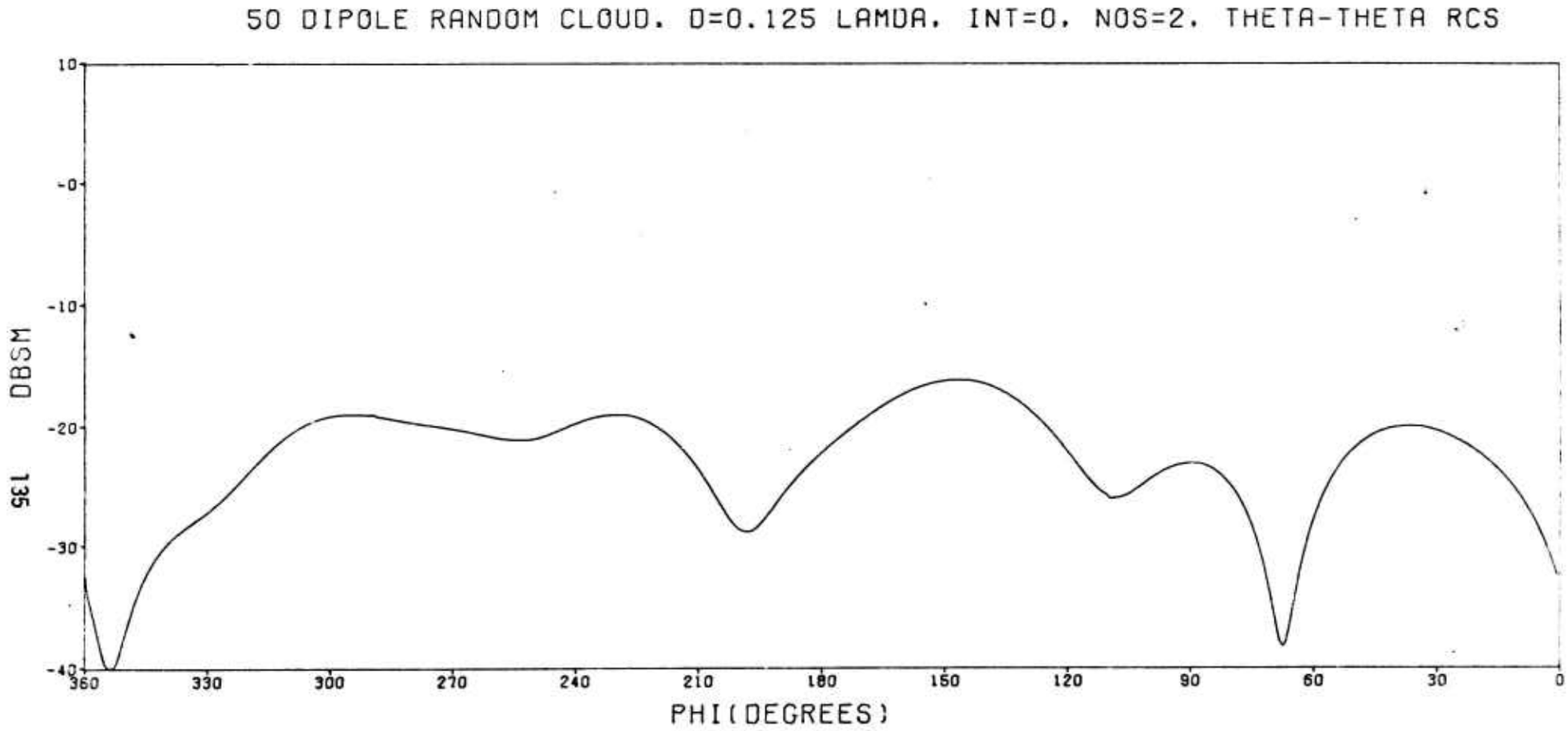


Figure 86. θ-θ backscattering pattern, N=50 dipoles, P=2 segments, $d/\lambda=0.125$, exact integration.

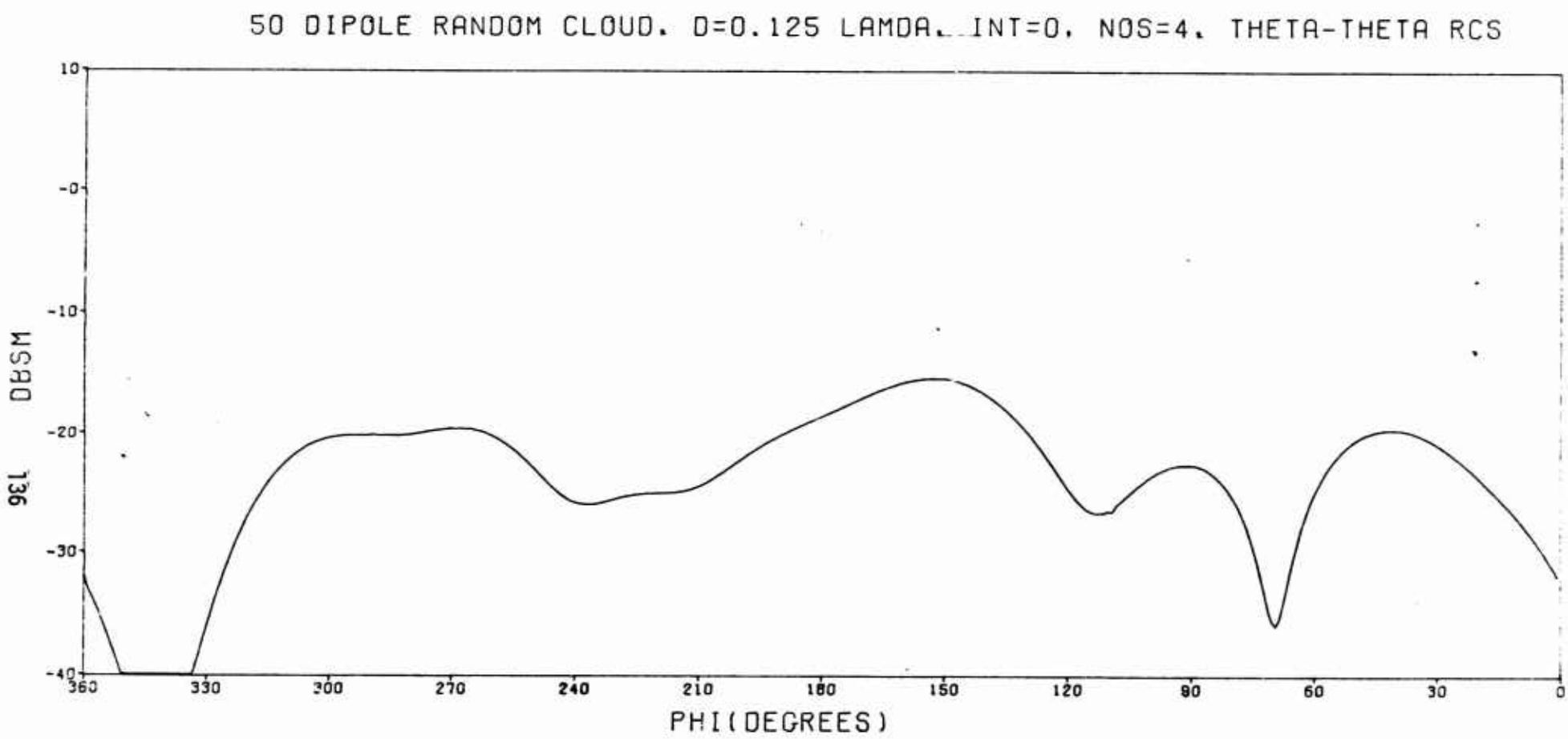


Figure 87. θ-θ backscattering pattern, N=50 dipoles, P=4 segments, d/λ=0.125, exact integration.

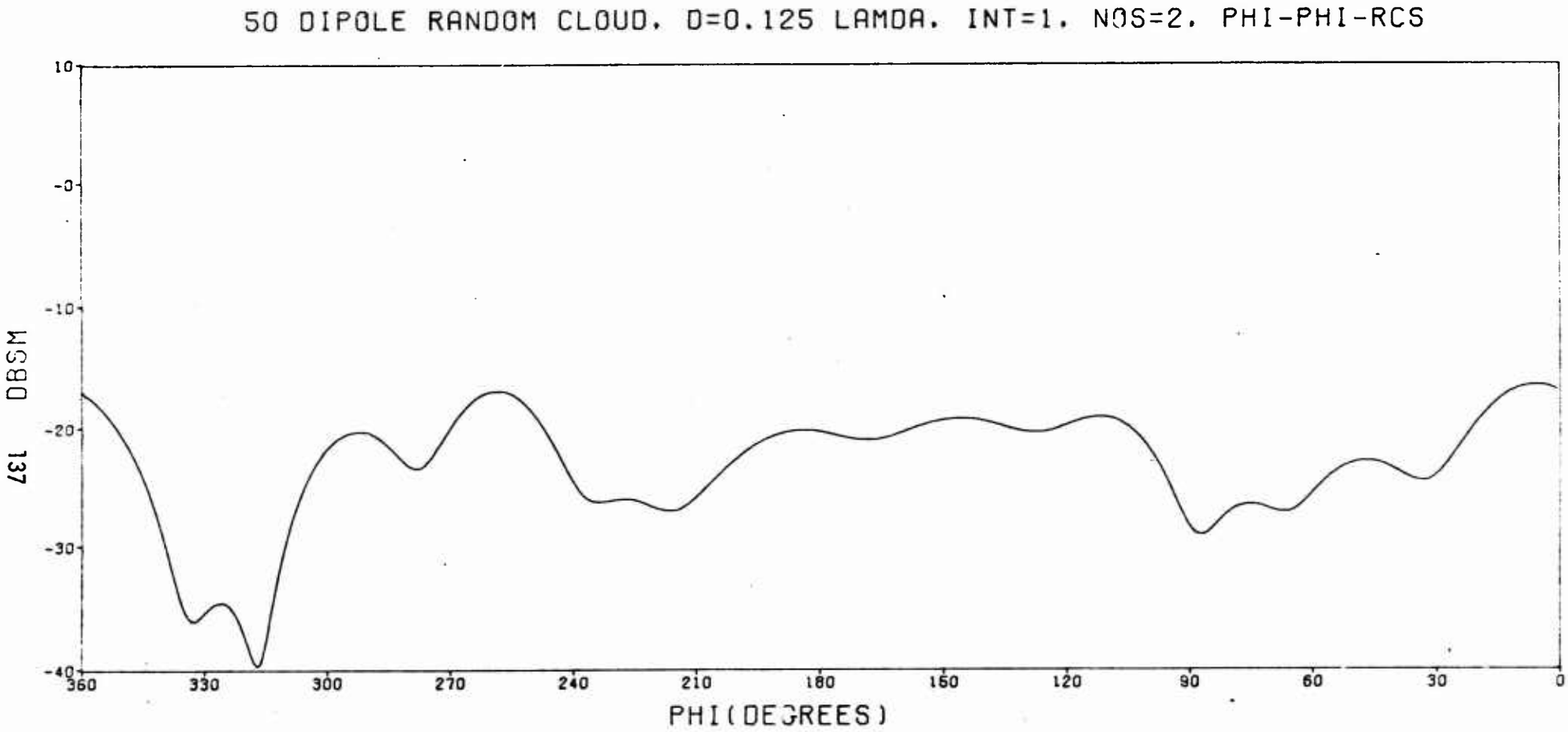


Figure 88. ϕ-ϕ backscattering pattern, N=50 dipoles, P=2 segments, d/λ=0.125, 12-point integration.

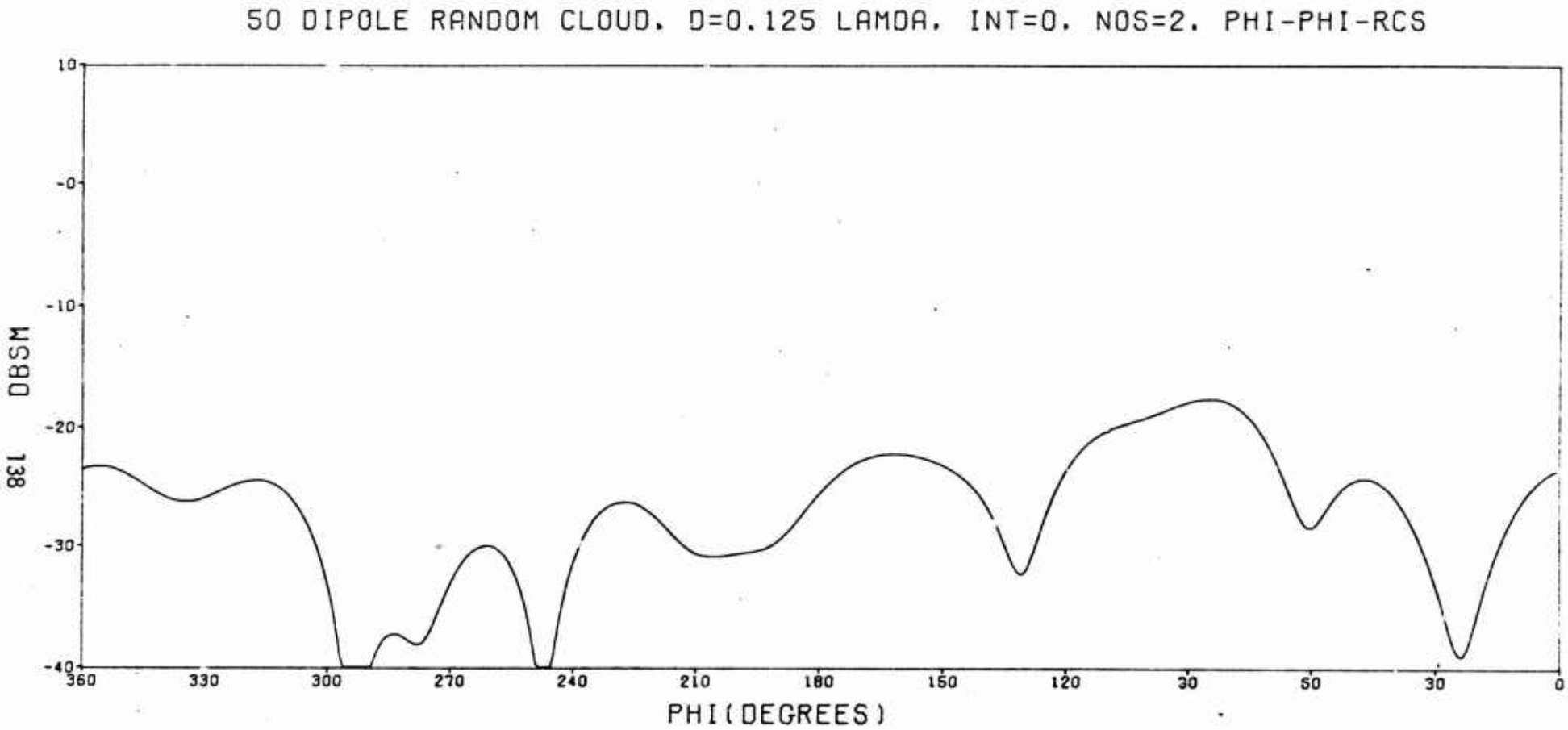


Figure 89. ϕ-ϕ backscattering pattern, N=50 dipoles, P=2 segments, d/λ=0.125, exact integration.

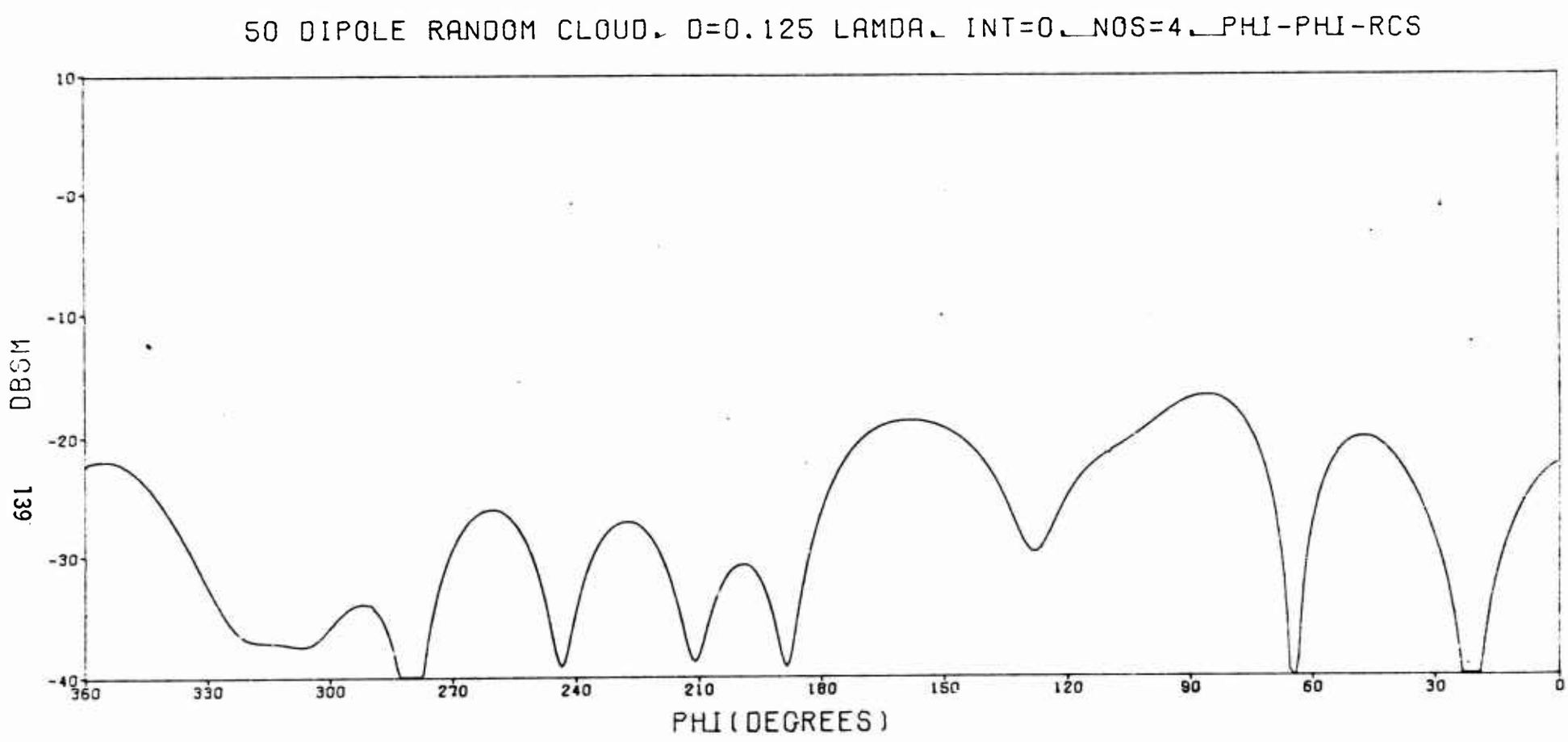


Figure 90. ϕ-ϕ backscattering pattern, N=50 dipoles, P=4 segments, d/λ=0.125, exact integration.

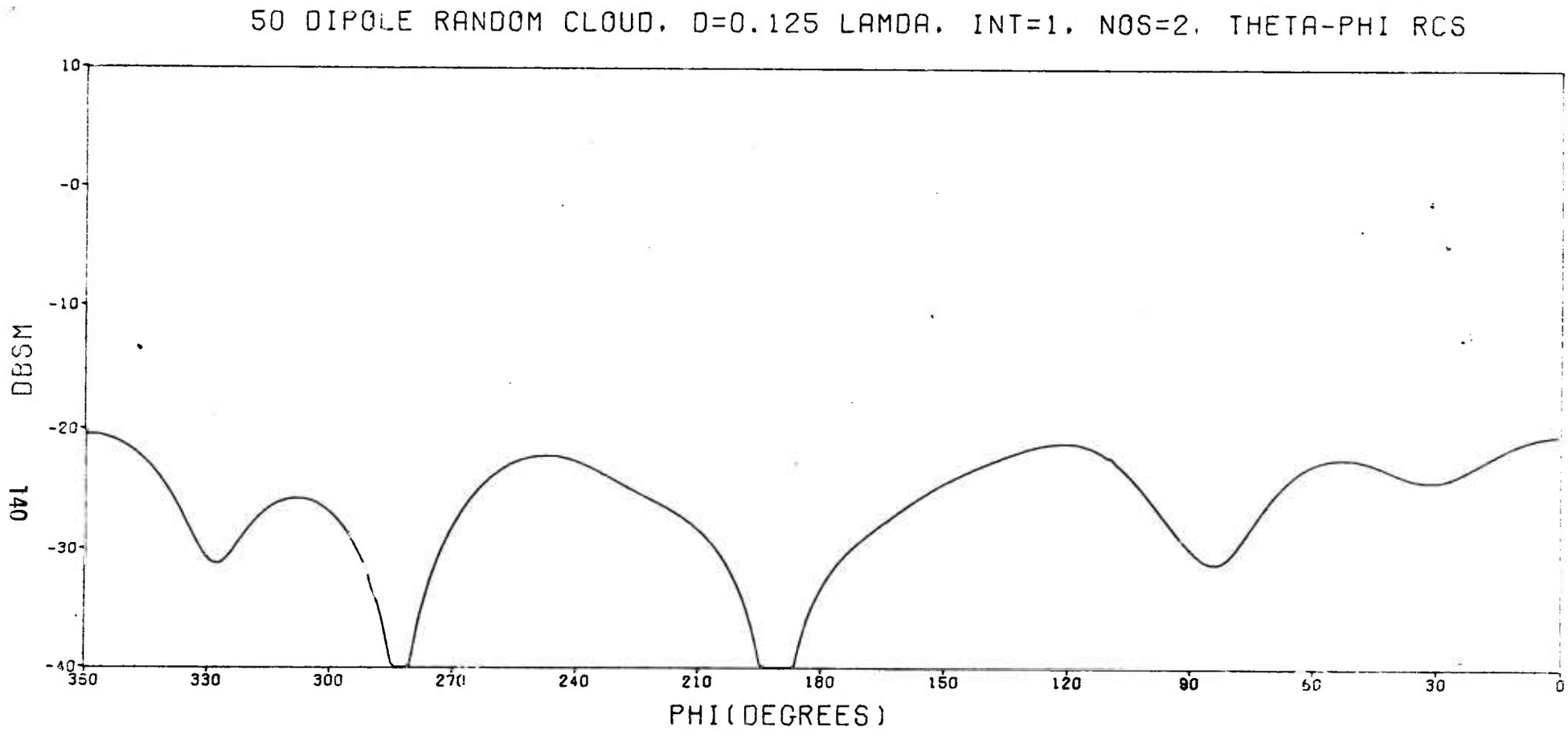


Figure 91. θ-ϕ backscattering pattern, N=50 dipoles, P=2 segments, d/λ=0.125, 12-point integration.

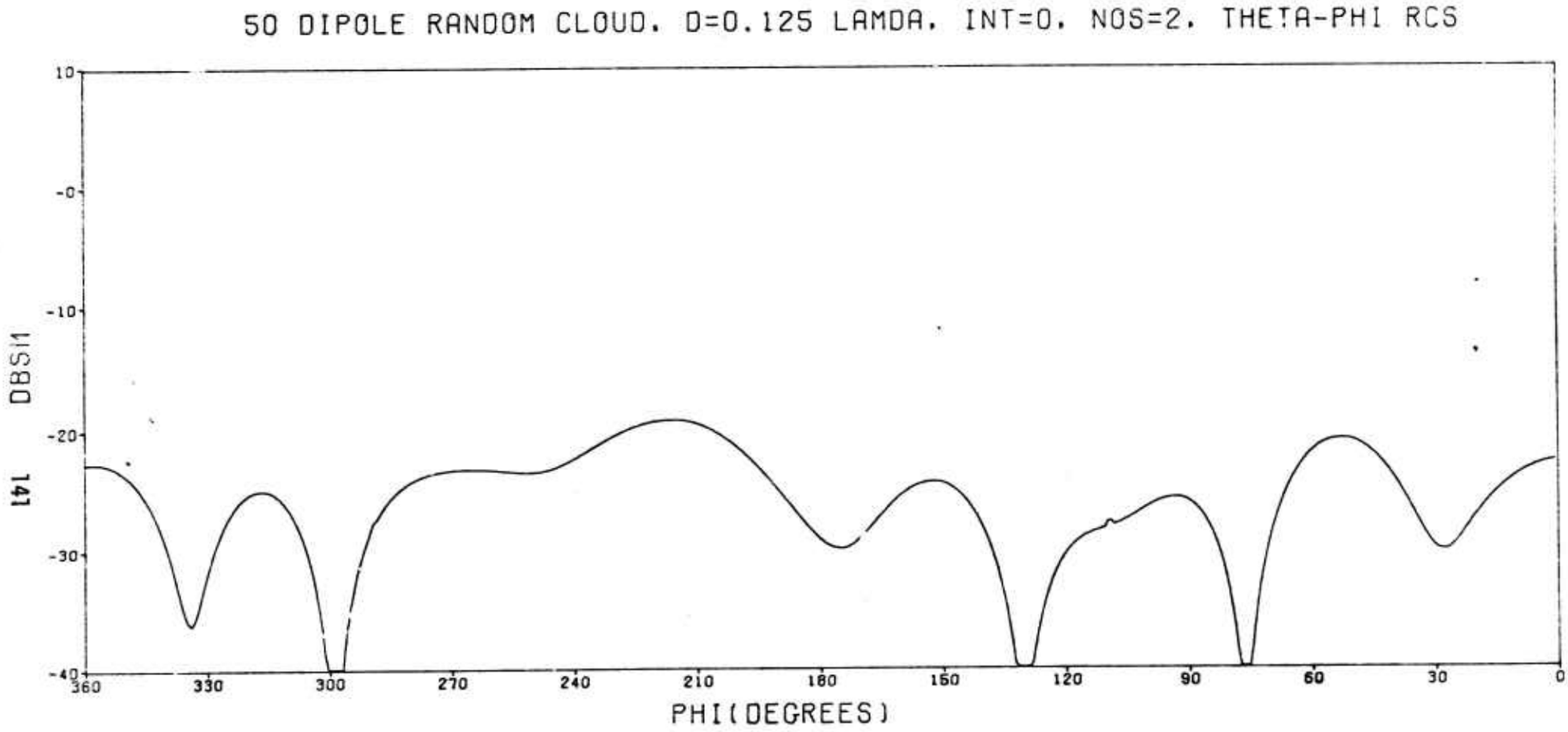


Figure 92. θ-ϕ backscattering pattern, N=50 dipoles, P=2 segments, d/λ=0.125, exact integration.

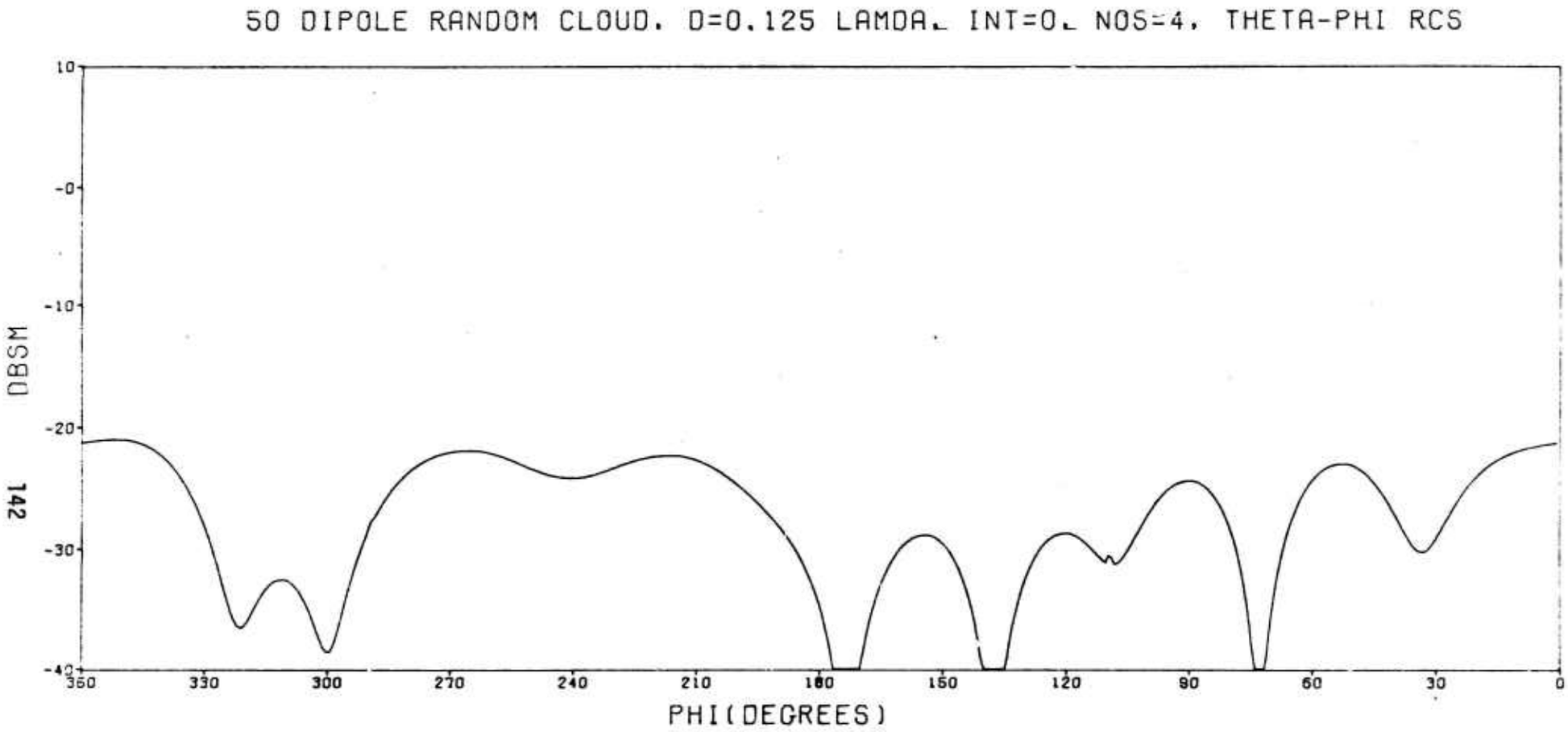


Figure 93. θ-ϕ backscattering pattern, N=50 dipoles, P=4 segments, d/λ=0.125, exact integration.

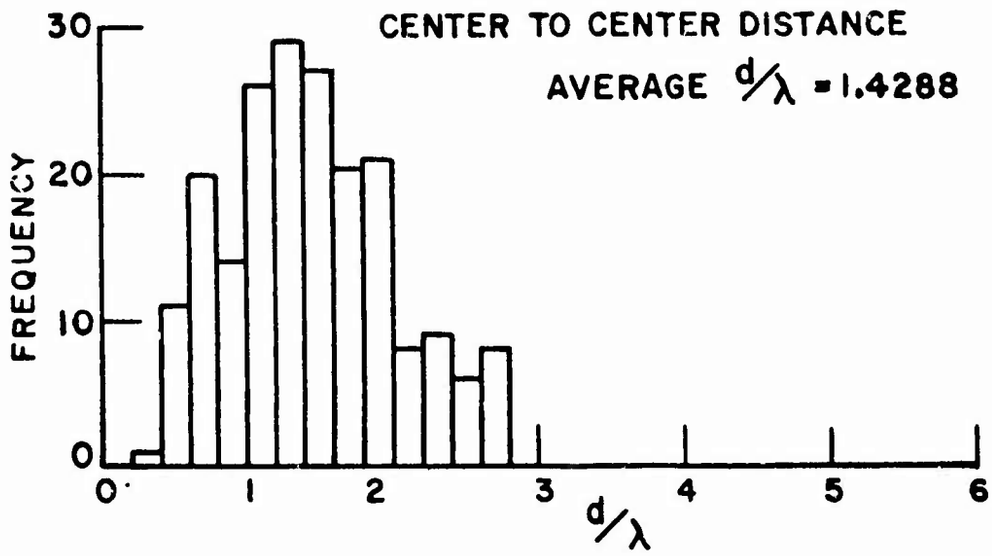


Figure 94. Histogram of the center-to-center distances of random two-dipole clouds. Average spacing $d/\lambda \simeq 1.43$.

If a very small average spacing is assumed, and accordingly, a 4-segment model with exact integration is used for each dipole, the curves of Figs. 97-98 result. Figure 97 is the distribution of the spacing, with the rather small value of average spacing $d/\lambda = 0.286$. Figures 98a,b present the relative frequencies of the cross sections averages over 512 look angles with and without coupling. Again, the exponential trend of the histograms is evident.

From Figs. 95b and 98b, for the two dipoles uncoupled, we note that for far spacings, the average echo is about $2\langle\sigma_0\rangle$ or about $0.35\lambda^2$, whereas for close spacing the average echo exceeds this ($\sim 0.47\lambda^2$). This is expected because for such a close average spacing the two dipoles cannot be excited incoherently and their echo therefore lies above that for totally incoherent scatterers. This effect for two dipoles variously spaced is shown in Fig. 99. The $2\langle\sigma_0\rangle$ law does not appear to be reached until $d/\lambda \simeq 1.4$. For clouds containing larger numbers of uncoupled dipoles, the limit $N\langle\sigma_0\rangle$ is expected to be reached for smaller spacings due to the larger overall extent of the clouds.

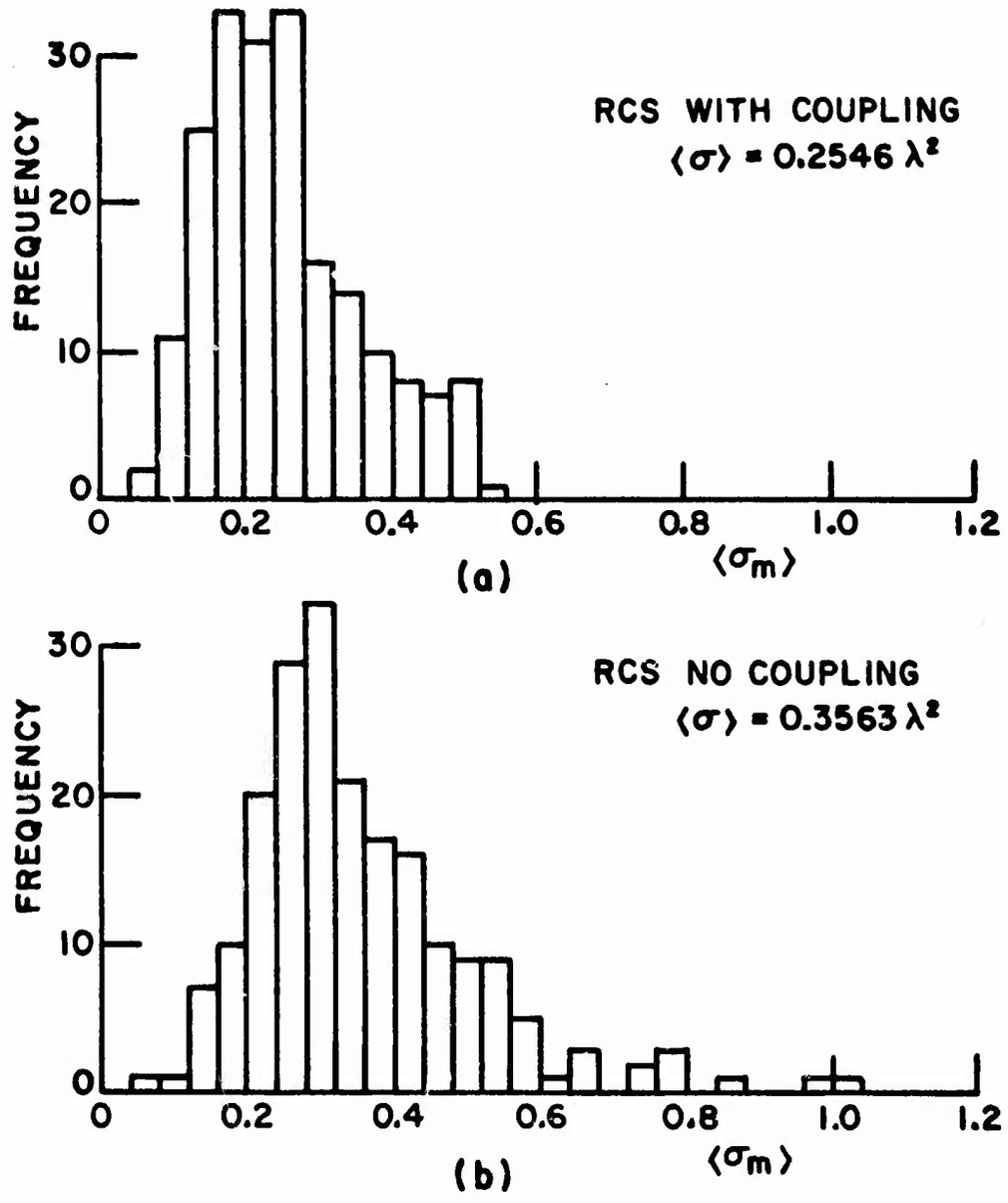


Figure 95. Histograms of the spatially averaged radar cross sections of the clouds generated for Fig. 94: (a) with coupling (b) with no coupling.

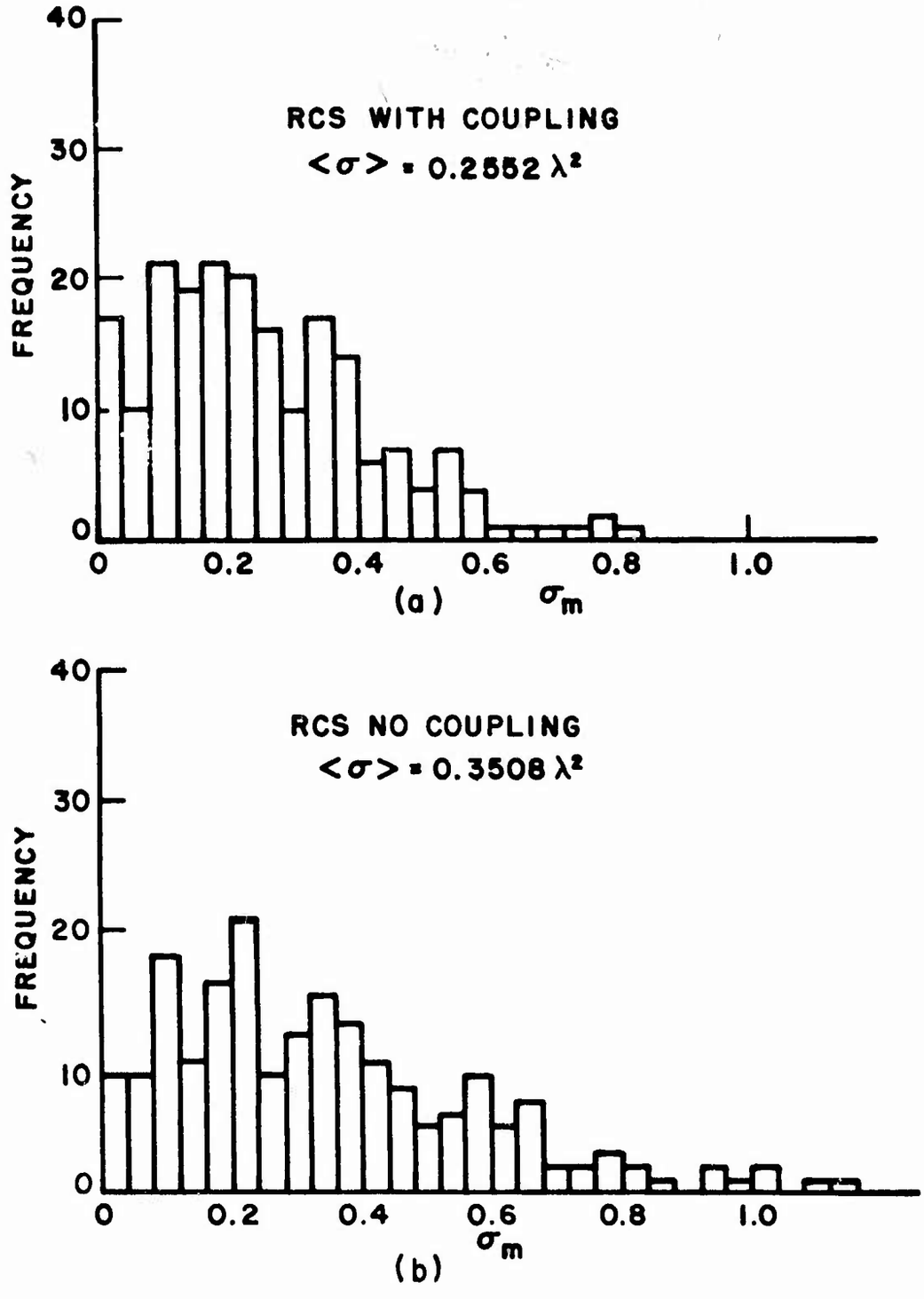


Figure 96. Histograms of the radar cross sections (single aspect) of the clouds generated for Fig. 94. (a) with coupling (b) with no coupling.

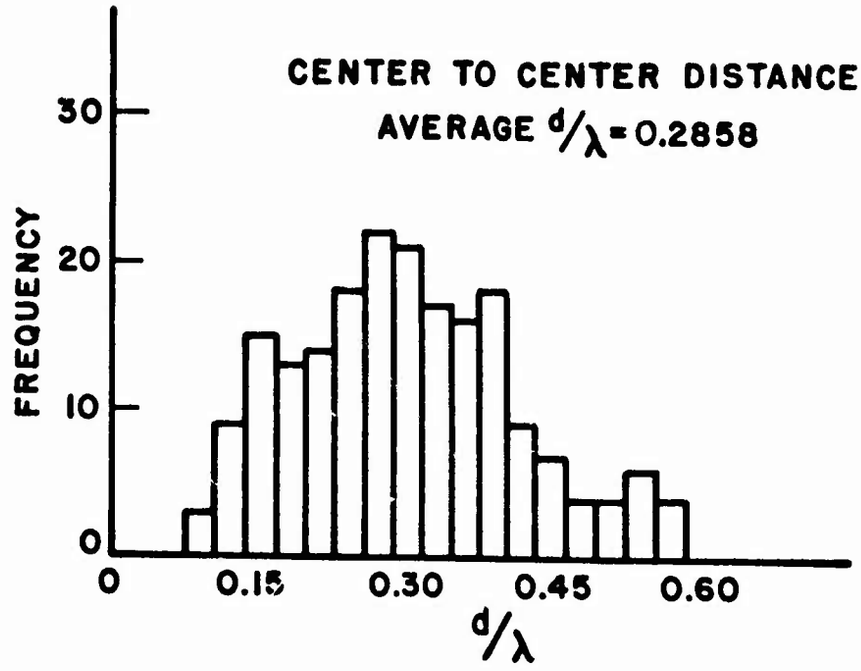


Figure 97. Histogram of the center-to-center distance of random two-dipole clouds. Average spacing $d/\lambda \simeq 0.29$.

From Figs. 95a and 98a, for the two dipoles coupled, we see a trend consistent with what has been said in the above paragraph. These figures show an average cross section of $\sim 0.25\lambda^2$ for both the far-spaced and the near-spaced dipoles, seemingly violating the earlier conclusion that the near-spaced dipoles, because they are more strongly coupled, should present a smaller average cross section than do the far-spaced dipoles. What we are seeing however is the effect of coherent excitation due to the close proximity of the wires, an effect which increases the average cross section. It is not increased as high as the uncoupled wires, however, due to the coupling which tends to reduce the average cross section, and so ends up with a value which in our example happens to about equal that for the coupled wires with larger spacing.

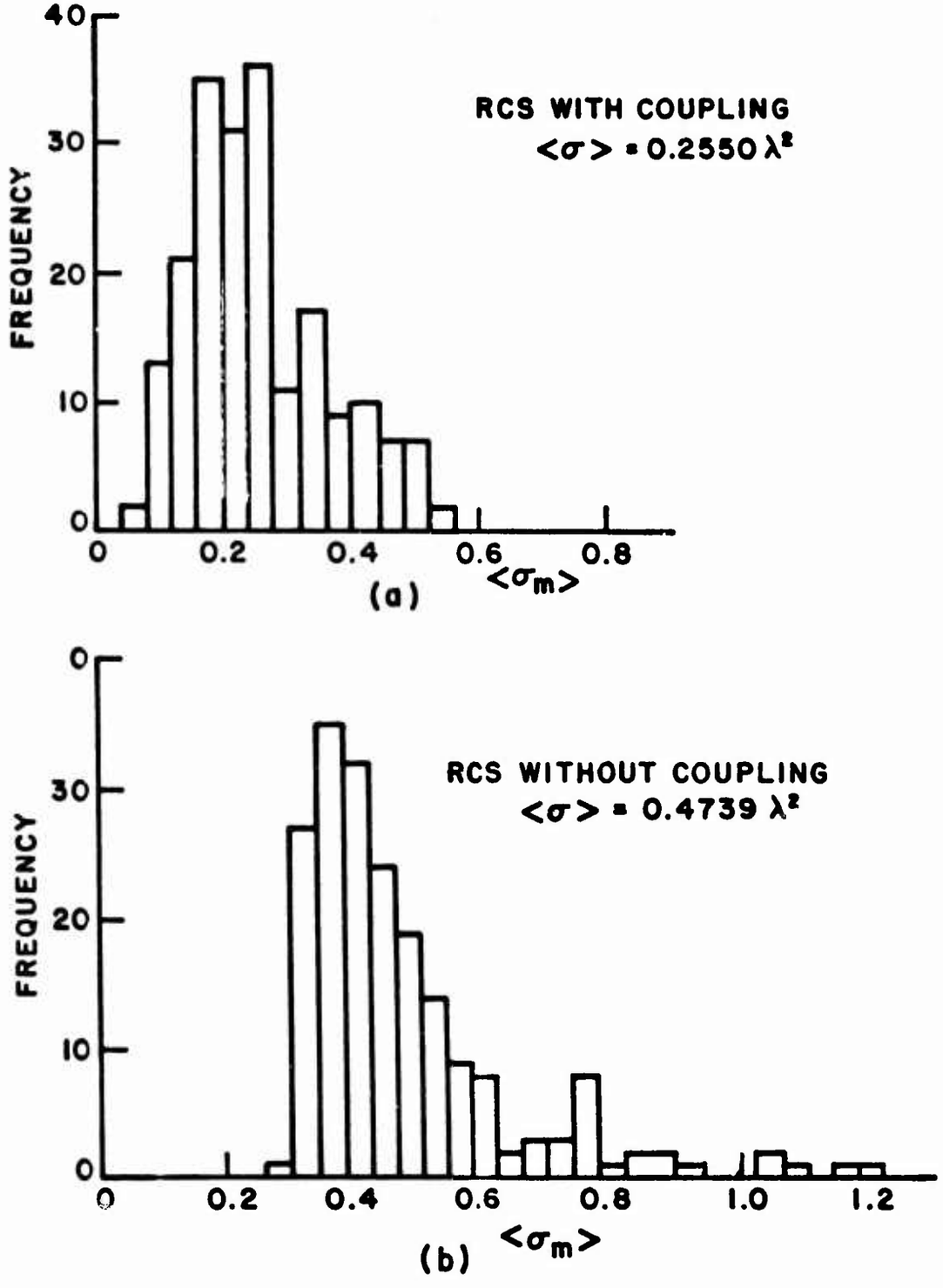


Figure 98. Histograms of the spatially averaged radar cross sections of the clouds generated for Fig. 97. (a) with coupling (b) with no coupling.

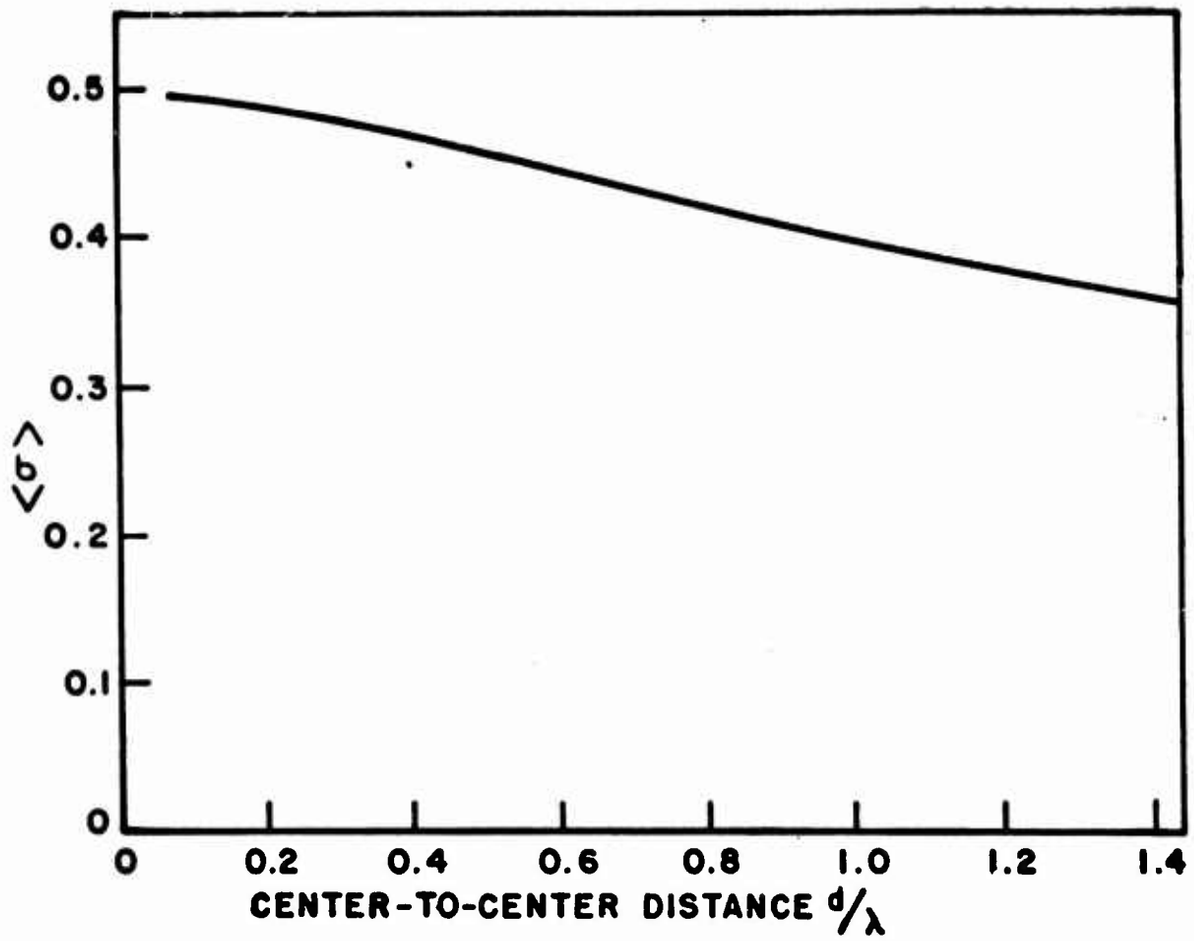


Figure 99. Average cross section of random two-dipole clouds as a function of average center-to-center spacing between dipoles.

E. The Question of Mixed Lengths

Tenuous chaff clouds consisting of wires of identical length resonating at about a half wavelength display a bandwidth of about 20%, or somewhat greater for denser clouds (see Fig. 15). Threats which are expected over a greater bandwidth than this demand the use of a variety of wire lengths within the same chaff cloud. This poses the question, what is the influence of coupling on chaff elements of mixed lengths? If the bandwidth ratio of interest is 10 to 1 for example, at the low end of the spectrum some wires will be 0.05 λlong and others will be 0.5λ long. At the high end of the spectrum, some wires will be 0.5λ long and others will be 5λ long. The first case presents little difficulty - the shorter

wires are very ineffective scatterers, coupled only weakly to nearby neighbors and their presence can be ignored to a very good approximation. The second case does present problems, however, because the larger wires require a large number of segments to model them accurately. This has the undesirable effect of reducing the total number of wires permitted in a cloud, assuming a fixed matrix size.

During the period of this contract, we made a brief investigation of small clouds containing wires of two lengths, 0.475λ and 0.703λ long. The shorter wires were resonant, each one in isolation having a maximum tumble average cross section of about $\langle\sigma_1\rangle \simeq 0.15\lambda^2$; the longer wires were antiresonant, each one in isolation having a minimum tumble average cross section of about $\langle\sigma_2\rangle \simeq 0.038\lambda^2$ [61]. These two lengths were purposely chosen to take advantage of their maximum disparity in tumble average echo. Four curves, shown in Figs. 100-103, were generated, each showing the calculated averaged backscattering cross section $\langle\sigma_m\rangle$ of clouds containing N = 2, 4, 6, and 8 wires, averaging being over 512 look angles in the θ-θ and ϕ-ϕ polarizations of transmitter and receiver. Ten clouds, each with these numbers of wires, were calculated and the $\langle\sigma_m\rangle$ of each are plotted as a dot. The ensemble average of these ten values are plotted as a cross. Four different values of average spacing, d/λ = 4.0, 1.0, 0.5, 0.25, as defined for inhomgeneous clouds, were assumed, and in all cases equal numbers $N_1=N_2=N/2$ of wires of the two lengths were assumed. On each of the curves, the straight line $N_1\langle\sigma_1\rangle+N_2\langle\sigma_2\rangle$ vs N appears in order to give the reader an estimate for the average echo in the absence of coupling. The trends are the same as those observed in previous work treating clouds of identical resonant wires. Average echo is reduced by coupling but not by as large a percentage as in the case where all wires are resonant. For example, for d/λ = 0.5, if all N wires are resonant, one expects the average echo of N wires with coupling to be about 60% of that with these same wires uncoupled. If, however, the N wires are half resonant and half antiresonant, one can expect the average echo of the N wire mixture to be about 78% of that with the same wires uncoupled. Evidently, the antiresonant dipoles, whose tumble average echo is naturally low, are not so severely influenced by coupling.

The computer program used to generate data for multiple length chaff is given in Appendix D.

F. Additional Experimental Results

During the earlier phases of this program a few experiments were performed to gather data, to check, or to complement calculated data. Some of the results of these experiments have already been presented where appropriate; in this section we document whatever other experimental data were recorded.

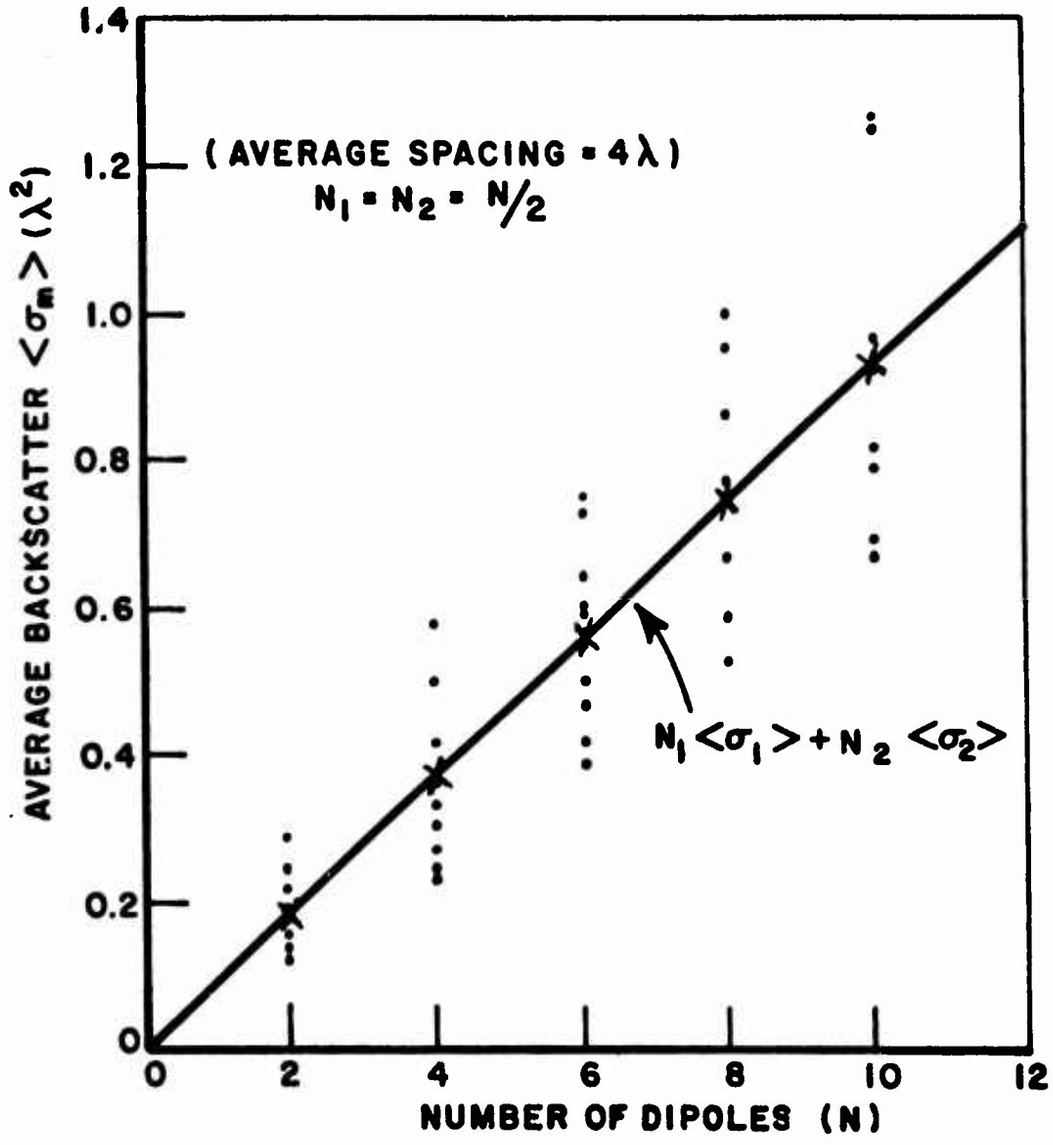


Figure 100. Calculated average backscattering cross sections for ensembles of clouds containing equal numbers of two wire lengths, $\ell_1/\lambda = 0.495$, $\ell_2/\lambda = 0.703$, with average spacing $d/\lambda = 4.0$. Straight line represents decoupled dipoles, crosses are averages of the data.

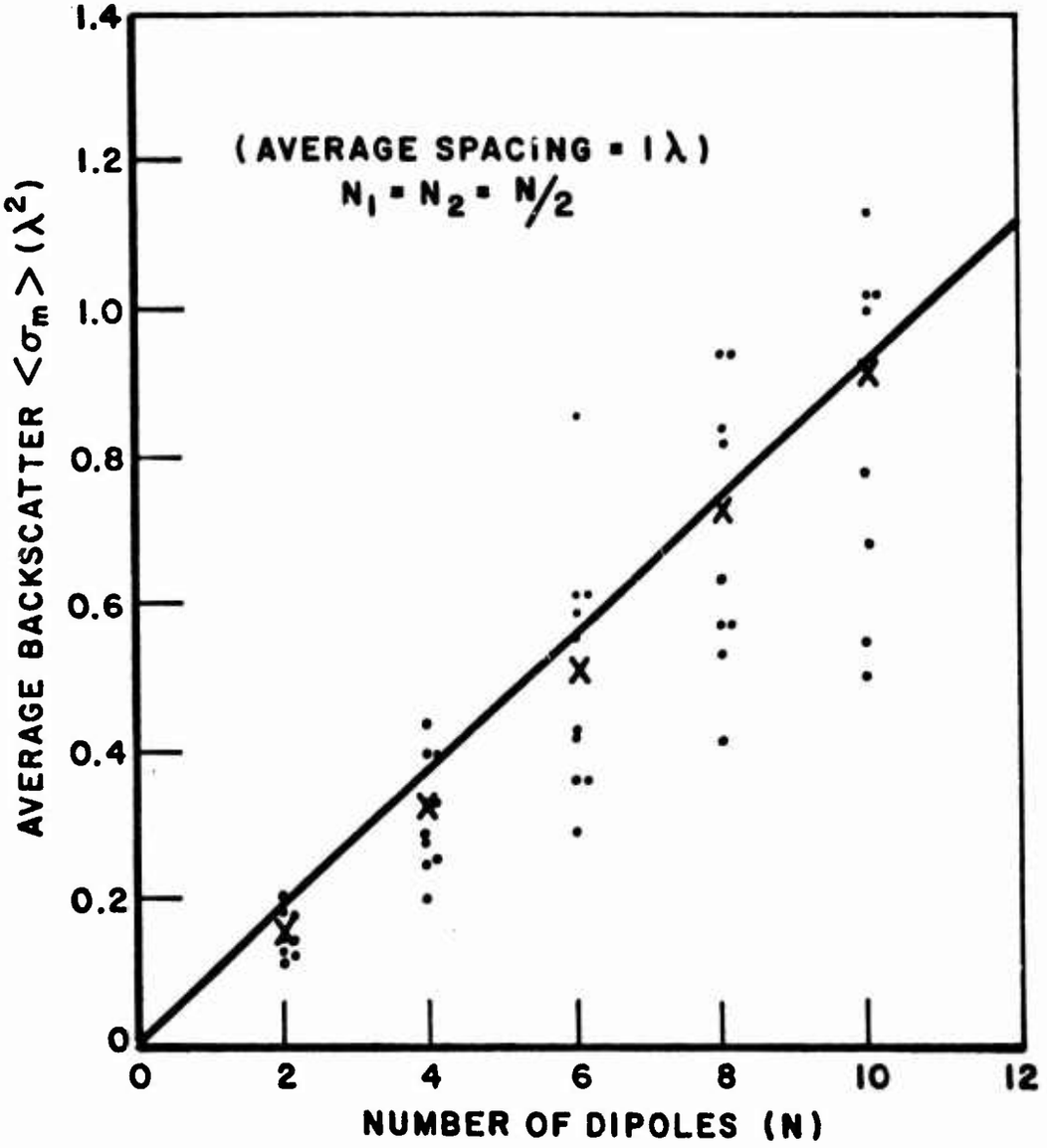


Figure 101. Calculated average backscattering cross sections for ensembles of clouds containing equal numbers of two wire lengths, $\ell_1/ = 0.495$, $\ell_2/ = 0.703$, with average spacing $d/\lambda = 1.0$. Straight line represents decoupled dipoles, crosses are averages of the data.

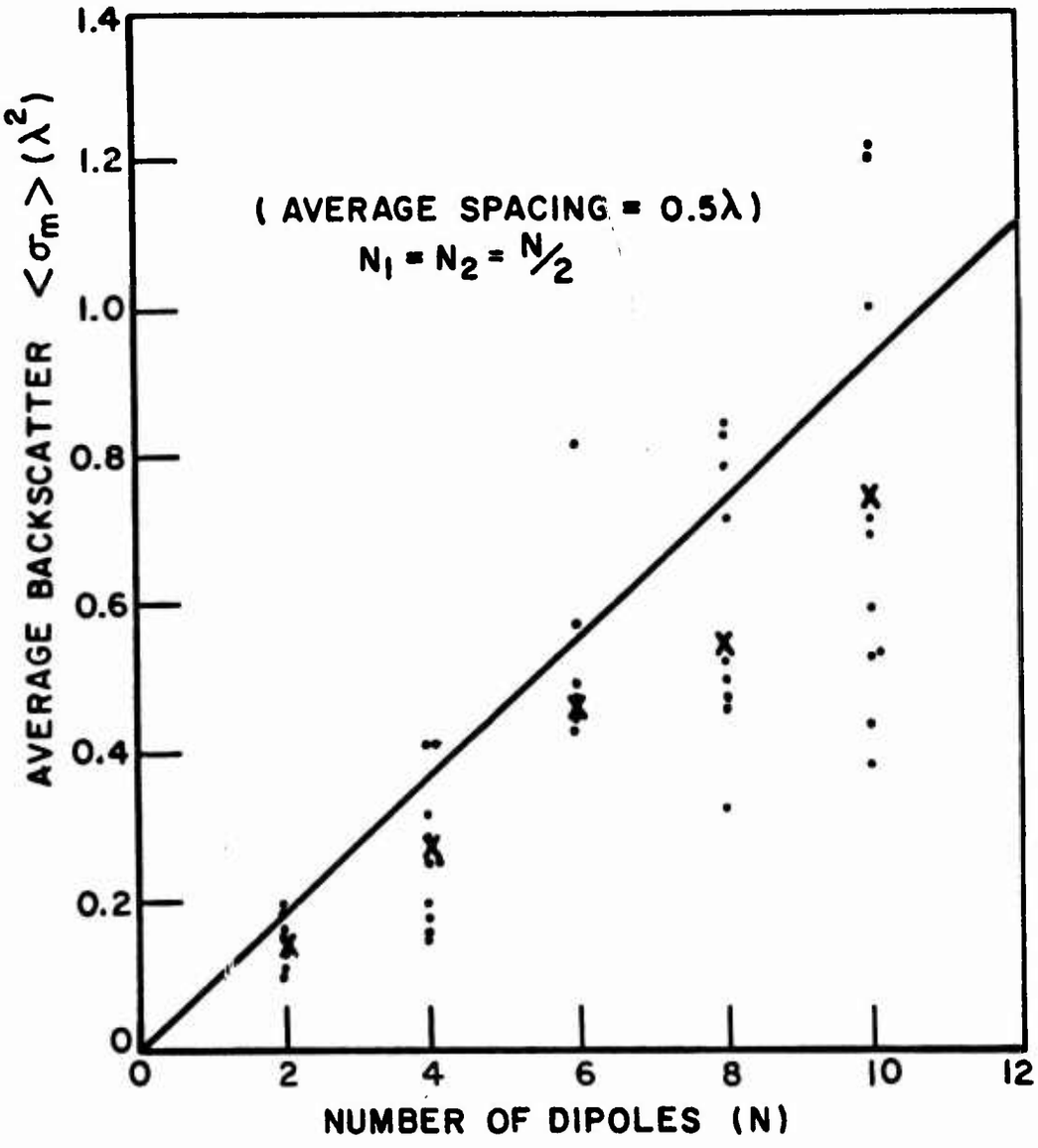


Figure 102. Calculated average backscattering cross sections for ensembles of clouds containing equal numbers of two wire lengths, $\ell_1/\lambda = 0.495$, $\ell_2/\lambda = 0.703$, with average spacing $d/\lambda = 0.5$. Straight line represents decoupled dipoles, crosses are averages of the data.

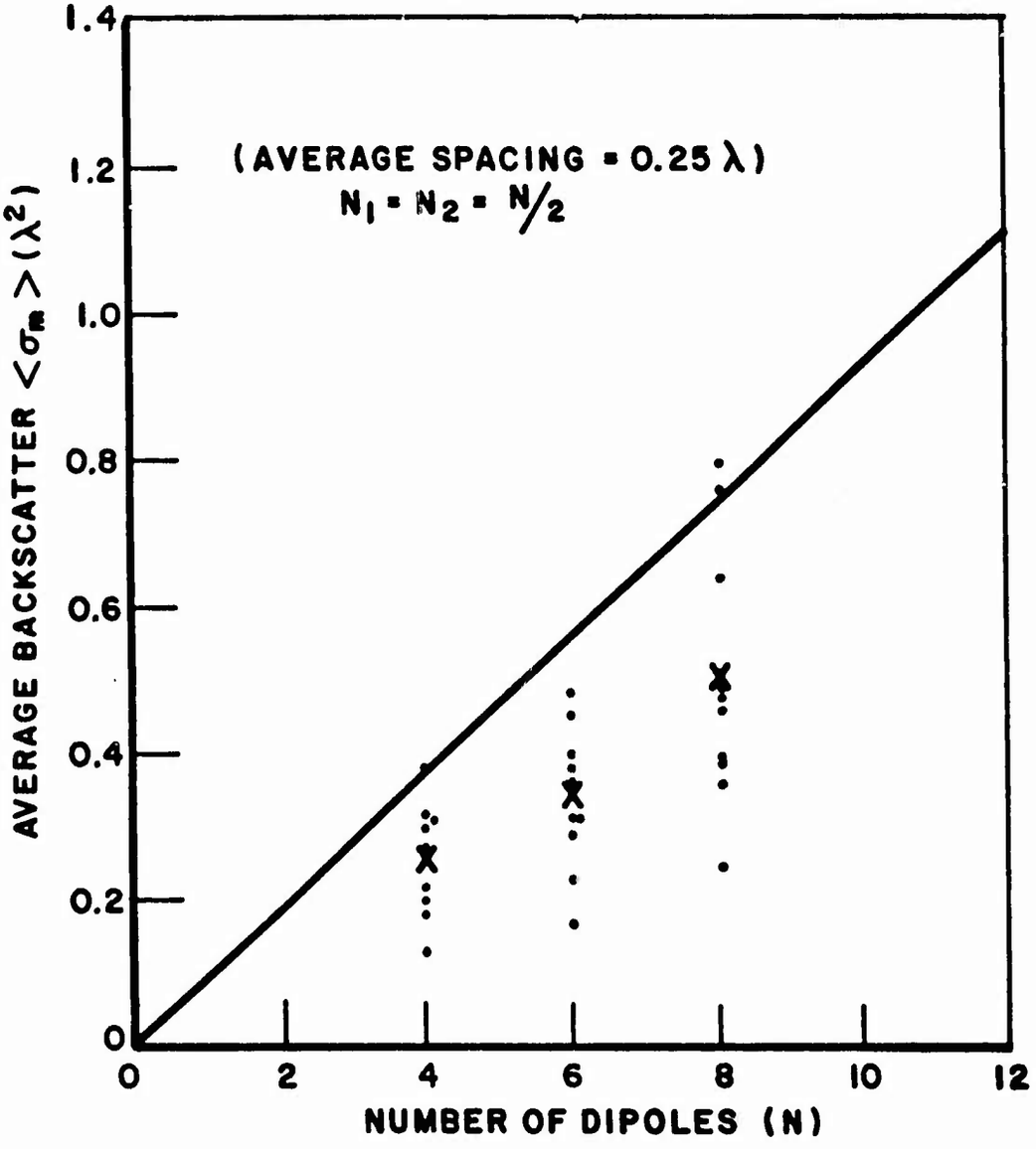


Figure 103. Calculated average backscattering cross sections for ensembles of clouds containing equal numbers of two wire lengths, $\ell_1/\lambda = 0.495$, $\ell_2/\lambda = 0.703$, with average spacing $d/\lambda = 0.25$. Straight line represents decoupled dipoles, crosses are averages of the data.

(1) Experimental Verification

It is important to verify that the data calculated by computer are indeed good approximations to what would be measured in controlled laboratory experiments. Such experiments were performed on selected frozen models whch were considered to be severe tests of the computer predictions. Figures 104 and 105 show photographs of a matrix of 125 carefully dimensional 2.0" x 2.0" styrofoam cubes. Each cube can have embedded in it (by means of an accurately machined jig) a rod, located near one edge. By orienting each cube in one of 12 different possible positions and placing it in a styrofoam box, a cloud of 125 dipoles can be built up. Although this scheme does not allow all possible orientations of the dipoles, there is a sufficient variety of orientations and spacings to create a rather aperiodic structure.

The most severe test of the accuracy of the computer routines is to compare experimental and calculated backscattering patterns (using full matrix Scrout) when all dipoles are oriented parallel in a regular periodic array and closely spaced. Results for two such cases are presented in Figs. 106 and 107 for 27 dipoles and 125 dipoles, respectively. In both cases all dipoles were horizontal and the backscattering cross sections for horizontal polarizations of transmitter and receiver were recorded in a horizontal 360° cut around the cloud. A frequency of 3.13 GHz was chosen to bring each dipole to its free space resonance, and caused a spacing between adjacent dipoles of $d/\lambda = 0.53$. Typical disparities of about 2 dB are evident, but the overall pattern is well predicted. Some of the disparity is due to imperfections in the mathematical model and round off error, but most of the error is caused by errors in measurement. We made several experimental runs, tearing the cloud apart and reconstructing it as identically as possible between each run, and found that the experimental data had a variance which enclosed the calculated curves. This convinced us that the computer routines are accurate and the data based on them are as valid as if measured. Another 27-dipole cloud was constructed with the dipoles randomly oriented with an average spacing of $d/\lambda = 0.53$. Calculated and measured results are shown in Fig. 108 and again they compare very well.

In the above experiments, the 125-cube styrofoam matrix without dipoles had an echo below the internal noise level of the measuring system.

(2) Extinction Measurements Through an Artificial Chaff Cloud

A beam proceeding through a random medium such as a chaff cloud suffers energy loss through scattering by each particle into directions other than forward and into polarizations other than incident. This

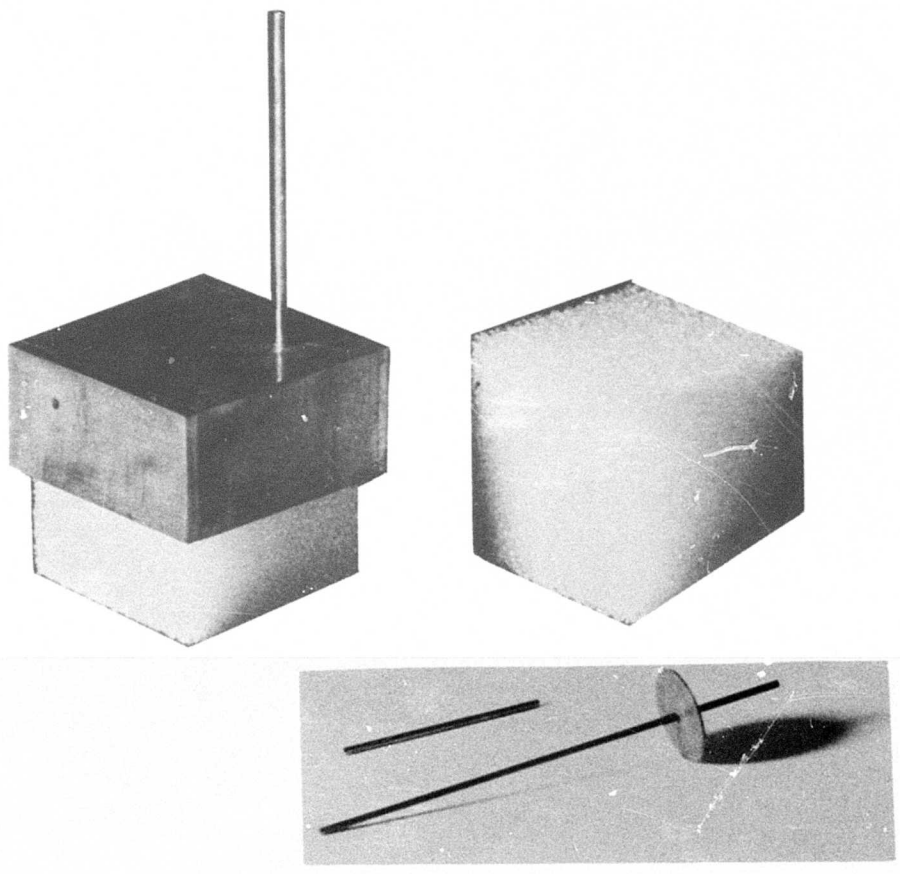


Figure 104. A typical styrofoam 2" cube, dipole, and jig for accurately inserting the dipole.

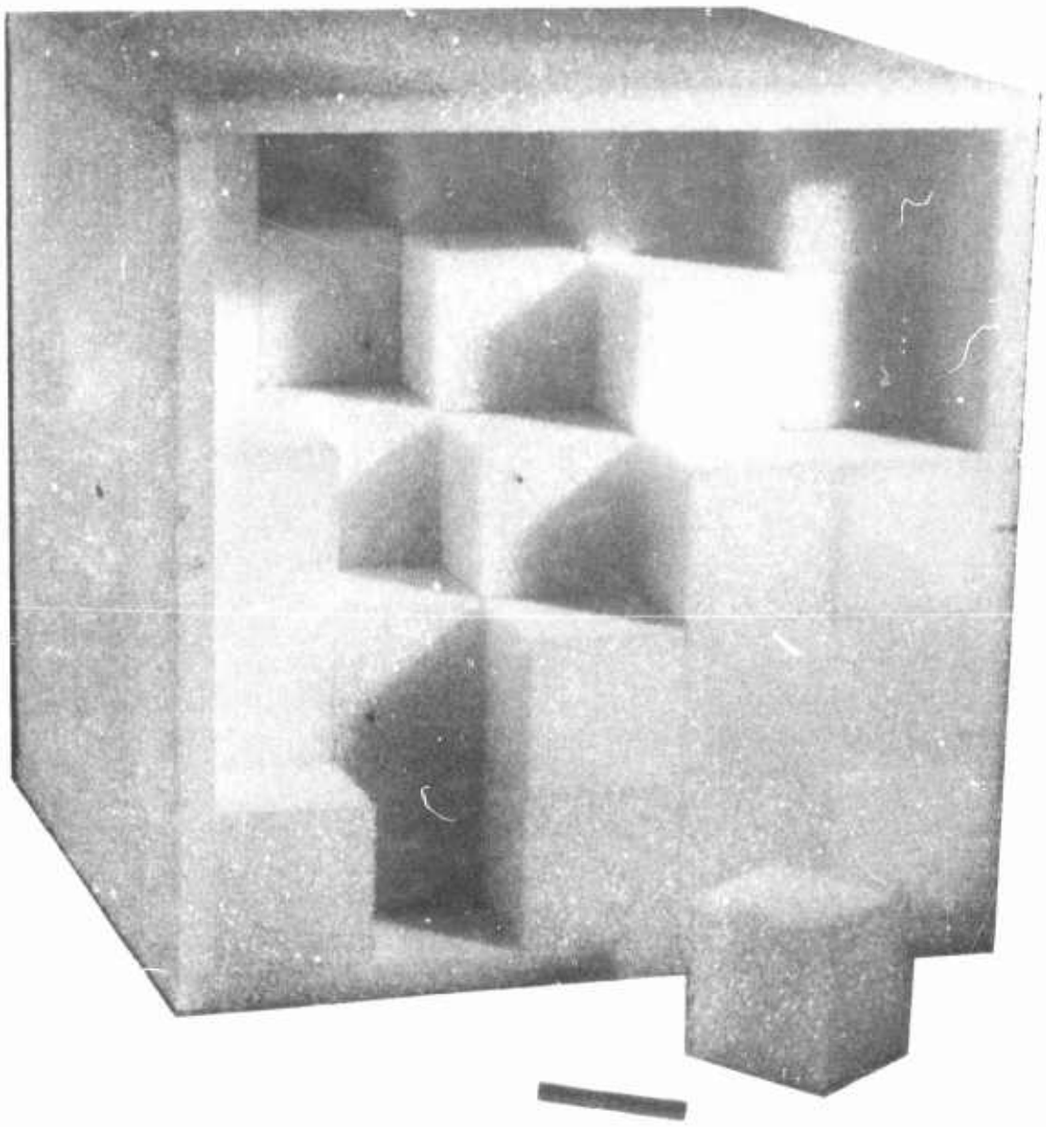


Figure 105. Styrofoam cubes in styrofoam box for creating a cloud of dipoles.

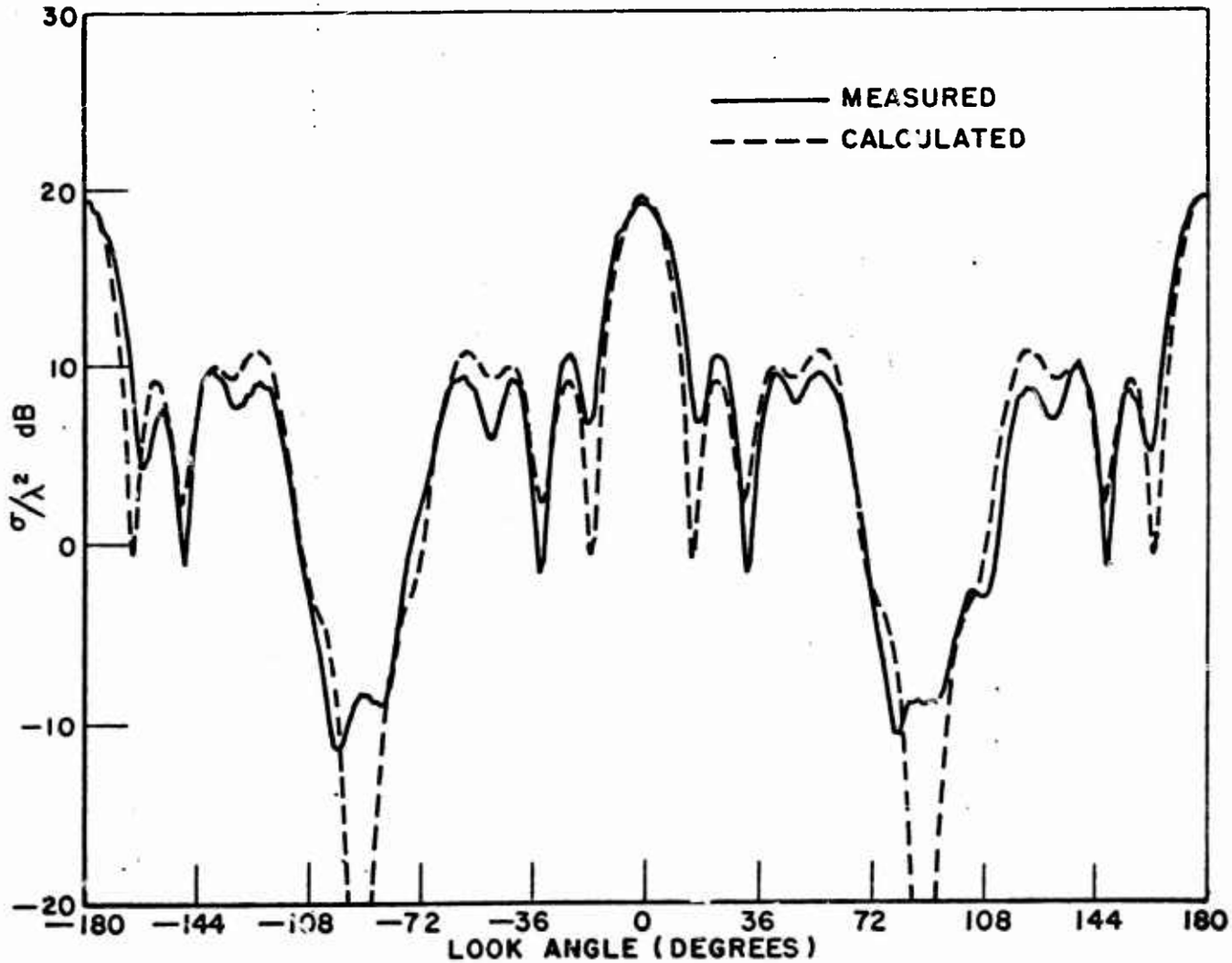


Figure 106. Measured and calculated E-plane backscattering cross-section patterns for a 3 x 3 x 3 array of 27 parallel dipoles spaced about 0.53λ apart.

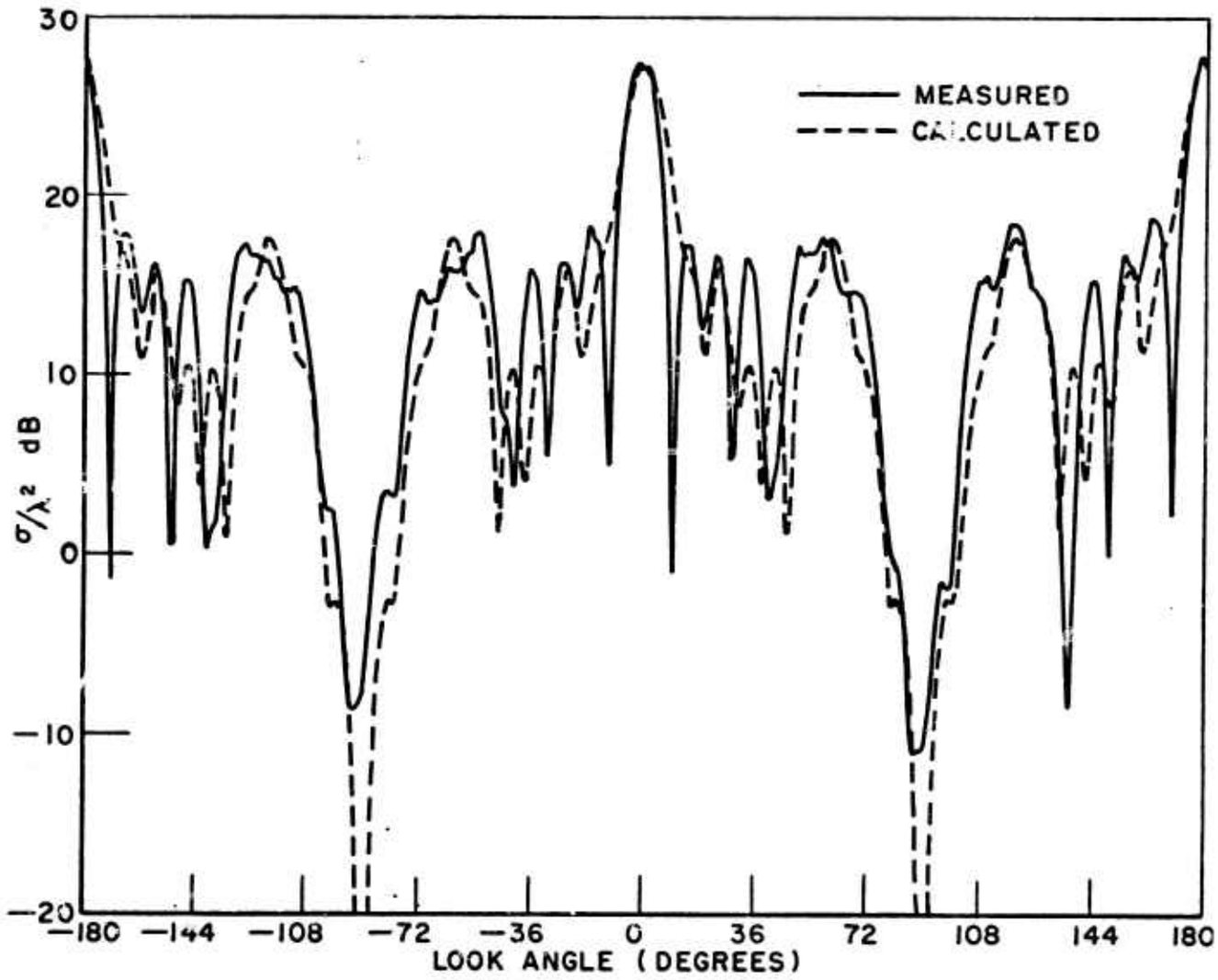


Figure 107. Measured and calculated E-plane backscattering cross-section patterns for a 5 x 5 x 5 array of 125 parallel resonant dipoles spaced about 0.53λ apart.

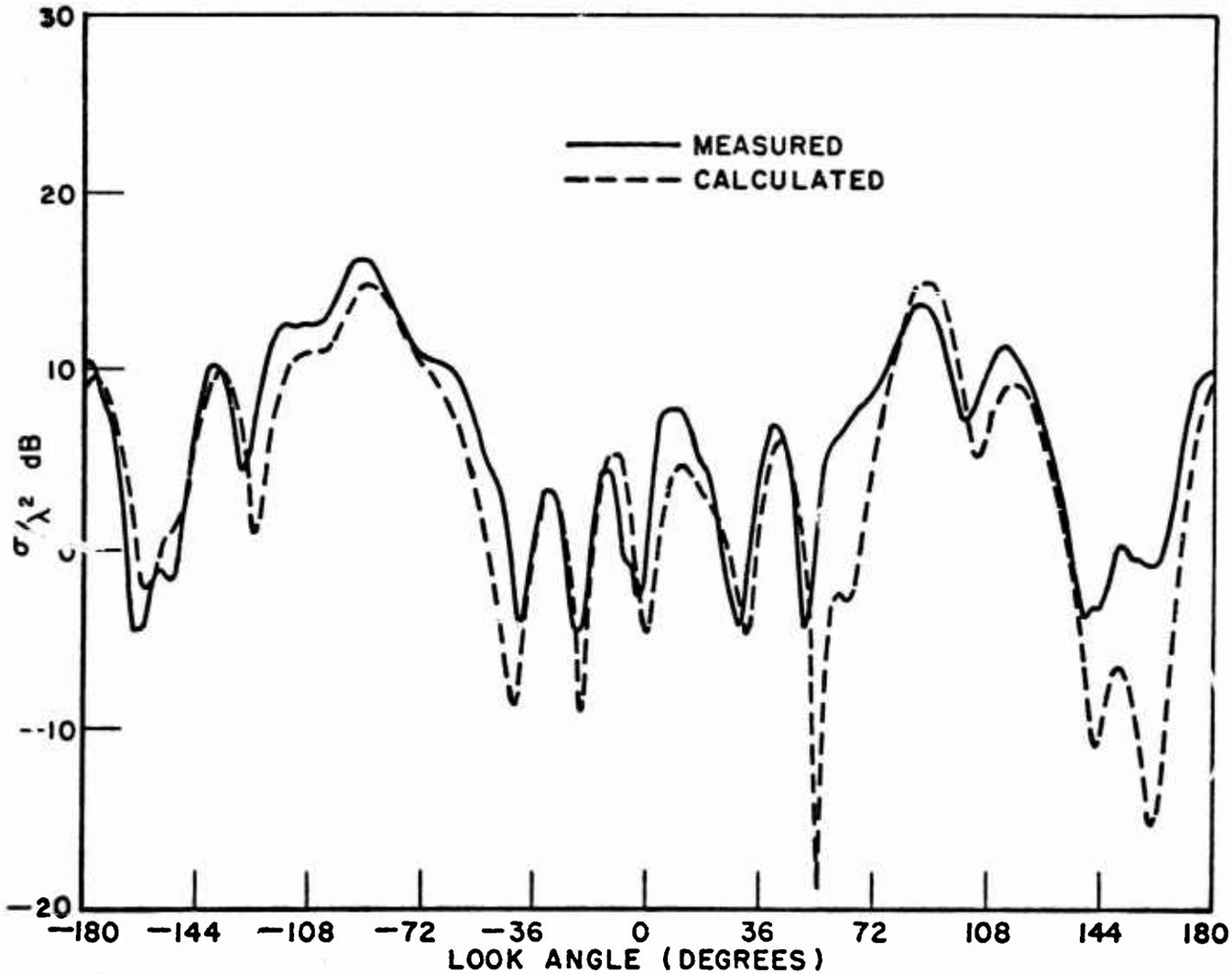


Figure 108. Measured and calculated E-plane backscattering cross-section patterns for a cloud of 27 dipoles randomly oriented in styrofoam cubes. Average dipole spacing was 0.53λ.

extinction of the beam is important to chaff cloud investigations because, clearly, if it causes a significant diminution of energy toward the rear of the cloud, chaff elements there will be relatively ineffective contributors to backscatter and might be better used elsewhere. To observe extinction and extinction rate through a chaff cloud as functions of dipole density is therefore of interest to us. However, to obtain significant extinction requires a sizeable cloud containing thousands of dipoles, and considering that the problem is a statistical one in which many similar clouds must be generated to obtain averages, the computer generation of extinction data becomes a formidible task. Moreover, in the early part of this program we did not have the capability of generating such large clouds, so we turned to a few experiments to observe the extinction through artificial chaff clouds. In this section we document some of the results of these experiments.

Figures 109-112 show the average insertion loss observed between a horn antenna and a receiving dipole probe situated in various line-of-sight positions within a medium of tumbling resonant dipoles. The dipoles were enclosed in a wooden "box", 30" long in the direction of propagation and having a 24" square cross section, with foam "windows" at both ends and hairflex absorber (or aluminum foil) lining on the other four walls. The box was supported on circular rims such that it could be rotated continuously about the line-of-sight axis, thereby tumbling its contents in a random manner. The dipole probe, encapsulated in a protective foam sphere, was drawn along the line-of-sight from the front window to the rear one along a slender dielectric-tube containing the coaxial line exiting through the rear window and feeding the receiver. A horn antenna disposed about 33" from the front window served as the illuminating source. The signal received from the probe was measured for several minutes duration of tumbling and averaged over this time period for selected probe positions between the windows.

The curves shown in Figs. 109-112 show the averaged difference (in dB) between the received signals without and with dipoles in the box, i.e., insertion loss, for polarization of the probe parallel to and orthogonal to the incident linearly polarized wave. Ideally, this differencing scheme should remove the effect of range on energy decrease and leave only the extinction due to particle scattering. All data were taken at about 2.9 GHz. Figure 109 shows results for 1000 resonant dipoles (nails), each one encapsulated completely in a 2.38" foam sphere. Figure 110 shows results for 1000 resonant dipoles (copper wire), each one encapsulated completely in a 2" diameter foam sphere, tumbled together with 850 free dipoles. Figure 111 shows results for $\sim$3200 resonant dipoles (nails) in 1-1/2" foam spheres while Fig. 112 shows results for $\sim$7200 resonant dipoles (nails) in 1" foam spheres. (In these latter two cases the dipoles protruded outside the spheres.) The effects of progressively higher dipole densities is evident in the progressively increasing

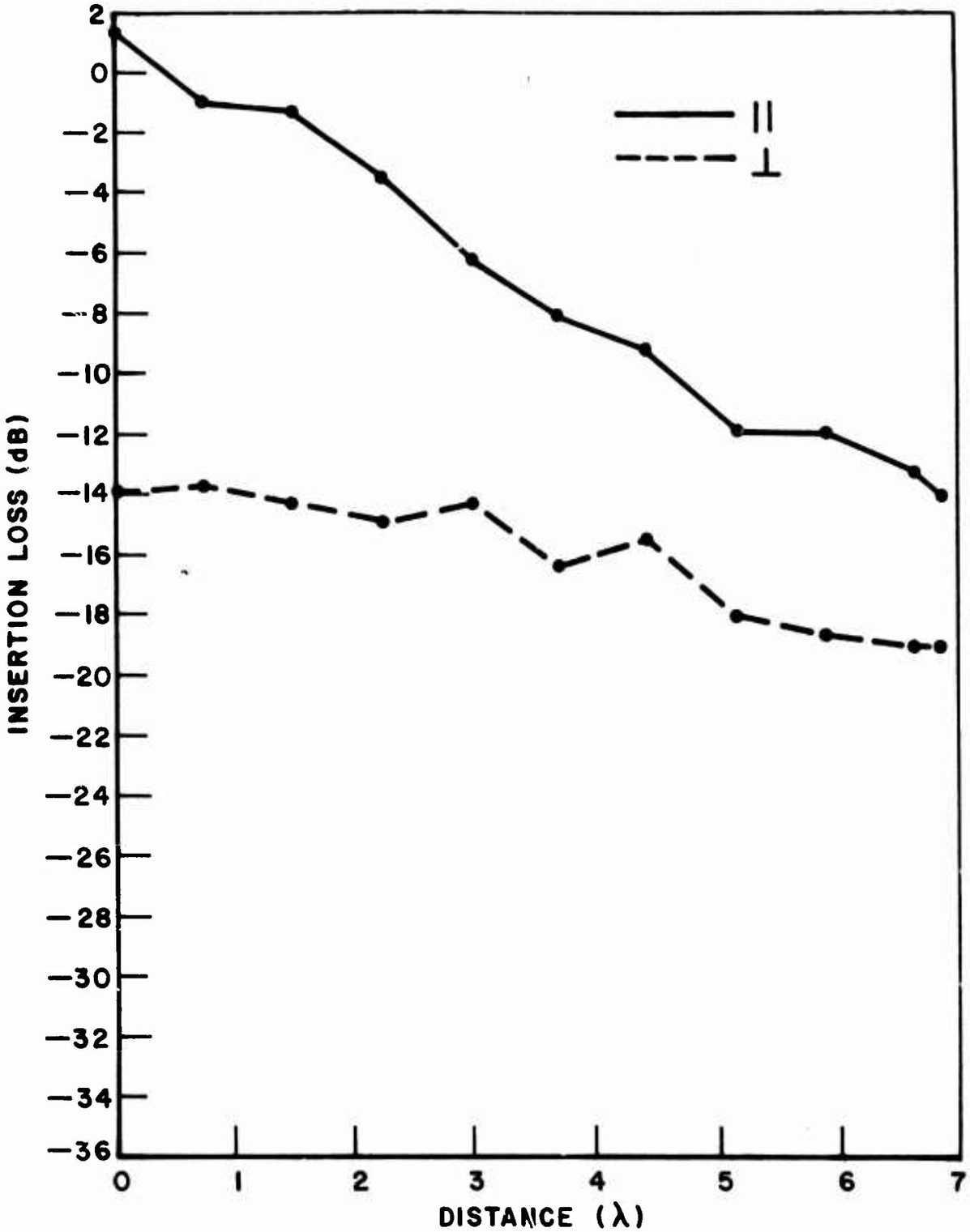


Figure 109. Measured insertion loss of same-sense and cross-sense polarizations for 1000 resonant dipoles encapsulated in 2.38" foam spheres. $f \simeq 2.9$ GHz.

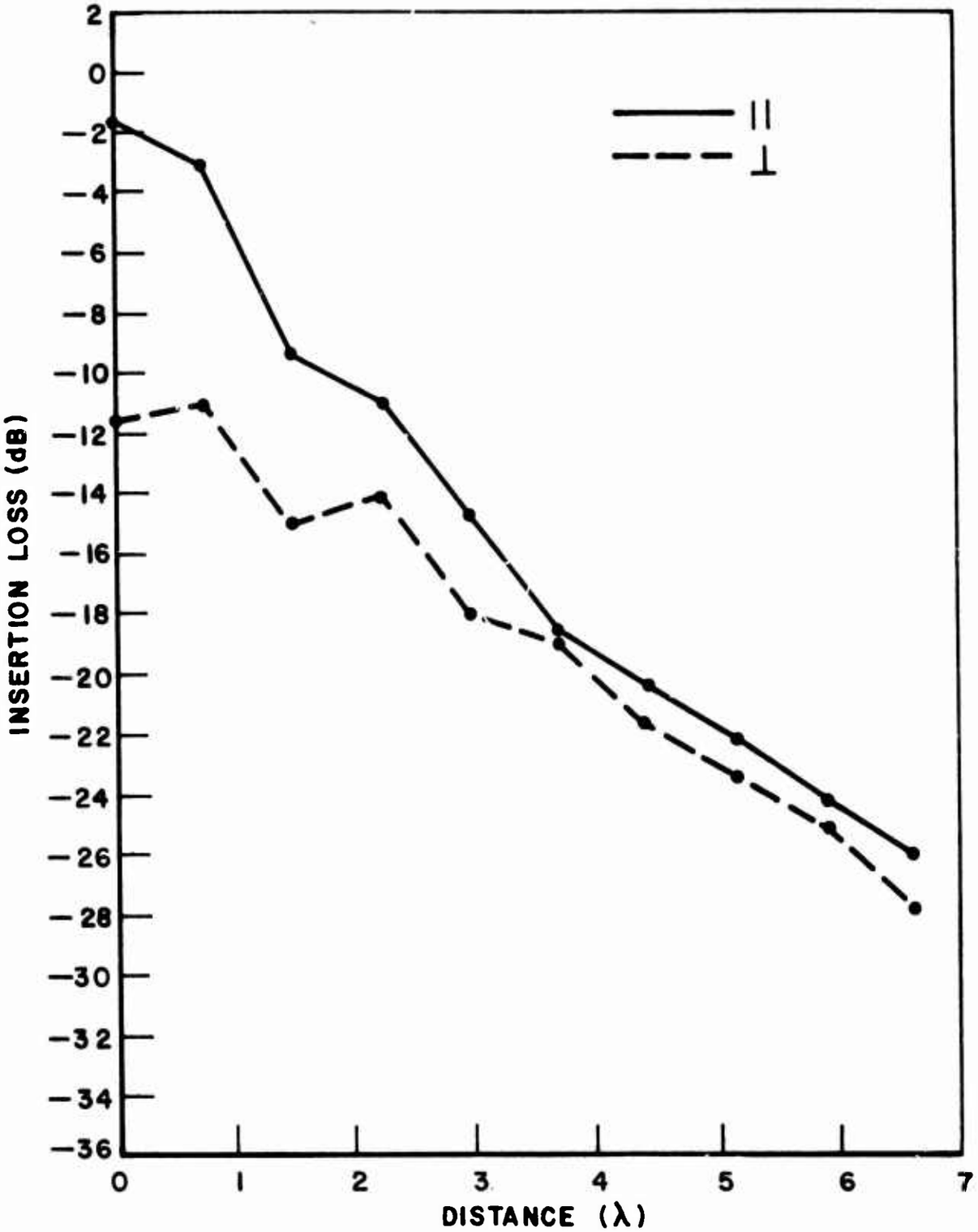


Figure 110. Measured insertion loss of same-sense and cross-sense polarizations for 1000 resonant dipoles encapsulated in 2" foam spheres, plus 250 free dipoles. $f \simeq 2.9$ GHz.

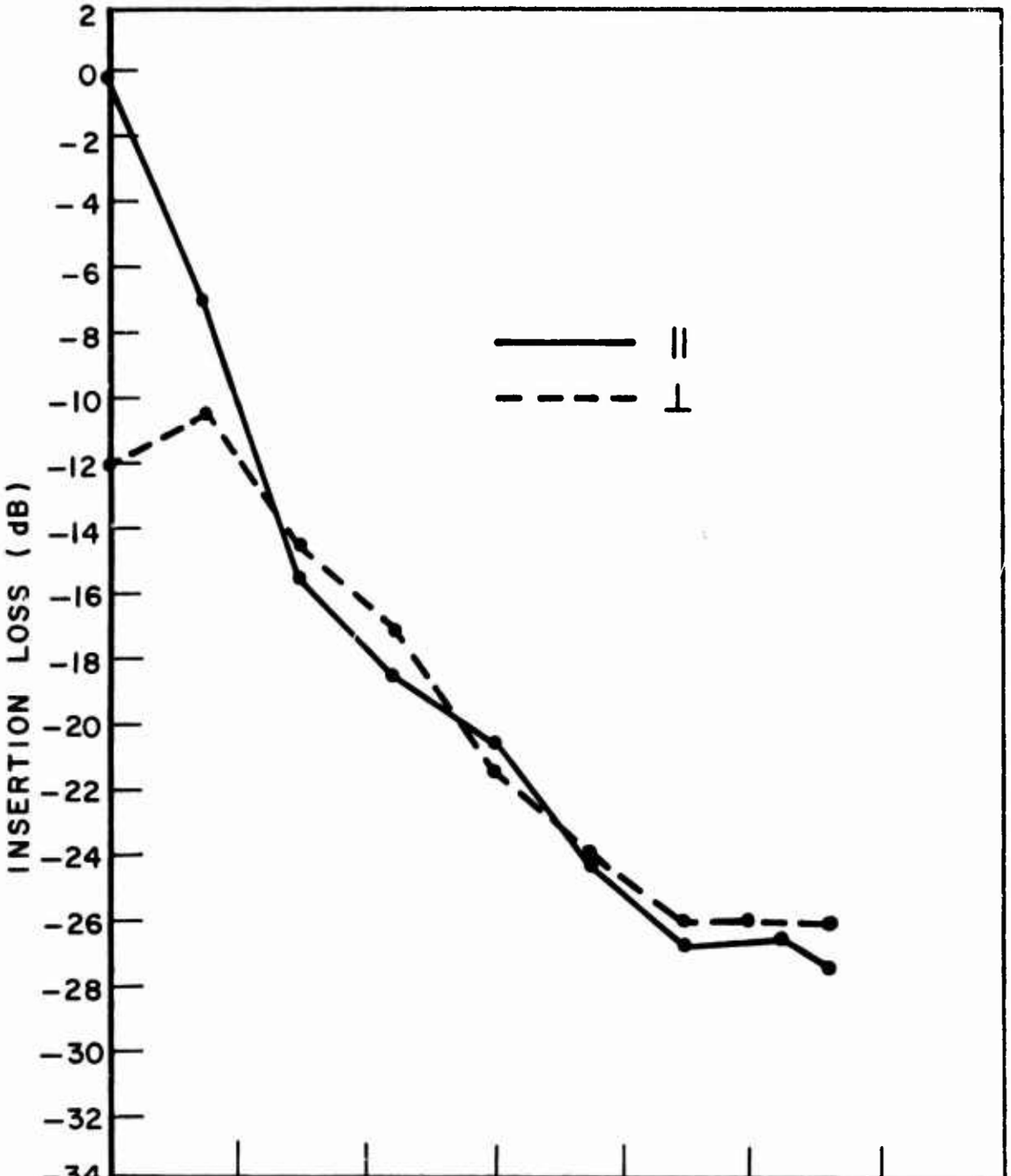


Figure 111. Measured insertion loss of same-sense and cross-sense polarizations for ∿3200 resonant dipoles encapsulated in 1-1/2" foam spheres. f ≃ 2.9 GHz.

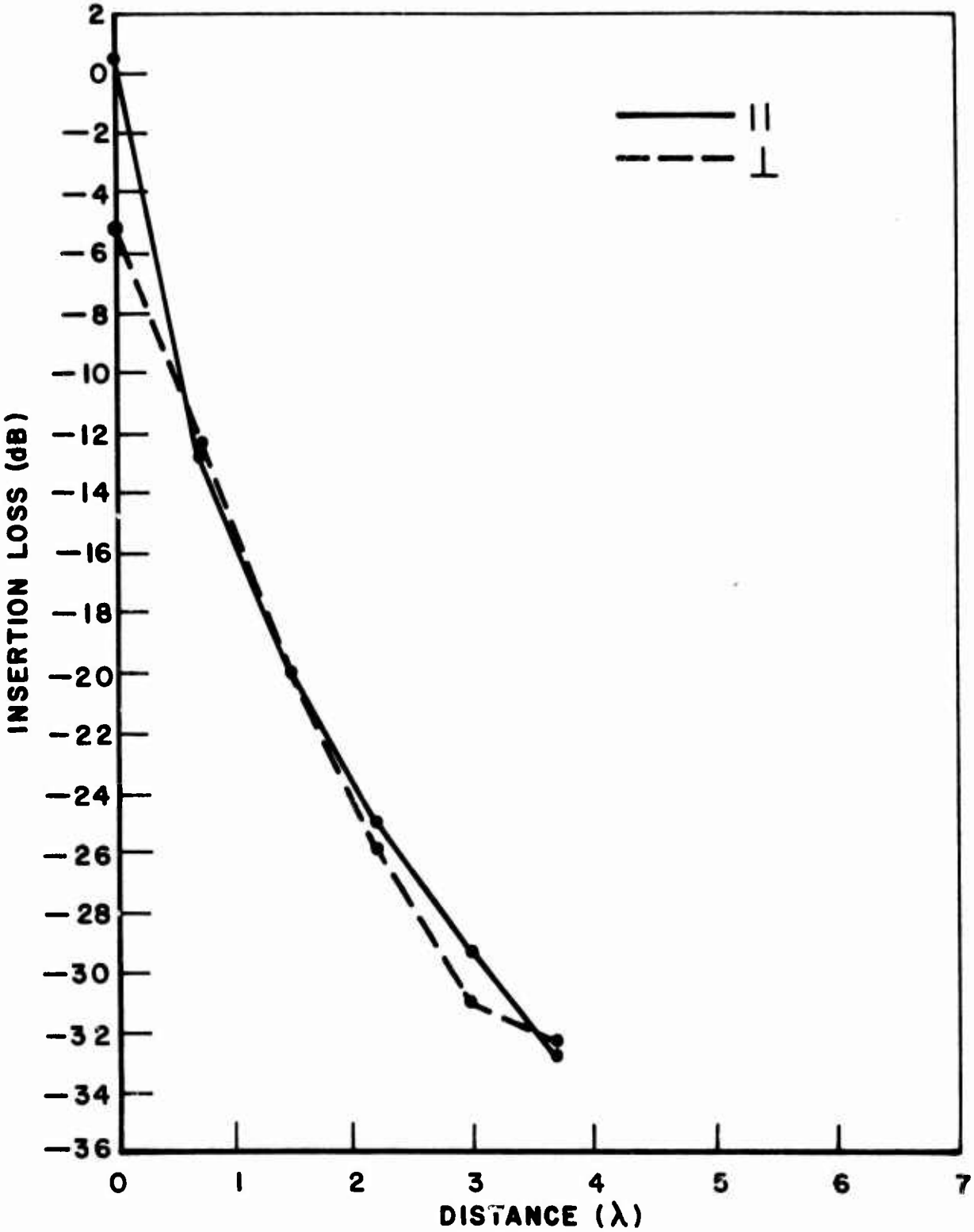


Figure 112. Measured insertion loss of same-sense and cross-sense polarizations for ∿7200 resonant dipoles encapsulated in 1" foam spheres. f ≃ 2.9 GHz.

extinction rates shown in this sequence of figures. Some standing wave effects between the windows are also evident in the non-monotonically decreasing character of the curves. Averaging this undesired effect out, one observes extinction rates which are higher in the region within 3 or 4 wavelengths of the front interface than they are deeper in the cloud. Deep in the cloud the parallel and orthogonally polarized components exhibit the same extinction rates as well as the same extinction. This is to be expected because deep in the cloud the incident wave has been so severely depolarized by the many random scattering interactions that no polarization preference exists. Table 8 lists the approximate extinction rates deep in the cloud.

TABLE 8
EXPERIMENTALLY DETERMINED EXTINCTION RATES

Fig. No.	Density (dip/λ^3)	Extinction rate (dB/λ)
109	$\sim$ 5	$\sim$2.9
110	$\sim$ 7	$\sim$3
111	$\sim$ 9	$\sim$3.6
112	$\sim$12	$\sim$6

The dipole densities listed here are rough estimates.

Some additional tests were performed with metal sheets covering two and then four of the hairflex-coated walls. Energy scattered by the particles out of the incident beam into the walls is absorbed there if all walls are hairflex. This would simulate a cloud smaller in transverse extent than the incident beamwidth, any energy exiting the sides of the cloud being lost to space. Energy scattered by the particles out of the incident beam into the walls is reflected back into the beam if all the walls are metal. This would simulate a very large cloud illuminated by a plane wave. As expected, the extinction rate was reduced in this latter case; for example, in one set of tests the observed rates deep in the clouds were 4.1 dB/λ, 3 dB/λ, and 1.8 dB/λ with 0, 2, and 4 walls covered with metal, respectively.

(3) Scattering from Touching Chaff Elements

A series of measurements were made of the radar backscatter from 4" foam spheres sprinkled with aluminum strip chaff and aluminized glass chaff provided by the Avionics Laboratory. Figures 113-116 relate to the aluminum strip chaff and Figs. 117 and 118 relate to the glass chaff. In all figures, the solid curves represent the echo of a 4" foam sphere silver-painted (its imperfect surface explaining the echo fluctuations with 360° of rotation). The other curves represent foam spheres coated with the chaff elements. Each figure presents two such curves, for two 360° cuts about the sphere, each labelled with the curve average in λ^2.

The chaff elements were applied by sprinkling them randomly upon a sphere made tacky with a spray solvent. The aluminum strip chaff density was quite low - on the order of a 100 elements distributed over the whole surface. (A surface density of about 0.5 dipole/λ^2.) They were 1.5 cm long, implying resonance of an individual dipole at about 9.5 GHz. Because so few elements touched, the sequence of curves (Figs. 113-116) show a resonance effect about this same frequency. The average echo at resonance (9.53 GHz) was on the order of $6.6\lambda^2$, about equal to the cross section of a solid metal sphere. It's also about equal to 50 (i.e., one half the total number of dipoles) times the tumble average cross section of a single dipole at resonance, giving the impression that the front hemisphere looks almost like a solid metal hemisphere, shielding the elements on the back hemisphere. A short effort (described in a monthly letter to the sponsor) was devoted to investigating this shielding effect, but results were inconclusive.

G. The Aircraft-Chaff-Tracker Interaction Problem

During the last month of the contract, the sponsor supported an effort to computer-simulate the interaction of a combined aircraft-chaff cloud target with a split-gate missile tracking radar. Because the research effort is concurrent with the writing of this final report, only its general outline is described here. Details are presented in Appendix G.

A computer software routine has been developed which presents on a CRT the simulated aircraft radar plot of both the PPI indicator and a height finding indicator. Relative orientation of both aircraft and missile are displayed as a visual aid. Evasive aircraft maneuvers in all three dimensions and in time incorporating preselected aircraft response characteristics (due to inertia and stress limitations) can controlled by the operator. Chaff clouds are deployed at will by action of the operator. The missile radar, incorporating either a split-gate range tracker or a leading edge range tracker with selected gate width and time response, dictates the trajectory of the missile under selected time response.

characteristics of the angle tracker and missile inertial system. (This program includes a missile velocity increase with time and variable maneuverability with altitude and shift of its center of gravity with burn.) Presently, the aircraft presents to the radar an echo composed of three Gaussian pulses of selected durations and amplitudes. Also, deployed chaff clouds remain fixed in space (but fall behind the aircraft as the aircraft moves ahead) and present to the tracker an echo which remains constant in time and aspect. The integrated radar signal returned from the aircraft and chaff clouds, if any are present, is calculated as a function of time and relative missile position and compared with an assumed thermal noise signal. If the resulting signal-to-noise ratio dips below a selected threshold, break-lock conditions apply and the missile continues on a ballistic flight. Presently, a numerical printout is made of the time-space-S/N history of assumed tactical manuevers. This is difficult to interpret at a glance, so some consideration is being given to a graphical plot of the same data so that successful tactics can be discerned and modified easily and quickly. Also the data will be taped for later retrieval and analysis. A sketch of this work is given in Appendix G.

Ultimately, the success of the effort described to simulate the radar interaction problem depends upon the validity of the input data, i.e., good radar echo from aircraft and chaff clouds as functions of aspect and time, accurate dynamic response characteristics of the aircraft and missile and of the radar, and realistic tactical maneuvers. It is toward this goal that the present chaff contract and complementary ones are directed.

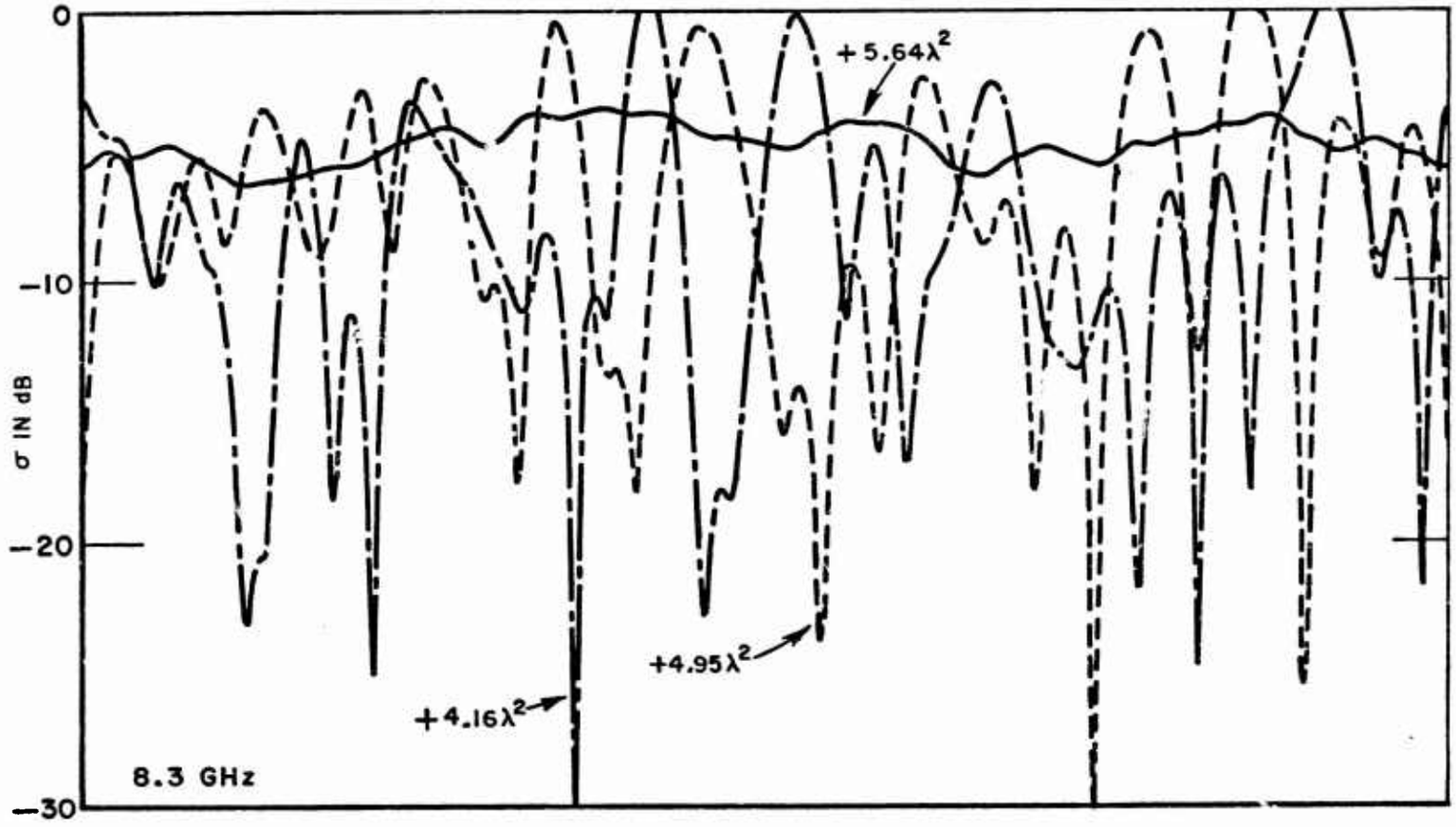


Figure 113. Radar cross section vs 360° of rotation of 1.5 cm Aℓ chaff on 4" foam sphere.

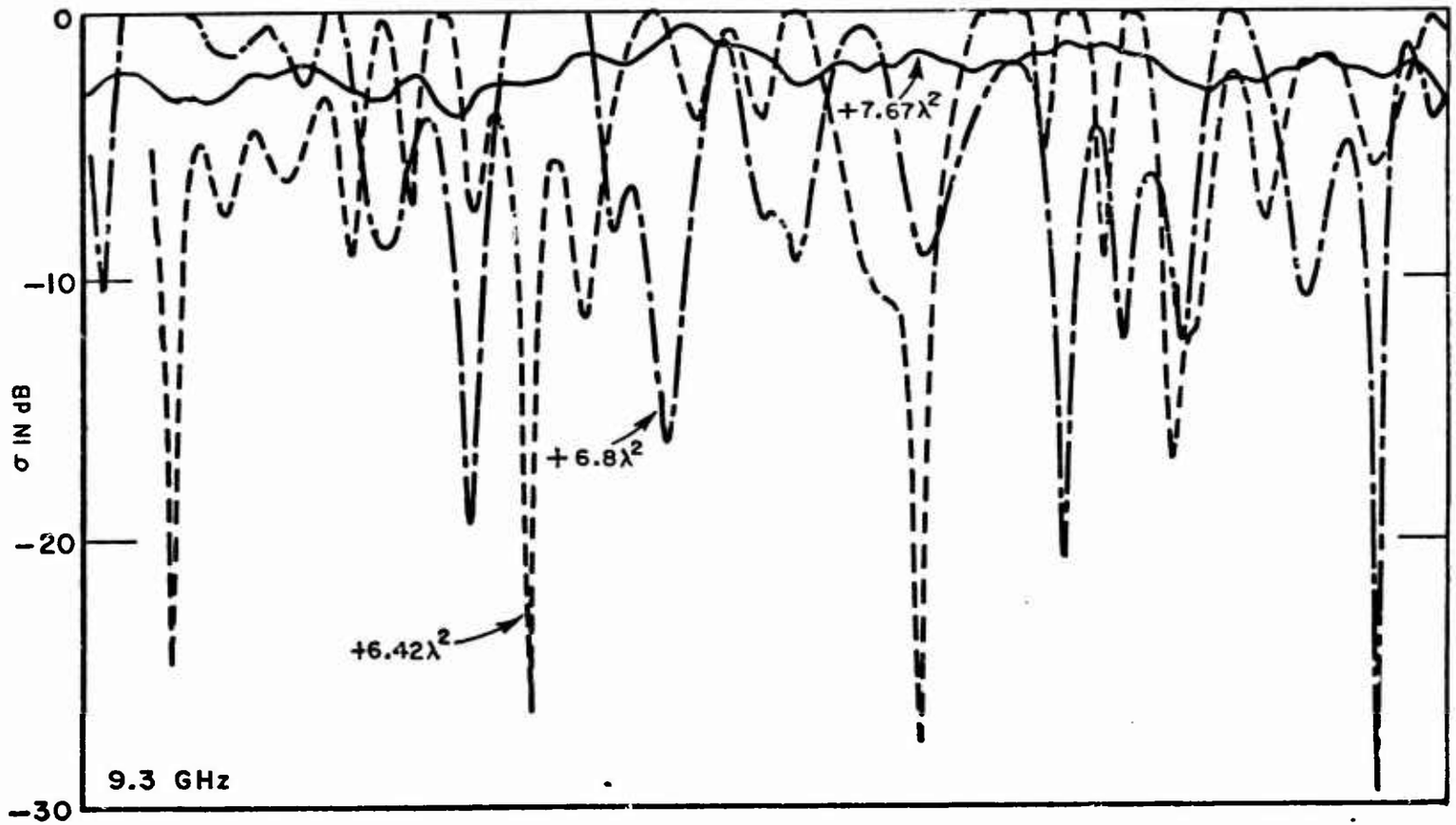


Figure 114. Radar cross section vs 360° of rotation of 1.5 cm Aℓ chaff on 4" foam sphere.

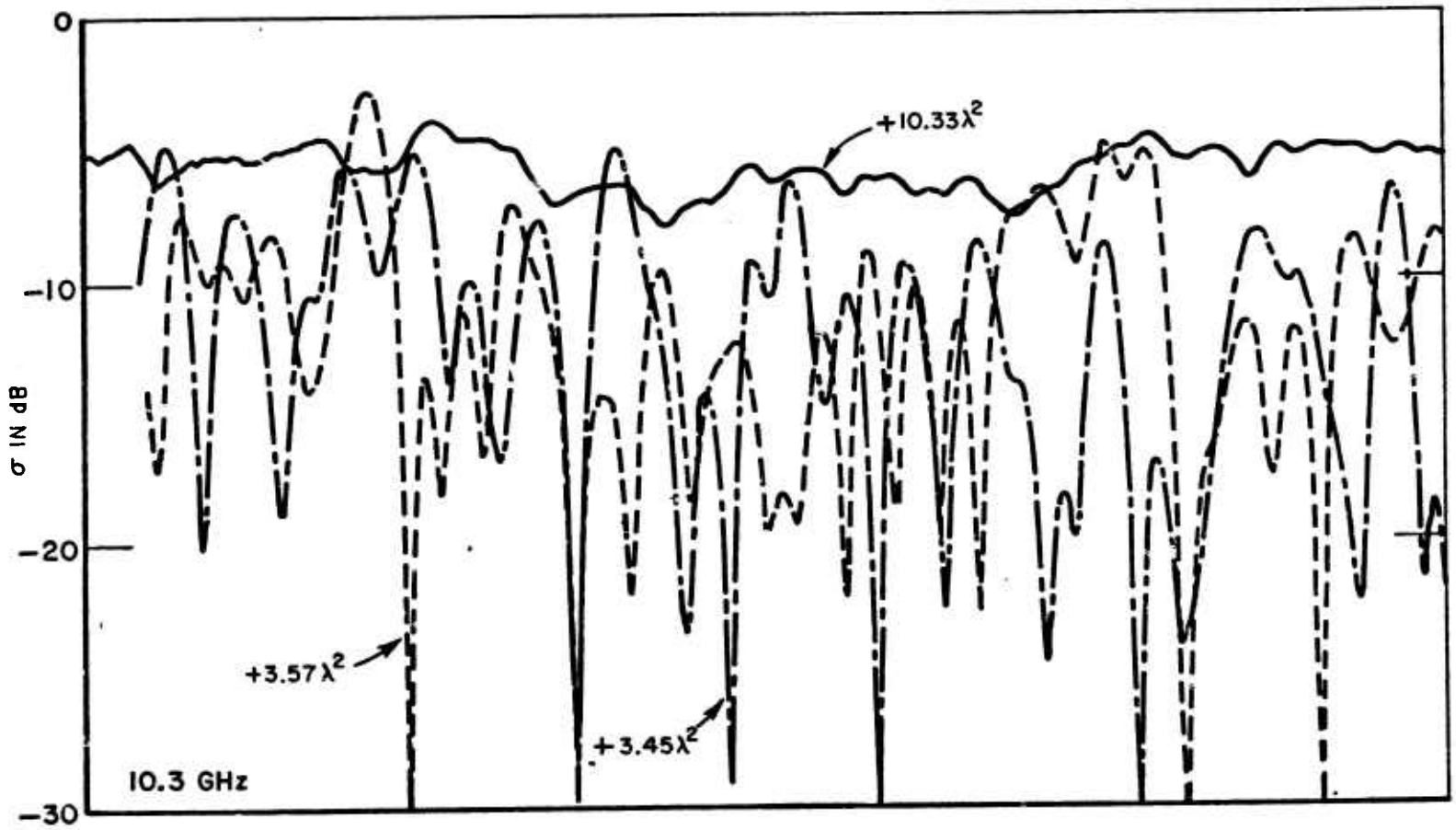


Figure 115, Radar cross section vs 360° of rotation of 1.5 cm Aℓ chaff on 4" foam sphere.

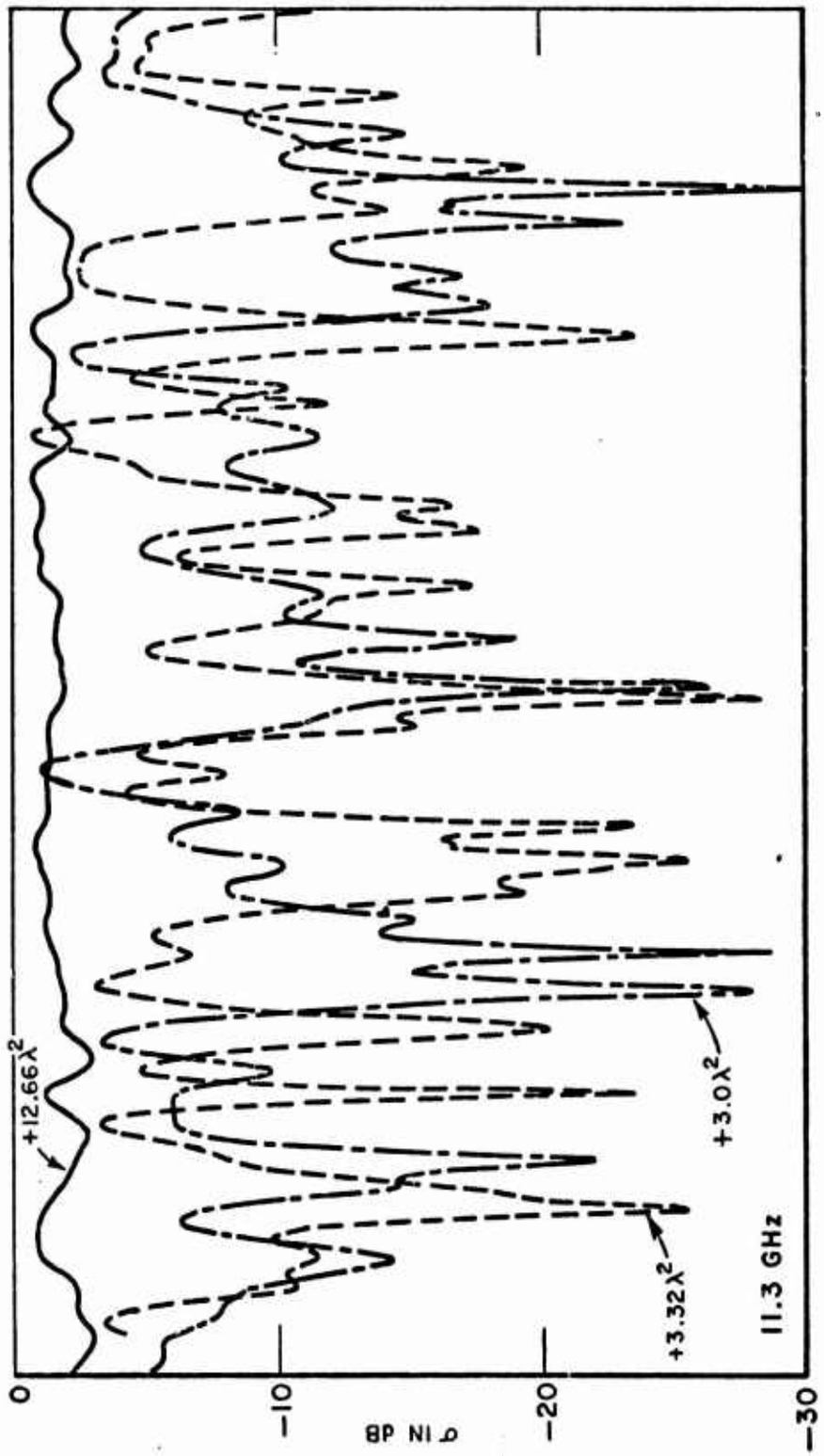


Figure 116. Radar cross section vs 360° of rotation of 1.5 cm Aℓ chaff on 4" foam sphere.

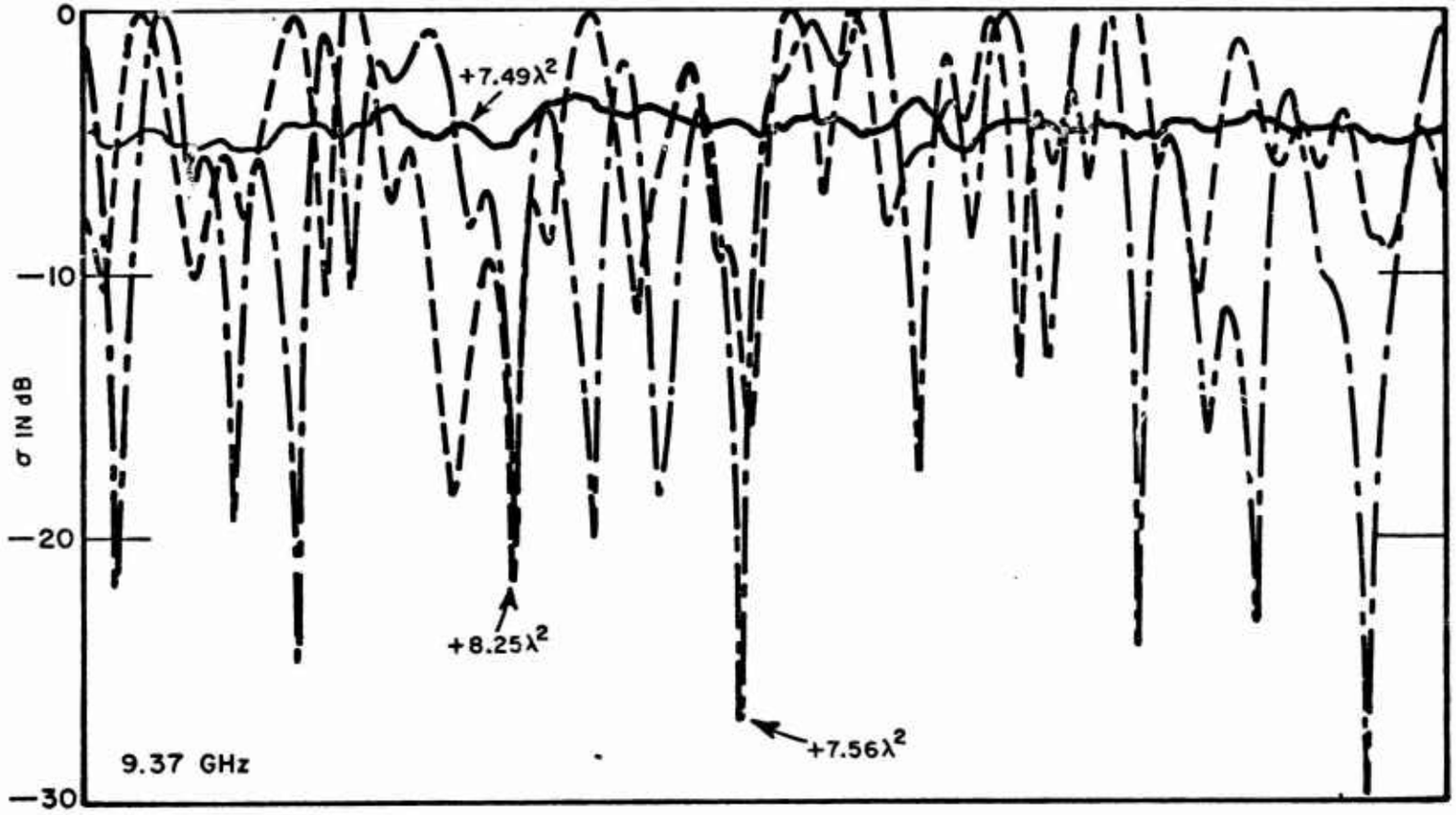


Figure 117. Radar cross section vs 360° of rotation of 1.75 cm glass chaff on 4" foam sphere.

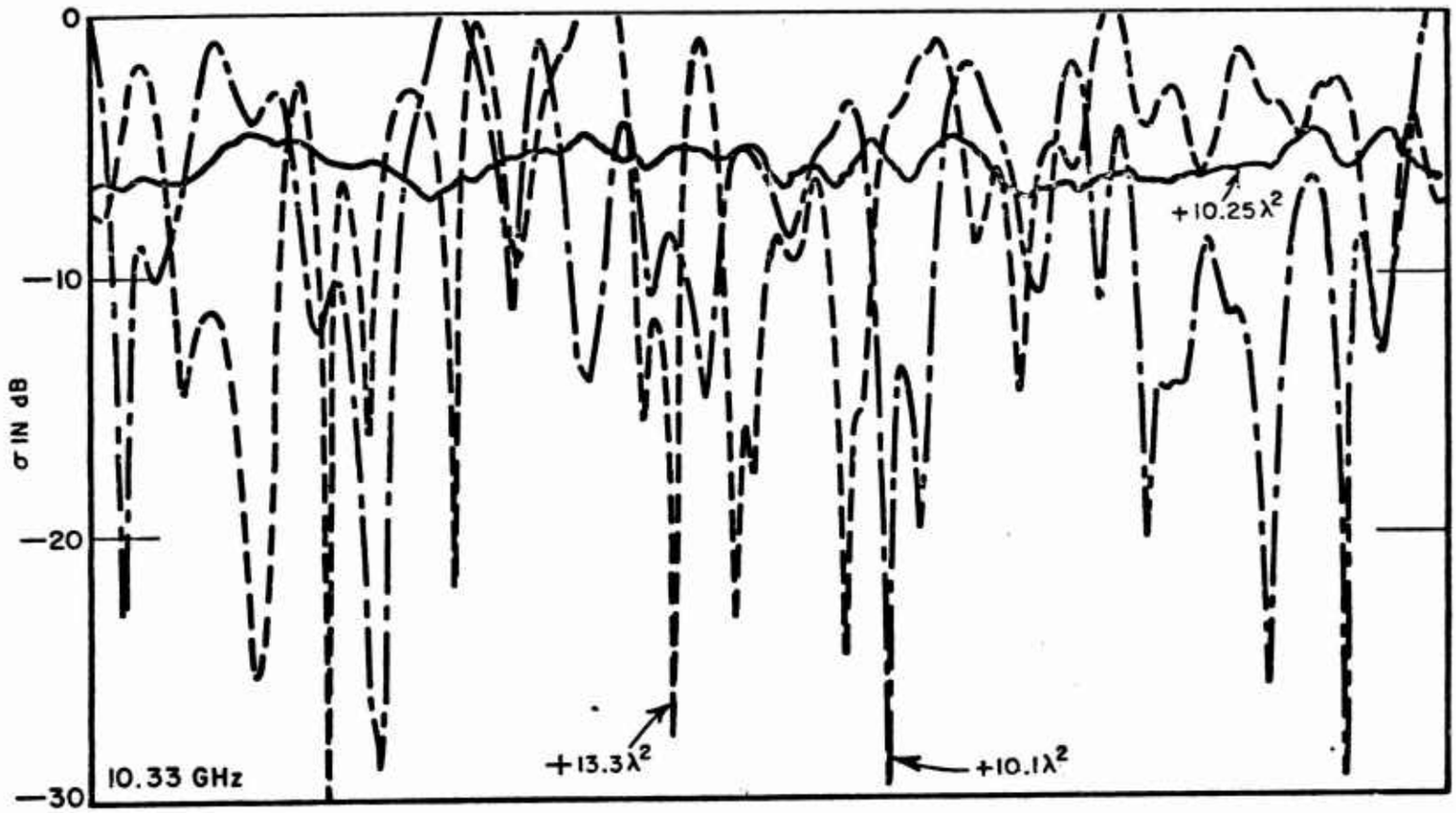


Figure 118. Radar cross section vs 360° of rotation of 1.75 cm glass chaff on 4" foam sphere.

III. DISCUSSION

Using the frozen model of a chaff cloud together with an experimentally verified computer routine we conclude that for clouds of up to 200 coupled resonant dipoles, the backscattering cross section is, on the average, reduced from that predicted by uncoupled theory when the average dipole spacing falls below 2 wavelengths. It is important to remind the reader that these conclusions are based upon data calculated for inhomogeneous clouds, i.e., spherical clouds whose dipole number density varied along the radial direction according to a Gaussian function. The array dipole spacings were maximum at the center of the cloud and equal to about 1/3 those of the average dipole spacings defined and used here. Dipole orientations were always assumed equally likely. Using data gathered for all such clouds containing up to 30 dipoles, the ratios of the average backscattering cross section calculated with coupling to that calculated without coupling are summarized in Fig. 119 for average dipole spacings down to 0.5 wavelengths. It appears that at an average spacing of about 0.4 wavelengths the average radar return can be expected to be reduced about 3 dB by coupling effects. We have not included spacings smaller than 0.5 because approximations concerning the coupling terms in the computer routine come into question beyond this point. More exact relations are available if needed and are discussed in this report (see below), but they require more time and expense to implement on the computer. Moreover, closer spacings increase the probability that more and more dipoles touch, a feature which can be incorporated into the computer but not without some encumberance.

Although less data were gathered for bistatic angles up to 135° (all of it experimental), both same-sense and crossed-sense linear polarizations showed trends similar to backscatter - closer average spacings effected reduced cross sections. In addition, several frequency runs by computer showed that even at the smallest average spacing of 0.5 wavelength the dipole resonance frequency remains essentially unchanged from the free space resonance frequency. For the spacings investigated, apparently the loading effects on a typical dipole in the cloud due to all of its neighbors essentially influence only the amplitude of the current and not its shape, thereby causing reduced scatter but maintaining about the same resonance frequency. Thus, we conclude that for average spacings down to 0.5λ ($\sim$ 8 dipoles/λ^3) each chaff element should be cut to its free-space resonant length to achieve best performance from the cloud.

In additional scattering measurements, some effect was devoted to an experimental evaluation of the extinction through a cloud of dipoles, averaged over time as the dipoles were set into motion. In this report curves are presented of the insertion loss incurred by the presence of the dipoles as functions of depth and for several densities.

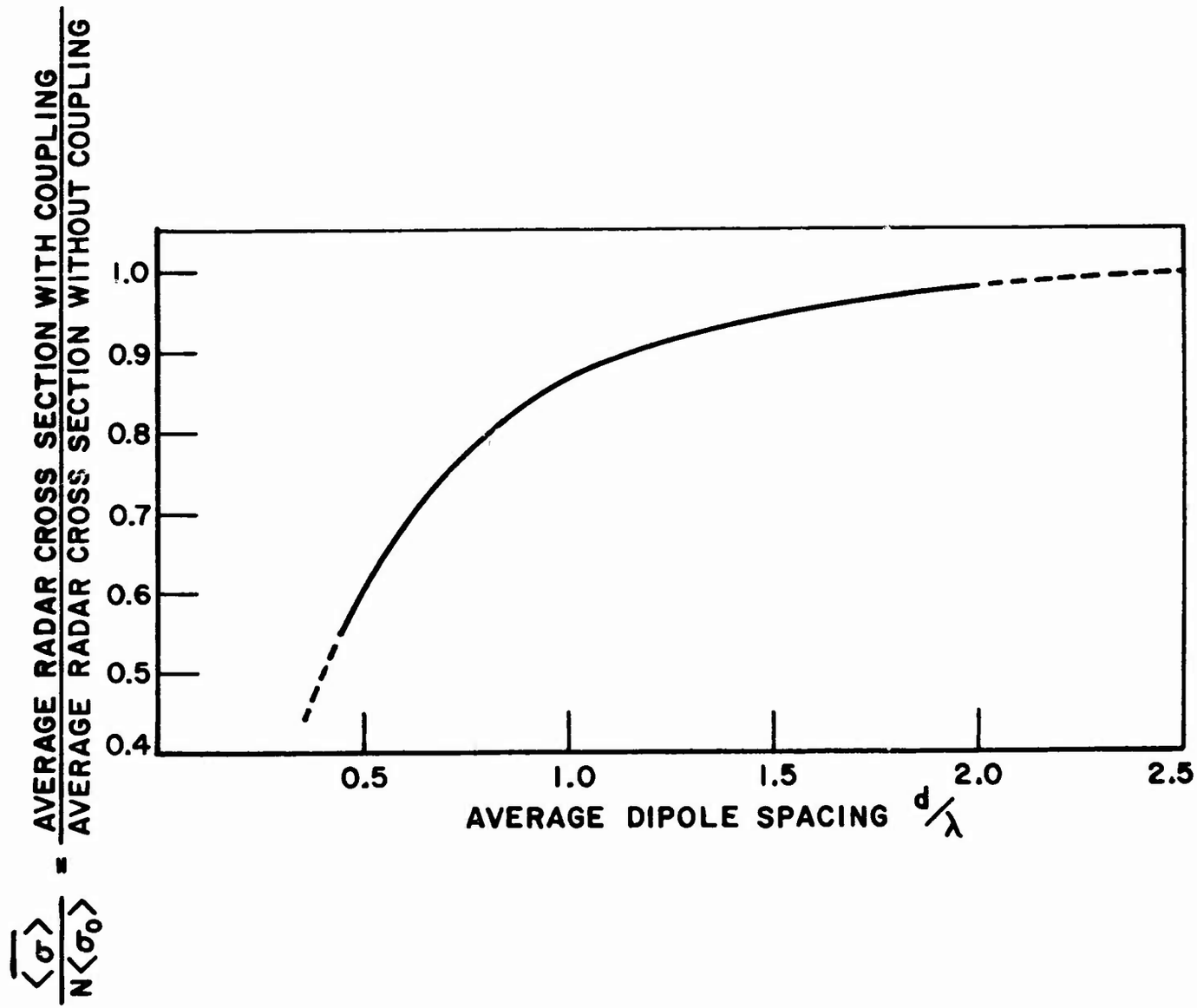


Figure 119. The decrease in average backscattering cross section due to coupling for a range of average dipole spacings (Gaussian density distribution assumed).

A table summarizing the extinction rates is given on page 165. These rates indicate a rather rapid extinction of energy as the wave proceeds into the cloud, even for rather tenuous clouds. Thus, if one is to design efficient chaff clouds, i.e., place the chaff dipoles where they will create the most effective echo, it is important to account for the effect of extinction and to predict it. Other experiments were performed to obtain backscattering patterns for a range of frequencies of foam spheres covered with aluminum and glass-chaff. The patterns show that at the resonance frequency of the individual dipoles, the sphere displays an average cross section essentially the same as that of conducting sphere of the same size, even when the dipoles are rather sparsely distributed over the sphere surface.

A study was made of clouds with dipoles spaced on the average less than 0.5λ - down to 0.125λ - to investigate the accuracy of the two-segment model employed in almost all the cases discussed here. It was found that the 2-segment model appeared satisfactory down to average spacings of 0.25λ if an exact (rather than a 12-point Gaussian) integration was employed to find the mutual impedances. A penalty of a 60% increase in calculation time was incurred, however. For average spacings as small as 0.125λ, even a four segment model with exact integration did not yield sufficiently stable results. Thus, the programs presented in this report are not considered reliable for inhomogeneous clouds with average spacings less than 0.25λ. Of course, the programs can easily be modified to increase segmentation but the resulting consumption of rapid-access memory becomes intolerable.

A small amount of data were calculated for clouds containing two chaff lengths and the results show the same effect of coupling as was observed with single length chaff. It appears though that the elements which have a lower tumble average cross section (because they are anti-resonant) are less influenced by mutual coupling. Not enough data were accumulated to give an empirical mixture rule to estimate the average cross section of any combination of any two element lengths.

In order to extend our capability to calculate radar cross sections of clouds with more than 200 dipoles without exceeding the fast access memory capabilities of even the largest computers two investigations were initiated. One of the use of the sparse matrix technique. This method takes into account the physical fact that dipoles which are electrically far apart in the cloud are only weakly coupled; this in turn implies that many elements of the impedance matrix relating the fields and currents are almost null. If such elements are arbitrarily set to zero and their number exceeds about 80% of the total number of matrix elements, sparse matrix algorithms may be applied which effect large savings in computer memory - so much so that matrices of much larger order than normally possible can be inverted. This method has been applied to the chaff scattering problem with some success, but it was found to be more time consuming than expected, particularly in the matrix reordering portion of the algorithm. Also, the arbitrary

sparsing of the impedance matrix causes approximations which make a study of the extinction through the cloud impossible using this technique. A second method which does not suffer from this latter disadvantage is the iterative scatter technique which assumes initial currents on the dipoles as if they were uncoupled and updates all these currents in successive steps corresponding to what might be thought of as successive orders of inter-dipole scatter. We have found the successive overrelaxation (SOR) method together with the Gauss-Seidel algorithm to be the most successful iterative method we have used on clouds of resonant wires. (It was found to be less successful on solid obstacles.) Details will be found in a separate technical report [38]; in this report we show scattering results for clouds containing 1000 dipoles calculated using SOR. Some check cases are also presented to validate the method. Although, like the sparse matrix technique, the iterative method is time consuming, it does permit calculation of scattering data for much larger clouds than can be conveniently handled any other way, and should yield accurate extinction data.

One other topic which was investigated briefly during this contract period was the analysis of the aircraft-tracker radar interaction in the presence of chaff. Detailed results will be found in a separate technical report [62].

Computer programs used to calculate the data generated for this contract are given in appendices.

IV. RECOMMENDATIONS FOR FUTURE EFFORT

(1) We have observed that as closer and closer spacings between dipoles are assumed, the current distribution on each dipole is not only changed in amplitude, but also in shape along the length of the dipole. To represent this distorted shape requires more than a two-segment model of the dipole if a piecewise sinusoidal basis is used. We suggest the use of two basis functions, each defined over the entire length of the dipole, one being even, the other odd with respect to the dipole center. The even function appears as a cosine function blunted at the ends while the odd function appears as a sine function whose peaks are shifted toward the dipole extremeties. We feel that such basis functions should be sufficient to account for the current distortions due to the influence of nearby neighbors without

increasing the order of the matrix equations. How much more dense this technique will allow the clouds to become beyond the 8 dipoles/λ^3 number is not known.

(2) We know that for extremely dense clouds where many dipoles are touching, the method of moments is not a viable technique. The cloud appears in some sense as a solid body of conducting material, whose surface is almost fur-like and changing with time. We suggest that such a surface be modelled as a random surface with only incoherent scatter and for chosen cloud shapes, the calculated echo patterns be compared with experimentally derived patterns.

(3) Beyond the problem of dense clouds is the fact that most chaff clouds contain dipoles of various lengths to meet threats over a range of frequencies. The low frequencies present no new problems, but at the higher frequencies, those dipoles resonant at low frequencies become electrically long and require many segments (or modes) to adequately describe the currents induced on them. This enlarges the matrix to sizes which cannot be handled by computers. Thus, we suggest the use of basis functions which are travelling waves rather than standing waves and thereby reduce the number needed. The longer wires would of necessity be assumed straight, uncoupled to each other and to the short elements.

(4) The computer simulation of an aircraft-missile intercept problem in the presence of chaff should be continued. Better models should be developed for the echo return from a typical aircraft as a function of its aspect with respect to the incident wave. Realistic radar models, including effects of doppler and angle tracking should be incorporated. And, probably most difficult, more accurate chaff cloud returns should be simulated.

APPENDIX A
STATISTICAL ANALYSIS EMPLOYED IN THIS REPORT

A. Definitions

We assume a frozen model for a chaff cloud and shall denote the backscattering cross section of the mth cloud in the ensemble illuminated from an angle θ, by $\sigma_m(\theta)$ often leaving the θ-dependence implicit for convenience. Averages with respect to angle θ (here called spatial averages) will be denoted by Poisson brackets, $\langle\ \rangle$, while ensemble averages over a set of clouds will be denoted by the overbar, $-$.

It can be shown[13,14] that the backscattering cross section $\sigma_m(\theta)$ of the mth cloud in the ensemble of clouds forming the frozen model, under the assumption of no coupling among dipoles, follows an exponential probability density function,* sketched in Fig. I-1a,

$$(I\text{-}1) \qquad p_m(\sigma_m) = \begin{cases} \dfrac{1}{\langle\sigma_m\rangle}\, e^{-\frac{\sigma_m}{\langle\sigma_m\rangle}}, & \sigma_m \geq 0 \\ 0, & \sigma_m < 0 \end{cases}$$

where $\langle\sigma_m\rangle$ is the spatial average of $\sigma_m(\theta)$ over all θ given by

$$(I\text{-}2) \qquad \langle\sigma_m\rangle = \int \sigma_m(\theta)\, d\theta\ .$$

Evidence that indeed Eq. (I-1) is valid is presented in Ref. 11 for clouds of up to 30 dipoles spaced on the average by two wavelengths (negligible coupling case). There also exist some actual radar chaff measurements which indicate an exponential distribution of backscattering cross section.[15]

The standard deviation[16] of $\sigma_m(\theta)$ is by definition

$$(I\text{-}3) \qquad s_m = \sqrt{\langle(\sigma_m - \langle\sigma_m\rangle)^2\rangle} = \left[\int (\sigma_m(\theta) - \langle\sigma_m\rangle)^2\, d\theta\right]^{1/2} = \langle\sigma_m\rangle$$

and the variance of $\sigma_m(\theta)$ is equal to s_m^2.

The cumulative probability function associated with σ_m is of interest and sketched in Fig. I-1b, is

*It can be shown that the exponential probability density function is strictly applicable only if the cloud density is uniform, which in the present case is not true. However, for the clouds considered here, it is a very good approximation.

$$(I\text{-}4) \qquad P_m(\sigma_m) = \int_{-\infty}^{\sigma_m} p_m(x)dx = \begin{cases} 1 - e^{-\frac{\sigma_m}{\langle\sigma_m\rangle}}, & \sigma_m \geq 0 \\ 0, & \sigma_m < 0 \end{cases}$$

This function, evaluated at, say $\sigma_m = \sigma'_m$, gives the fraction of all possible values of σ_m which lie in the range, $0 \leq \sigma_m \leq \sigma'_m$. Special values of σ'_m are given names which we will refer to later. For example if $P_m(\sigma'_m) = 1/5$, $\sigma'_m = \sigma_{m\ 1/5}$ is called the 20% or first quintile; if $P_m(\sigma'_m) = 1/2$, $\sigma'_m = \sigma_{m\ 1/2}$ is called the 50% or median (as distinct from the mean or average value we have symbolized by $\langle\sigma_m\rangle$); if $P_m(\sigma'_m) = 4/5$, $\sigma'_m = \sigma_{m\ 4/5}$ is called the 80% or fourth quintile.

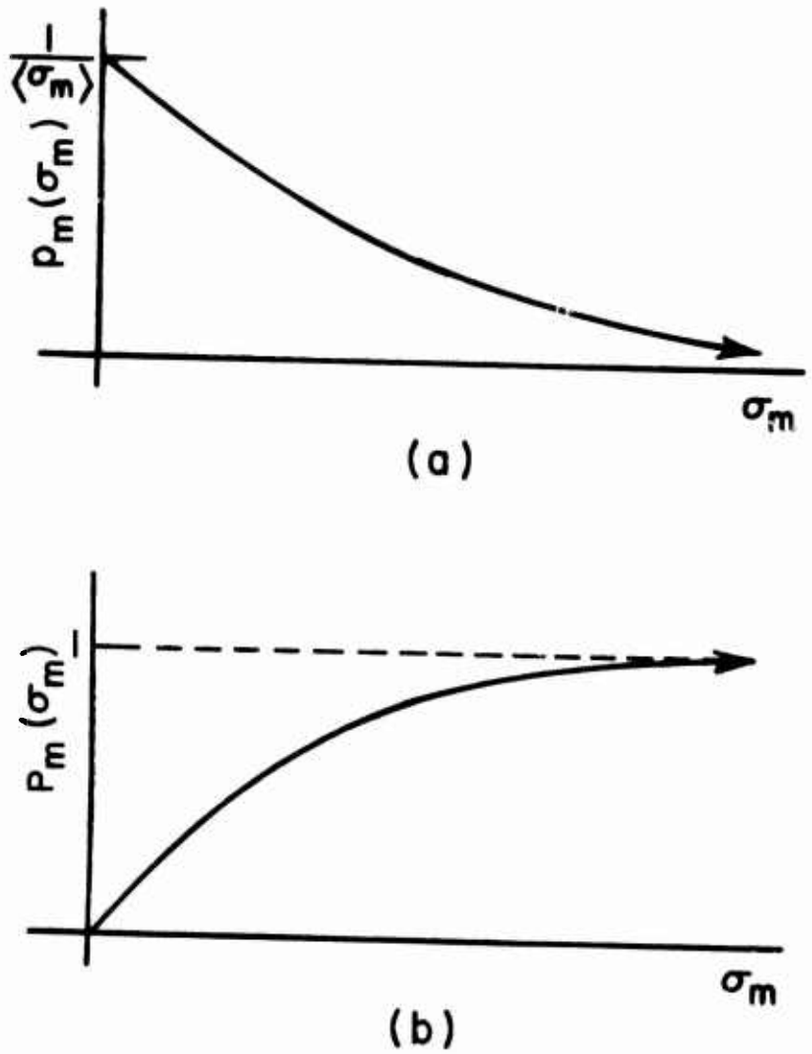


Figure I-1. Sketches of the exponential probability density function and corresponding cumulative probability function for the backscattering cross section of a chaff cloud.

In the frozen cloud model, many different sample clouds are generated (each one, of course, with the same number of dipoles spaced on the average the same). If the number of such clouds is $M \gg 1$, we obtain M sample functions $\sigma_m(\theta)$, each one representing the backscattering cross section as a function of look angle θ of the mth cloud in the ensemble, where $1 \leq m \leq M$. Since the spatial average cross section $\langle\sigma_m\rangle$ will in general differ for each m we obtain a distribution of M values of $\langle\sigma_m\rangle$. If M is large enough, we can obtain a relative frequency histogram[17] of $\langle\sigma_m\rangle$ which may be fitted to a Gaussian probability density function $q_{mean}(\langle\sigma_m\rangle)$ since the means of the exponential process are Gaussianly distributed. If, for convenience, the symbol $\langle\sigma\rangle$ is used in place of $\langle\sigma_m\rangle$, we depict the relative frequency histogram of $\langle\sigma\rangle$ by a bar graph and the Gaussian probability density function of the sampling distribution by a smooth curve as sketched in Fig. I-2. The Gaussian density function[18] is

$$(I\text{-}5) \qquad q_{mean}(\langle\sigma\rangle) = \frac{1}{\sqrt{2\pi}\, s_{mean}} e^{-\frac{(\langle\sigma\rangle - \langle\bar{\sigma}\rangle)^2}{2 s_{mean}^2}}$$

where $\langle\bar{\sigma}\rangle$ is the ensemble mean of $\langle\sigma\rangle$ and s_{mean} is the standard deviation of $\langle\sigma\rangle$. On an ensemble basis they may be expressed as

$$(I\text{-}6) \qquad \langle\bar{\sigma}\rangle \equiv \int_{-\infty}^{\infty} \langle\sigma\rangle\, q_{mean}(\langle\sigma\rangle) d\langle\sigma\rangle \quad ,$$

and

$$(I\text{-}7) \qquad s_{mean} = \overline{(\langle\sigma\rangle - \langle\bar{\sigma}\rangle)^2} \equiv \int_{-\infty}^{\infty} (\langle\sigma\rangle - \langle\bar{\sigma}\rangle)^2\, q_{mean}(\langle\sigma\rangle) d\langle\sigma\rangle .$$

The mean value $\langle\bar{\sigma}\rangle$ is the arithmetic average of all the values of $\langle\sigma\rangle$ (i.e., $\langle\sigma_m\rangle$) and the standard deviation (since $q(\langle\sigma\rangle)$ is Gaussian) determines the range of values, $(\langle\bar{\sigma}\rangle - s_{mean}) < (\langle\sigma\rangle) < (\langle\bar{\sigma}\rangle + s_{mean})$, between which lie 68.27% of all the possible values of $\langle\sigma\rangle$ (i.e., $\langle\sigma_m\rangle$). The two curves in Fig. I-2 both are normalized to unit area and the one may be fitted to the other by, for example, a chi-square test.[19] Of course, since $\langle\sigma\rangle$ cannot be negative, the fit of a Guassian distribution(which admits negative values) can be accomplished only in the region of $\langle\sigma\rangle$ values about the $\langle\bar{\sigma}\rangle$ value.

In a manner very similar to that for treating $\langle\sigma\rangle$, we may fit Gaussian curves to histograms of $\sigma_{m\ 1/5}$, $\sigma_{m\ 1/2}$, $\sigma_{m\ 4/5}$ (using the simplified symbols, $\sigma_{1/5}$, $\sigma_{1/2}$, $\sigma_{4/5}$),

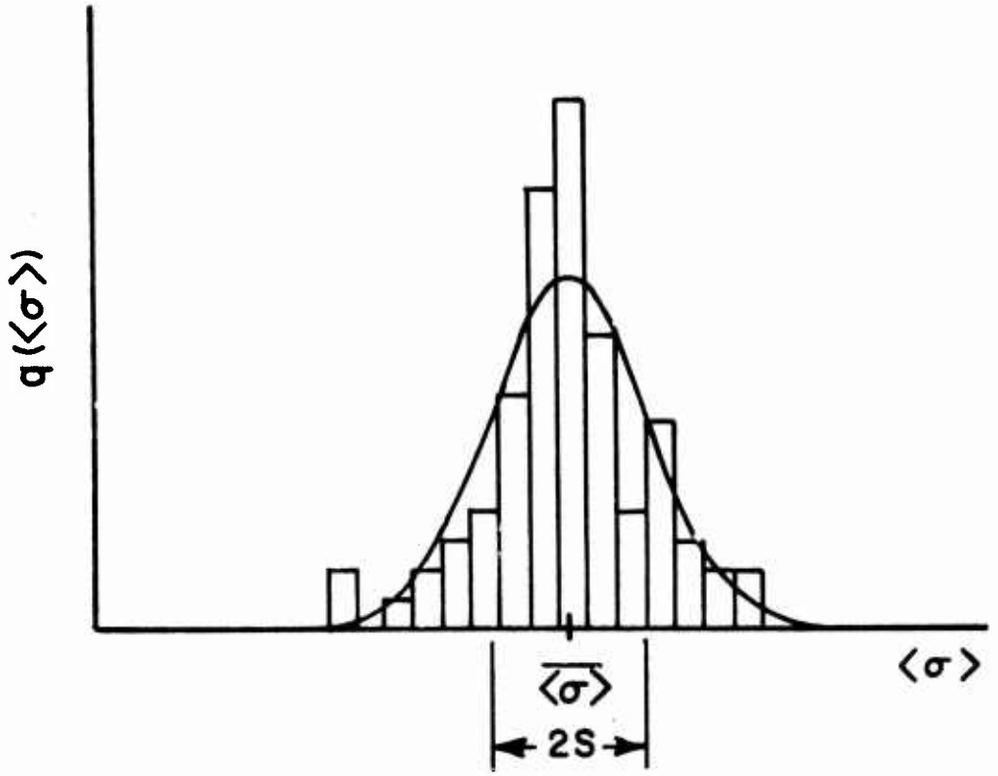


Figure I-2. Sketches of a histogram and associated Gaussian probability distribution of spatial average cross sections of frozen chaff clouds.

$$(I\text{-}8) \qquad q_{1/5}(\sigma_{1/5}) = \frac{1}{\sqrt{2\pi}\, s_{1/5}} e^{-\frac{(\sigma_{1/5}-\bar{\sigma}_{1/5})^2}{2 s_{1/5}^2}},$$

$$(I\text{-}9) \qquad q_{1/2}(\sigma_{1/2}) = \frac{1}{\sqrt{2\pi}\, s_{1/2}} e^{-\frac{(\sigma_{1/2}-\bar{\sigma}_{1/2})^2}{2 s_{1/2}^2}},$$

$$(I\text{-}10) \qquad q_{4/5}(\sigma_{4/5}) = \frac{1}{\sqrt{2\pi}\, s_{4/5}} e^{-\frac{(\sigma_{4/5}-\bar{\sigma}_{4/5})^2}{2 s_{4/5}^2}},$$

where the mean and standard deviation of $\sigma_{1/5}$, for example, taken on an ensemble basis, are

$$\text{(I-11)} \qquad \overline{\sigma_{1/5}} \equiv \int_{-\infty}^{\infty} \sigma_{1/5}\, q_{1/5}(\sigma_{1/5})\, d\sigma_{1/5},$$

$$\text{(I-12)} \qquad s_{1/5}^2 = \overline{(\sigma_{1/5} - \overline{\sigma}_{1/5})^2} \equiv \int_{-\infty}^{\infty} (\sigma_{1/5} - \overline{\sigma}_{1/5})^2\, q_{1/5}(\sigma_{1/5})\, d\sigma_{1/5}.$$

Similar expressions may be used for $\overline{\sigma}_{1/2}$, $s_{1/2}$, $\overline{\sigma}_{4/5}$, and $s_{4/5}$.

Once a value for $\langle\overline{\sigma}\rangle$ has been obtained, we hypothesize that this value may be used in Eqs. (I-1) and (I-2) to obtain the probability density function and cumulative probability of the backscattering cross section σ of the frozen model, even in the presence of coupling,

$$\text{(I-13)} \qquad p(\sigma) = \begin{cases} \dfrac{1}{\langle\sigma\rangle}\, e^{-\frac{\sigma}{\langle\sigma\rangle}}, & \sigma \geq 0 \\ 0, & \sigma < 0 \end{cases}$$

$$\text{(I-14)} \qquad P(\sigma) = \begin{cases} 1 - e^{-\frac{\sigma}{\langle\sigma\rangle}}, & \sigma \geq 0 \\ 0, & \sigma < 0 \end{cases}$$

If these functions indeed do characterize the frozen model then it should be true that

$$\text{(I-15a)} \qquad P(\overline{\sigma_{1/5}}) = 0.2,$$

$$\text{(I-15b)} \qquad P(\overline{\sigma_{1/2}}) = 0.5\ ,$$

$$\text{(I-15c)} \qquad P(\overline{\sigma_{4/5}}) = 0.8\ .$$

One can test the data to see if equalities (I-15) are approximately true, in which case we have some assurance that Eq. (I-13) is valid for coupled clouds.

We considered the following reasoning to obtain one other indicator that Eq. (I-13) is valid. The standard deviation of σ is found from Eq. (I-13) to be

$$(I\text{-}16) \qquad s = \left[\int_{-\infty}^{\infty} (\sigma - \langle\bar{\sigma}\rangle)^2 \, p(\sigma) d\sigma\right]^{1/2} = \langle\bar{\sigma}\rangle$$

This standard deviation of σ should be related to the standard deviation s_{mean} of the sampling distribution of means by the relationship[20]

$$(I\text{-}17) \qquad s = s_{mean}\sqrt{N_s}\sqrt{\frac{N_p - 1}{N_p - N_s}}$$

where the numbers N_p and N_s are defined as the population and sample size, respectively, and may be obtained in our case as follows.

If, in the frozen model, there are M clouds, each viewed at 512 angles with two polarizations, then the number of pieces of backscattering data, called the population, is $N_p = 1024M$. By sampling each cloud at 512 look angles with two polarizations and considering these data as independent, we form 2M samples of size $N_S = 512$ data points each. However, the 512 look angles are probably not independent, i.e., we have oversampled $\sigma_m(\theta)$. To obtain an estimate of the number of independent samples, we observe the highest frequency in the spectrum $W(N, d/\lambda)$ (discussed below) and consider the sample size to be $N_S = 2W\,(N, d/\lambda)$. Using these values for N_p and N_S and s_{mean} as obtained from $q_{mean}(\langle\sigma\rangle)$, Eq. (I-17) gives a value $s \approx s_{mean}\, 2w$ which may be compared with the value $s = \langle\bar{\sigma}\rangle$ of Eq. (I-16). We applied the above numbers to Eq. (I-17) and did not arrive at relations between s and s_{mean} which were consistent with Eq. (I-16). We can only conclude that the above method for estimating sample size is invalid probably due to the inhomogeneity of the population data from cloud to cloud.

Another parameter which may be of use in characterizing the frozen cloud model is the spatial frequency spectrum of the backscattering cross section. If $\sigma_m(\theta)$ is the backscattering cross section of the mth cloud in the frozen model, then we define $F_m(\omega)$ as the Fourier transform of one period of $\sigma_m(\theta)$, $0 \leq \theta \leq 360°$, where ω is the spatial spectral variable (in say, Hz/degrees of θ).

A typical $F_m(\omega)$ might appear as sketched in Fig. I-3, where W_m is the highest frequency in the spectrum. One finds that W_m varies with N, the number of dipoles in a cloud, and d/λ, the average spacing between dipoles, so we signify this dependence by writing $W_m(N, d/\lambda)$. In addition, one finds that for a fixed $(N, d/\lambda)$ pair, different members of the ensemble of clouds, i.e.,

different values for $1 \leq m \leq M$, yield slightly different $W_m(N, d/\lambda)$. If we denote by $W(N, d/\lambda)$ the average of these values over 22 different clouds, we can derive Table I. In this table, $W(N, d/\lambda)$ appear in the upper triangles, while in the lower triangles appear the values of $4\delta(N,d/\lambda)/\lambda$; i.e., diameters (in wavelengths) of spheres encompassing 95% of the dipoles in a typical cloud associated with the pair $(N, d/\lambda)$, If the parallel dipole scatterers are assumed to exist at the extremeties of these diameters, then frequency $W'(N, d/\lambda)$ can be calculated on the basis of the beamwidth of the broadside lobe according to

$$W'(N, \frac{d}{\lambda}) = \frac{180}{\sin^{-1} \frac{1}{2} \cdot \frac{\lambda}{4\delta}} .$$

Both $W(N, d/\lambda)$ and $W'(N, d/\lambda)$ are plotted in Fig. I-4. The 2-dipole model appears to predict values for $W(N, d/\lambda)$ which are too low and do not decrease rapidly enough with decrease in N, but considering the simplicity of the model, the comparisons are not bad.

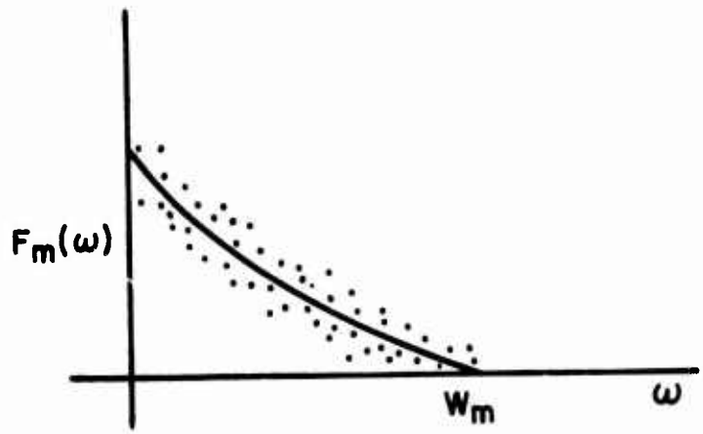


Figure I-3. Sketch of a typical spatial frequency spectrum of a frozen chaff cloud.

B. Statistical Analysis of Backscattering Data

A large amount of calculated backscattering data, based on the frozen model, have been obtained for several cases. To examine the properties of these data, statistical methods must be employed. The usual procedures for dealing with this kind of statistical problem are as follows:

1. Data are first classified into small intervals and by counting the relative frequencies of occurance in each interval, a histogram can be drawn. By inspecting the histogram, it is then possible to select a suitable mathematical model, namely, the frequency distribution function. The unknown parameters are then estimated by the Maximum Likelihood Method.

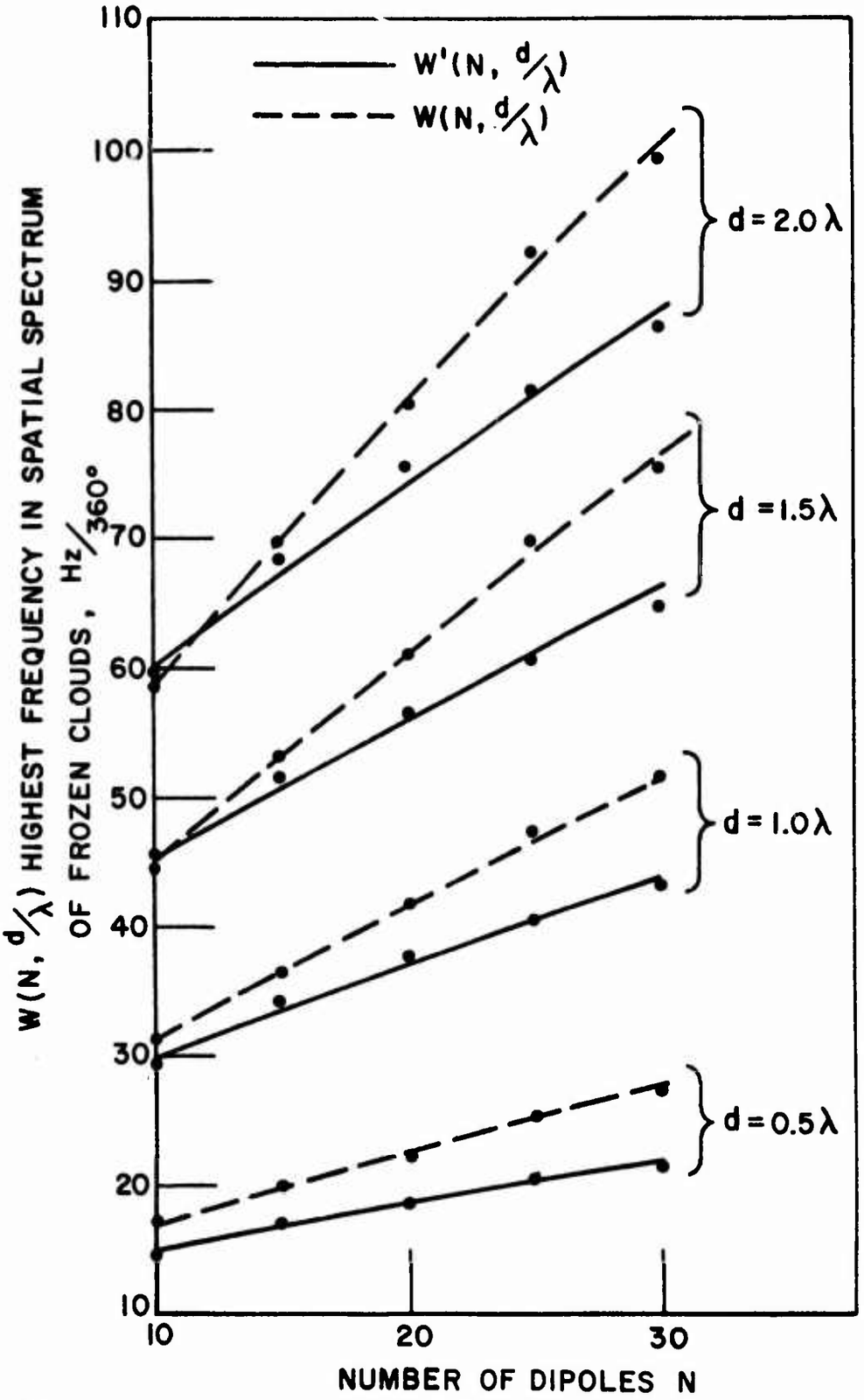


Figure I-4. The highest spatial frequency W_n in the spectrum of a frozen chaff cloud, as function of N and d/λ. The approximant W'_n is derived as discussed in the text.

2. The reasonableness of the mathematical model can be checked by the Chi-Square Test.[19] Basically, the quantity

$$\sum_{i=1}^{k} \frac{(o_i - e_i)^2}{e_i},$$

where o_i, e_i are the observed and expected frequencies respectively, is compared to a χ_0^2 variable with ν degree of freedom. This serves as the criterion for the goodness of fit.

3. Once the assumption of the model is justified, our interest is in the confidence limits of the parameters. This gives some idea of the expected variation of the parameters of interest. Given the size of the confidence interval, it is then possible to determine the number of data points sufficient to describe the statistical behavior of the cloud.

To illustrate the technique, we analyze the data obtained for the case N=30 dipoles, d=2.0λ in detail here.

Table II shows the spatial average backscattering cross-section $\langle\sigma_m(\theta)\rangle$ for m=1, 2, ..., 80 clouds. These data are then classified into 24 classes, from class mark 2.0 to 6.8 with interval size 0.2. The resultant histogram is shown in Fig. I-5. The symmetry and skewness of the histogram suggests that the data are likely to be Gaussian-distributed. We therefore assume that $\langle\sigma_m(\theta)\rangle$ has a Gaussian distribution with mean μ and variance σ^2. Since these are not known a priori, they must be estimated from the data. It can be shown[19] that the maximum likelihood estimators of μ and σ^2 are given by the sample mean $\overline{\langle\sigma_m(\theta)\rangle}$ and sample variance s^2_{mean}. Thus,

$$\text{(I-18)} \quad q_{mean}(\langle\sigma_m(\theta)\rangle) = \frac{A}{\sqrt{2\pi}\, s_{mean}} e^{-\frac{(\langle\sigma_m(\theta)\rangle - \overline{\langle\sigma_m(\theta)\rangle})^2}{2 s^2_{mean}}},$$

where the sample mean $\overline{\langle\sigma_m(\theta)\rangle} = \sum_{m=1}^{80} \frac{\langle\sigma_m(\theta)\rangle}{80} = 4.2798,$

$$\text{sample variance } s^2_{mean} = \sum_{m=1}^{80} \frac{(\langle\sigma_m(\theta)\rangle - \overline{\langle\sigma_m(\theta)\rangle})^2}{79} = 0.288$$

and total area of the histogram A = 16.0.

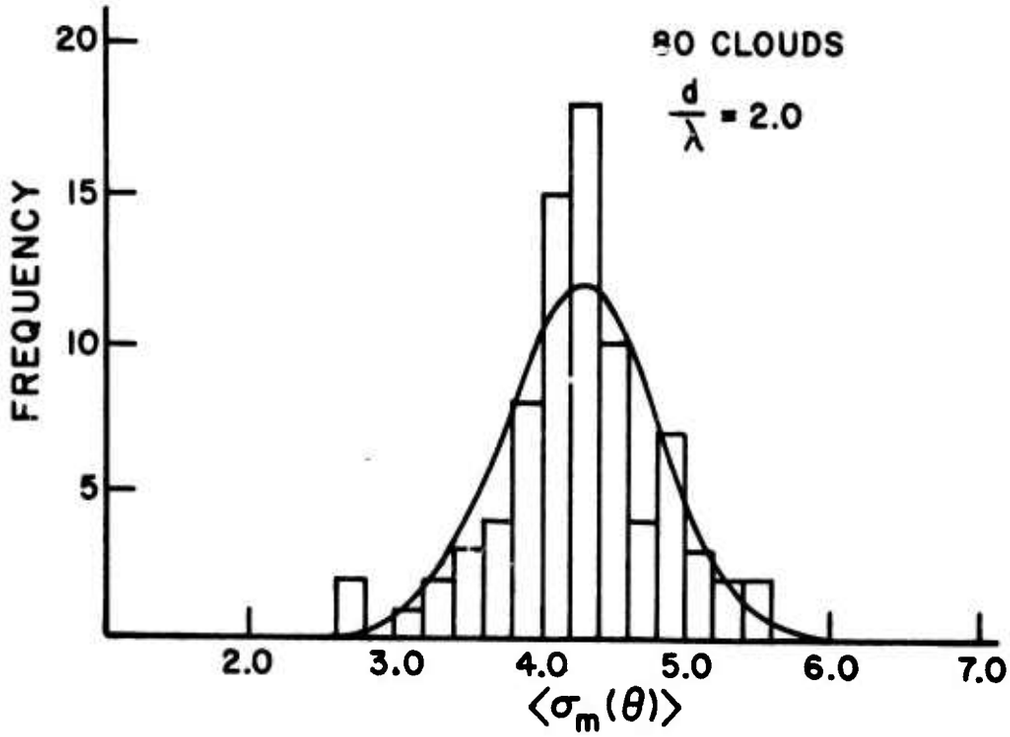


Figure I-5. The histogram and associated Gaussian probability distribution of the spatial averages of 80 frozen chaff clouds containing 30 dipoles each.

Equation (I-18) is also plotted in Fig. I-5. Here we have fit the histogram of the backscattering data to a Gaussian curve. One measure of fit is the Chi-square test which evaluates the deviation χ^2

$$\chi^2 = \sum_{i=1}^{k} \frac{(o_i - e_i)^2}{e_i}$$

between the observed frequencies o_i and the expected frequencies e_i in $i=1,2,\cdots k$ intervals. The expected frequency e_i is obtained by integrating the area under the curve and the observed frequency o_i is obtained by counting the number of occurances of the backscattering data in the ith interval. The results are shown in Table III.

In computing the deviation χ^2, it is necessary that $e_i \geq 5$ and $k \geq 5$.[19] Several intervals can be combined until the above condition is satisfied. This is indicated in the left column of Table III.

Calculations show that

$$\chi^2 = \sum_{i=1}^{9} \frac{(o_i - e_i)^2}{e_i} = \frac{(7-5.87)^2}{5.87} + \frac{(4-5.35)^2}{5.35} + \frac{(8-7.96)^2}{7.96}$$

$$+ \frac{(15-10.33)^2}{10.33} + \frac{(18-11.68)^2}{11.68} + \frac{(10-11.52)^2}{11.52}$$

$$+ \frac{(4-9.91)^2}{9.91} + \frac{(7-7.43)^2}{7.43} + \frac{(7-9.94)^2}{9.94}$$

$$= 10.7088$$

This value is compared with a χ_0^2 variable with $=K-1-\ell$ degrees of freedom, where ℓ is the number of parameters that the interval probability depends upon; since $q_{mean}(<\sigma_m(\theta)>)$ depends on two unknown parameters, we have $\ell=2$ in this case.

It is found that $\chi_0^2=12.592$ with $\nu=6$ for a 5% significant level, and since $\chi^2 < \chi_0^2$, we conclude that the model is satisfactory.

The 95% confidence interval for the mean is given by:

$$\left(\overline{<\sigma_m(\theta)>} - \frac{bS}{m-1}, \overline{<\sigma_m(\theta)>} + \frac{bS}{\sqrt{m-1}}\right) = (4.1613, 4.3983)$$

where b can be obtained from the Student-t[19] distribution table. For example, b=1.96 for $m > 30$. The confidence interval of the mean is then given by $L = 2bS/\sqrt{m-1}$, or solving for m, we obtain:

$$(I\text{-}19) \qquad m = 1 + \left(\frac{2bS}{L}\right)^2 .$$

For L=0.1 $<\sigma_m(\theta)>$ = 0.42798, Eq. (I-19) can be used to obtain the value of m by trial and error. Assume m=26, b=2.056, Eqs. (I-19) gives m = 26. We summarize the results as follows:

1. Evidence is shown that the spatial averages are Gaussian distributed.

2. The mean value will lie inside the interval (4.1613, 4.3983).

3. We predict that 95% of the data will fall in the interval (3.2050, 5.3546) in the long run.

4. Only 26 clouds are needed to determine the statistical behavior of the spatial variation of the Chaff cloud if the size of the confidence interval is allowed to be 10% of its mean value.

The backscattering data obtained for the cases N=30 dipoles, $d=0.5\lambda$ and N=10 dipoles, $d=0.5\lambda$, 2.0λ were treated in the same manner as above. 80 clouds for each of these cases were used in the analysis. It turns out that in all cases, the spatial averages are Gaussian distributed to a good approximation. The results of the analysis are shown in Table IV and in Figs. I-6 and I-7.

It was mentioned in Section A that the backscattering cross-section $\langle\sigma_m(\theta)\rangle$ under the assumption of no coupling among dipoles, follows the exponential probability density function of Eq. (I-13). If the exponential density function also holds for coupled elements, then Eq. (I-15a), (15b), (15c) should be approximately true even for small spacings d/λ. We now want to show that this is indeed the case.

Backscattering data were obtained for 4 cases, namely, N=30 dipoles, $d=0.5\lambda$, 2.0λ and N=10 dipoles, $d=0.5\lambda$, 2.0λ at 20%, 50%, 80% levels. Again, 80 clouds of each case were used in the statistical analysis. The assumption that the data were obtained from sampling a Gaussian population is good except for the case of 20% level. However, it is found that the variance in these cases are so small that even if more clouds are included in the analysis, the sample mean will not change significantly. We thus include them for comparison.

The 20%, 50% and 80% levels are obtained by substituting into Eq. (I-14) with the appropriate value of $\overline{\langle\sigma\rangle}$ used and in Table IV they are compared with the value obtained for forming histograms and approximating these with Gaussian distributions. The same results are shown in Fig. I-8, where the curves are calculated using Eq. (I-14) and dots are values calculated using histograms. In Fig. I-8 our additional curve for N=50 dipoles $d/\lambda = 2.0$ is given. The good comparison leads us to conclude that the backscattering cross section, even with severe coupling effects, appears to obey the exponential distribution when the associated value of mean cross section $\overline{\langle\sigma\rangle}$ is incorporated.

Again, 80 clouds of each case were used in the statistical analysis. The assumption that the data were obtained from sampling a Gaussian population is good except for the case of 20% level. However, it is found that the variance in these cases are so small that even if more clouds are included in the analysis, the sample mean will not change significantly. We thus include them for comparison.

The 20%, 50% and 80% levels are obtained by substituting into Eq. (I-14) with the appropriate value of $\langle\sigma\rangle$ used and in Table IV they are compared with the value obtained by forming histograms and approximating these with Gaussian distributions. The same results are shown in Fig. I-8, where the curves are calculated using Eq. (I-14) and dots are values calculated using histograms. The good comparison leads us to conclude that the backscattering cross section, even with severe coupling effects, appears to obey the exponential distribution when the associated value of mean cross section $\langle\sigma\rangle$ is incorporated.

TABLE I - HIGHEST SPECTRAL FREQUENCIES

N / d/λ	10	15	20	25	30
2.0	58.7 / 4.76	69.4 / 5.45	80.3 / 6.00	92.0 / 6.46	99.0 / 6.86
1.5	45.2 / 3.57	53.0 / 4.08	61.0 / 4.50	69.7 / 4.84	75.2 / 5.15
1.0	31.2 / 2.38	36.3 / 2.72	41.5 / 3.00	47.3 / 3.23	51.4 / 3.43
0.5	17.3 / 1.19	20.0 / 1.36	22.7 / 1.50	25.6 / 1.61	27.7 / 1.71

TABLE II - SPATIAL AVERAGES OF 80 CLOUDS

4.3842	4.4985	4.49505	5.50184
4.40	4.8349	4.17054	3.9871
4.5801	4.0279	4.64392	4.09925
4.63293	4.3948	4.18171	3.63187
4.67246	4.0599	4.34841	4.89497
3.52468	5.4608	3.31524	5.15151
3.84084	4.3303	3.90761	4.11126
5.07415	3.7248	2.62660	4.90393
3.77322	3.4994	4.45712	4.34125
3.23141	4.38457	4.12578	4.08328
2.6788	4.52723	4.38605	4.81494
4.03150	4.26477	4.02188	4.03012
4.2399	3.74232	4.22126	3.83469
5.2056	4.29156	4.18910	4.28673
4.2795	3.45858	4.30804	4.47537
4.1598	5.11655	3.9152	3.91778
4.2642	4.30620	4.93505	4.56063
4.5817	4.60128	3.84363	4.84562
4.1245	4.90631	5.36615	4.16605
4.4971	4.45655	3.95543	4.26969

TABLE III - THE DATA OF TABLE II CLASSIFIED INTO RELATIVE FREQUENCIES OF OCCURRENCE

k	Level	Freq Dist o_i	Theo Freq e_i
1	2.100	0	.00
	2.300	0	.01
	2.500	0	.03
	2.700	2	.09
	2.900	0	.28
	3.100	[illegible]	.72
	3.300	[illegible]	1.61
	3.500	3	3.14
2	3.700	4	5.35
3	3.900	8	7.96
4	4.100	15	10.33
5	4.300	18	11.68
6	4.500	10	11.22
7	4.700	4	9.91
8	4.900	7	7.43
	5.100	3	4.86
	5.300	2	2.77
9	5.500	2	1.38
	5.700	0	.60
	5.900	0	.23
	6.100	0	.07
	6.300	0	.02
	6.500	0	.00
	6.700	0	.00

TABLE IV - CUMULATIVE PROBABILITY VALUES

	N d/λ	30 2.0	30 0.5	10 2.0	10 0.5
	$\langle\overline{\sigma_m(\theta)}\rangle$	4.2798	2.6513	1.42440	1.0219
	$\langle\overline{\sigma_m(\theta)}\rangle_{20\%}$	0.9484	0.5749	0.3127	0.2220
20%	$P(\sigma)$	0.9550	0.5916	0.3177	0.2280
	$\langle\overline{\sigma_m(\theta)}\rangle_{50\%}$	3.0555	1.8215	0.9634	0.6932
50%	$P(\sigma)$	2.9665	1.8377	0.9870	0.7083
	$\langle\overline{\sigma_m(\theta)}\rangle_{80\%}$	7.1347	4.2469	2.2597	1.6384
80%	$P(\sigma)$	6.8881	4.2671	2.2918	1.6447

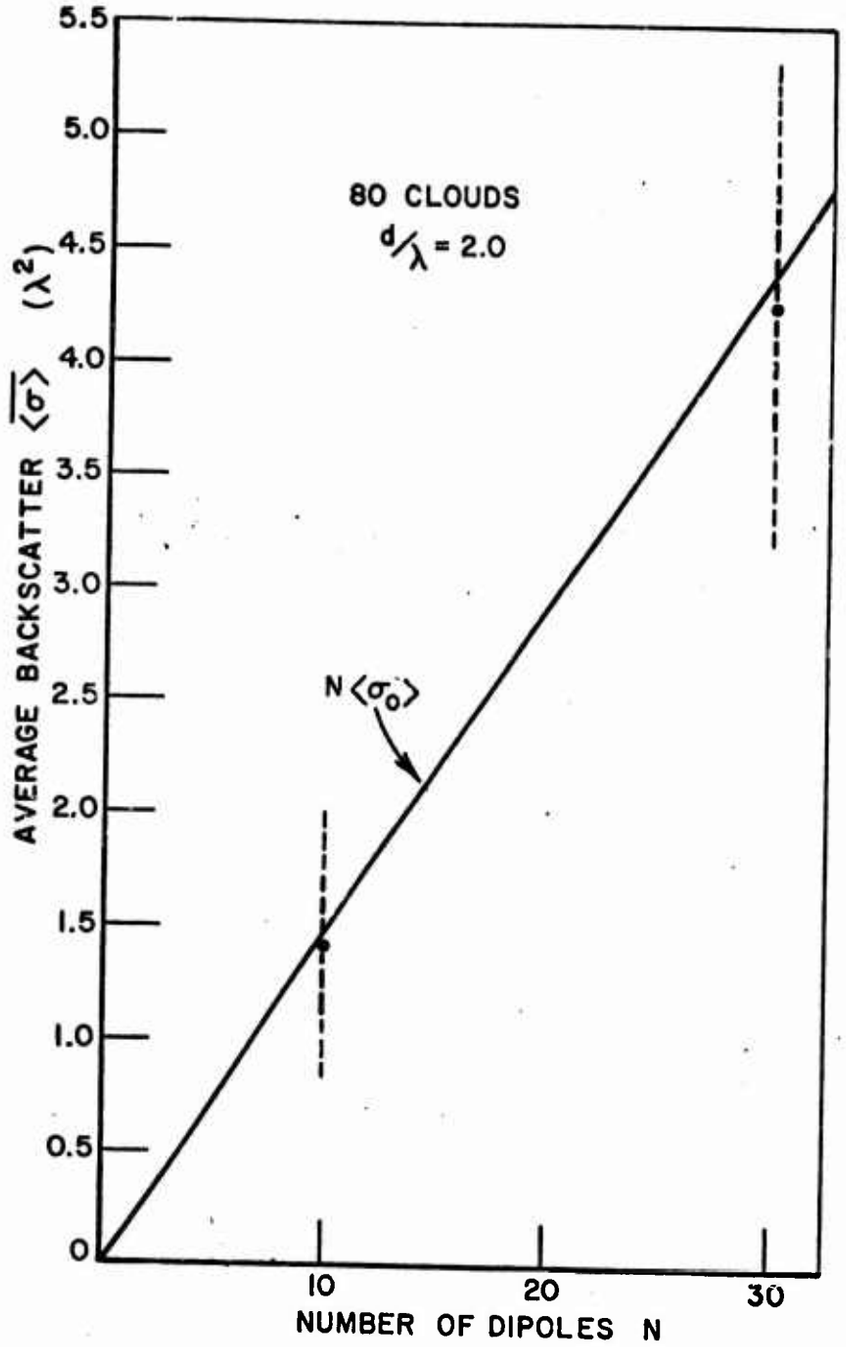


Figure I-6. Ensemble average backscatter $\overline{\langle\sigma\rangle}$ over 80 frozen chaff clouds with average spacing $d/\lambda = 2$. Two cases are considered, N = 10 dipoles and N = 30 dipoles as indicated by the heavy dots. The heavy dashes indicate the range containing 68% of the individual spatial averages (within one standard deviation either side of $\overline{\langle\sigma\rangle}$) and the light dashes indicate the range containing 95% of the individual spatial averages (within two standard deviations either side of $\overline{\langle\sigma\rangle}$). The straight line represents uncoupled dipoles.

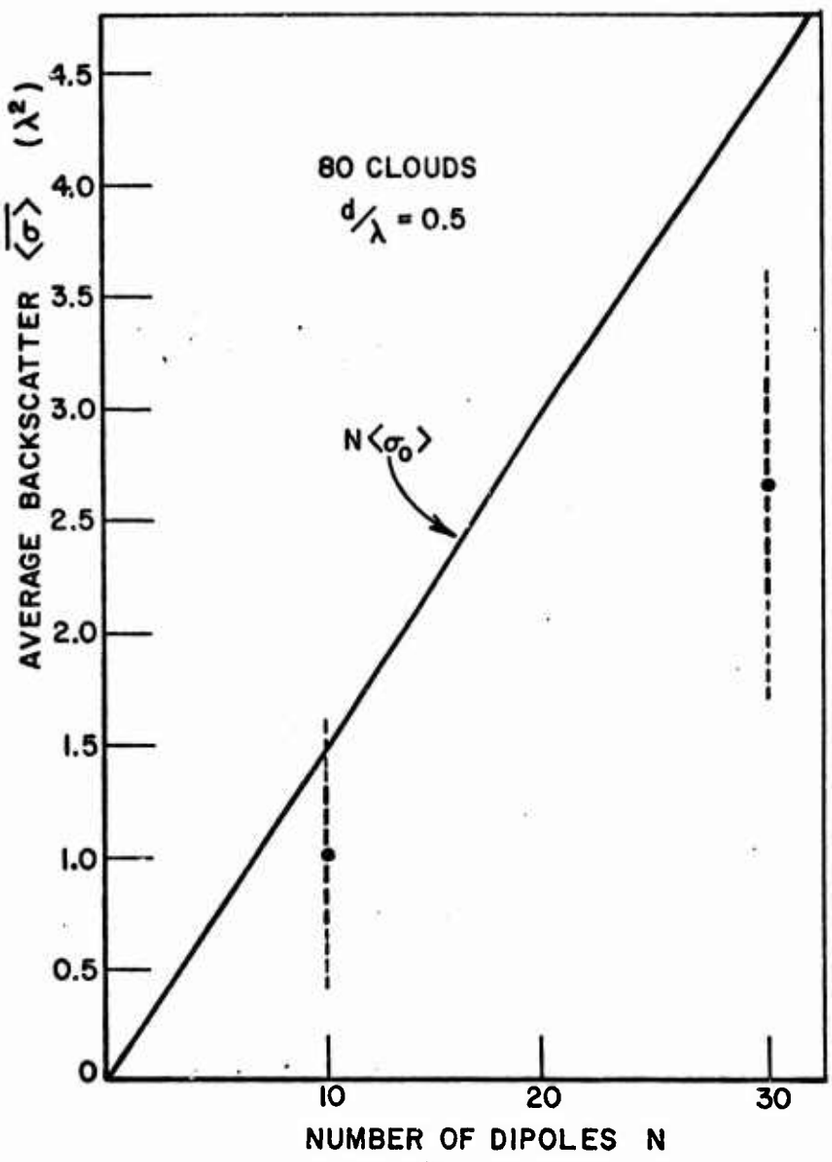


Figure I-7. Ensemble average backscatter $\overline{\langle\sigma\rangle}$ over 80 frozen chaff clouds with average spacing $d/\lambda = 0.5$. Two cases are considered, N = 10 dipoles and N = 30 dipoles as indicated by the heavy dots. The heavy dashes indicate the range containing 68% of the individual spatial averages (within one standard deviation either side of $\overline{\langle\sigma\rangle}$) and the light dashes indicate the range containing 95% of the individual spatial averages (within two standard deviations either side of $\overline{\langle\sigma\rangle}$). The straight line represents uncoupled dipoles.

APPENDIX B

REACTION MATCHING IN ELECTROMAGNETIC PROBLEMS

The objective of this work is to determine the electromagnetic scattering properties of large (random) clouds of perfectly conducting wires (dipoles) illuminated by a monochromatic plane wave. The emphasis is on applying an integral equation solution to this problem for those cases where the number of volume density of dipoles is large (1000 dipoles, 8 dipoles/λ^3) and the mutual couplings among all dipoles must be taken into account. The purpose of this appendix is to review the reaction [21] technique for developing an integral equation for the currents induced on these dipoles and to consider the transformation of this integral formulation via Moment Methods [22] to a system of algebraic equations more suitable for numerical solution by digital computer.

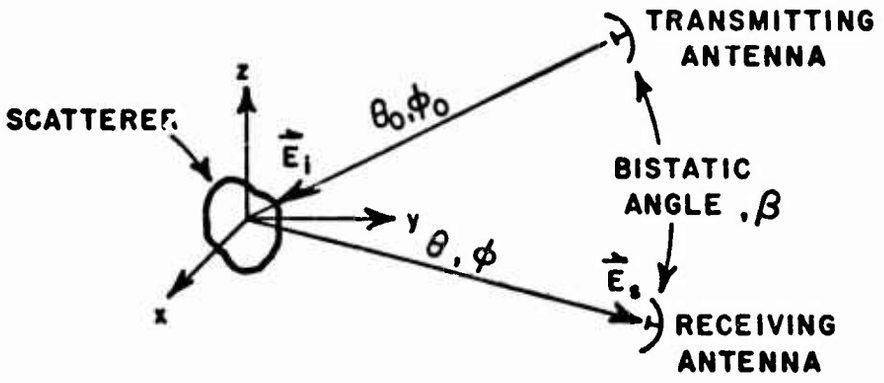
A. Scattering Properties of Obstacles

One measure often used to characterize scattering properties is radar cross section or echo area σ defined by

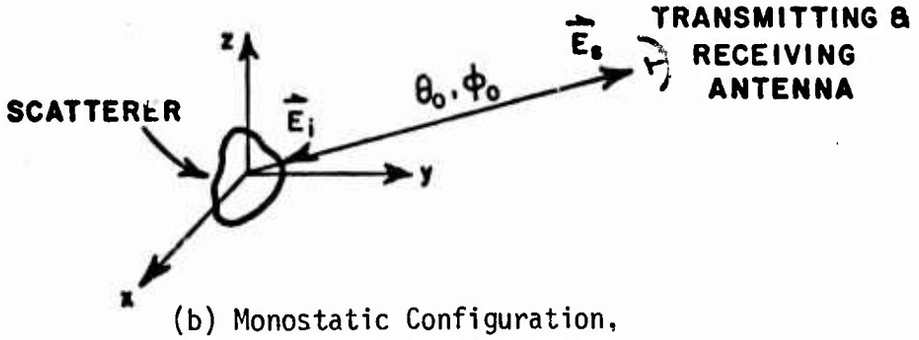
$$\text{(II-1)} \qquad \sigma = \lim_{R\to\infty} 4\pi R^2 \frac{|\bar{E}_s \cdot \hat{h}_r|^2}{|\bar{E}_i|^2}$$

where $\bar{E}_i$ is the electric field intensity of an incident plane wave of fixed polarization arriving from a particular direction (say θ_o,ϕ_o) and $\bar{E}_s$ is the electric field intensity of the scattered field a large distance R from the obstacle in an arbitrary direction (θ,ϕ). The quantity h_r is a unit vector spacifying the direction of the vector effective height of the receiving antenna which fixes the polarization component of the scattered field intercepted by this antenna. The reader is referred to the work by Kennaugh [23] for a complete discussion of the characterization of polarization properties for arbitrary scatterers.

The units for σ in the MKS system are meters^2 and radar cross section obviously represents an area. More specifically, σ is the area normal to the incoming plane wave which would intercept enough power from the incident fields so that if this power were reradiated isotropically the power intercepted by the receiving antenna would be identical to that caused by the obstacle itself. Figure II-1 illustrates the two basic types of radar



(a) Bistatic Configuration.



(b) Monostatic Configuration,

$\phi = \phi_0$

Figure II-1. Scattering cross-section configurations.

cross section measurements; bistatic cross section where transmitter and receiver are separateed by angle $\beta \neq 0$, and monostatic cross section where transmitter and receiver are coincident, $\beta = 0$. Monostatic cross section is commonly referred to as backscatter cross section and this terminology will be adhered to in all following discussions.

B. Scattering by Perfectly Conducting Bodies

Calculation of the scattered electric field appearing in Eq. (II-1) normally requires knowledge of the "secondary sources" induced in or on the scattering obstacle. A perfectly conducting obstacle will obviously have only a secondary source of the electric type induced on its surface and an integral equation for this surface distribution can be derived by applying the usual boundary conditions and the "zero reaction" theorem of Rumsey [21]. A detailed treatment of this approach is given by Richmond [24] and is summarized here.

Consider the basic geometry for the problem shown in Fig. II-2. The arbitrary metallic scatterer is located about the origin O in a right hand coordinate system and the primary electric and magnetic source distributions $\bar{J}_i$, $\bar{M}_i$, of finite extent and with $e^{j\omega t}$ time dependence, are located by position vector $\bar{R}$. R here is assumed large ($R \to \infty$), thus assuring that the free space fields of $\bar{J}_i, \bar{M}_i$, in the absence of the scatterer, produce a plane wave in the vicinity of O. For convenience consider these plane wave fields to be θ polarized with components given by

$$\bar{E}_i = e^{jk_o R} \hat{\theta} \qquad \text{(II-2)}$$

$$\bar{H}_i = -\frac{1}{\eta_o} e^{jk_o R} \hat{\phi} \qquad \text{(II-3)}$$

where $\eta_o = 120\pi$ is the free space wave impedance and $k_o = \sqrt{\mu_o \varepsilon_o}$ is the free space propagation constant.

The surface of the perfectly conducting obstacle, defined by S in Fig. II-2, separates the interior source free region V from the exterior region containing $\bar{J}_i, \bar{M}_i$. Consider the total fields in the presence of the scattering obstacle. These fields are (E,H) outside V and $(0,0)$ inside V and are the superposition of the free space fields of $\bar{J}_i, \bar{M}_i$ and the free space field of new secondary source J_s on S. J_s here is precisely the surface conduction current distribution induced on S by $(\bar{E}_i, \bar{H}_i)$. It is convenient at this point to define the scattered fields $\bar{E}_s, \bar{H}_s$ in terms of the difference fields given by

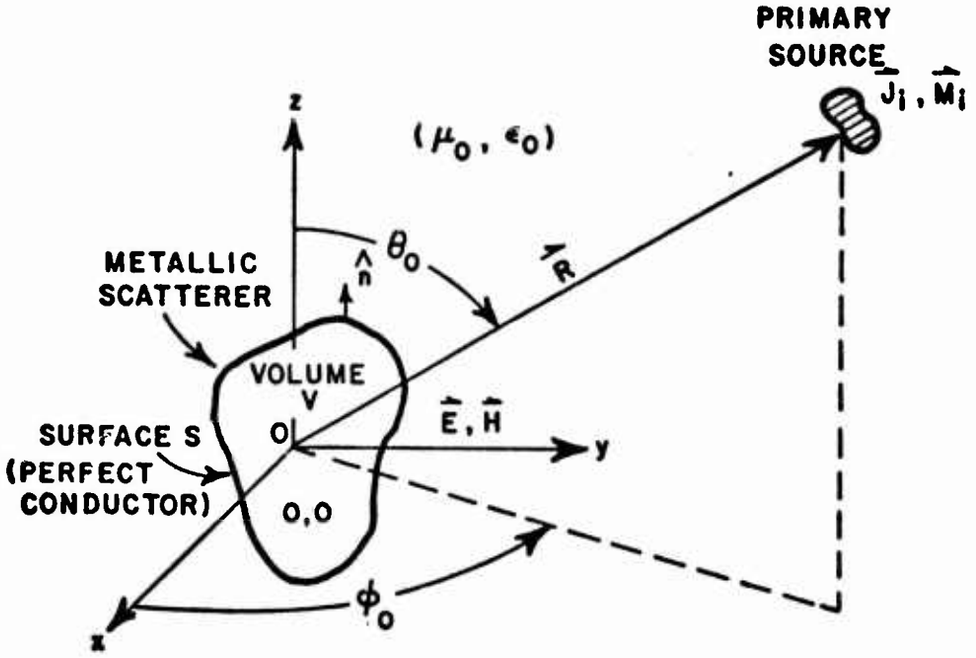


Figure II-2. Arbitrary metallic scatterer in presence of primary sources.

(II-4) $\bar{E}_s \equiv \bar{E} - \bar{E}_i$

(II-5) $\bar{H}_s \equiv \bar{H} - \bar{H}_i$.

From this definition of fields, the surface distribution $\bar{J}_s$, radiating in free space, must generate (E_s, H_s) outside V and $(-\bar{E}_i, -\bar{H}_i)$ inside V . $\bar{J}_s$ can be written in terms of the boundary conditions on S (a perfect conductor) as

(II-6) $\bar{J}_s = \hat{n} \times \bar{H}$,

where n is taken to be the unit outward normal. The equivalent problem stated in terms of $(\bar{J}_i, \bar{M}_i)$ and $\bar{J}_s$ radiating in free space is illustrated in Fig. II-3.

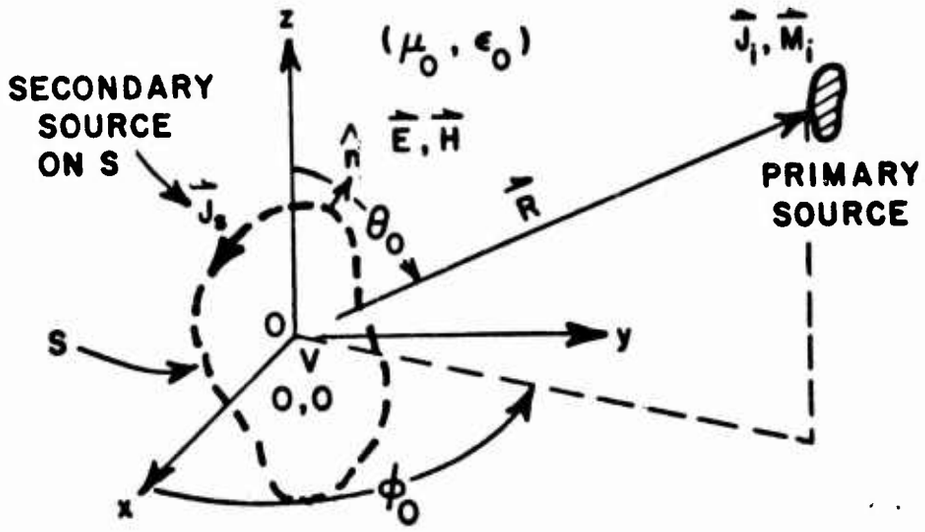


Figure II-3. Equivalent problem.

The term "reaction" was first introduced by Rumsey to describe certain measurable effects between sources and their fields. Consider two sources $(\bar{J}_a, \bar{M}_a)$ and $(\bar{J}_b, \bar{M}_b)$ of finite extent and radiating fields $(\bar{E}_a, \bar{H}_a)$ and $(\bar{E}_b, \bar{H}_b)$, respectively. Also consider the region of space containing these sources and their fields to be isotropic but not necessarily homogeneous. The reaction of source b on source a, denoted by <a,b> (mutual reaction), is defined by the scalar quantity

$$\text{(II-7)} \qquad \langle a,b \rangle = \iiint_a (\bar{J}_a \cdot \bar{E}_b - \bar{M}_a \cdot \bar{H}_b) da'$$

where the region of integration in this case, is over the "a" sources (e.g., volumetric, surface, or filamentary). In reciprocity media the reciprocity theorem of Carson [25] can be applied to Eq. (II-7) to show equality of mutual reactions; i.e.,

$$\text{(II-8)} \qquad \langle a,b \rangle = \langle b,a \rangle \ .$$

Sources can also be reacted with themselves to yield "self-reactions" denoted <a,a< or <b,b>.

Application of the reaction concept to the present scattering problem leads readily to the required integral equation formulation for the unknown surface current J_s and to a variational solution for this current. Consider placing an arbitrary but known "test" distribution of electric current J' ("b" source) inside V and let this source radiate free space fields (E',H'). These fields of the "b" source can be reacted with the superposition of sources $\bar{J}_i$ and $\bar{J}_s$ ("a" sources) to yield

$$(II\text{-}9) \qquad \langle a,b \rangle = \iiint_{\substack{\text{primary} \\ \text{sources}}} \bar{J}_i \cdot \bar{E}' dv + \iint_S \bar{J}_s \cdot \bar{E}' ds \ .$$

Similarly, the fields of J_i, J_s can be reacted with the test source J' and because the resulting reaction integral is taken over the J' source located in the null field region V, this reaction is identically zero; i.e.,

$$(II\text{-}10) \qquad \langle b,a \rangle = \iiint_{\substack{\text{test} \\ \text{source}}} \bar{J}' \cdot (\bar{E}_i - \bar{E}_i) dv' \equiv 0 \ .$$

However, by way of the reciprocity between mutual reactions (Eq. II-8), Eq. (II-9) is also identically zero and can be rearranged to give

$$(II\text{-}11) \qquad -\iint_S \bar{J}_s \cdot \bar{E}' \, ds = \iiint_{\substack{\text{primary} \\ \text{source}}} \bar{J}_i \cdot \bar{E}' \, dv \ ,$$

and by applying Carson's reciprocity theorem to the right hand side of Eq. (II-11), the final form for the integral equation becomes

$$(II\text{-}12) \qquad -\iint_S \bar{J}_s \cdot \bar{E}' \, ds = \iiint_{\substack{\text{test} \\ \text{source}}} \bar{J}' \cdot \bar{E}_i \, dv' .$$

This is now in a convenient form where $\bar{E}_i$ is well defined (Eq. II-2) and $\bar{J}'$ can be specified so that $\bar{E}'$ can be calculated. That leaves $\bar{J}_s$ as the only unknown quantity in this expression. Integral equation Eq. (II-12) is a special case of a more general Reaction Integral Equation (RIE) formulation discussed by Richmond [26]. Furthermore the "zero reaction" test was applied with an electric test source $\bar{J}'$ only and the result

was the electric field integral of Eq.(II-1); however, if a magnetic test source M' had been used, the result would have been a magnetic field integral form. This use of a zero reaction test appears to have first been used by Kouyoumjian [27] and later developed by Rumsey and Richmond.

C. Numerical Solutions

Solutions of the electromagnetic integral equation, Eq. (II-12), have in the past been obtained via a number of classic procedures, e.g., modal (eigenfunction) expansions, low frequency expansions (powers of k_0), high frequency expansions (powers of $1/k_0$), variational methods, physical and geometrical optics, etc. However, with the advent of the numerical computer, the most common method of solution, especially for complicated resonant sized obstacles, has become the method of moments [22]. This is the technique alluded to earlier by which the integral equation is converted to a system of simultaneous algebraic equations; the computer being admirably suited to compute the "inversion" type solution to this system of equations.

Consider a generalized set of vector functions Φ_n, $n=1,2,\cdots$ defined on S to be suitable for expanding the induced surface currents on S; i.e.,

$$(II\text{-}13) \qquad \bar{J}_s = \sum_{n=1}^{\infty} J_n \bar{\Phi}_n ,$$

where J_n are unknown (complex) coefficients to be determined. Also assume the nth expansion "mode" Φ_n of this set radiates fields $(\bar{E}_n, \bar{H}_n)$ in free space.

Consider another set of normalized vector modes, $\bar{\theta}_m$, $m=1,2,\cdots$ as a set of arbitrary test sources J'; i.e.,

$$(II\text{-}14) \qquad \bar{J}' = \bar{\theta}_m, \quad m=1,2,\cdots .$$

Let mode m of this set, θ_m, radiate $(\bar{E}_m, \bar{H}_m)$ in free space. Equation (II-12), rewritten in terms of these expansions, becomes the doubly infinite set of algebraic equations given by

$$(II\text{-}15) \qquad -\sum_{n=1}^{\infty} J_n \iint_n \bar{\Phi}_n \cdot \bar{E}_m \, ds = \iiint_m \bar{\theta}_m \cdot \bar{E}_i \, dv' ,$$

$$m = 1,2,\cdots,$$

where the orders of integration and summation has been interchanged and the regions of integration are over the respective domains for each mode function. The practical choice of mode functions which will be used here leads to more manageable finite systems of equations than implied by Eq. (II-15).

Recall, the reaction test sources $\bar{J}'$ have not yet been specified. Consider now the particular choice for the J' distributions $\bar{\theta}_m = \bar{\Phi}_m$, which is known as Galerkin's method and let this mode set consist of a finite number of functions $\bar{\Phi}_n$, n=1,2,...,N where each function is nonzero only over a specific interval in space (e.g., volume region, surface area, or section of a contour); the $\bar{\Phi}_n$'s in this case constitute an incomplete subsectional basis set. Figure II-4 illustrates one method of subsectionalizing the surface domain of J_s where J' is defined to flow on surface S'. Surface S', in the case of a general scatterer, recall, must be located "inside" S as shown in the figure. However, for the perfectly conducting obstacles, S' may coincide with S and the zero reaction test will remain a valid test. The system of algebraic equations defined in terms of this finite subsectional mode expansion now takes the form

$$(II\text{-}16) \quad -\sum_{n=1}^{N} J_n \iint_n \bar{\Phi}_n \cdot \bar{E}_m \, ds = \iint_m \bar{\Phi}_m \cdot \bar{E}_i \, ds', \quad m=1,2,\ldots,N,$$

where $\bar{E}_m$ denotes the electric field of test source $\bar{\Phi}_m$ located on S'. This algebraic system of N equations with N unknowns J_n is commonly represented in the electromagnetics literature by the matrix formulation

$$(II\text{-}1) \quad ZI = V,$$

where $Z = [Z_{mn}]$ represents the N x N matrix of generalized mode impedances with elements Z_{mn} given by

$$(II\text{-}18) \quad Z_{mn} = -\iint_n \bar{\Phi}_n \cdot \bar{E}_m ds \; ; \qquad m,n=1,2,\cdots,N,$$

$I = (J_n)$ is the N x I vector of unknown mode currents and $V = (V_m)$ is the N x I vector of known generalized mode voltages given by

$$(II\text{-}19) \quad V_m = \iint_m \bar{\Phi}_m \cdot \bar{E}_i \, ds', \quad m=1,2,\ldots,N \, .$$

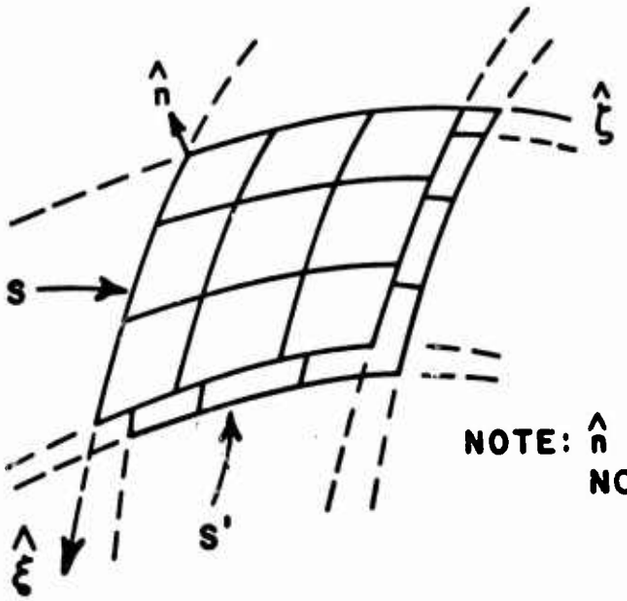


Figure II-4. Subsectionalization of S and S' and convenient surface coordinate system (ξ,ζ).

The expansion which defines $\overline{J}_s$ is given by

$$(II\text{-}20) \quad \overline{J}_s \approx \sum_{n=1}^{N} J_n \, \overline{\Phi}_n \text{ on } S$$

and the test sources are given by

$$(II\text{-}2) \quad \overline{J}' = \overline{\Phi}_m, \; m=1,2,\ldots,N \text{ on } S'.$$

The indicated approximation of $\overline{J}_s$ in Eq. (II-20), under suitable conditions, will approach the true distribution when, in the limit, the subsectioning becomes infinitely "fine" and $N\to\infty$. This of course defeats the purpose of numerical modeling and the assumption here is that a reasonable number of samples (4-10 per λ^2) will give enough information to successfully interpolate $\overline{J}_s$. The use of testing functions on S' instead of S when S and S' are separated, also has the particular advantage of avoiding the singular nature of the self-reaction of a source with its own field. Normal separations between S and S' should be less than 0.01λ to give good numerical results for the types of EM problems discussed here.

D. Examples of Bases for Surface-Patch and Wire-Grid Modeling

The order N of the system of equations represented in Eq. (II-16) is obviously dependent on the geometry and electrical size of the scatterer, the choice of basis set and the degree and type of subsectioning required to achieve a desired numerical accuracy. The purpose of this section will be to present certain examples of basis sets for the continuous conducting obstacle and to discuss some advantages and disadvantages of each.

1. Surface-Patch Bases (Patch subsectioning)

Figure II-5 shows examples of two basis functions suitable for the surface-patch model. Basis functions of this type were first considered by Wang, Richmond, et al. [28]. A specific example of the use of the cosine modes on a flat plate scatterer is shown in Fig. II-6 where only $\hat{\xi}$ directed modes are considered; however, for more accurate results and/or the case of an arbitrarily polarized incident wave, $\hat{\zeta}$ directed modes would also be included. The approximation to J_S is then computed as a linear combination of modes in the two vector directions $\hat{\xi}$ and $\hat{\zeta}$.

2. Wire-Grid Subsectional Modeling and the Piecewise Sinusoidal Basis Functions

One particular geometry of considerable interest in EM theory is the thin cylindrical antenna or scatterer and its applications to the modeling of arbitrarily shaped conducting obstacles. First developments in the use of wires for numerical modeling of continuous conducting shapes were advanced by Richmond [29] and this approach was later used extensively by Lin and Richmond [30] and Thiele [31]. The basic technique of wire-grid modeling is to define a suitable number of points on the surface of the obstacle and then interconnect these points with straight wire segments. These segments serve as approximate paths for the induced surface currents and the integral equation of Eq. (II-16) now becomes one for solving for the unknown surface currents on these wires.

One possible set of basis modes which are amenable to the wire-grid structure are the overlapping piecewise sinusoidal dipole modes introduced by Richmond [32]. Other types of subsectional bases often appear in the literature [33]; e.g., pulse bases, piecewise linear bases, etc. The literature also refers to trigonometric whole bases [34] from time to time. However, the piecewise sinusoidal basis functions have been shown [35] to have certain superior properties, making them well suited to numerical solution of wire structure problems.

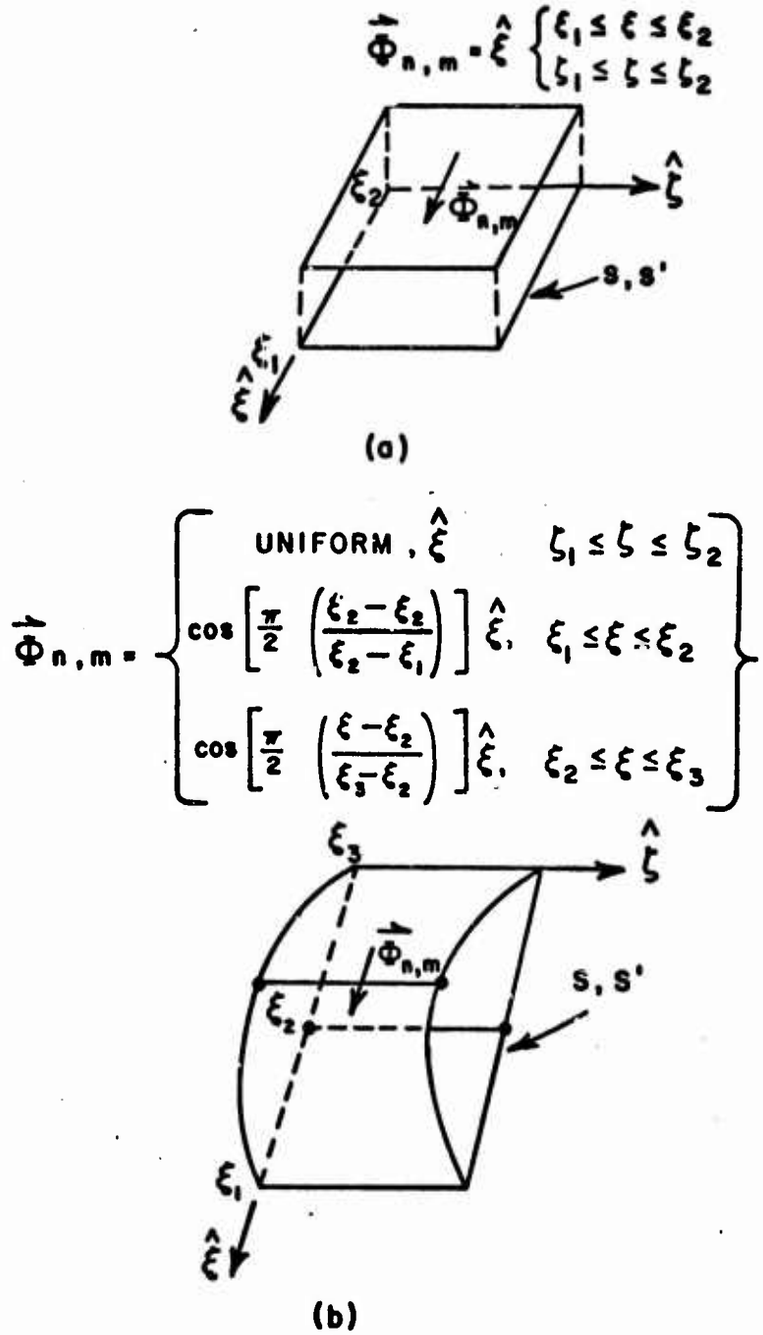


Figure II-5. Examples of subsectional basis functions for surface scatterer. (a) Uniform rectangular pulse basis functions, one pulse per subsectional region; (b) Overlapping cosinusoidal basis functions, one cosine mode per two subsections in ξ, uniform in ζ.

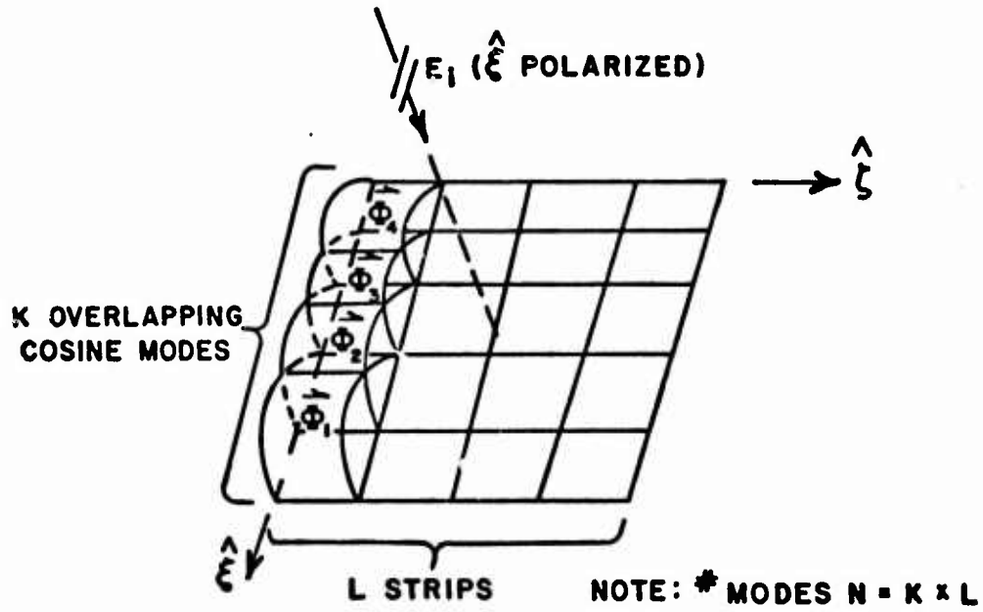
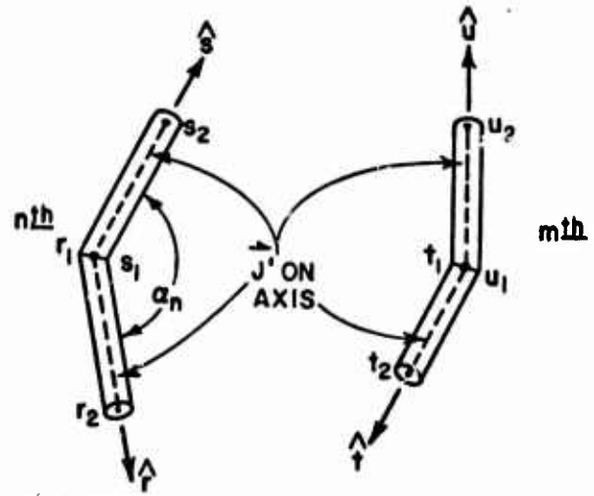


Figure II-6. Mode structure for computing backscatter cross section from thin square flat plate (perfect conductor) using overlapping cosine modes (see Fig. II-5).

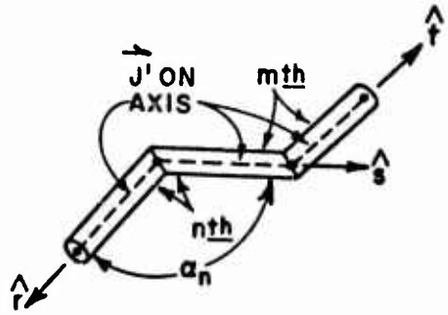
Figure II-7 shows two examples of pairs of interconnecting segments - separated pairs and overlapping pairs. Consider the nth dipole mode $\overline{\Phi}_n$ given by

$$\text{(II-22)} \qquad \overline{\Phi}_n = \begin{cases} \dfrac{\sin k_o(r_2-r)}{\sin k_o(r_2-r_1)}\ \hat{r}, & r_1 \le r \le r_2 \\ \dfrac{\sin k_o(s_2-s)}{\sin k_o(s_2-s_1)}\ \hat{s}, & s_1 \le s \le s_2 \,. \end{cases}$$

This mode flows as a tubular surface current density on the nth pair of intersecting segments (v-dipole) with arms in the r and s directions. Now consider the test source $\overline{J}^i$ (Eq. (II-21) to be a filamentary source on the axes of these segments. It can be shown that the reaction of this axial test source with any colinear tubular surface current mode is identical to the reaction between this same axial test source and a filamentary current mode $2\pi a\ \overline{\Phi}_n$ located one radius "a" away from the segment axis. Figure II-8 illustrates the



(a) SEPARATE DIPOLES



(b) OVERLAPPING DIPOLES

Figure II-7. Nonoverlapping dipole segments and overlapping dipole segments.

equivalent cases. Numerical calculations have indicated [36] that for the non-colinear cases (Fig. II-7), the errors introduced into the self and mutual reactions, by using the axial test sources and filamentary approximations for the surface modes, can be neglected when segment lengths exceed 20 radii and spacings between separate dipoles exceed a/λ or the angle α_n between two intersecting (overlapping) segments exceeds $\sim 30°$. Figure II-9 illustrates a section of wire-grid modeling for an arbitarily shaped conducting obstacle and shows a portion of an overlapping piecewise sinusoidal mode structure. Only a few $\hat{\xi}$ directed modes are shown; however, for an arbitarily directed surface current, modes must be included in the $\hat{\zeta}$ direction and enforcement of continuity of the currents at each junction of multiple segments assures that a junction having k intersecting segments will have only k-1 independent dipole modes passing through it.

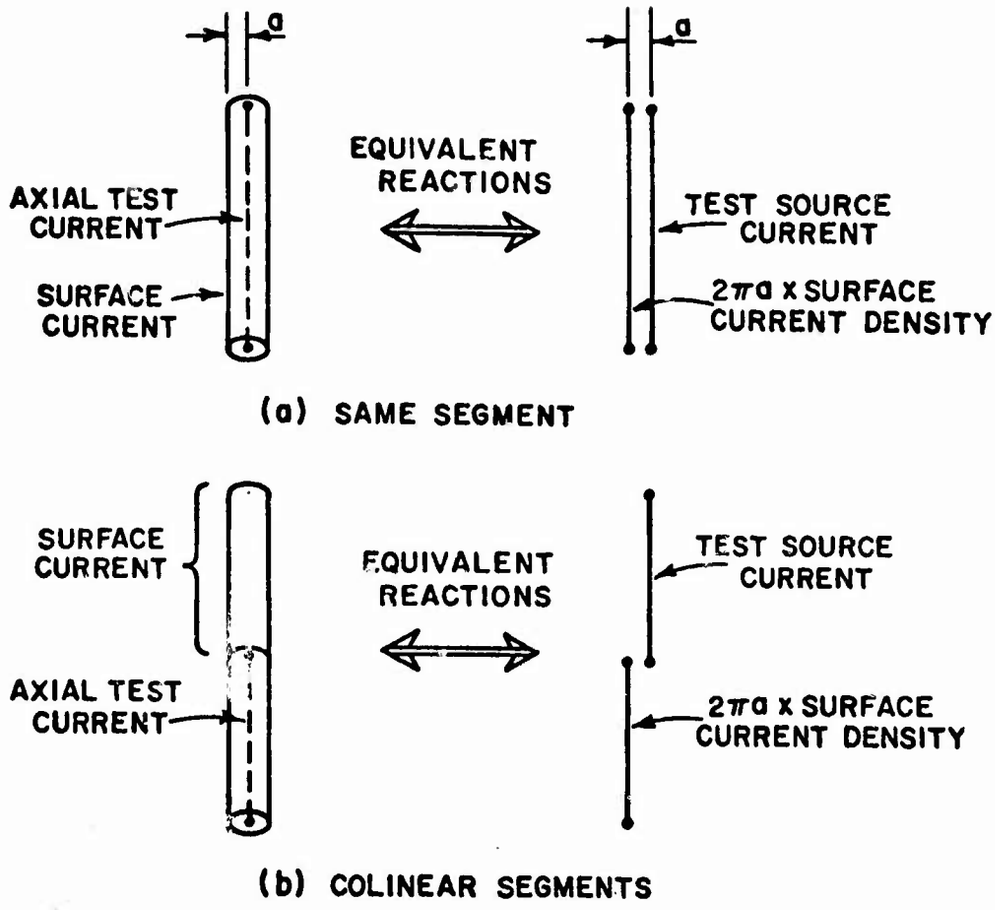


Figure II-8. Equivalence of reactions between colinear axial test source and tubular surface current and equivalent parallel filamentary cases.

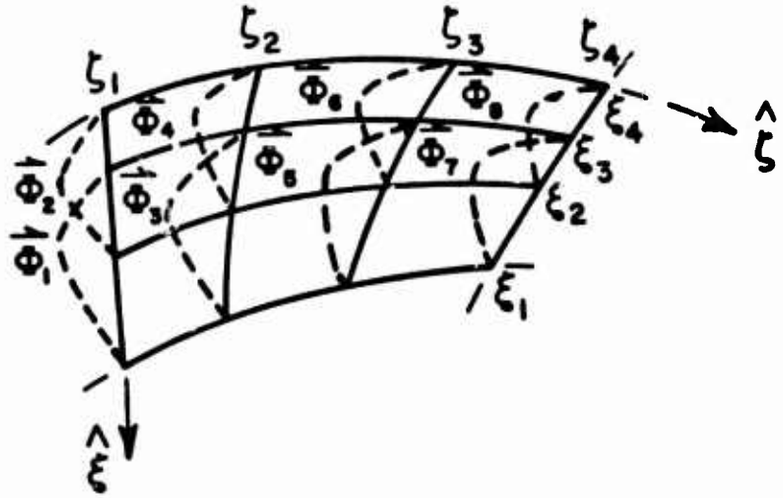


Figure II-9. Sample mode structure on wire-grid model of conducting surface. Sample of $\hat{\xi}$ basis functions shown; however, both $\hat{\xi}$ and $\hat{\zeta}$ function required in general.

The integrals in Eq. (II-16) become line integrals for this type of modeling and successful application of the piecewise sinusoidal expansion modes normally requires wire segment lengths not to exceed 0.25λ.

3. Advantages and Disadvantages

Both surface patch and wire-grid modeling are generally considered suitable for continuous conducting obstacles. However, if the obstacle includes a protruding section; e.g., antenna (monopole), then the wire-grid type structure is usually more convenient. The surface patch technique, on the other hand, will model the same size surface with fewer modes but computations of the wire-grid mutual impedances are performed much faster than for the patches. If computing time is critical, then the wire-grid model might be considered to have the advantage, even though it may require a larger number of modes.

E. Chaff Clouds

The discussion so far has emphasized the more general cases of arbitrarily shaped conducting scatterers; however, it also serves as the basic background needed for the problem at hand, namely, scattering by random clouds of thin conducting wires. Here, the wires are assumed to be of resonant length $\sim 0.5\lambda$ and the piecewise sinusoidal modes are used. Each wire can then be modeled as a p=2 segment dipole requiring only one mode per wire.

Possible exceptions to this will occur for those cases where the wire lengths are significantly greater than resonant length, a situation briefly treated in this study.

Figure II-10 shows two typical 2-segment wires and also illustrates the approximate filamentary models used; test expansion mode $_m$ on the axis of dipole s and unknown expansion mode I_n $_n$ on the surface of dipole t. The actual random array will consist of many of these resonant wires with the centers of all wires chosen with uniform or nonuniform probability in a spherical volume of space and each randomly oriented according to a uniform spherical probability density function. The technique used to generate the array is discussed in detail in Appendix III.

F. A Convenient Change in Notation

A rather more convenient form for the matrix equation presented in Eq. (II-17) can be expressed using slightly different matrix and vector notation. The following definitions, while perhaps unconventional from the standpoint of electromagnetic theory, are in standard usage in numerical analysis and will be used throughout the remaining chapters of this study. The self and mutual reactions or generalized model impedances previously defined in Eq. (II-18) will be denoted here by the N x N matrix $A = [a_{mn}]$ with the elements a_{mn} given by

$$(II\text{-}23) \quad a_{mn} = -\iint_n \overline{\Phi}_n \cdot \overline{E}_m \, ds; \quad m,n = 1,2,\cdots,N,$$

and the generalized mode voltages previously defined in Eq. (II-19) are now denoted by the N x 1 vector $b = (b_m)$ with elements b_m given by

$$(II\text{-}24) \quad b_m = \iint_m \overline{\Phi}_m \cdot \overline{E}_i \, ds', \; m = 1,2,\cdots N.$$

The unknown mode coefficients J_n, representing samples of the distribution $\overline{J}_s$, will be denoted by the N x 1 vector $x = (x_n)$, $n = 1,2,\cdots,N$. This whole system of equations is now expressed in the new notation as

$$(II\text{-}25) \quad Ax = b \; .$$

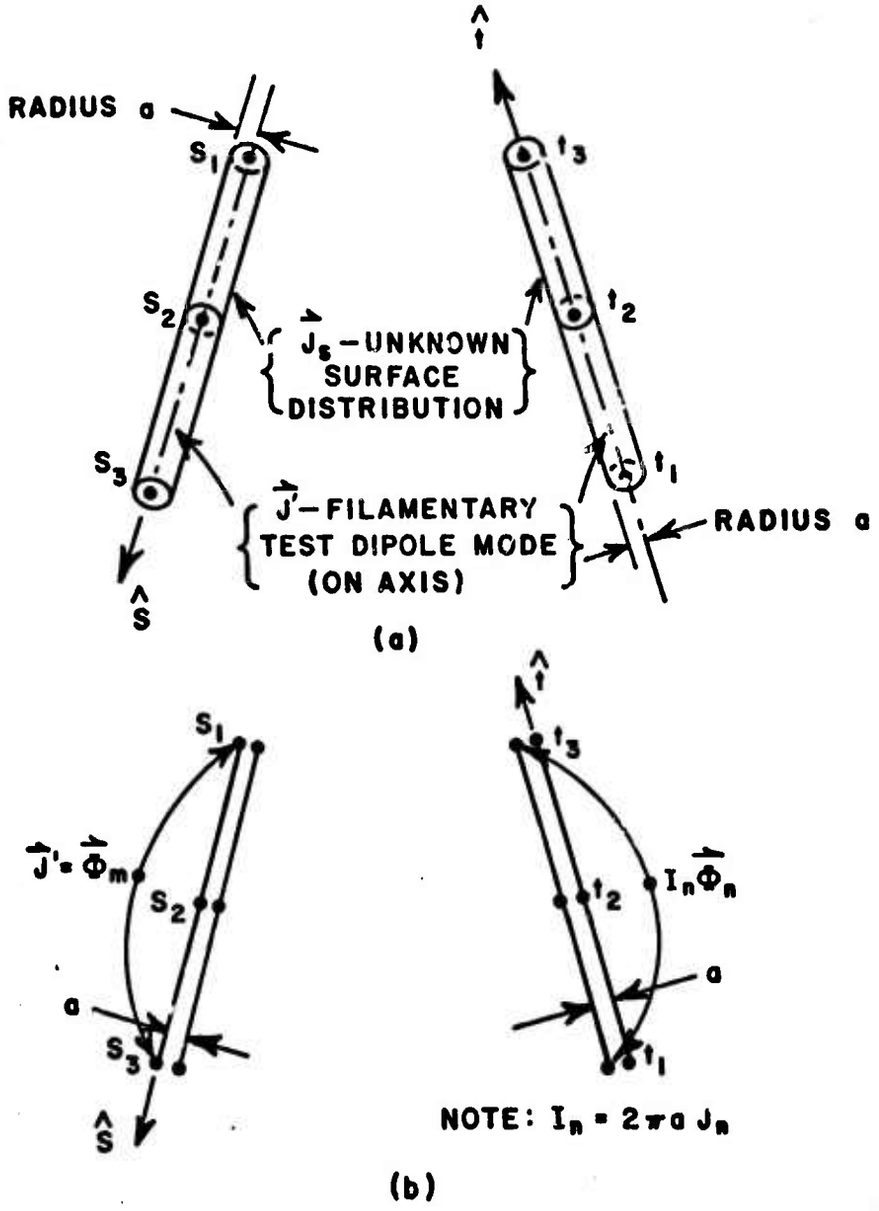


Figure II-10. a) Thin cylindrical wires, b) Approximate filementary model using piecewise sinusoidal expansions Φ_n on surface and Φ_m on axis.

The N x N impedance matrix A in the case of these random arrays of thin wires will contain all possible interactions among N wires and will not be a "thin" or sparse matrix. Also, the number of wires considered will be as large as N = 1000 and hence, the equation to be solved, Eq. (II-25), will be a "full" matrix equation of up to order 1000. All elements of Eq. (II-25) will be complex numbers and the impedances given by Eq. (II-23) will be complex symmetric, i.e., $a_{mn} = a_{nm}$ for all m and n. This last condition results from the reciprocity relation of Eq. (II-8) and the use of Galerkin's method.

APPENDIX C
CLOUD GEOMETRY

A. The Radially Inhomogeneous Cloud

To create a chaff cloud, N dipoles are randomly positioned in space and oriented according to certain statistical rules. Their orientations are specified so that all possible orientations are equally likely, i.e., a spherical probability density function for orientation is implied. Their positions are specified by the Cartesian coordinates (x,y,z) of their centers according to the following rules:

1. The probability of finding the x-coordinate of a dipole center in a small increment Δx about x is

$$(III\text{-}1) \qquad g(x)\Delta x = \frac{\Delta x}{\sqrt{2\pi\delta^2}}\, e^{\frac{1}{2}\left(\frac{x}{\delta}\right)^2}$$

2. The probability of finding the y coordinate of a dipole center in a small increment Δy about y is

$$(III\text{-}2) \qquad g(y)\Delta y = \frac{\Delta y}{\sqrt{2\pi\delta^2}}\, e^{-\frac{1}{2}\left(\frac{y}{\delta}\right)^2}$$

3. The probability of finding the z coordinate of a dipole center in a small increment Δz about z is

$$(III\text{-}3) \qquad g(z)\Delta z = \frac{\Delta z}{\sqrt{2\pi\delta^2}}\, e^{-\frac{1}{2}\left(\frac{z}{\delta}\right)^2}$$

4. The process by which the coordinates (x,y,z) of a dipole center are selected are statistically independent.

Note that the three probability density functions are Gaussian with zero mean and identical standard deviation δ, implying that the cloud is most dense in the center and spherically symmetric.

Because of the statistical independence property, the probability of finding a dipole center in a small cube of volume $v=\Delta x\Delta y\Delta z$ about the point (x_1,y_1,z_1) is

$$(III\text{-}4)\ \Delta P(x_1,y_1,z_1;\Delta v) = \int_{z_1-\frac{\Delta z}{2}}^{z_1+\frac{\Delta z}{2}} \int_{y_1-\frac{\Delta y}{2}}^{y_1+\frac{\Delta y}{2}} \int_{x_1-\frac{\Delta x}{2}}^{x_1+\frac{\Delta x}{2}} g(x)g(y)g(z)\,dx\,dy\,dz$$

If there are a total of N dipoles in the cloud, the number of dipoles expected to lie in the small cube v about the point (x_1,y_1,z_1) is on the average,

$$\text{(III-5)} \qquad \Delta N(x_1,y_1,z_1;\ v) = N\ \Delta P(x_1,y_1,z_1;\Delta v)$$

so the fraction of the total number of dipoles lying in v about (x_1,y_1,z_1) is on the average,

$$\text{(III-6)} \qquad \frac{\Delta N(x_1,y_1,z_1;\Delta v)}{N} = g(x_1)g(y_1)g(z_1)\Delta v$$

If N is very large, or a large ensemble of clouds with the same N and standard deviation δ is assumed, and if the sample volume Δv is made very small, we can define in the limit the relative density of dipoles at a point (x_1,y_1,z_1) by

(III-7)

$$n(x_1,y_1,z_1) = \lim_{\substack{N\to\infty \\ \Delta v\to 0}} \frac{\Delta N(x_1,y_1,z_1;\Delta v)}{N\ \Delta v} = g(x_1)g(y_1)g(z_1)$$

$$= n(r_1) = \frac{1}{2\pi\delta^2}\left[\frac{1}{\sqrt{2\pi\delta^2}}\ e^{-\frac{1}{2}\left(\frac{r_1}{\delta}\right)^2}\right]$$

or

$$\text{(III-8)} \qquad n(r\ ) = \frac{1}{2\pi\delta^2}\ g(r)$$

where $r = (x^2+y^2+z^2)^{1/2}$ is the radial distance from the center of the cloud. (From (III-8), we see that the dipole density is independent of (θ,ϕ) (a spherical symmetric cloud) and is proportional to a Gaussian function in the radial direction.

In our work, we chose to characterize a cloud by a constant which we call "the average spacing between dipoles," d/λ, defined as follows.

1. For a given spherical volume $V=(4/3)\pi R^3$ over which the average is desired, calculate the expected number of dipoles in V; call this number kN where N is the total number of dipoles in the cloud and $0\leq k\leq 1$ is the fraction of the total number of dipoles contained in V.

2. Consider V to be divided into kN equal cubes, and call the edge dimension of each cube d. In this manner we obtain the relationship

$$\text{(III-9a)} \qquad \frac{3}{4}\pi R^3 = kN\, d^3,$$

or

$$\text{(III-9b)} \qquad \frac{d}{\lambda} = \left(\frac{4\pi}{3}\right)^{1/3} \frac{1}{k^{1/3}} \frac{1}{N^{1/3}} \frac{R}{\lambda}$$

In our case,

$$\text{(III-10)} \qquad kN = N \int_0^{2\pi}\int_0^{\pi}\int_0^{R} \eta(r) r^2 \sin\theta \, dr \, d\theta \, d\phi$$

$$= N \int_{-R}^{R} \frac{r^2 e^{-\frac{1}{2}\left(\frac{r}{\delta}\right)^2}}{\delta^3\sqrt{2\pi}} dr ,$$

so

$$\text{(III-11)} \qquad k = \frac{1}{\delta^2\sqrt{2\pi\delta^2}} \int_{-R}^{R} r^2 e^{-\frac{1}{2}\left(\frac{r}{\delta}\right)^2} dr$$

which, evaluated by integration by parts, is

$$\text{(III-12)} \qquad k = \frac{1}{\sqrt{2\pi}} \int_{-R/\delta}^{R/\delta} e^{-\frac{1}{2}t^2} dt - \sqrt{\frac{2}{\pi}} \frac{R}{\delta} e^{-\frac{1}{2}\left(\frac{R}{\delta}\right)^2}$$

The first term in Eq. (III-12) is the integral of the normalized Gaussian function and can be evaluated from tables. Values of k are plotted vs R/δ in Fig. III-1.

If Eq. (III-9b) is written in the form,

$$\text{(III-13)} \qquad N^{1/3} \frac{d}{\delta} = \left(\frac{4\pi}{3k}\right)^{1/3} \frac{R}{\delta}$$

the quantity $N^{1/3}d/\delta$ may be plotted vs R/δ, using the values of k corresponding to values of R/δ according to Eq. (III-12). This plot is also shown in Fig. III-1. In this report, a value of $R = 2.05\delta$, corresponding to $k = 0.76$, has been chosen as the radius of the sphere over which an average is taken to obtain the relationship between d/λ and δ/λ. For this choice Eq. (III-9b) is

$$\text{(III-14)} \quad \frac{d}{\lambda} = \left(\frac{4\pi}{3}\right)^{1/3} \frac{1}{(0.76)^{1/3}} \frac{1}{N^{1/3}} 2.05 \frac{\delta}{\lambda}$$

$$= \frac{3.62}{N^{1/3}} \frac{\delta}{\lambda}$$

It was by selecting convenient values of d/λ, such as 2.0, 1.5, 1.0, 0.5 in this report, that corresponding values of δ/λ were obtained for use in Eqs. (III-1), (III-2), (III-3).

Note that the choice $R/\delta = 2.05$ is rather arbitrary. If, for example we chose to average over smaller and smaller spheres, in the limit as $R/\delta \to 0$ and $k \to 0$, we obtain the relationship between a new average spacing d'/λ and δ/λ,

$$\text{(III-15)} \quad \frac{d'}{\lambda} = \frac{\sqrt{2\pi}}{N^{1/3}} \frac{\delta}{\lambda}$$

$$= \frac{2.51}{N^{1/3}} \frac{\delta}{\lambda}$$

Assuming that the δ/λ values calculated from Eq. (III-14) are used in (III-15), we see that d'/λ is about $0.69d/\lambda$, yielding the corresponding table

d/λ	d'/λ
2.00	1.38
1.50	1.04
1.00	0.69
0.50	0.35
0.25	0.173

$\frac{\delta}{\lambda}$ same for the two cases

Thus, the values of d/λ presented in this report are conservatively large, i.e., substantially smaller average spacings are encountered in the center of each cloud.

The quantity $(1/d')^3$ is equal to the density of dipoles in the center of the cloud expressed in dipoles per cubic wavelength if d' is in wavelengths. Similarly, the quantity $(1/d)^3$ is the density of dipoles averaged over the sphere containing 76% of the dipoles. Some typical plots of dipole density versus radius for selected values of N and d/λ are shown in Figs. III-2,3. The dashed lines represent the values of δ/λ as related to d/λ by Eq. (III-14).

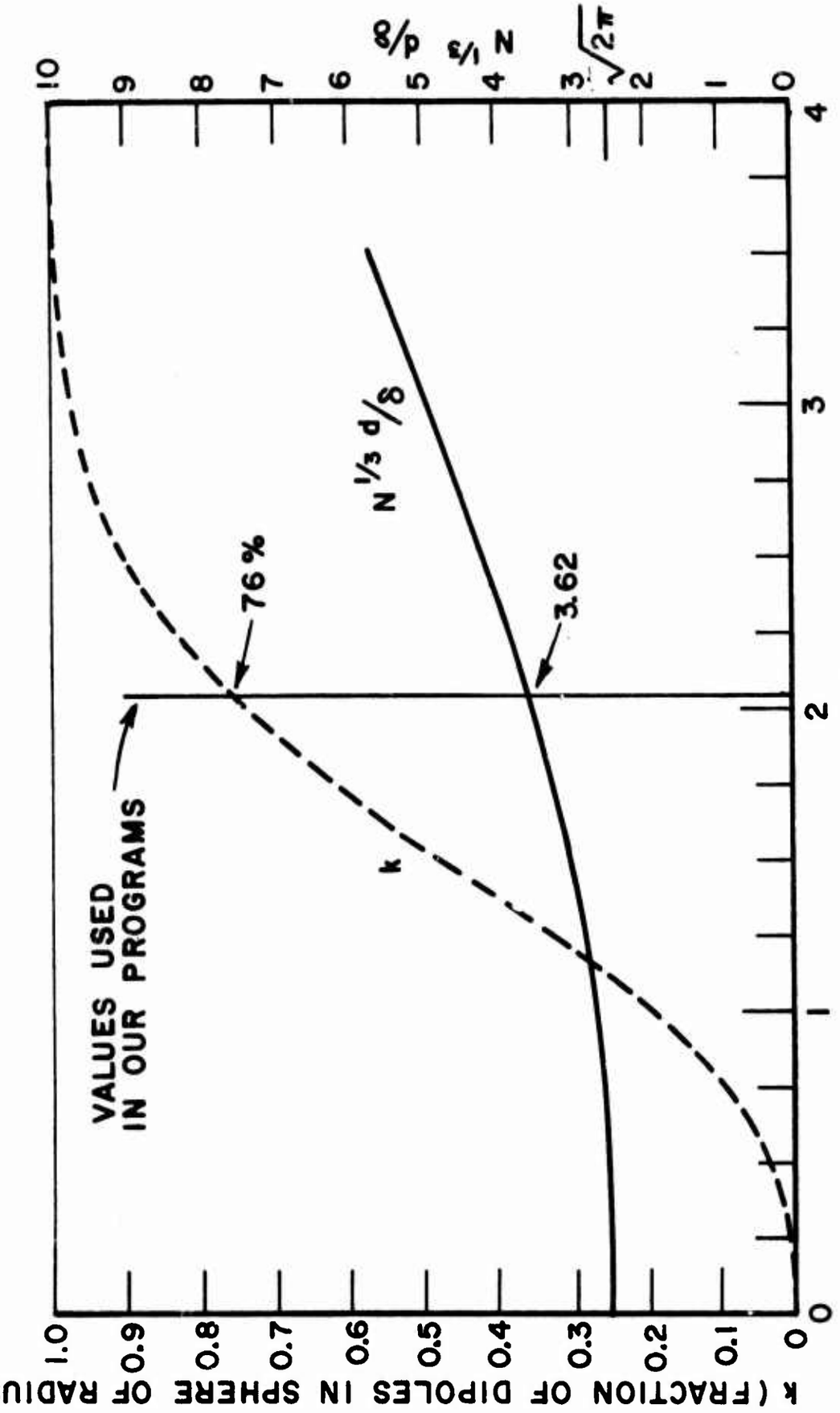

k (FRACTION OF DIPOLES IN SPHERE OF RADIUS R/δ)
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
0
1
2
3
4
R/δ
δ/p
N^1/3 d/δ
3 √2π
N^1/3
0
1
2
3
4
5
6
7
8
9
10
3.62
76%
k
VALUES USED
IN OUR PROGRAMS

Figure III-1.

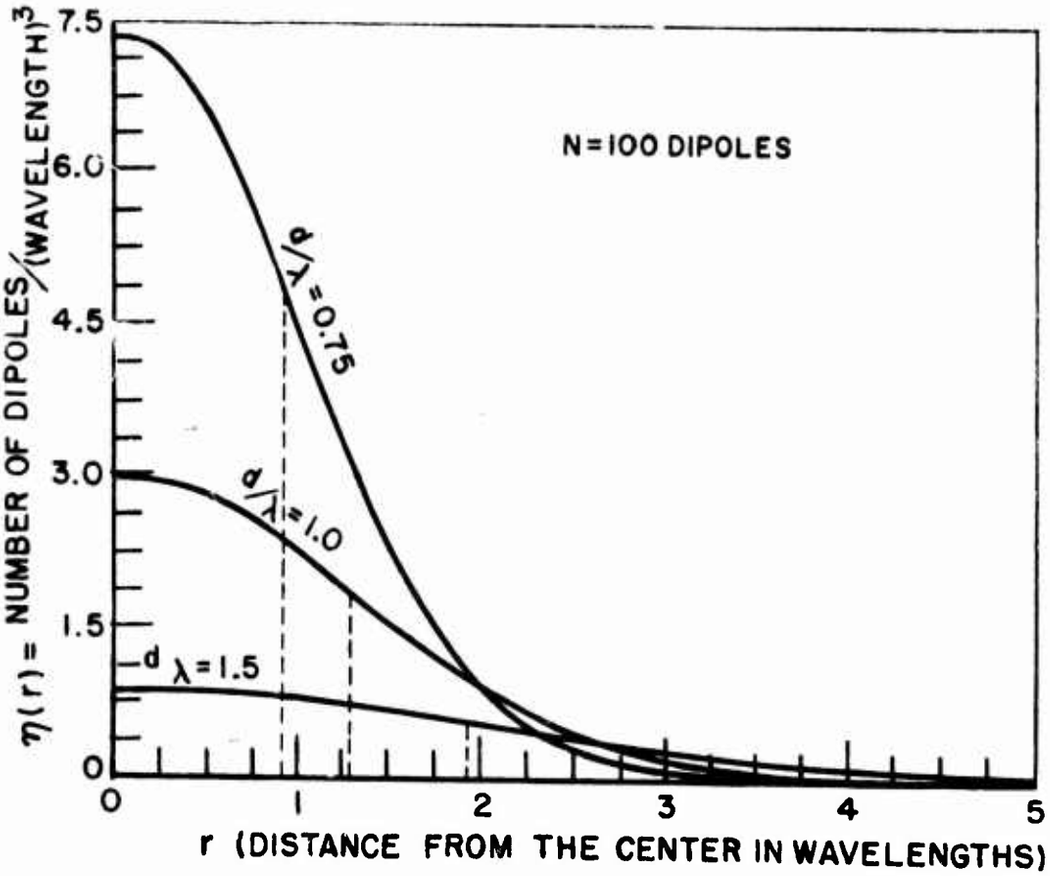

N=100 DIPOLES
$\eta(r)$ = NUMBER OF DIPOLES/(WAVELENGTH)^3
7.5
6.0
4.5
3.0
1.5
0
$d/\lambda = 0.75$
$d/\lambda = 1.0$
$d/\lambda = 1.5$
0
1
2
3
4
5
r (DISTANCE FROM THE CENTER IN WAVELENGTHS)

Figure III-2.

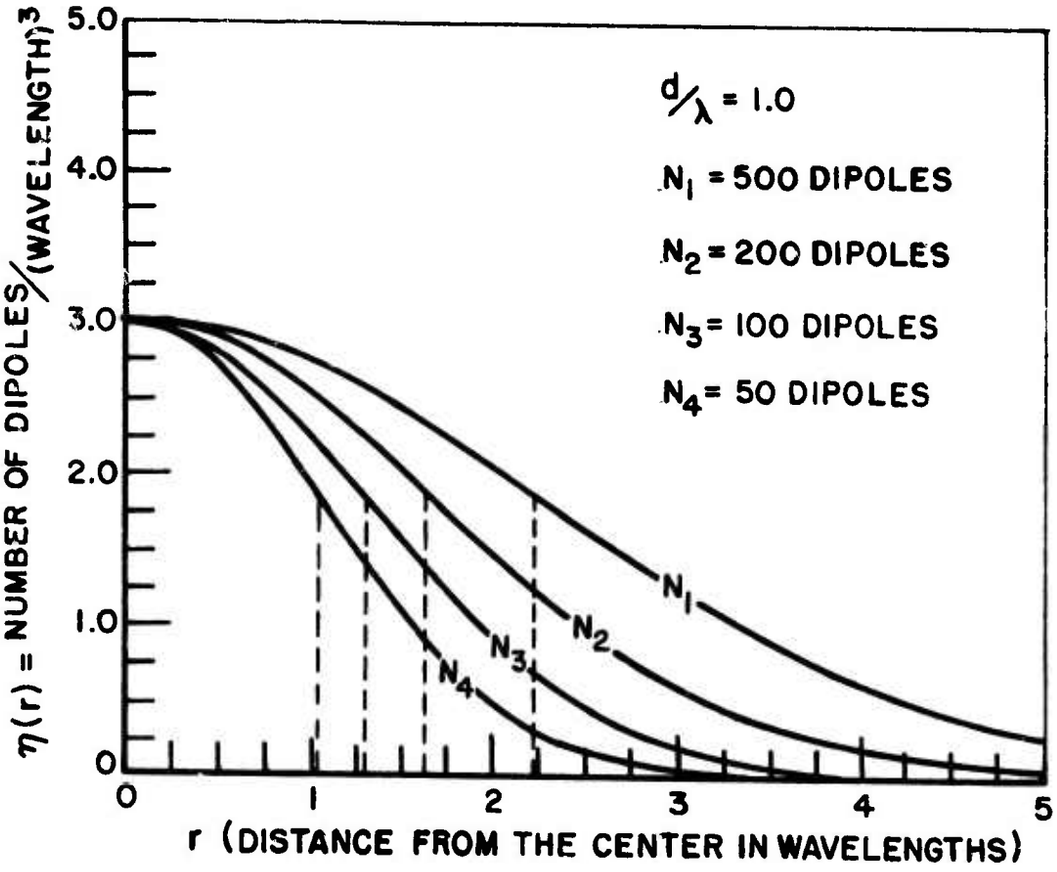

5.0
4.0
3.0
2.0
1.0
0
η(r) = NUMBER OF DIPOLES/(WAVELENGTH)³
d/λ = 1.0
N1 = 500 DIPOLES
N2 = 200 DIPOLES
N3 = 100 DIPOLES
N4 = 50 DIPOLES
N1
N2
N3
N4
0
1
2
3
4
5
r (DISTANCE FROM THE CENTER IN WAVELENGTHS)

Figure III-3.

B. The Homogeneous Cloud

Consider the generation of N randomly distributed points representing center coordinates of N dipoles. If these N points are distributed according to a uniform probability density function and are confined to a spherical volume region V around the origin with an average volume density D, then the radius of V is given by

$$R_o = \left(\frac{3N}{4\pi D}\right)^{1/3} . \tag{III-16}$$

Consider these points to be defined in terms of statistically independent random variables $\tilde{r},\tilde{\theta},\tilde{\phi}$ in the usual spherical coordinate systems. The probability of finding one of these points inside the incremental volume element dv must be given by

$$p(\tilde{r},\tilde{\theta},\tilde{\phi})\, dr\, d\theta\, d\phi = \begin{cases} \frac{3}{4\pi R_o^3}\, \tilde{r}^2 \sin\tilde{\theta}\, dr\, d\theta d\phi, & 0 \le \tilde{r} \le R_o \\ 0, & R_o < \tilde{r} \end{cases} \tag{III-17}$$

to insure these points will be uniformly distributed throughout V. Since the random variables $\tilde{r},\tilde{\theta},\tilde{\phi}$ are statistically independent, the independent probability density functions become

$$p(\tilde{r}) = \frac{3}{R_o^3}\, \tilde{r}^2 , \tag{III-18}$$

$$p(\tilde{\theta}) = \frac{1}{2} \sin\tilde{\theta} \tag{III-19}$$

and

$$p(\tilde{\phi}) = \frac{1}{2\pi} . \tag{III-20}$$

The two angular density functions above can be computed in terms of direction cosines cos α, cosβ and cosγ as follows:

(III-21) $$\cos \tilde{\theta} = 2\tilde{A}(1) - 1$$

(III-22) $$\sin \tilde{\theta} = (1 - \cos^{2}\tilde{\theta})^{1/2}$$

(III-23) $$\tilde{\phi} = 2\pi \tilde{A}(z)$$

(III-24) $$\cos \alpha - \sin \tilde{\theta} \cos \tilde{\phi}$$

(III-25) $$\cos \beta = \sin \tilde{\theta} \sin \tilde{\phi}$$

(III-26) $$\cos \gamma = \cos \tilde{\theta}$$

where the $\tilde{A}(i)$'s are obtained by independent calls to IBM-SSP subroutine RANDU: $\tilde{A}(i)$, i=1,2,... forms a sequence of uniformly distributed psuedo random numbers in the range $0 \leq A(i) \leq 1$. The properly distributed radial variable is given by

(III-27) $$\tilde{r} = R_{o}(\tilde{A}(3))^{1/3}$$

where $\tilde{A}(3)$ corresponds to another call to RANDU. Finally, orientations of the N dipoles are each chosen independently according to the same sequence of Eqs. (III-21) to (III-26), again using independent calls to RANDU. Once the midpoints and orientations are specified, this fully specifies the modeled chaff cloud used here.

APPENDIX D

FULL MATRIX COMPUTER PROGRAM FOR MULTIPLE LENGTHS

In Reference 12, Appendix II is presented a computer program for full matrix solution (scrout) of chaff clouds with single length elements. This appendix presents a program (still using scrout) extended in two ways: it permits the analysis of clouds containing three different element lengths in any combination of numbers and lengths; and it utilizes improved algorithms for obtaining the elements z_{mn} of the impedance matrix.

The computer program in this appendix is used to calculate the random backscattering cross section of "ND" randomly distributed dipoles. These dipoles form three groups and each group has a different dipole length.

Since the dipoles are randomly distributed, one can assume that dipole No. 1 through No. N1 are in group 1 with length DL1, dipole No. N1+1 through N2 are in group 2 with length DL2, and dipole No. N2+1 through ND are in group 3 with length DL3. Dipoles within each group are further divided into segments according to the accuracy desired. Segmentation for dipoles in each group are denoted by NOS1, NOS2, and NOS3. Set NOS1 equal to 3 means all the dipoles in group 1 are divided into 3 segments, etc. If DL1=DL2=DL3 and $N_1=N/3$, $N_{2/3}$, the cloud is made up of N identical dipoles.

All the input parameters for this program are specified as follows:

1. N1: last dipole number in group 1.
2. N2: last dipole number in group 2.
3. ND: last dipole number (which is identical to the total number of dipoles) in group 3.
4. DL1: dipole length (in wavelengths) for group 1.
5. DL2: dipole length (in wavelengths) for group 2.
6. DL3: dipole length (in wavelengths) for group 3.
7. NOS1: segmentation used for dipoles in group 1.
8. NOS2: segmentation used for dipoles in group 2.
9. NOS3: segmentation used for dipoles in group 3.
10. INT: integration sampling constant (usually 10)
11. AL: wire radius for all the dipoles.
12. NSETS: number of clouds to be studied.
13. Spc: average spacing between dipoles.
14. IZ: starting point of the random generator.

This program is set up to plot the echo (in dB) for DBPP, DBTT, and DBTP. One can easily obtain the following quantities as defined in previous Report 3401-1: AVTT, AVPP, AVTP, AV11, VARTT, VARTP, VARPP and VAR11 using the outputs (from SUBROUTINE BKCD) ECTT, ECTP and FCPP.

```
      OPTIONS 32K
      COMPLEX C(5050),ETT(100),EPP(100),S(100)
      DIMENSION DBTT(360),DBPP(360),DBTP(360)
      DIMENSION CA(30),CB(30),CG(30),X(30),Y(30),Z(30)
      DIMENSION XX(100),YY(100),ZZ(100),CCA(100),CCB(100),CCG(100)
      DIMENSION HL123(3),HK(100)
      DATA PI/3.141592/
      DATA IDM,MAXND/100,30/
      IDN=(IDM*IDM-IDM)/2+IDM
      TP=2.*PI
      DR=0.01745329
      ALMDA2=((1.8/.475)*0.0254)**2
      ALMDA2=1.0
      READ(8,*) DL1,DL2,DL3,NOS1,NOS2,NOS3,N1,N2,ND,INT,AL,NSETS
      MODE1=(NOS1-1)*N1
      MODE2=(NOS2-1)*(N2-N1)
      MODE3=(NOS3-1)*(ND-N2)
      NMODE=MODE1+MODE2+MODE3
      INC=(NMODE*NMODE-NMODE)/2+NMODE
      READ(8,*) SPC,IZ
      STDX=SPC/2.05/(4.*3.141592/(3.*.76*ND))**(1./3.)
      STDY=STDX
      STDZ=STDX
      DO 80 NSET=1,NSETS
      IX=IZ
      CALL CLDGEO(ND,STDX,STDY,STDZ,IX,X,Y,Z,CA,CB,CG)
      CALL CLDMOD(DL1,DL2,DL3,NOS1,NOS2,NOS3,N1,N2,ND,IDM
     2,X,Y,Z,CA,CB,CG,XX,YY,ZZ,CCA,CCB,CCG,HK,HL123)
C     WRITE(6,2) ((I,XX(I),YY(I),ZZ(I)),I=1,NMODE)
    2 FORMAT(5X,I5,3E15.4)
      CALL ZIJ(NMODE,XX,YY,ZZ,CCA,CCB,CCG,HK,HL123,AL,INT,
     1IDM,MODE1,MODE2,C,IDN)
C     WRITE(8,3) ((I,C(I)),I=1,INC)
    3 FORMAT(5X,I5,2E15.4)
      CALL SQROT1(C,NMODE,IDN)
      PH=0.0
      CPH=1.0
      SPH=0.0
      NPHI=360
      DPH=1.0
      XNTT=-1000.0
      XNTP=-1000.0
      XNPP=-1000.0
      AVTT=0.0
      AVPP=0.0
      DO 66 NPH=1,NPHI
      CALL BKCU(CPH,SPH,0.0,ECTT,ECTP,ECPP,
     2XX,YY,ZZ,CCA,CCB,CCG,HK,NMODE,IDM,C,ETT,EPP,S,IDN)
      GO TO 77
      IF(ECTT.LT.0.000000001) ECTT=0.00000001
      IF(ECTP.LT.0.000000001) ECTP=0.00000001
      IF(ECPP.LT.0.000000001) ECPP=0.00000001
      DBTT(NPH)=10.*ALOG10(ECTT*ALMDA2)
      DBTP(NPH)=10.*ALOG10(ECTP*ALMDA2)
      DBPP(NPH)=10.*ALOG10(ECPP*ALMDA2)
```

```
      IF(XNTT.LT.DBTT(NPH)) XNTT=DBTT(NPH)
      IF(XNTP.LT.DBTP(NPH)) XNTP=DBTP(NPH)
      IF(XNPP.LT.DBPP(NPH)) XNPP=DBPP(NPH)
77    CONTINUE
      AVTT=AVTT+ECTT
      AVPP=AVPP+ECPP
      PH=PH+DPH
      PHR=PH*DR
      CPH=COS(PHR)
      SPH=SIN(PHR)
 66   CONTINUE
      AV11=(AVTT+AVPP)/FLOAT(NPHI)/2.
      WRITE(8,-) SPC,AV11
      GO TO 2000
      DO 999 NPH=1,NPHI
      DBTT(NPH)=DBTT(NPH)-XNTT
      DBTP(NPH)=DBTP(NPH)-XNTP
      DBPP(NPH)=DBPP(NPH)-XNPP
999   CONTINUE
      WRITE(8,1000)
      READ(8,-) ICC
      CALL PLOT1(N1,N2,ND,DL1,DL2,DL3,NOS1,NOS2,NOS3,
     2INT,ICC,DBPP,IZ,XNPP,STDX)
      WRITE(8,1000)
      READ(8,-) ICC
      CALL PLOT1(N1,N2,ND,DL1,DL2,DL3,NOS1,NOS2,NOS3,
     2INT,ICC,DBTT,IZ,XNTT,STDX)
      WRITE(8,1000)
      READ(8,-) ICC
      CALL PLOT1(N1,N2,ND,DL1,DL2,DL3,NOS1,NOS2,NOS3,
     2INT,ICC,DBTP,IZ,XNTP,STDX)
1000  FORMAT(5X,'READY TO PLOT?,1,3,2')
2000  CONTINUE
      IZ=IZ*8709
      IF(IZ) 76,80,80
 76   CONTINUE
      IZ=IZ*8388607+1
 80   CONTINUE
      CALL EXIT
      END
      SUBROUTINE CLDMOD(DL1,DL2,DL3,NOS1,NOS2,NOS3,N1,N2,ND,IDM
     2,X,Y,Z,CA,CB,CG,XX,YY,ZZ,CCA,CCB,CCG,HK,HL123)
      DIMENSION X(1),Y(1),Z(1),CA(1),CB(1),CG(1),XX(1),YY(1),ZZ(1)
      DIMENSION CCA(1),CCB(1),CCG(1),HK(1)
      DIMENSION NM123(3),IL123(3),IU123(3),DL123(3),HL123(3)
      TP=2.*3.141592
      DL123(1)=DL1
      DL123(2)=DL2
      DL123(3)=DL3
      HL123(1)=DL1/NOS1
      HL123(2)=DL2/NOS2
      HL123(3)=DL3/NOS3
      NM123(1)=NOS1-1
      NM123(2)=NOS2-1
      NM123(3)=NOS3-1
```

```
      IL123(1)=1
      IL123(2)=1+N1
      IL123(3)=1+N2
      IU123(1)=N1
      IU123(2)=N2
      IU123(3)=ND
      KK=3
      IF(ND.LT.3) KK=ND
      DO 2 K=1,KK
      IA=IL123(K)
      IB=IU123(K)
      DO 1 I=IA,IB
      NM=NM123(K)
      DO 1 II=1,NM
      L=II+(I-IA)*NM
      IF(K.GT.1) L=L+IR
      XX(L)=X(1)-(DL123(K)*0.5-FLOAT(II)*HL123(K))*TP*CA(I)
      YY(L)=Y(1)-(DL123(K)*0.5-FLOAT(II)*HL123(K))*TP*CB(I)
      ZZ(L)=Z(1)-(DL123(K)*0.5-FLOAT(II)*HL123(K))*TP*CG(I)
      CCA(L)=CA(I)
      CCB(L)=CB(I)
      CCG(L)=CG(I)
      HK(L)=HL123(K)*TP
    1 CONTINUE
      IR=L
    2 CONTINUE
      RETURN
      END
      SUBROUTINE CLOGEO(N,STDX ,STDY ,STDZ ,IZ,X,Y,Z,CA,CB,CG)
      DIMENSION X(1),Y(1),Z(1),CA(1),CB(1),CG(1)
      DATA PI/3.141592/
      TP=2.*PI
      IX=IZ
      STDXK=STDX*TP
      STDYK=STDY*TP
      STDZK=STDZ*TP
      DO 30 I=1,N
      CALL GAUSS(IX,STDXK,0.0,X(I))
      CALL GAUSS(IX,STDYK,0.0,Y(I))
      CALL GAUSS(IX,STDZK,0.0,Z(I))
      CALL RANDU(IX,IY,A1)
      IX=IY
      PHI=TP*A1
      CALL RANDU(IX,IY,A2)
      IX=IY
      COSTH=2.*A2-1.0
      SINTH=SQRT(1.-COSTH*COSTH)
      CA(I)=SINTH*COS(PHI)
      CB(I)=SINTH*SIN(PHI)
      CG(I)=COSTH
   30 CONTINUE
      RETURN
      END
      SUBROUTINE GAUSS(IX,S,AM,V)
      A=0.0
```

```
      DO 50 I=1,12
      CALL RANDU(IX,IY,Y)
      IX=IY
   50 A=A+Y
      V=(A-6.0)*S+AM
      RETURN
      END
      SUBROUTINE RANDU(IX,IY,YFL)
      IY=IX*16645
      IF(IY)5,6,6
    5 IY=IY+8388607+1
    6 YFL=IY
      YFL=YFL*.1192093E-6
      RETURN
      END
      SUBROUTINE ZIJ(N,XX,YY,ZZ,CCA,CCB,CCG,HK,HL123,AL,INT,
     1IDM,MODE1,MODE2,C,IDN)
      COMPLEX CAA,CBB,CCC
      COMPLEX P11,P12,P21,P22,ZMN,CIJ,C(1)
      DIMENSION XX(1),YY(1),ZZ(1),CCA(1),CCB(1),CCG(1),HK(1)
      DIMENSION HL123(3)
      DATA PI/3.141592/
      TP=2.*PI
      AK=AL*TP
      CAA=ZMN(AL,HL123(1),0,0)
      CBB=ZMN(AL,HL123(2),0,0)
      CCC=ZMN(AL,HL123(3),0,0)
      DO 40 I=1,N
      II=(I-1)*N-(I*I-I)/2+I
      IF(I.LE.MODE1) C(II)=CAA
      IF(I.GT.MODE1.AND.I.LE.(MODE1+MODE2)) C(II)=CBB
      IF(I.GT.(MODE1+MODE2)) C(II)=CCC
   40 CONTINUE
      N1=N-1
      IF(N1.LT.1) N1=1
      DO 45 I=1,N1
      DS=HK(I)
      COS=COS(DS)
      SDS=SIN(DS)
      X1=XX(I)-DS*CCA(I)
      Y1=YY(I)-DS*CCB(I)
      Z1=ZZ(I)-DS*CCG(I)
      X2=XX(I)
      Y2=YY(I)
      Z2=ZZ(I)
      X3=XX(I)+DS*CCA(I)
      Y3=YY(I)+DS*CCB(I)
      Z3=ZZ(I)+DS*CCG(I)
      ID=(I-1)*N-(I*I-I)/2
      IP=I+1
      IF(IP.GT.N) RETURN
      DO 45 J=IP,N
      DT=HK(J)
      SDT=SIN(DT)
      IJ=ID+J
```

```
      XA=XX(J)-DT*CCA(J)
      YA=YY(J)-DT*CCB(J)
      ZA=ZZ(J)-DT*CCG(J)
      XB=XX(J)
      YB=YY(J)
      ZB=ZZ(J)
      XC=XX(J)+DT*CCA(J)
      YC=YY(J)+DT*CCB(J)
      ZC=ZZ(J)+DT*CCG(J)
      CIJ=(0.0,0.0)
      CALL ZGS(X1,Y1,Z1,X2,Y2,Z2,XA,YA,ZA,XB,YB,ZB,
     1AK,DS,CDS,SDS,DT,SDI,INT,P11,P12,P21,P22)
      CIJ=CIJ+P22
      CALL ZGS(X1,Y1,Z1,X2,Y2,Z2,XB,YB,ZB,XC,YC,ZC,
     1AK,DS,CDS,SDS,DT,SDI,INT,P11,P12,P21,P22)
      CIJ=CIJ+P21
      CALL ZGS(X2,Y2,Z2,X3,Y3,Z3,XA,YA,ZA,XB,YB,ZB,
     1AK,DS,CDS,SDS,DT,SDI,INT,P11,P12,P21,P22)
      CIJ=CIJ+P12
      CALL ZGS(X2,Y2,Z2,X3,Y3,Z3,XB,YB,ZB,XC,YC,ZC,
     1AK,DS,CDS,SDS,DT,SDI,INT,P11,P12,P21,P22)
      C(IJ)=CIJ+P11
   45 CONTINUE
      RETURN
      END
      SUBROUTINE ZFFD(X,Y,Z,CA,CB,CG,CTH,STH,CPH,SPH,
     2SDK,CDK,HK  ,FT,EP)
      COMPLEX ET,EP,EJB,ES
      G=(CA*CPH+CB*SPH)*STH+CG*CTH
      GK=1.-G*G
      ET=(0.0,0.0)
      EP=(0.0,0.0)
      IF(GK.LT.0.001) GO TO 200
      B=(X*CPH+Y*SPH)*STH+Z*CTH
      EJB=CMPLX(COS(B),SIN(B))
      ES=(0.0,60.0)*EJB*(CDK-COS(G*HK))/GK/SDK
      T=(CA*CPH+CB*SPH)*CTH-CG*STH
      P=-CA*SPH+CB*CPH
      ET=T*ES
      EP=P*ES
  200 CONTINUE
      RETURN
      END
      SUBROUTINE BKCD(CPH,SPH,CTH,ECTT,ECTP,ECPP,
     2X,Y,Z,CA,CB,CG,HK,N,IDM,C,ETT,EPP,S,IDN)
      DIMENSION X(1),Y(1),Z(1),CA(1),CB(1),CG(1),HK(1)
      COMPLEX   C(1),ETT(1),EPP(1),S(1),ETH,EPH
      DATA PI/3.141592/
      TP=2.*PI
      STH=SQRT(1.-CTH*CTH)
      DO 70 I=1,N
      SDK=SIN(HK(I))
      CDK=COS(HK(I))
      CALL ZFFD(X(I),Y(I),Z(I),CA(I),CB(I),CG(I),CTH,STH,CPH,SPH,
     2SDK,CDK,HK(I)  ,ETT(I),EPP(I))
```

```
      S(I)=ETT(I)*(0.0,1.0)/TP/30.0
   70 CONTINUE
      CALL SQROT2(C,S,N,IDM,IDN)
      ETH=(0.0,0.0)
      EPH=(0.0,0.0)
      DO 80 I=1,N
      ETH=ETH+S(I)*ETT(I)
      EPH=EPH+S(I)*EPP(I)
   80 CONTINUE
      CETH=CABS(ETH)
      CEPH=CABS(EPH)
      ECTT=2.0*TP*CETH*CETH
      ECTP=2.0*TP*CEPH*CEPH
      DO 90 I=1,N
      S(I)=EPP(I)*(0.0,1.0)/TP/30.0
   90 CONTINUE
      CALL SQROT2(C,S,N,IDM,IDN)
      EPH=(0.0,0.0)
      DO 100 I=1,N
      EPH=EPH+S(I)*EPP(I)
  100 CONTINUE
      CEPH=CABS(EPH)
      ECPP=2.0*TP*CEPH*CEPH
      RETURN
      END
      SUBROUTINE PLOT1(NA,NB,NC,DL1,DL2,DL3,NOS1,NOS2,NOS3,
     2INT,ICASE,F,IX,XNORM,STDX)
      DIMENSION LX(9)
      DIMENSION IBUF(100),L1(4),L2(5),L3(6),LD(1),LINT(2),
     2LNOS(2),LN(7),LPHI(5),LL(2),LLAMDA(3),F(360),X(360)
      DATA L1,L2,L3/12H PHI-PHI RCS,15H  THETA-PHI RCS,
     218H   THETA-THETA RCS/
      DATA LLAMDA,LINT,LNOS/9H  LAMDA, ,6H  INT=,6H  NOS=/
      DATA LN,LPHI/21H DIPOLE RANDOM CLOUD, ,15H   PHI(DEGREES)/
      DATA LL,LD/6H  OBSW,3H L=/
      DATA LX/25H NORM FACTOR=           DB/
      CALL PLOTS(IBUF, 100, 3)
      CALL PLOT(0.0,0.0,-3)
      CALL AXIS(0.0,1.5,LPHI,-15,15.0,0.0,0.0,24.0,1.25,-1)
      CALL AXIS(0.0,1.5,LL,+6,6.25,90.0,-40.0,8.0,1.25,-1)
      CALL PLOT(0.0,7.75,3)
      CALL PLOT(15.0,7.75,2)
      CALL PLOT(15.0,1.5,2)
      YH=8.25
      W=0.2
      CALL NUMBER(0.1,7.50,0.15,FLOAT(IX),0.0,-1)
      CALL SYMBOL(4.75,7.0,.15,LX,0.0,25)
      CALL NUMBER(7.00,7.0,.15,XNORM,0.0,+2)
      SPC=STDX*2.05*(4.*3.141592/(3.*.76*NC))**(1./3.)
      CALL NUMBER(13.6,7.50,0.15,STDX,0.0,+4)
      CALL NUMBER(13.6,7.20,0.15,SPC,0.0,+4)
      CALL NUMBER(0.8,8.55,.15,FLOAT(NA),0.0,-1)
      CALL NUMBER(0.8,8.25,.15,FLOAT(NB),0.0,-1)
      CALL NUMBER(0.8,7.95,.15,FLOAT(NC),0.0,-1)
      CALL SYMBOL(1.50,YH,W,LN,0.0,21)
```

```
      CALL SYMBOL(5.75,YH,W,LD,0.0,3)
      CALL NUMBER(6.35,8.55,.15,DL1,0.0,+3)
      CALL NUMBER(6.35,8.25,.15,DL2,0.0,+3)
      CALL NUMBER(6.35,7.95,.15,DL3,0.0,+3)
      CALL SYMBOL(7.20,YH,W,LLAMDA,0.0,9)
      CALL SYMBOL(8.5,YH,W,LINT,0.0,6)
      CALL NUMBER(9.78,YH,W,FLOAT(INT),0.0,-1)
      CALL SYMBOL(10.4,YH,W,LNOS,0.0,6)
      CALL NUMBER(11.6,8.55,.15,FLOAT(NOS1),0.0,-1)
      CALL NUMBER(11.6,8.25,.15,FLOAT(NOS2),0.0,-1)
      CALL NUMBER(11.6,7.95,.15,FLOAT(NOS3),0.0,-1)
      IF(ICASE.EQ.1) CALL SYMBOL(11.8,YH,W,L1,0.0,12)
      IF(ICASE.EQ.2) CALL SYMBOL(11.8,YH,W,L2,0.0,15)
      IF(ICASE.EQ.3) CALL SYMBOL(11.8,YH,W,L3,0.0,18)
      CALL PLOT(0.0,1.5,-3)
      DO 20 I=1,360
      IF(F(I).LT.-40.) F(I)=-40.
   20 X(I)=I
      CALL LINE(X,0.0,24.0,F,-40.,8.0,360,0.52)
      CALL PLOT(17.0,-1.5, 999)
      RETURN
      END
      SUBROUTINE SQROT1(C,N,IDN)
C*****************************************************************************
C*                                                                           *
C*       PURPOSE                                                             *
C*          TO TRANSFORM A SYMMETRIC MATRIX INTO AN AUXILIARY                *
C*          MATRIX (IMPLICIT INVERSE)                                        *
C*                                                                           *
C*       USAGE                                                               *
C*          CALL SQROT1(C,N,IDN)                                             *
C*                                                                           *
C*       DESCRIPTION OF PARAMETERS                                           *
C*          C         -  THE ARRAY CONTAINING THE MATRIX IN COMPRESSED       *
C*                       FORM ON ENTRY AND ITS AUXILIARY IN COMPRESSED       *
C*                       FORM ON EXIT                                        *
C*          N         -  THE NUMBER OF ROWS OR COLUMNS IN THE MATRIX         *
C*          IDN       -  THE DIMENSION OF THE ARRAY C                        *
C*                                                                           *
C*       REMARKS                                                             *
C*          THE UPPER TRIANGLE OF THE MATRIX IS STORED BY ROWS IN THE        *
C*          ARRAY C. ONE DIMENSIONAL SUBSCRIPTS ARE RELATED TO               *
C*          CORRESPONDING TWO DIMENSIONAL SUBSCRIPTS BY                      *
C*                   IJ=(I-1)*N-(I*I-I)/2+J                                  *
C*          WHERE IJ IS THE ONE DIMENSIONAL SUBSCRIPT AND I AND J            *
C*          ARE THE TWO DIMENSIONAL SUBSCRIPTS                               *
C*                                                                           *
C*       METHOD                                                              *
C*          "SQUARE ROOT" METHOD FOR SOLUTION OF A SYMMETRIC MATRIX          *
C*          EQUATION. THE ORIGINAL SYMMETRIC MATRIX M AND THE UPPER          *
C*          TRIANGULAR AUXILIARY MATRIX A ARE RELATED BY                     *
C*                   M=TRANSPOSE(A)*A                                        *
C*                                                                           *
C*       REFERENCES                                                          *
C*          FADDEEV, D. K. AND FADDEEVA, V. N., COMPUTATIONAL                *
```

```
C*          METHODS OF LINEAR ALGEBRA, W. H. FREEMAN AND CO., SAN      *
C*          FRANCISCO, 1963, P. 144-147                                *
C*                                                                     *
C***********************************************************************
      COMPLEX C(IDN)
      C(1)=CSQRT(C(1))
      DO 1 K=2,N
    1 C(K)=C(K)/C(1)
      DO 2 I=2,N
      IMO=I-1
      IPO=I+1
      ID=(I-1)*N-(I*I-I)/2
      II=ID+I
      DO 3 L=1,IMO
      LI=(L-1)*N-(L*L-L)/2+I
    3 C(II)=C(II)-C(LI)*C(LI)
      C(II)=CSQRT(C(II))
      IF(IPO.GT.N)GO TO 2
      DO 5 J=IPO,N
      IJ=ID+J
      DO 6 M=1,IMO
      MD=(M-1)*N-(M*M-M)/2
      MI=MD+I
      MJ=MD+J
    6 C(IJ)=C(IJ)-C(MJ)*C(MI)
    5 C(IJ)=C(IJ)/C(II)
    2 CONTINUE
      RETURN
      END
      SUBROUTINE SQROT2(C,S,N,IDM,IDN)
C***********************************************************************
C*                                                                     *
C*       PURPOSE                                                       *
C*          TO OBTAIN A SOLUTION TO THE SYMMETRIC MATRIX EQUATION      *
C*          MX=Y USING THE AUXILIARY OF M CALCULATED BY SQROT1         *
C*                                                                     *
C*       USAGE                                                         *
C*          CALL SQROT2(C,S,N,IDM,IDN)                                 *
C*                                                                     *
C*       DESCRIPTION OF PARAMETERS                                     *
C*          C        -  AN ARRAY CONTAINING THE UPPER TRIANGULAR       *
C*                      AUXILIARY MATRIX IN COMPRESSED FORM            *
C*          S        -  AN ARRAY CONTAINING THE RIGHT HAND SIDE VECTOR *
C*                      OF THE EQUATION ON ENTRY AND THE SOLUTION      *
C*                      VECTOR ON EXIT                                 *
C*          N        -  THE NUMBER OF SIMULTANEOUS EQUATIONS           *
C*          IDM      -  THE DIMENSION OF THE ARRAY S                   *
C*          IDN      -  THE DIMENSION OF THE ARRAY C                   *
C*                                                                     *
C*       REMARKS                                                       *
C*          THE UPPER TRIANGLE OF THE AUXILIARY MATRIX IS STORED BY    *
C*          ROWS IN THE ARRAY C. ONE DIMENSIONAL SUBSCRIPTS ARE        *
C*          RELATED TO CORRESPONDING TWO DIMENSIONAL SUBSCRIPTS BY     *
C*                   IJ=(I-1)*N-(I*I-I)/2+J                            *
C*          WHERE IJ IS THE ONE DIMENSIONAL SUBSCRIPT AND I AND J      *
```

```
C*          ARE THE TWO DIMENSIONAL SUBSCRIPTS                              *
C*                                                                          *
C*          SQROT1 MUST BE CALLED BEFORE THE FIRST ENTRY TO SQROT2          *
C*                                                                          *
C*       METHOD                                                             *
C*          "SQUARE ROOT" METHOD FOR SOLUTION OF A SYMMETRIC MATRIX         *
C*          EQUATION. THE ORIGINAL SYMMETRIC MATRIX M AND THE UPPER         *
C*          TRIANGULAR AUXILIARY MATRIX A ARE RELATED BY                    *
C*                M=TRANSPOSE(A)*A                                          *
C*                                                                          *
C*       REFERENCES                                                         *
C*          FADDEEV, D. K. AND FADDEEVA, V. N., COMPUTATIONAL               *
C*          METHODS OF LINEAR ALGEBRA, W. H. FREEMAN AND CO., SAN           *
C*          FRANCISCO, 1963, P. 140-147                                     *
C*                                                                          *
C****************************************************************************
      COMPLEX S(100),C(100)
      S(1)=S(1)/C(1)
      DO 10 I=2,N
      IM1=I-1
      DO 11 L=1,IM1
      LI=(L-1)*N-(L*L-L)/2+I
   11 S(I)=S(I)-C(LI)*S(L)
      II=(I-1)*N-(I*I-I)/2+I
   10 S(I)=S(I)/C(II)
      NN=((N+1)*N)/2
      S(N)=S(N)/C(NN)
      N1O=N-1
      DO 25 I=1,N1O
      K=N-I
      KP1=K+1
      KD=(K-1)*N-(K*K-K)/2
      DO 26 L=KP1,N
      KL=KD+L
   26 S(K)=S(K)-C(KL)*S(L)
      KK=KD+K
      S(K)=S(K)/C(KK)
   25 CONTINUE
      RETURN
      END
      SUBROUTINE SICI(SI,CI,X)                                          SICI000
C*************************************************************************** SICI001
C*                                                                       * SICI002
C*       PURPOSE                                                         * SICI003
C*          COMPUTES THE SINE AND COSINE INTEGRALS                       * SICI004
C*                                                                       * SICI005
C*       USAGE                                                           * SICI006
C*          CALL SICI(SI,CI,X)                                           * SICI007
C*                                                                       * SICI008
C*       DESCRIPTION OF PARAMETERS                                       * SICI009
C*          SI     - THE RESULTANT VALUE SI(X)                           * SICI010
C*          CI     - THE RESULTANT VALUE CI(X)                           * SICI011
C*          X      - THE ARGUMENT OF SI(X) AND CI(X)                     * SICI012
C*                                                                       * SICI013
C*       REMARKS                                                         * SICI014
```

```
C*           THE ARGUMENT VALUE REMAINS UNCHANGED                              * SICI015
C*                                                                             * SICI016
C*        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                        * SICI017
C*           NONE                                                              * SICI018
C*                                                                             * SICI019
C*        METHOD                                                               * SICI020
C*           DEFINITION                                                        * SICI021
C*           SI(X)=INTEGRAL(SIN(T)/T)                                          * SICI022
C*           CI(X)=INTEGRAL(COS(T)/T)                                          * SICI023
C*           EVALUATION                                                        * SICI024
C*           REDUCTION OF RANGE USING SYMMETRY                                 * SICI025
C*           DIFFERENT APPROXIMATIONS ARE USED FOR ABS(X) GREATER              * SICI026
C*           THAN 4 AND FOR ABS(X) LESS THAN 4.                                * SICI027
C*                                                                             * SICI028
C*        REFERENCES                                                           * SICI029
C*           IBM SCIENTIFIC SUBROUTINE PACKAGE P. 370                          * SICI030
C*           LUKE AND WIMP, 'POLYNOMIAL APPROXIMATIONS TO INTEGRAL             * SICI031
C*           TRANSFORMS', MATHEMATICAL TABLES AND OTHER AIDS TO                * SICI032
C*           COMPUTATION, VOL. 15, 1961, ISSUE 74, P. 174-178                  * SICI033
C                                                                              * SICI034
C*******************************************************************************SICI035
      Z=ABS(X)                                                                   SICI036
      IF(Z-4.0)3,3,4                                                             SICI037
    1 Y=(4.0-Z)*(4.0+Z)                                                          SICI038
      S1=-1.577797E0                                                           SICI039
      IF(Z)5,2,5                                                               SICI040
    2 C1=-1.0E38                                                               SICI041
      RETURN                                                                   SICI042
    3 S1=X*(((((1.753141E-9*Y+1.568908E-7)*Y+1.374160E-5)*Y+8.939689E-4)       SICI043
     2*Y+1.964082E-2)*Y+4.995509E-1+S1/Y)                                      SICI044
      C1=((5.772156E-1+ALOG(Z))/Z-Z*(((((1.080985E-10*Y+1.584956E-8)*Y         SICI045
     2+1.725752E-6)*Y+1.185999E-4)*Y+4.599920E-3)*Y+1.555308E-1))*Z            SICI046
      RETURN                                                                   SICI047
    4 S1=SIN(Z)                                                                SICI048
      Y=COS(Z)                                                                 SICI049
      Z=4.0/Z                                                                  SICI050
      U=(((((((((4.048062E-3*Z-2.279143E-2)*Z+5.515070E-2)*Z-7.261642E-2)      SICI051
     2*Z+4.907716E-2)*Z-3.632519E-3)*Z-2.314617E-2)*Z-1.134950E-5)*Z           SICI052
     3+6.250011E-2)*Z+2.568982E-10                                             SICI053
      V=(((((((((-5.106699E-5*Z+2.819179E-2)*Z-6.537285E-2)*Z                  SICI054
     2+7.902634E-2)*Z-4.000416E-2)*Z-7.945556E-3)*Z+2.661293E-2)*Z            SICI055
     3-3.764000E-4)*Z-3.122418E-2)*Z-6.646441E-7)*Z+2.5000005-1                SICI056
      C1=Z*(S1*V-Y*U)                                                          SICI057
      S1=-Z*(S1*U+Y*V)                                                         SICI058
      IF(X)5,6,6                                                               SICI059
    5 S1=-3.141593E0-S1                                                        SICI060
    6 RETURN                                                                   SICI061
      END                                                                      SICI062
      SUBROUTINE ZGS(XA,YA,ZA,XB,YB,ZB,X1,Y1,Z1,X2,Y2,Z2,AR,
     2DS,CDS,SDS,DT,SDT,IMT,P11,P12,P21,P22)
      COMPLEX CST,EJ1,EJ2,EJA,EJB,ER1,ER2,E11,ET2,P11,P12,P21,P22,GAM
      COMPLEX SGDS,SGDT
      DATA ETA,GAM,PI/376.727,(.0,1.),3.14159/
      CA=(X2-X1)/DT
      CB=(Y2-Y1)/DT
```

```
      CG=(Z2-Z1)/DT
      CAS=(XB-XA)/DS
      CBS=(YB-YA)/DS
      CGS=(ZB-ZA)/DS
      CC=CA*CAS+CB*CBS+CG*CGS
      IF(ABS(CC).GT.0.997)GO TO 200
      SZ=(X1-XA)*CAS+(Y1-YA)*CBS+(Z1-ZA)*CGS
      IF(INT.EQ.0)GO TO 300
      CGDS=CDS
      SGDS=CMPLX(.0,SDS)
      SGDT=CMPLX(.0,SDT)
      INS=2*(INT/2)
      IF(INS.LT.2)INS=2
      IP=INS+1
      DELT=DT/INS
      T=.0
      DSZ=CC*DELT
      P11=(.0,.0)
      P12=(.0,.0)
      P21=(.0,.0)
      P22=(.0,.0)
      AKS=AK*AK
      SGN=-1.
      DO 100 IN=1,IP
      ZZ1=SZ
      ZZ2=SZ-DS
      XXZ=X1+T*CA-XA-SZ*CAS
      YYZ=Y1+T*CB-YA-SZ*CBS
      ZZZ=Z1+T*CG-ZA-SZ*CGS
      RS=XXZ**2+YYZ**2+ZZZ**2
      R1=SQRT(RS+ZZ1**2)
      EJA=CMPLX(COS(R1),-SIN(R1))
      EJ1=EJA/R1
      R2=SQRT(RS+ZZ2**2)
      EJB=CMPLX(COS(R2),-SIN(R2))
      EJ2=EJB/R2
      ER1=EJA*SGDS+ZZ1*EJ1*CGDS-ZZ2*EJ2
      ER2=-EJB*SGDS+ZZ2*EJ2*CGDS-ZZ1*EJ1
      FAC=.0
      IF(RS.GT.AKS)FAC=(CA*XXZ+CB*YYZ+CG*ZZZ)/RS
      ET1=CC*(EJ2-EJ1*CGDS)+FAC*ER1
      ET2=CC*(EJ1-EJ2*CGDS)+FAC*ER2
      C=3.+SGN
      IF(IN.EQ.1 .OR. IN.EQ.IP)C=1.
      C1=C*SIN(DT-T)
      C2=C*SIN(T)
      P11=P11+ET1*C1
      P12=P12+ET1*C2
      P21=P21+ET2*C1
      P22=P22+ET2*C2
      T=T+DELT
      SZ=SZ+DSZ
  100 SGN=-SGN
      CST=-(.0,1.)*ETA*DELT/(12.*PI*SGDS*SGDT)
      P11=CST*P11
```

```
      P12=CST*P12
      P21=CST*P21
      P22=CST*P22
      RETURN
  200 SZ1=(X1-XA)*CAS+(Y1-YA)*CBS+(Z1-ZA)*CGS
      RH1=SQRT((X1-XA-SZ1*CAS)**2+(Y1-YA-SZ1*CBS)**2+(Z1-ZA-SZ1*CGS)**2)
      SZ2=SZ1+DT*CC
      RH2=SQRT((X2-XA-SZ2*CAS)**2+(Y2-YA-SZ2*CBS)**2+(Z2-ZA-SZ2*CGS)**2)
      DDK=(RH1+RH2)/2.
      IF(DDK.LT.AK)DDK=AK
      CALL ZGMM(.0,DS,SZ1,SZ2,DDK,CDS,SDS,SDT,1.,P11,P12,P21,P22)
      RETURN
  300 SS=SQRT(1.-CC*CC)
      CAD=(CGS*CB-CBS*CG)/SS
      CBD=(CAS*CG-CGS*CA)/SS
      CGD=(CBS*CA-CAS*CB)/SS
      DK=(X1-XA)*CAD+(Y1-YA)*CBD+(Z1-ZA)*CGD
      DK=ABS(DK)
      IF(DK.LT.AK)DK=AK
      XZ=XA+SZ*CAS
      YZ=YA+SZ*CBS
      ZZ=ZA+SZ*CGS
      XP1=X1-DK*CAD
      YP1=Y1-DK*CBD
      ZP1=Z1-DK*CGD
      CAP=CBS*CGD-CGS*CBD
      CBP=CGS*CAD-CAS*CGD
      CGP=CAS*CBD-CBS*CAD
      P1=CAP*(XP1-XZ)+CBP*(YP1-YZ)+CGP*(ZP1-ZZ)
      T1=P1/SS
      S1=T1*CC-SZ
      CALL ZGMM(S1,S1+DS,T1,T1+DT,DK,CDS,SDS,SDT,CC,P11,P12,P21,P22)
      RETURN
      END
      SUBROUTINE ZGMM(S1,S2,T1,T2,D,CGDS,SGD1,SGD2,CPSI,P11,P12,P21,P22)
      COMPLEX E(2,2),F(2,2),GAM,P11,P12,P21,P22
      COMPLEX EB,EC,EK,EL,EKL,EGZI,ES1,ES2,ET1,ET2,EXPA,EXPB
      COMPLEX EGZ(2,2),GM(2),GP(2)
      COMPLEX EXA(2),EXB(2)
      DATA ETA,GAM,PI/376.727,(.0,1.),3.14159/
      DSQ=D*D
      SGDS=SGD1
      IF(S2.LT.S1)SGDS=-SGD1
      SGDT=SGD2
      IF(T2.LT.T1)SGDT=-SGD2
      IF(ABS(CPSI).GT.0.997)GO TO 110
      ES1=CEXP(GAM*S1)
      ES2=CEXP(GAM*S2)
      ET1=CEXP(GAM*T1)
      ET2=CEXP(GAM*T2)
      C=D/SQRT(1.-CPSI*CPSI)
      B=C*CPSI
      EB=CEXP(GAM*CMPLX(.0,B))
      EC=CEXP(GAM*CMPLX(.0,C))
      DO 10 K=1,2
```

```
      DO 10 L=1,2
   10 F(K,L)=(.0,.0)
      EK=EB
      DO 50 K=1,2
      FK=(-1)**K
      EL=EC
      DO 40 L=1,2
      FL=(-1)**L
      EKL=EK*EL
      XX=FK*B+FL*C
      S1=S1
      DO 30 I=1,2
      R1=SQRT(DSQ+S1*S1+T1*T1-2.*S1*T1*CPSI)
      R2=SQRT(DSQ+S1*S1+T2*T2-2.*S1*T2*CPSI)
      CALL EXPJ(GAM*CMPLX(R1+FK*S1+FL*T1,-XX),
     2          GAM*CMPLX(R2+FK*S1+FL*T2,-XX),EXA(I))
      CALL EXPJ(GAM*CMPLX(R1+FK*S1+FL*T1,XX),
     2          GAM*CMPLX(R2+FK*S1+FL*T2,XX),EXB(I))
      IF(K.EQ.2 .OR. L.EQ.2)GO TO 30
      ZC=S1*CPSI
      EGZI=CEXP(GAM*ZC)
      CALL EXPJ(GAM*(R1+ZC-T1),GAM*(R2+ZC-T2),EXPB)
      CALL EXPJ(GAM*(R1-ZC+T1),GAM*(R2-ZC+T2),EXPA)
      F(I,1)=2.*SGDS*(.0,1.)*EXPA/EGZI
      F(I,2)=2.*SGDS*(.0,1.)*EXPB*EGZI
   30 S1=S2
      E(K,L)=E(K,L)+(EXA(2)-EXA(1))*EKL+(EXB(2)-EXB(1))/EKL
   40 EL=1./EC
   50 EK=1./EB
      CST=-ETA/(16.*PI*SGDS*SGDT)
      P11=CST*(( F(1,1)+E(2,2)*ES2-E(1,2)/ES2)*ET2
     A        +(-F(1,2)-E(2,1)*ES2+E(1,1)/ES2)/ET2)
      P12=CST*((-F(1,1)-E(2,2)*ES2+E(1,2)/ES2)*ET1
     B        +( F(1,2)+E(2,1)*ES2-E(1,1)/ES2)/ET1)
      P21=CST*((-F(2,1)-E(2,2)*ES1+E(1,2)/ES1)*ET2
     C        +( F(2,2)+E(2,1)*ES1-E(1,1)/ES1)/ET2)
      P22=CST*(( F(2,1)+E(2,2)*ES1-E(1,2)/ES1)*ET1
     D        +(-F(2,2)-E(2,1)*ES1+E(1,1)/ES1)/ET1)
      RETURN
  110 IF(CPSI.LT.0.)GO TO 120
      TA=T1
      TB=T2
      GO TO 130
  120 TA=-T1
      TB=-T2
      SGDT=-SGDT
  130 S1=S1
      DO 150 I=1,2
      TJ=TA
      DO 140 J=1,2
      ZIJ=TJ-S1
      R=SQRT(DSQ+ZIJ*ZIJ)
      W=R+ZIJ
      IF(ZIJ.LT.0.)W=DSQ/(R-ZIJ)
      V=R-ZIJ
```

```
      IF(ZIJ.GT.0.)V=DSQ/(R+ZIJ)
      IF(J.EQ.1)V1=V
      IF(J.EQ.1)W1=W
      EGZ(I,J)=CEXP(GAM*ZIJ)
  140 TJ=TB
      CALL EXPJ(GAM*V1,GAM*V,GP(I))
      CALL EXPJ(GAM*W1,GAM*W,GM(1))
  150 S1=S2
      CST=ETA/(8.*PI*SGDS*SGDT)
      P11=CST*(GM(2)*EGZ(2,2)+GP(2)/EGZ(2,2)
     2-CGDS*(GM(1)*EGZ(1,2)+GP(1)/EGZ(1,2)))
      P12=CST*(-GM(2)*EGZ(2,1)-GP(2)/EGZ(2,1)
     2+CGDS*(GM(1)*EGZ(1,1)+GP(1)/EGZ(1,1)))
      P21=CST*(GM(1)*EGZ(1,2)+GP(1)/EGZ(1,2)
     2-CGDS*(GM(2)*EGZ(2,2)+GP(2)/EGZ(2,2)))
      P22=CST*(-GM(1)*EGZ(1,1)-GP(1)/EGZ(1,1)
     2+CGDS*(GM(2)*EGZ(2,1)+GP(2)/EGZ(2,1)))
      RETURN
      END
      SUBROUTINE EXPJ(V1,V2,W12)
      COMPLEX EC,E15,S,T,UC,VC,V1,V2,W12,Z
      DIMENSION V(21),W(21),D(16),E(16)
      DATA V/   0.22284667E 00,
     20.11889321E 01,0.29927363E 01,0.57751436E 01,0.98374674E 01,
     20.15982874E 02,0.93307812E-01,0.49269174E 00,0.12155954E 01,
     20.22699495E 01,0.36676227E 01,0.54253366E 01,0.75659152E 01,
     20.10120228E 02,0.13130282E 02,0.16654408E 02,0.20776479E 02,
     20.25623894E 02,0.31407519E 02,0.38539683E 02,0.48026086E 02/
      DATA W/   0.45896460E 00,
     20.41700083E 00,0.11337338E 00,0.10399197E-01,0.26101720E-03,
     20.89854791E-06,0.21823487E 00,0.34221017E 00,0.26302758E 00,
     20.12642582E 00,0.40206865E-01,0.85638770E-02,0.12124361E-02,
     20.11167440E-03,0.64599267E-05,0.22263169E-06,0.42274304E-08,
     20.39218973E-10,0.14565152E-12,0.14830270E-15,0.16005949E-19/
      DATA U/   0.22495942E 02,
     2 0.74411568E 02,-0.41431576E 03,-0.78754339E 02, 0.11254744E 02,
     2 0.16021761E 03,-0.23862195E 03,-0.50094687E 03,-0.68487854E 02,
     2 0.12254778E 02,-0.10161976E 02,-0.47219591E 01, 0.79729681E 01,
     2-0.21069574E 02, 0.22046490E 01, 0.89728244E 01/
      DATA E/   0.21103107E 02,
     2-0.37959787E 03,-0.97489220E 02, 0.12900672E 03, 0.17949226E 02,
     2-0.12910981E 03,-0.55705574E 03, 0.13524801E 02, 0.14696721E 03,
     2 0.17949528E 02,-0.32981014E 00, 0.31028836E 02, 0.81657657E 01,
     2 0.22236961E 02, 0.39124892E 02, 0.81636799E 01/
      Z=V1
      DO 100 JIM=1,2
      X=REAL(Z)
      Y=AIMAG(Z)
      E15=(.0,.0)
      AB=CABS(Z)
      IF(AB.EQ.0.)GO TO 90
      IF(X.GE.0. .AND. AB.GT.10.)GO TO 80
      YA=ABS(Y)
      IF(X.LE.0. .AND. YA.GT.10.)GO TO 80
      IF(YA-X.GE.17.5.OR.YA.GE.6.5.OR.X+YA.GE.5.5.OR.X.GE.3.)GO TO 20
```

```
771         IF(X.LE.-9.)GO TO 40
772         IF(YA-X.GE.2.5)GO TO 50
773         IF(X+YA.GE.1.5)GO TO 30
774    10   N=6.+3.*AB
775         E15=1./(N-1.)-Z/N**2
776    15   N=N-1
777         E15=1./(N-1.)-Z*E15/N
778         IF(N.GE.3)GO TO 15
779         E15=Z*E15-CMPLX(.577216+ALOG(AB),ATAN2(Y,X))
780         GO TO 90
781    20   J1=1
782         J2=6
783         GO TO 31
784    30   J1=7
785         J2=21
786    31   S=(.0,.0)
787         YS=Y*Y
788         DO 32 I=J1,J2
789         XI=V(I)+X
790         CF=W(I)/(XI*XI+YS)
791    32   S=S+CMPLX(XI*CF,-YA*CF)
792         GO TO 54
793    40   T3=X*X-Y*Y
794         T4=2.*X*YA
795         T5=X*T3-YA*T4
796         T6=X*T4+YA*T3
797         UC=CMPLX(D(11)+D(12)*X+D(13)*T3+T5-E(12)*YA-E(13)*T4,
798        2          E(11)+E(12)*X+E(13)*T3+T6+D(12)*YA+D(13)*T4)
799         VC=CMPLX(D(14)+D(15)*X+D(16)*T3+T5-E(15)*YA-E(16)*T4,
800        2          E(14)+E(15)*X+E(16)*T3+T6+D(15)*YA+D(16)*T4)
801         GO TO 52
802    50   T3=X*X-Y*Y
803         T4=2.*X*YA
804         T5=X*T3-YA*T4
805         T6=X*T4+YA*T3
806         T7=X*T5-YA*T6
807         T8=X*T6+YA*T5
808         T9=X*T7-YA*T8
809         T10=X*T8+YA*T7
810         UC=CMPLX(D(1)+D(2)*X+D(3)*T3+D(4)*T5+D(5)*T7+T9-(E(2)*YA+E(3)*T4
811        2+E(4)*T6+E(5)*T8),E(1)+E(2)*X+E(3)*T3+E(4)*T5+E(5)*T7+T10+
812        3(D(2)*YA+D(3)*T4+D(4)*T6+D(5)*T8))
813         VC=CMPLX(D(6)+D(7)*X+D(8)*T3+D(9)*T5+D(10)*T7+T9-(E(7)*YA+E(8)*T4
814        2+E(9)*T6+E(10)*T8),E(6)+E(7)*X+E(8)*T3+E(9)*T5+E(10)*T7+T10+
815        3(D(7)*YA+D(8)*T4+D(9)*T6+D(10)*T8))
816    52   EC=UC/VC
817         S=EC/CMPLX(X,YA)
818    54   EX=EXP(-X)
819         T=EX*CMPLX(COS(YA),-SIN(YA))
820         E15=S*T
821    56   IF(Y.LT.0.)E15=CONJG(E15)
822         GO TO 90
823    80   E15=.409319/(Z+.193044)+.421831/(Z+1.02666)+.147126/(Z+2.56788)+
824        2.206335E-1/(Z+4.90035)+.107401E-2/(Z+8.18215)+.158654E-4/(Z+
825        312.7342)+.317031E-7/(Z+19.3957)
```

```
      E15=E15*CEXP(-Z)
   90 IF(JIM.EQ.1)W12=E15
  100 Z=V2
      Z=V2/V1
      TH=ATAN2(AIMAG(Z),REAL(Z))-ATAN2(AIMAG(V2),REAL(V2))
     2+ATAN2(AIMAG(V1),REAL(V1))
      AB=ABS(TH)
      IF(AB.LT.1.)TH=.0
      IF(TH.GT.1.)TH=6.2831853
      IF(TH.LT.-1.)TH=-6.2831853
      W12=W12-E15*CMPLX(.0,TH)
      RETURN
      END                                                              0158
      COMPLEX FUNCTION ZMN(DL,HL,SL)
      REAL L,LL,LL                                                     ZMN050
      B=6.2831853                                                      ZMN051
      D=DL
      L=HL
      LL=HL
      HC=SL
      BLL=B*LL                                                         ZMN052
      H=ABS(HC)-L                                                      ZMN053
      LL=LL                                                            ZMN054
      HPL=H+LL                                                         ZMN055
      HP2L=H+2.0*LL                                                    ZMN056
      HP3L=H+3.0*LL                                                    ZMN057
      HML=H-LL                                                         ZMN058
      SBL=SIN(BLL)                                                     ZMN059
      CBL=COS(BLL)                                                     ZMN060
      SBH=SIN(B*H)                                                     ZMN041
      CBH=COS(B*H)                                                     ZMN042
      SBHML=SIN(B*HML)                                                 ZMN043
      CBHML=COS(B*HML)                                                 ZMN044
      SBHPL=SIN(B*HPL)                                                 ZMN045
      CBHPL=COS(B*HPL)                                                 ZMN046
      SBHP2L=SIN(B*HP2L)                                               ZMN047
      CBHP2L=COS(B*HP2L)                                               ZMN048
      SBHP3L=SIN(B*HP3L)                                               ZMN049
      CBHP3L=COS(B*HP3L)                                               ZMN050
      TEMP=SQRT(D*D+H*H)+H                                             ZMN051
      V1=B*D*D/TEMP                                                    ZMN052
      U1=B*TEMP                                                        ZMN062
      TEMP=SQRT(D*D+HML*HML)+HML                                       ZMN063
      U0=B*TEMP                                                        ZMN064
      V0=B*D*D/TEMP                                                    ZMN065
      TEMP=SQRT(D*D+HPL*HPL)+HPL                                       ZMN066
      U3=B*TEMP                                                        ZMN067
      V3=B*D*D/TEMP                                                    ZMN068
      TEMP=SQRT(D*D+HP2L*HP2L)+HP2L                                    ZMN069
      U2=B*D*D/TEMP                                                    ZMN070
      V2=B*TEMP                                                        ZMN071
      TEMP=SQRT(D*D+HP3L*HP3L)+HP3L                                    ZMN072
      U4=B*D*D/TEMP                                                    ZMN073
      V4=B*TEMP                                                        ZMN074
      CALL SIC1 (SI00,CI00,U0)                                         ZMN100
```

```
      CALL SIC1 (SIU1,CIU1,U1)                                          ZMN101
      CALL SIC1 (SIV2,CIV2,V2)                                          ZMN102
      CALL SIC1 (SIV4,CIV4,V4)                                          ZMN103
      CALL SIC1 (SIU3,CIU3,U3)                                          ZMN104
      IF (D.LE.0.0) GO TO 60                                            ZMN075
      CALL SIC1 (SIV1,CIV1,V1)                                          ZMN076
      CALL SIC1 (SIV0,CIV0,V0)                                          ZMN079
      CALL SIC1 (SIV3,CIV3,V3)                                          ZMN081
      CALL SIC1 (SIU2,CIU2,U2)                                          ZMN082
      CALL SIC1 (SIU4,CIU4,U4)                                          ZMN084
      R=15.0*(CBHML*(CIU0+CIV0-CIU1-CIV1)+SBHML*(-SIU0+SIV0+SIU1-SIV1)+C ZMN086
     2BHPL*(2.*CIV3+2.*CIU3-CIU2-CIV2-CIU1-CIV1)+SBHPL*(-SIV3+SIU3+SIU2- ZMN087
     3SIV2-SIU1+SIV1+SIU3-SIV3)+CBHP3L*(-CIU2-CIV2+CIU4+CIV4)+SBHP3L*(SI ZMN088
     4U2-SIV2-SIU4+SIV4) +2.*CBL*CBH*(-CIV1-CIU1+CIV3+CIU3)+2.*CBL*SBH*( ZMN089
     5SIV1-SIU1-SIV3+SIU3)+2.*CBL*CBHP2L*(CIV4+CIU3-CIU2-CIV2)+2.*CBL*SB ZMN090
     6HP2L*(-SIV3+SIU3+SIU2-SIV2))                                      ZMN091
      X=15.0*(CBHML*(-SIU0-SIV0+SIU1+SIV1)-SBHML*(-CIU0+CIV0+CIU1-CIV1)+ ZMN092
     2CBHPL*(-2.*SIV3-2.*SIU3+SIU2+SIV2+SIU1+SIV1)+SBHPL*(-2.* CIV3+2.*C ZMN093
     3IU3+CIU2-CIV2-CIU1+CIV1)+CBHP3L*(SIU2+SIV2-SIU4-SIV4)+SBHP3L*(CIU2 ZMN094
     4-CIV2-CIU4+CIV4)+2.*CBL*CBH*(SIV1+SIU1-SIV3-SIU3)+ 2.*CBL*SBH*(CIV ZMN095
     51-CIU1-CIV3+CIU3)+2.*CBL*CBHP2L*(-SIV3-SIU3+SIU2+SIV2)+2.*CBL*SBHP ZMN096
     6P2L*(-CIV3+CIU3+CIU2-CIV2))                                       ZMN097
      GO TO 70                                                          ZMN098
   60 CONTINUE                                                          ZMN099
      R=15.0*(CBHML*(CIU0-CIU1+ALOG(H /HML))+SBHML*(SIU0-SIU1)+SBHPL*   ZMN105
     2 (2.*SIU3 -SIV2-SIU1)+CBHPL*(2.*CIU3-CIV2-CIU1+ALOG(HP2L/HPL)+    ZMN106
     3ALOG(H /HPL))+CBHP3L*(CIV4-CIV2+ALOG(HP2L/HP3L))+SBHP3L*(SIV4-SIV2 ZMN107
     4 )+2.*CBL*CBH*(CIU3-CIU1+ALOG(H /HPL))+2.*CBL*SBH*(SIU3-SIU1) +   ZMN108
     5 2.*CBL*CBHP2L*(CIU3-CIV2+ALOG(HP2L/HPL))+2.*CBL*SBHP2L*(SIU3-    ZMN109
     6 SIV2))                                                           ZMN110
      X=15.0*(CBHML*(SIU1-SIU0)+SBHML*(CIU0-CIU1+ALOG(HML/H ))+CBHPL*   ZMN111
     2 (SIV2+SIU1-2.*SIU3)+SBHPL*(2.*CIU3-CIV2-CIU1+ALOG(HPL/HP2L)+     ZMN112
     3ALOG(HPL/H ))+CBHP3L*(SIV2-SIV4)+SBHP3L*(CIV4-CIV2+ALOG(HP3L/HP2L) ZMN113
     4)+2.*CBL*CBH*(SIU1-SIU3)+2.*CBL*SBH*(CIU3-CIU1+ALOG(HPL/H ))+2.*CB ZMN114
     5L*CBHP2L*(SIV2-SIU3)+2.*CBL*SBHP2L*(CIU3-CIV2+ALOG(HPL/HP2L)))     ZMN115
   70 ZMN=CMPLX(R,X)/(SBL*SBL)                                          ZMN116
      RETURN                                                            ZMN117
      END
```

APPENDIX E
SPARSE MATRIX COMPUTER PROGRAM

The advantage of reducing the computer storage requirement can be achieved in solving a sparse matrix equation using high-speed computers if only the non-zero terms are stored. Computation time can also be reduced if only those operations (associated with solution techniques) involving nonzero terms are performed. However, most direct methods of solving systems of linear equations (e.g., square-root, Crout, Gaussian elimination, etc.) operate on the original matrix to produce an auxiliary matrix which in general is not sparse even though the original matrix was sparse.

Sparse matrix techniques require that this new auxiliary matrix be sparse as well. To accomplish this goal, special schemes are used to renumber the original matrix in order to ensure that the number of generated non-zero elements is minimum and to index the stored elements which include not only the original but also the newly-generated non-zero elements. Consequently, the advantages of reducing computer storage and computation time mentioned previously are only relative, since additional time must be devoted to the renumbering part and more storage space has to be allocated for the newly-generated non-zero elements. With these facts in mind, we proceed to describe, in general terms, one sparse-matrix method given by Berry [44].

For efficient utilization of high-speed memory and to allow for practical solution of a very large matrix equation, storage is allocated for only the non-zero elements of the original matrix. These terms are collapsed into two columnar arrays. The diagonal elements are stored by rows in a linear array D with dimensions N where N is the number of linear equations. The off-diagonal, non-zero elements of the upper triangular portion of the matrix are stored by rows in a linear array U with dimensions less or equal to $N(N+1)/2$. An efficient set of pointers for locating these terms in the array U is an absolute necessity. For a symmetric matrix, only the pointers associated with the upper triangular array of the matrix are retained. Two pointer arrays II and J are used to index the array U. It has dimension equal to N. The number stored in position k of this array represents the starting location in the pointer array J of terms associated with row k of the original matrix. J has dimensions equal to $N(N-1)/2$. This is a column identifier. The number stored in position k of this array represents the column index of the element U(k). Using the information contained in II and J, two additional pointer arrays, IUR and IUC, are set up. They record the same information contained in II and J, but this time the full matrix is being considered. Note that IUR has dimension $N+1$ and IUC has dimension

less or equal to N(N-1). An example should help clarify this scheme. For the original Z matrix given below, the arrays would be as follows:

$$Z = \begin{bmatrix} Z_{11} & 0 & Z_{13} & 0 & Z_{15} \\ 0 & Z_{22} & Z_{23} & Z_{24} & 0 \\ Z_{31} & Z_{32} & Z_{33} & Z_{34} & 0 \\ 0 & Z_{42} & Z_{43} & Z_{44} & 0 \\ Z_{51} & 0 & 0 & 0 & Z_{55} \end{bmatrix}$$

II(1) = 1	J(1) = 3	NUMOFF(1) = 2
II(2) = 3	J(2) = 5	NUMOFF(2) = 2
II(3) = 5	J(3) = 3	NUMOFF(3) = 3
II(4) = 6	J(4) = 4	NUMOFF(4) = 2
II(5) = 6	J(5) = 4	NUMOFF(5) = 1

N=5

Row Locator	Column Identifier	Term Identified
IUR(1)=1	IUC(1)=3	Z_{13}
IUR(2)=3	IUC(2)=5	Z_{15}
IUR(3)=5	IUC(3)=3	Z_{23}
IUR(4)=8	IUC(4)=4	Z_{24}
IUR(5)=10	IUC(5)=1	Z_{31}
IUR(6)=11	IUC(6)=2	Z_{32}
	IUC(7)=4	Z_{34}
	IUC(8)=2	Z_{47}
	IUC(9)=3	Z_{43}
	IUC(10)=1	Z_{51}

A specialized matrix decomposition known as the "square-root method" is used to solve the system of equations. Before decomposition, the algorithm given by Berry is used to renumber the unknowrs such that the number of non-zero elements in the auxiliary matrix produced by the decomposition is minimum.

There are three basic parts to the renumbering algorithm. All parts search the non-zero structure recorded by the pointer arrays IUR and IUC. An array NUMOFF with dimension N+1 is set up to record the total number of non-zero off-diagonal terms associated with each equation. NUMOFF(k) equals the total number of these terms that would appear in the Z matrix in row k.

Part I of the algorithm searches the array NUMOFF once to see if there are any equations with only one non-zero off-diagonal term. If one is found, it is number 1 and the array NUMOFF is altered. A single sweep through the array NUMOFF will rapidly pick off every equation that has only one or fewer effective off-diagonal terms. Decomposition of these single off-diagonal term equations will cause no new non-zero terms in the matrix.

Part II of the algorithm searches the remaining equations (those not renumbered in Part I) for equations which can be decomposed without increasing the number of non-zero terms. As each equation is checked, an array IFILL with dimension N+1 is set up which records the number of new positions that would become non-zero if that particular equation were renumbered next. If any equations were renumbered in this part, the algorithm is repeated because now the effective number of non-zero off-diagonal terms is different from the time Part II is first entered. When a complete Part II search is made without finding any equations for renumbering, then Part III is entered.

Part III finds the equation that would cause the fewest new non-zero terms by searching the array IFILL. After the choice is made and that equation renumbered, the new non-zero topology caused by decomposition of that equation is recorded in the system of pointers. After bookkeeping operations have been completed for renumbering an equation from Part III, Part II is again entered at the beginning.

After all of the equations are renumbered into the order in which the linear equations finally will appear in the matrix, the II and J pointer arrays are reorganized. For the reorganization all of the ponters are changed to correspond to the new system of equation numbers and include all non-zero terms that will ultimately be found in the upper triangular matrix U.

Finally, the solution of the matrix equation is readily obtained via the square-root method. This essentially is the very same program presented in Appendix II in Reference 12. The format sheet presented there is repeated here.

There are six input cards which are specified as follows:

Data card	Variables	Format	Descriptions
1	NSETS	I5	Number of clouds requested to be calculated
2	N	I5	Number of dipoles in a cloud
3	TL	F10.5	Length of a dipole in wavelengths
4	STDX,STDY,STD2,CF	4F10.5	Standard deviation of a Gaussian random generator for x,y,z coordinates respectively. CF is a coupling factor which weights the off-diagonal z-matrix elements. Usually set to unity.
5	IZ	I14	Starting point of the random generator
6	L2NPHI,ANGMIN, ANGMAX	I5,2F10.6	(2) is the number of look angles taken in the angle range (ANGMIN, ANGMAX)

The computer output consists of two parts: first print out of the input data with proper headings; second the average backscattering radar cross section calculations as described below.

Variables	Descriptions
AVTT	Average echo in θ-θ polarization
VARTT	Variance of the echo in θ-θ polarization
AVTP	Average echo in θ-ϕ polarization
VARTP	Variance of the echo in θ-ϕ polarization
AVPP	Average echo in ϕ-ϕ polarization
VARPP	Variance of the echo in ϕ-ϕ polarization
AV11	Average echo in both θ-θ and ϕ-ϕ polarizations
VAR11	Variance of the echo in both θ-θ and ϕ-ϕ polarizations
$S_{11}(20\%)$	The level under which 20% of the return signals belong
$S_{11}(50\%)$	The level under which 50% of the return signals belong
$S_{11}(80\%)$	The level under which 80% of the return signals belong

Finally the starting number of the random generator for the next computer run is indicated.

```
C*****************************************************************************
C*                                                                           *
C*       CHAXS - MAIN PROGRAM                                                *
C*                                                                           *
C*       PURPOSE                                                             *
C*          CALCULATION OF SOME STATISTICAL PARAMETERS OF THE                *
C*          BACKSCATTER FROM A "RANDOM" CHAFF CLOUD. THE AVERAGE             *
C*          VALUES AND VARIANCES OF THETA-THETA, THETA-PHI, AND              *
C*          PHI-PHI POLARIZATIONS ARE OBTAINED. AN AVERAGE AND               *
C*          VARIANCE FOR LINEAR-SAME SENSE LINEAR POLARIZATION               *
C*          ARE ALSO ESTIMATED. THREE POINTS ON THE CUMULATIVE               *
C*          PROBABILITY CURVE ARE ALSO CALCULATED.                           *
C*                                                                           *
C*       INPUT DATA                                                          *
C*                                                                           *
C*          NSETS           - THE NUMBER OF DATA SETS (CLOUDS) TO BE         *
C*                            RUN.                                           *
C*          N               - THE NUMBER OF DIPOLES IN THE CLOUD             *
C*          TL              - THE LENGTH OF THE DIPOLES IN WAVELENGTHS       *
C*          STDX,STDY,STDZ  - STANDARD DEVIATIONS OF THE DIPOLE              *
C*                            COORDINATES ALONG THE THREE PRINCIPAL          *
C*                            AXES IN WAVELENGTHS                            *
C*          CF              - A SCALING FACTOR FOR THE COUPLING              *
C*                            BETWEEN DIPOLES; USUALLY SET EQUAL TO 1.0      *
C*          IZ              - A STARTING NUMBER FOR THE RANDOM               *
C*                            NUMBER GENERATORS USED TO SET UP THE           *
C*                            CLOUDS. THIS ALLOWS A GIVEN "RANDOM"           *
C*                            CLOUD TO BE REGENERATED AT ANY TIME.           *
C*          L2NPHI          - LOG BASE 2 OF THE NUMBER OF "LOOK              *
C*                            ANGLES" TO BE USED.                            *
C*          ANGMIN,ANGMAX   - RANGE OVER WHICH THESE LOOK ANGLES             *
C*                            WILL BE SPACED; USUALLY 0.0-360.0              *
C*       OTHER PARAMETERS                                                    *
C*                                                                           *
C*          D,U             - COMPLEX ARRAYS CONTAINING THE COUPLING         *
C*                            MATRIX IN SPARSE MATRIX FORM                   *
C*          X,Y,Z           - ARRAYS CONTAINING THE COORDINATES OF           *
C*                            THE CENTERS OF THE DIPOLES                     *
C*          CA,CB,CG        - ARRAYS CONTAINING THE DIRECTION COSINES        *
C*                            OF THE DIPOLE ORIENTATIONS                     *
C*          II,J            - POINTER ARRAYS FOR THE SPARSE MATRIX           *
C*          AVTT,AVTP,AVPP  - CALCULATED AVERAGE BACKSCATTER FOR             *
C*          AV11              THETA-THETA, THETA-PHI, PHI-PHI, AND           *
C*                            LINEAR-SAME SENSE LINEAR POLARIZATIONS         *
C*          VARTT,VARTP     - CALCULATED VARIANCES ABOUT THE ABOVE           *
C*          VARPP,VAR11       AVERAGES                                       *
C*          PARAMETERS IN /SORT/ ARE USED BY THE REORDERING ROUTINES         *
C*                                                                           *
C*       REMARKS                                                             *
C*          DATA IS ONLY CALCULATED IN THE THETA EQUALS 90 DEGREES           *
```

```
C*          PLANE. THE UPPER TRIANGLE OF THE MATRIX C IS STORED BY      *
C*          ROWS IN SPARSE MATRIX FORM. SEE SUBROUTINES EXPAND, ORDER,  *
C*          SPSGT1 & SPSGT2 FOR DETAILS ON THIS STORAGE MODE.           *
C*                                                                      *
C**********************************************************************
      INCLUDE FFTPB,LIB
      COMPLEX C(200),U(513),ET1(201),EPP(201),CAA,ZMN,S(201),ZIJ
      DIMENSION X(201),Y(201),Z(201),CA(201),CB(201),CG(201),II(201),J(1
     2513),FREQ(128),SIG(128)
      DIMENSION II(6),CC(6)
      COMMON PK,SPK,CPK/FFZ/U,X,Y,Z,CA,CB,CG,II,J,M/SOIL/IORDER(201),NO
     2E(201),IUK(201),IUC(1513),NUMOFF(201),ITA(201)
    3 FORMAT(1415)                                                      CHAX0005
      CALL FFRK(1)
    4 FORMAT(7F10.5)                                                    CHAX0006
C**********************************************************************
C*                                                                      *
C*        READ INPUT DATA AND INITIALIZE PARAMETERS                     *
C*                                                                      *
C**********************************************************************
      PK=0.0174532D0                                                    CHAX0007
      PI=3.1415927                                                      CHAX0008
      TP=6.2831853                                                      CHAX0009
      READ(5,3)NSETS                                                    CHAX0010
      READ(5,3)                                                         CHAX0012
      READ(5,4)TL                                                       CHAX0013
      HL=TL/2.0
      AL=HL/100.0
      HK=PI*TL                                                          CHAX0014
      AK=HK/100.0                                                       CHAX0015
      READ(5,4)STDX,STDY,STDZ,CF
      WRITE(6,10)TL,STDX,STDY,STDZ,CF
   10 FORMAT(3HOH=,F5.2,8H LENGTH=,F10.5,30H POSITION STANDARD DEVIATIONS=
     2,1P3E15.6/1H ,16HCOUPLING FACTOR=,E15.8)
      STDXK=TP*STDX
      STDYK=TP*STDY
      STDZK=TP*STDZ
      READ(5,20)IZ
   20 FORMAT(I7)
      READ(5,22)L2MPHI,ANGMIN,ANGMAX
   22 FORMAT(I5,2F10.6)
      ANGDIF=ANGMAX-ANGMIN
      MPHI=2**L2MPHI
      DPH=ANGDIF/(FLOAT(MPHI))
      WRITE(6,23)MPHI,L2MPHI,ANGMIN,ANGMAX
   23 FORMAT(25HONUMBER OF LOOK ANGLES=,I5,4H=2**,I2,13H  ANGLE RANGE=,F6
     2.1,3H - ,F6.1//)
      CALL GITA(6,IT,CC)                                                CHAX0047
C**********************************************************************
C*                                                                      *
```

```
C*        CALCULATE SELF IMPEDANCE AND SET THRESHOLD FOR MUTUALS          *
C*                                                                        *
C************************************************************************
      CAA=ZMN(AL,0.0,HL,HL)
      THRSHD=CABS(CAA)*0.1
      CSK=COS(HK)                                                       CHAX0045
      SNK=SIN(HK)                                                       CHAX0046
      DO 80 NSET=1,NSETS
      IX=IZ
      WRITE(6,25)IX
   25 FORMAT(32H0RANDOM GENERATOR INITIALIZED AT,I15//)
      SIG(1)=0.0
      DSIG=0.06*FLOAT(N)
      DO 29 NSIG=1,100
      FREQ(NSIG)=0.0
   29 SIG(NSIG+1)=SIG(NSIG)+DSIG
      FREQ(101)=0.0
C************************************************************************
C*                                                                      *
C*        SET UP CLOUD (GENERATE X,Y,Z,CA,CB,CG)                        *
C*                                                                      *
C************************************************************************
      DO 30 I=1,N                                                       CHAX0026
      CALL GAUSS(IX,STDXK,0.0,X(I))
      CALL GAUSS(IX,STDYK,0.0,Y(I))
      CALL GAUSS(IX,STDZK,0.0,Z(I))
      CALL RANDU(IX,IY,A1)
      IX=IY
      PHI=TP*A1                                                         CHAX0054
      CALL RANDU(IX,IY,A2)
      IX=IY
      COSTH=2.0*A2-1.0                                                  CHAX0056
      SINTH=SQRT(1.0-COSTH*COSTH)                                       CHAX0057
      CA(I)=SINTH*COS(PHI)                                              CHAX0058
      CB(I)=SINTH*SIN(PHI)                                              CHAX0059
   30 CG(I)=COSTH                                                       CHAX0040
      DO 40 I=1,N                                                       CHAX0042
   40 IT(I)=CAA
C************************************************************************
C*                                                                      *
C*        SET UP INITIAL VALUES FOR POINTER ARRAYS AND PARAMETERS       *
C*        NEEDED FOR THE REORDERING                                     *
C*                                                                      *
C************************************************************************
      DO 41 I=1,N
      IORDER(I)=I
      NODE(I)=1
   41 NUMOFF(I)=0
      IC=1
      IT(1)=1
```

```
      NM1=N-1
      DO 45 I=1,NM1
      IP1=I+1
      DO 43 JC=IP1,N
      CALL ZGAUS(X(I),Y(I),Z(I),X(JC),Y(JC),Z(JC),CA(I),CB(I),CG(I),CA(J
     2C),CB(JC),CG(JC),ZIJ,6,TT,CC)
      IF(CABS(ZIJ).LT.THRSHD)GO TO 43
      J(IC)=JC
      IC=IC+1
      NUMOFF(I)=NUMOFF(I)+1
      NUMOFF(JC)=NUMOFF(JC)+1
   43 CONTINUE
   45 II(I+1)=IC
      IF(IC.GT.1500)WRITE(0,46)IC
   46 FORMAT(39HOARRAY J OVERRUN DURING INITIALIZATION/1X,I5,15H CELLS R
     2EQUIRED)
      CALL EXPAND(II,J,N)
C*****************************************************************************
C*                                                                           *
C*       GENERATE POINTER ARRAYS FOR THE REORDERED SYSTEM                    *
C*                                                                           *
C*****************************************************************************
      CALL ORDER(II,J,N)
C*****************************************************************************
C*                                                                           *
C*       CALCULATE MUTUAL IMPEDANCES AND STORE IN D AND U                    *
C*                                                                           *
C*****************************************************************************
      DO 49 I=1,NM1
      JFST=II(I)
      JLST=II(I+1)-1
      ISUB=IORDER(I)
      IF(JFST.GT.JLST)GO TO 49
      DO 47 JC=JFST,JLST
      JJC=J(JC)
      JSUB=IORDER(JJC)
   47 CALL ZGAUS(X(ISUB),Y(ISUB),Z(ISUB),X(JSUB),Y(JSUB),Z(JSUB),CA(ISUB
     2),CB(ISUB),CG(ISUB),CA(JSUB),CB(JSUB),CG(JSUB),U(JC),6,TT,CC)
   49 CONTINUE
C*****************************************************************************
C*                                                                           *
C*       GENERATE "SQUARE ROOT" METHOD AUXILIARY MATRIX OF C                 *
C*                                                                           *
C*****************************************************************************
      CALL SPSQT1(D,U,II,J,N)
C*****************************************************************************
C*                                                                           *
C*       ACCUMULATE SUMS FOR AVERAGES AND VARIANCES                          *
C*                                                                           *
C*****************************************************************************
```

```
      PH=0.0
      CPH=1.0
      SPH=0.0
      AVTT=0.0
      AVTP=0.0
      AVPP=0.0
      VARTT=0.0
      VARTP=0.0
      VARPP=0.0
      DO 66 NPH=1,NPHI
      CALL FPT(CPH,SPH,0.0,ECTT,ECTP,ECPP,IORDER)
      AVTT=AVTT+ECTT
      AVTP=AVTP+ECTP
      AVPP=AVPP+ECPP
      VARTT=VARTT+ECTT*ECTT
      VARTP=VARTP+ECTP*ECTP
      VARPP=VARPP+ECPP*ECPP
C*****************************************************************************
C*                                                                           *
C*       CONSTRUCT HISTOGRAM OF THE BACKSCATTER FOR THE CUMULATIVE           *
C*       DISTRIBUTION CALCULATION                                            *
C*                                                                           *
C*****************************************************************************
      DO 50 NSIG=1,101
      TEMP=SIG(NSIG)
      IF(ECTT-TEMP)60,50,50
   50 CONTINUE
      GO TO 62
   60 FREQ(NSIG)=FREQ(NSIG)+1.0
   62 CONTINUE
      DO 63 NSIG=1,101
      TEMP=SIG(NSIG)
      IF(ECPP-TEMP)64,63,63
   63 CONTINUE
      GO TO 65
   64 FREQ(NSIG)=FREQ(NSIG)+1.0
   65 CONTINUE
      PH=PH+DPH
      PHR=PH*DR
      CPH=COS(PHR)
   66 SPH=SIN(PHR)
C*****************************************************************************
C*                                                                           *
C*       CALCULATE BACKSCATTER AVERAGES AND VARIANCES                        *
C*                                                                           *
C*****************************************************************************
      NPH2=2*NPHI
      AV11=AVTT+AVPP
      VAR11=(VARTT+VARPP-AV11*AV11/FLOAT(NPH2))/FLOAT(NPH2-1)
      AV11=AV11/FLOAT(NPH2)
```

```
      VARTT=(VARTT-AVTT*AVTT/FLOAT(NPH))/FLOAT(NPH-1)
      AVTT=AVTT/FLOAT(NPH)
      VARTP=(VARTP-AVTP*AVTP/FLOAT(NPH))/FLOAT(NPH-1)
      AVTP=AVTP/FLOAT(NPH)
      VARPP=(VARPP-AVPP*AVPP/FLOAT(NPH))/FLOAT(NPH-1)
      AVPP=AVPP/FLOAT(NPH)
      FREQ(1)=FREQ(1)/FLOAT(NPH1)/2.0
      FX=FREQ(1)
C*****************************************************************
C*                                                               *
C*       CALCULATE THREE POINTS ON THE CUMULATIVE DISTRIBUTION CURVE   *
C*                                                               *
C*****************************************************************
      DO 70 NSIG=2,101
      FREQ(NSIG)=FREQ(NSIG)/FLOAT(NPH1)/2.0
      FX=FX+FREQ(NSIG)
      IF(FX.LE.0.2)SIG20=SIG(NSIG)
      IF(FX.LE.0.5)SIG50=SIG(NSIG)
      IF(FX.LE.0.8)SIG80=SIG(NSIG)
   70 FREQ(NSIG)=FX
      WRITE(6,71)AVTT,VARTT,AVTP,VARTP,AVPP,VARPP,AV11,VAR11,SIG20,SIG50
     2,SIG80
   71 FORMAT(6H0AVTT=,1PE15.8,7H VARTT=,E15.8,6H AVTP=,E15.8,7H VARTP=,E
     215.8/6H AVPP=,E15.8,7H VARPP=,E15.8,6H AV11=,E15.8,7H VAR11=,E15.8
     3/11H0S11 BELOW ,E15.8,19H 20% OF THE SAMPLES/11H S11 BELOW ,E15.8,
     419H 50% OF THE SAMPLES/11H S11 BELOW ,E15.8,19H 80% OF THE SAMPLES
     5//)
      IZ=IZ*9709
      IF(IZ)76,80,80
   76 IZ=IZ+2147483647+1
   80 CONTINUE
      WRITE(6,85)IZ
   85 FORMAT(79H0A GOOD NUMBER TO USE FOR INITIALIZING THE RANDOM GENERA
     2TOR ON THE NEXT RUN IS ,I7)
      CALL EXIT
      END
      SUBROUTINE GITAB(N,T,C)                                    GTAB0000
C*****************************************************************GTAB0001
C*                                                               *GTAB0002
C*       PURPOSE                                                 *GTAB0003
C*          SETS UP COEFFICIENT TABLES FOR ZGAUS                 *GTAB0004
C*                                                               *GTAB0005
C*       USAGE                                                   *GTAB0006
C*          CALL GITAB(N,T,C)                                    *GTAB0007
C*                                                               *GTAB0008
C*       DESCRIPTION OF PARAMETERS                               *GTAB0009
C*          N    -  THE NUMBER OF POINTS ZGAUS IS TO USE IN      *GTAB0010
C*                  INTEGRATION                                  *GTAB0011
C*          T    -  THE ABSCISSA VALUES FOR ZGAUS                *GTAB0012
C*                  (MUST BE DIMENSIONED IN MAIN PROGRAM)        *GTAB0013
```

```
C*          C     -  THE WEIGHTING COEFFICIENTS FOR ZGAUS                    *GTAB0014
C*                   (MUST BE DIMENSIONED IN MAIN PROGRAM)                   *GTAB0015
C*          HK    -  THE HALF LENGTH OF THE DIPOLES IN ELECTRIC/L            *GTAB0016
C*                   RADIANS                                                 *GTAB0017
C*          COK   -  COS(HK)                                                 *GTAB0018
C*          SOK   -  SIN(HK)                                                 *GTAB0019
C*                                                                           *GTAB0020
C*************************************************************************GTAB0021
      DIMENSION T(N),C(N),V(40),A(40)                                        GTAB0022
      COMMON HK,SOK,COK                                                      GTAB0023
      DATA V/    .33998104,.86113631,.23861919,.66120939,.93246951,          GTAB0024
     2.1834346,.52553241,.79666648,.96028985,.14887433,.43339539,           GTAB0025
     3.67940957,.86506337,.97390653,.12523340,.36783150,.58731795,          GTAB0026
     4.76990267,.90411726,.98156063,.05012851,.28160354,.45801678,          GTAB0027
     5.61787825,.75540441,.88563120,.94457502,.98940093,.06405689,          GTAB0028
     6.19111887,.41504267,.65379651,.84642147,.64809365,.74012419,          GTAB0029
     7.82000199,.87641353,.93827455,.97472856,.99517227/                    GTAB0030
      DATA W/    .65214515,.34785485,.46791393,.36076157,.17132448,          GTAB0031
     2.04605376,.13370689,.22238103,.10122854,.29552422,.26926672,          GTAB0032
     3.21908636,.14945135,.06667134,.24914704,.23349254,.20316742,          GTAB0033
     4.16007832,.10693933,.04717533,.12948497,.15620852,.16915651,          GTAB0034
     5.14959599,.12462897,.09515851,.06225352,.02715246,.12796120,          GTAB0035
     6.25857404,.12367047,.11550367,.10744426,.09761865,.08619016,          GTAB0036
     7.07334664,.05929857,.04427744,.02853139,.01234122/                    GTAB0037
      NU=(N*N-N)/2                                                           GTAB0038
      IF(N.EQ.6)NU=21                                                        GTAB0039
      IF(N.EQ.12)NU=99                                                       GTAB0040
      NI=NU+N-1                                                              GTAB0041
      J=1                                                                    GTAB0042
      DO 100 I=NU,NI                                                         GTAB0043
      T(J)=HK*V(I)                                                           GTAB0044
      C(J)=SIN(HK-T(J))*W(I)                                                 GTAB0045
  100 J=J+1                                                                  GTAB0046
      RETURN                                                                 GTAB0047
      END                                                                    GTAB0048
      SUBROUTINE ZGAUS(XA,YA,ZA,XB,YB,ZB,CAS,CBS,CGS,CA,CB,CG,SM12,N,T,CZGUS0000
     2)                                                                      ZGUS0001
C*************************************************************************ZGUS0002
C*                                                                         *ZGUS0003
C*       PURPOSE                                                           *ZGUS0004
C*          CALCULATES THE MUTUAL IMPEDANCE BETWEEN TWO DIPOLES            *ZGUS0005
C*                                                                         *ZGUS0006
C*       USAGE                                                             *ZGUS0007
C*          CALL ZGAUS(XA,YA,ZA,XB,YB,ZB,CAS,CBS,CGS,CA,CB,CG,SM12,        *ZGUS0008
C*          N,T,C)                                                         *ZGUS0009
C*                                                                         *ZGUS0010
C*       DESCRIPTION OF PARAMETERS                                         *ZGUS0011
C*          XA,YA,ZA     - COORDINATES OF THE POSITION OF THE CENTER        *ZGUS0012
C*                         OF THE FIRST DIPOLE IN ELECTRICAL RADIANS       *ZGUS0013
C*          XB,YB,ZB     - COORDINATES OF THE POSITION OF THE CENTER        *ZGUS0014
```

```
C*                              OF THE SECOND DIPOLE IN ELECTRICAL RADIANS    *ZGLS0015
C*          CAS,CBS,CGS - DIRECTION COSINES OF THE ORIENTATION OF THE         *ZGLS0016
C*                        FIRST DIPOLE                                        *ZGLS0017
C*          CA,CB,CG    - DIRECTION COSINES OF THE ORIENTATION OF THE         *ZGLS0018
C*                        SECOND DIPOLE                                       *ZGLS0019
C*          SM12        - THE RESULTANT MUTUAL IMPEDANCE                      *ZGLS0020
C*          N           - THE NUMBER OF POINTS USED IN THE GAUSSIAN           *ZGLS0021
C*                        INTEGRATION                                         *ZGLS0022
C*          T           - THE ABSCISSA VALUES GENERATED BY GITAB              *ZGLS0023
C*          C           - THE WEIGHTING COEFFICIENTS GENERATED BY             *ZGLS0024
C*                        GITAB                                               *ZGLS0025
C*          HK          - THE HALF LENGTH OF THE DIPOLES IN                   *ZGLS0026
C*                        ELECTRICAL RADIANS                                  *ZGLS0027
C*          CHK         - COS(HK)                                             *ZGLS0028
C*          SHK         - SIN(HK)                                             *ZGLS0029
C*                                                                            *ZGLS0030
C*      REMARKS                                                               *ZGLS0031
C*          GITAB MUST BE CALLED WITH THE APPROPRIATE VALUE OF N              *ZGLS0032
C*          BEFORE ZGALS MAY BE CALLED. ONE CALL TO GITAB IS ALL THAT         *ZGLS0033
C*          IS REQUIRED FOR ANY NUMBER OF CALLS TO ZGALS AS LONG AS           *ZGLS0034
C*          N REMAINS UNCHANGED                                               *ZGLS0035
C*                                                                            *ZGLS0036
C*      SUBPROGRAMS AND FUNCTION SUBPROGRAMS REQUIRED                         *ZGLS0037
C*          GITAB                                                             *ZGLS0038
C*                                                                            *ZGLS0039
C*      METHOD                                                                *ZGLS0040
C*          INDUCED EMF EVALUATED BY GAUSSIAN INTEGRATION. PIECE WISE         *ZGLS0041
C*          SINUSOIDAL CURRENTS (TWO SEGMENTS PER DIPOLE) ARE                 *ZGLS0042
C*          ASSUMED                                                           *ZGLS0043
C*                                                                            *ZGLS0044
C*******************************************************************************ZGLS0045
      COMPLEX SM12,EJR1,EJR2,EJR3,ET1,ER1                                     ZGLS0046
      DIMENSION T(N),C(N)                                                     ZGLS0047
      COMMON HK,SHK,CHK                                                       ZGLS0048
      CC=CA*CAS+CB*CBS+CG*CGS                                                 ZGLS0049
      SM12=(0.0,0.0)                                                          ZGLS0050
      DO 90 I=1,N                                                             ZGLS0051
      TI=T(I)                                                                 ZGLS0052
      DO 90 J=1,2                                                             ZGLS0053
      X=XB+TI*CA                                                              ZGLS0054
      Y=YB+TI*CB                                                              ZGLS0055
      Z=ZB+TI*CG                                                              ZGLS0056
      ZZ=(X-XA)*CAS+(Y-YA)*CBS+(Z-ZA)*CGS                                     ZGLS0057
      Z1=ZZ+HK                                                                ZGLS0058
      Z2=ZZ-HK                                                                ZGLS0059
      XZ=XA+ZZ*CAS                                                            ZGLS0060
      YZ=YA+ZZ*CBS                                                            ZGLS0061
      ZZ=ZA+ZZ*CGS                                                            ZGLS0062
      XXZ=X-XZ                                                                ZGLS0063
      YYZ=Y-YZ                                                                ZGLS0064
```

```
401         ZZZ=Z-ZZ                                                        ZGUS0065
402         RS=XXZ*XXZ+YYZ*YYZ+ZZZ*ZZZ                                      ZGUS0066
403         R1=SQRT(RS+ZZ1*ZZ1)                                             ZGUS0067
404         EJR1=CMPLX(COS(R1),-SIN(R1))/R1                                 ZGUS0068
405         R2=SQRT(RS+ZZ2*ZZ2)                                             ZGUS0069
406         EJR2=CMPLX(COS(R2),-SIN(R2))/R2                                 ZGUS0070
407         R3=SQRT(RS+ZZ3*ZZ3)                                             ZGUS0071
408         EJR3=CMPLX(COS(R3),-SIN(R3))/R3                                 ZGUS0072
409         ET1=CL*(2.0*EJR2*COK-EJR1-EJR3)/SDK                             ZGUS0073
410         IF(RS.LT.1.0E-4*HK)GO TO 20                                     ZGUS0074
411         SKY=(CA*XXZ+CB*YYZ+CG*ZZZ)/RS                                   ZGUS0075
412         ER1=(EJR1*ZZ1+EJR3*ZZ3-2.0*COK*EJR2*ZZ2)/SDK                    ZGUS0076
413         ET1=ET1+ER1*SKY                                                 ZGUS0077
414      60 SM12=SM12-ET1*C(I)                                              ZGUS0078
415      90 TI=-TI                                                          ZGUS0079
416         SM12=SM12*(0.0,30.0)*HK/SLK                                     ZGUS0080
417         RETURN                                                          ZGUS0081
418         END                                                             ZGUS0082
419         SUBROUTINE FARFD(X,Y,Z,CA,CB,CG,CTH,STH,CPH,SPH,ET,EP)          FRFD0000
420 C*************************************************************************FRFD0001
421 C*                                                                       *FRFD0002
422 C*       PURPOSE                                                         *FRFD0003
423 C*          CALCULATES THE FAR ELECTRIC FIELDS OF A TWO SEGMENT          *FRFD0004
424 C*          PIECEWISE SINUSOIDAL DIPOLE CURRENT                          *FRFD0005
425 C*                                                                       *FRFD0006
426 C*       USAGE                                                           *FRFD0007
427 C*          CALL FARFD(X,Y,Z,CA,CB,CG,CTH,STH,CPH,SPH,ET,EP)             *FRFD0008
428 C*                                                                       *FRFD0009
429 C*       DESCRIPTION OF PARAMETERS                                       *FRFD0010
430 C*          X,Y,Z     -  COORDINATES OF THE CENTER OF THE DIPOLE         *FRFD0011
431 C*          CA,CB,CG  -  DIRECTION COSINES OF THE ORIENTATION OF         *FRFD0012
432 C*                       THE DIPOLE                                      *FRFD0013
433 C*          CTH       -  COS(THETA)                                      *FRFD0014
434 C*          STH       -  SIN(THETA)                                      *FRFD0015
435 C*          CPH       -  COS(PHI)                                        *FRFD0016
436 C*          SPH       -  SIN(PHI)                                        *FRFD0017
437 C*          ET        -  THETA COMPONENT OF THE CALCULATED E-FIELD       *FRFD0018
438 C*          EP        -  PHI COMPONENT OF THE CALCULATED E-FIELD         *FRFD0019
439 C*          HK        -  HALF LENGTH OF THE DIPOLE IN ELECTRICAL         *FRFD0020
440 C*                       RADIANS                                         *FRFD0021
441 C*          SDK       -  SIN(HK)                                         *FRFD0022
442 C*          COK       -  COS(HK)                                         *FRFD0023
443 C*                                                                       *FRFD0024
444 C*       REMARKS                                                         *FRFD0025
445 C*          THETA AND PHI ARE THE LOOK ANGLES FOR WHICH THE FIELDS ARE   *FRFD0026
446 C*          CALCULATED                                                   *FRFD0027
447 C*                                                                       *FRFD0028
448 C*************************************************************************FRFD0029
449         COMPLEX ET,EP,EJB,ES                                            FRFD0050
450         COMMON HK,SDK,COK                                               FRFD0051
```

```
      G=(CA*CPH+CB*SPH)*STH+CG*CTH                                      FRFD0052
      GK=1.0-G*G                                                        FRFD0053
      FT=(0.0,0.0)                                                      FRFD0054
      FP=(0.0,0.0)                                                      FRFD0055
      IF(GK.LT..001)GO TO 200                                           FRFD0056
      P=(X*CPH+Y*SPH)*STH+Z*CTH                                         FRFD0057
      EJR=CMPLX(COS(P),SIN(P))                                          FRFD0058
      ES=(0.0,0.5)*EJR*(CDK-COS(G*HK))/GK/SDK                           FRFD0059
      T=(CA*CPH+CB*SPH)*CTH-CG*STH                                      FRFD0060
      F=CA*SPH+CB*CPH                                                   FRFD0061
      FT=T*ES                                                           FRFD0062
      FP=F*ES                                                           FRFD0063
  200 CONTINUE                                                          FRFD0064
      RETURN                                                            FRFD0065
      END                                                               FRFD0066
      SUBROUTINE FPT(CPH,SPH,CTH,ECTT,ECTP,ECPP,IONER)                  FPT0001
C*****************************************************************    FPT0001
C*                                                               *    FPT0002
C*        PURPOSE                                                *    FPT0003
C*           CALCULATES THE BACKSCATTER FROM A CLOUD OF DIPOLES AT A *    FPT0004
C*           GIVEN LOOK ANGLE                                    *    FPT0005
C*                                                               *    FPT0006
C*        USAGE                                                  *    FPT0007
C*           CALL FPT(CPH,SPH,CTH,ECTT,ECTP,ECPP)                *    FPT0008
C*                                                               *    FPT0009
C*        DESCRIPTION OF PARAMETERS                              *    FPT0010
C*           CPH      -  COS(PHI)                                *    FPT0011
C*           SPH      -  SIN(PHI)                                *    FPT0012
C*           CTH      -  COS(THETA)                              *    FPT0013
C*           ECTT     -  THETA-THETA BACKSCATTER                 *    FPT0014
C*           ECTP     -  THETA-PHI BACKSCATTER                   *    FPT0015
C*           ECPP     -  PHI-PHI BACKSCATTER                     *    FPT0016
C*           HK       -  HALF LENGTH OF THE DIPOLES IN ELECTRICAL *    FPT0017
C*                       RADIANS                                 *    FPT0018
C*           SDK      -  SIN(HK)                                 *    FPT0019
C*           CDK      -  COS(HK)                                 *    FPT0020
C*           D,U      -  ARRAYS CONTAINING THE DIAGONAL TERMS AND THE *
C*                       OFF DIAGONAL TERMS RESPECTIVELY OF THE  *
C*                       AUXILIARY MATRIX CALCULATED FROM THE ACTUAL *
C*                       IMPEDANCE MATRIX BY SPSQT1. THE MATRIX IS IN *
C*                       THE SPARSE MATRIX STORAGE MODE.         *
C*           X,Y,Z    -  ARRAYS CONTAINING THE COORDINATES OF THE *    FPT0023
C*                       CENTERS OF THE DIPOLES                  *    FPT0024
C*           CA,CB,CG -  ARRAYS CONTAINING THE DIRECTION COSINES OF THE *    FPT0025
C*                       ORIENTATIONS OF THE DIPOLES             *    FPT0026
C*           IONER,   -  INDEXING ARRAYS FOR THE SPARSE MATRIX   *
C*           II,JJ       STORAGE MODE                            *
C*           N        -  THE NUMBER OF DIPOLES                   *    FPT0027
C*           NC       -  NOT USED                                *    FPT0028
C*                                                               *    FPT0029
```

```
C*        REMARKS                                                              * FPT0050
C*           SPSGT1 MUST BE CALLED TO GENERATE THE AUXILIARY MATRIX            * FPT0051
C*           BEFORE THE FIRST CALL TO FPT. THETA AND PHI ARE THE               * FPT0052
C*           LOOK ANGLES                                                       * FPT0053
C*                                                                             * FPT0054
C*        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                        * FPT0055
C*           SPSGT1, SPSGT2, FARFD                                             * FPT0056
C*                                                                             * FPT0057
C*******************************************************************************FPT0058
      COMPLEX D(201),U(1613),ETT(201),EPP(201),S(201),ETH,EPH
      DIMENSION X(201),Y(201),Z(201),CA(201),CB(201),CG(201),IORDER(201)
     2,T(201),U(1613)
      COMMON NK,SDK,CDK/FF/D,U,X,Y,Z,CA,CB,CG,II,J,N
      TP=6.2831853                                                           FPT0062
      STH=SQRT(1.0-CTH*CTH)                                                  FPT0063
      DO 70 I=1,N                                                            FPT0064
      ISUB=IORDER(I)
      CALL FARFD(X(ISUB),Y(ISUB),Z(ISUB),CA(ISUB),CB(ISUB),CG(ISUB),CTH,
     2STH,CPH,SPH,ETT(I),EPP(I))
   70 S(I)=ETT(I)*(0.0,1.0)/TP/50.0                                          FPT0067
      CALL SPSGT2(0.0,S,II,J,N)
      ETH=(0.0,0.0)                                                          FPT0069
      EPH=(0.0,0.0)                                                          FPT0070
      DO 80 I=1,N                                                            FPT0071
      ETH=ETH+S(I)*ETT(I)                                                    FPT0072
   80 EPH=EPH+S(I)*EPP(I)                                                    FPT0073
      CETH=CABS(ETH)                                                         FPT0074
      CEPH=CABS(EPH)                                                         FPT0075
      FCTT=2.0*TP*CETH*CETH                                                  FPT0076
      FCTP=2.0*TP*CEPH*CEPH                                                  FPT0077
      DO 90 I=1,N                                                            FPT0078
   90 S(I)=EPP(I)*(0.0,1.0)/TP/50.0                                          FPT0079
      CALL SPSGT2(0.0,S,II,J,N)
      EPH=(0.0,0.0)                                                          FPT0081
      DO 100 I=1,N                                                           FPT0082
  100 EPH=EPH+S(I)*EPP(I)                                                    FPT0083
      CEPH=CABS(EPH)                                                         FPT0084
      FCPP=2.0*TP*CEPH*CEPH                                                  FPT0085
      RETURN                                                                 FPT0086
      END                                                                    FPT0087
      COMPLEX FUNCTION ZMN(D,RC,LE,L)                                         ZMN000
C*******************************************************************************ZMN001
C*        PURPOSE                                                            * ZMN002
C*                                                                           * ZMN003
C*           COMPUTES THE MUTUAL IMPEDANCE BETWEEN TWO EQUAL LENGTH          * ZMN004
C*           PARALLEL DIPOLES.                                               * ZMN005
C*                                                                           * ZMN006
C*        USAGE                                                              * ZMN007
C*           Z=ZMN(D,RC,LE,L)                                                * ZMN008
C*                                                                           * ZMN009
```

```
C*       DESCRIPTION OF PARAMETERS                                          *  ZMNU10
C*          Z     -  THE RESULTANT VALUE OF THE MUTUAL IMPEDANCE            *  ZMNU11
C*          D     -  THE HORIZONTAL DISTANCE BETWEEN THE DIPOLES IN         *  ZMNU12
C*                   WAVELENGTHS.                                          *  ZMNU13
C*          HC    -  THE VERTICAL DISTANCE BETWEEN THE DIPOLES IN           *  ZMNU14
C*                   WAVELENGTHS.                                          *  ZMNU15
C*          LE    -  THE EFFECTIVE HALF LENGTH OF THE DIPOLES IN            *  ZMNU16
C*                   WAVELENGTHS (SEE FOR INSTANCE SCHELKUNOFF & FRIIS     *  ZMNU17
C*                   ANTENNAS THEORY AND PRACTICE P. 244-246 & P. 420)     *  ZMNU18
C*          L     -  THE PHYSICAL HALF LENGTH OF THE DIPOLES                *  ZMNU19
C*                                                                         *  ZMNU20
C*       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                      *  ZMNU21
C*          SICI  - SSP ROUTINE FOR SINE AND COSINE INTEGRALS              *  ZMNU22
C*                                                                         *  ZMNU23
C*       REFERENCES                                                        *  ZMNU24
C*          HOWARD E. KING, "MUTUAL IMPEDANCE OF UNEQUAL LENGTH           *  ZMNU25
C*          ANTENNAS IN ECHELON", IRE TRANS. ANTENNAS & PROPAGAT.,         *  ZMNU26
C*          VOL AP-5, JULY 1957 P. 306-313                                 *  ZMNU27
C*                                                                         *  ZMNU28
C*************************************************************************  ZMNU29
      REAL L,LE,LL                                                          ZMNU30
      B=6.283185                                                            ZMNU31
      BLE=B*LE                                                              ZMNU32
      H=ABS(HC)-L                                                           ZMNU33
      LL=LE                                                                 ZMNU34
      HPL=H+LL                                                              ZMNU35
      HP2L=H+2.0*LL                                                         ZMNU36
      HP3L=H+3.0*LL                                                         ZMNU37
      HML=H-LL                                                              ZMNU38
      SBLE=SIN(BLE)                                                         ZMNU39
      CBLE=COS(BLE)                                                         ZMNU40
      SBH=SIN(B*H)                                                          ZMNU41
      CBH=COS(B*H)                                                          ZMNU42
      SBHML=SIN(B*HML)                                                      ZMNU43
      CBHML=COS(B*HML)                                                      ZMNU44
      SBHPL=SIN(B*HPL)                                                      ZMNU45
      CBHPL=COS(B*HPL)                                                      ZMNU46
      SBHP2L=SIN(B*HP2L)                                                    ZMNU47
      CBHP2L=COS(B*HP2L)                                                    ZMNU48
      SBHP3L=SIN(B*HP3L)                                                    ZMNU49
      CBHP3L=COS(B*HP3L)                                                    ZMNU50
      TEMP=SQRT(L*D+H*H)+H                                                  ZMNU51
      V1=B*D*D/TEMP                                                         ZMNU52
C*************************************************************************  ZMNU53
C*                                                                         *  ZMNU54
C*       NOTE: SQRT(D*D+H*H)-H = D*D/(SQRT(D*D+H*H)+H)                     *  ZMNU55
C*                                                                         *  ZMNU56
C*          THE FORM ON THE RIGHT HAND SIDE OF THIS EQUATION HAS           *  ZMNU57
C*          SUPERIOR ROUND OFF CHARACTERISTICS AND WAS USED FOR THIS       *  ZMNU58
C*          REASON.                                                        *  ZMNU59
```

```
C*                                                                      *  ZMNU60
C**********************************************************************  ZMNU61
      U1=B*TEMP                                                            ZMNU62
      TEMP=SQRT(D*D+HML*HML)+HML                                           ZMNU63
      U0=B*TEMP                                                            ZMNU64
      V0=B*U*D/TEMP                                                        ZMNU65
      TEMP=SQRT(D*D+HPL*HPL)+HPL                                           ZMNU66
      U3=B*TEMP                                                            ZMNU67
      V3=B*U*D/TEMP                                                        ZMNU68
      TEMP=SQRT(D*D+HP2L*HP2L)+HP2L                                        ZMNU69
      U2=B*D*U/TEMP                                                        ZMNU70
      V2=B*TEMP                                                            ZMNU71
      TEMP=SQRT(D*D+HP3L*HP3L)+HP3L                                        ZMNU72
      U4=B*D*U/TEMP                                                        ZMNU73
      V4=B*TEMP                                                            ZMNU74
      IF (D.LE.0.0) GO TO 80                                               ZMNU75
      CALL SIC1 (SIV1,CIV1,V1)                                             ZMNU76
      CALL SIC1 (SIU1,CIU1,U1)                                             ZMNU77
      CALL SIC1(SIU0,CIU0,U0)                                              ZMNU78
      CALL SIC1 (SIV0,CIV0,V0)                                             ZMNU79
      CALL SIC1 (SIU3,CIU3,U3)                                             ZMNU80
      CALL SIC1 (SIV3,CIV3,V3)                                             ZMNU81
      CALL SIC1 (SIU2,CIU2,U2)                                             ZMNU82
      CALL SIC1 (SIV2,CIV2,V2)                                             ZMNU83
      CALL SIC1 (SIU4,CIU4,U4)                                             ZMNU84
      CALL SIC1 (SIV4,CIV4,V4)                                             ZMNU85
      P=15.0*(CBHPL*(CIU0+CIV0-CIU1-CIV1)-SBHML*(-SIU0+SIV0+SIU1-SIV1)+C   ZMNU86
     2BHPL*(2.*CIV3+2.*CIU3-CIU2-CIV2-CIU1-CIV1)+SBHPL*(-SIV3+SIU3+SIU2-   ZMNU87
     3SIV2-SIU1+SIV1+SIU3-SIV3)+CBHP3L*(-CIU2-CIV2+CIU4+CIV4)+SBHP3L*(SI   ZMNU88
     4U2-SIV2-SIU4+SIV4) +2.*CBL*CBH*(-CIV1-CIU1+CIV3+CIU3)+2.*CBL*SBH*(   ZMNU89
     5SIV1-SIU1-SIV3+SIU3)+2.*CBL*CBHP2L*(CIV3+CIU3-CIU2-CIV2)+2.*CBL*SB   ZMNU90
     6HP2L*(-SIV3+SIU3+SIU2-SIV2))                                         ZMNU91
      X=15.0*(CBHPL*(-SIU0-SIV0+SIU1+SIV1)-SBHML*(-CIU0+CIV0+CIU1-CIV1)+   ZMNU92
     2CBHPL*(-2.*SIV3-2.*SIU3+SIU2+SIV2+SIU1+SIV1)+SBHPL*(-2.* CIV3+2.*C   ZMNU93
     3IU3+CIU2-CIV2-CIU1+CIV1)+CBHP3L*(SIU2+SIV2-SIU4-SIV4)+SBHP3L*(CIU2   ZMNU94
     4-CIV2-CIU4+CIV4)+2.*CBL*CBH*(SIV1+SIU1-SIV3-SIU3)+ 2.*CBL*SBH*(CIV   ZMNU95
     51-CIU1-CIV3+CIU3)+2.*CBL*CBHP2L*(-SIV3-SIU3+SIU2+SIV2)+2.*CBL*SBHP   ZMNU96
     62L*(-CIV3+CIU3+CIU2-CIV2))                                           ZMNU97
      GO TO 90                                                             ZMNU98
   80 CONTINUE                                                             ZMNU99
      CALL SIC1 (SIU0,CIU0,U0)                                             ZMN100
      CALL SIC1 (SIU1,CIU1,U1)                                             ZMN101
      CALL SIC1 (SIV2,CIV2,V2)                                             ZMN102
      CALL SIC1 (SIV4,CIV4,V4)                                             ZMN103
      CALL SIC1 (SIU3,CIU3,U3)                                             ZMN104
      P=15.0*(CBHPL*(CIU0-CIU1+ALOG(H /HML))+SBHML*(SIU0-SIU1)+SBHPL*      ZMN105
     2 (2.*SIU3 -SIV2-SIU1)+CBHPL*(2.*CIU3-CIV2-CIU1+ALOG(HP2L/HPL))+      ZMN106
     3ALOG(H /HPL))+CBHP3L*(CIV4-CIV2+ALOG(HP2L/HP3L))+SBHP3L*(SIV4-SIV2   ZMN107
     4 )+2.*CBL*CBH*(CIU2-CIU1+ALOG(H /HPL))+2.*CBL*SBH*(SIU3-SIU1) +      ZMN108
     5 2.*CBL*CBHP2L*(CIU3-CIV2+ALOG(HP2L/HPL))+2.*CBL*SBHP2L*(SIU3-       ZMN109
```

```
     *  SIV2))                                                              ZMN110
        Y=10.0*(CPML*(SIU0-SIU0)+SBHML*(CIU0-CIU1+ALOG(HML/H ))+CBLPL*      ZMN111
       2 (SIV2+SIU1-2.*SIU3)+SBHPL*(2.*CIU3-CIV2-CIU1+ALOG(HPL/HP2L)+       ZMN112
       3ALOG(HPL/H ))+CBHP2L*(SIV2-SIV4)+SBHP2L*(CIV4-CIV2+ALOG(HP2L/HP2L)  ZMN113
       4)+2.*CBL*CBL*(SIU1-SIU3)+2.*CBL*SBH*(CIU3-CIU1+ALOG(HPL/H ))+2.*CB  ZMN114
       5L*CBHP2L*(SIV2-SIU2)+2.*CLL*SBHP2L*(CIU4-CIV2+ALOG(HPL/HP2L)))      ZMN115
    90 ZMN=CMPLX(F,X)/(SPL*SBL)                                             Z45116
       RETURN                                                               ZMN117
       END                                                                  ZMN118
       SUBROUTINE SICT(SI,CI,X)                                          SICI000
C*************************************************************************SICI001
C*                                                                       *SICI002
C*       PURPOSE                                                         *SICI003
C*          COMPUTES THE SINE AND COSINE INTEGRALS                       *SICI004
C*                                                                       *SICI005
C*       USAGE                                                           *SICI006
C*          CALL SICT(SI,CI,X)                                           *SICI007
C*                                                                       *SICI008
C*       DESCRIPTION OF PARAMETERS                                       *SICI009
C*          SI     - THE RESULTANT VALUE SI(X)                           *SICI010
C*          CI     - THE RESULTANT VALUE CI(X)                           *SICI011
C*          X      - THE ARGUMENT OF SI(X) AND CI(X)                     *SICI012
C*                                                                       *SICI013
C*       REMARKS                                                         *SICI014
C*          THE ARGUMENT VALUE REMAINS UNCHANGED                         *SICI015
C*                                                                       *SICI016
C*       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   *SICI017
C*          NONE                                                         *SICI018
C*                                                                       *SICI019
C*       METHOD                                                          *SICI020
C*          DEFINITION                                                   *SICI021
C*          SI(X)=INTEGRAL(SIN(T)/T)                                     *SICI022
C*          CI(X)=INTEGRAL(COS(T)/T)                                     *SICI023
C*          EVALUATION                                                   *SICI024
C*          REDUCTION OF RANGE USING SYMMETRY                            *SICI025
C*          DIFFERENT APPROXIMATIONS ARE USED FOR ABS(X) GREATER         *SICI026
C*          THAN 4 AND FOR ABS(X) LESS THAN 4.                           *SICI027
C*                                                                       *SICI028
C*       REFERENCES                                                      *SICI029
C*          IBM SCIENTIFIC SUBROUTINE PACKAGE P. 370                     *SICI030
C*          LUKE AND WIMP, 'POLYNOMIAL APPROXIMATIONS TO INTEGRAL        *SICI031
C*          TRANSFORMS', MATHEMATICAL TABLES AND OTHER AIDS TO           *SICI032
C*          COMPUTATION, VOL. 15, 1961, ISSUE 74, P. 174-178             *SICI033
C*                                                                       *SICI034
C*************************************************************************SICI035
       Z=ABS(X)                                                          SICI036
       IF(Z-4.0)1,1,4                                                    SICI037
     1 Y=(4.0-Z)*(4.0+Z)                                                 SICI038
       SI=-1.5707963E0                                                   SICI039
       IF(Z)5,2,5                                                        SICI040
```

```
      2 C1=-1.0E30                                                              SIC1041
        RETURN                                                                  SIC1042
      3 S1=X*(((((1.753141E-9*Y+1.56A988E-7)*Y+1.374168E-5)*Y+8.939869E-4)      SIC1043
       2*Y+1.964682E-2)*Y+4.395509E-1+SI/X)                                     SIC1044
        C1=((5.772156E-1+ALOG(Z))/Z-7*(((((1.A8E985E-10*Y+1.584996E-8)*Y        SIC1045
       2+1.725?54E-6)*Y+1.185999E-4)*Y+4.990920E-3)*Y+1.315308E-1))*Z           SIC1046
        RETURN                                                                  SIC1047
      4 S1=SIN(Z)                                                               SIC1048
        Y=COS(Z)                                                                SIC1049
        Z=4.0/Z                                                                 SIC1050
        U=(((((((4.040069E-3*Z-2.279143E-2)*Z+5.515470E-2)*Z-7.261642E-2)       SIC1051
       2*Z+4.987716E-2)*Z-3.552519E-2)*Z-2.314617E-2)*Z-1.134950E-5)*Z          SIC1052
       3+6.250011E-2)*Z+2.583089E-10                                            SIC1053
        V=(((((((((-5.108509E-4*Z+2.A13179E-2)*Z-5.527285E-2)*Z                 SIC1054
       2+7.902034E-2)*Z-4.400416E-2)*Z-7.945556E-3)*Z+2.6C1253E-2)*Z            SIC1055
       3-5.764000E-4)*Z-5.12241PE-2)*Z-6.6464?1E-7)*Z+2.500000E-1               SIC1056
        C1=Z*(S1*V-Y*U)                                                         SIC1057
        S1=-Z*(S1*U+Y*V)                                                        SIC1058
        IF(X)5,6,6                                                              SIC1059
      5 S1=-3.141594E0-S1                                                       SIC1060
      6 RETURN                                                                  SIC1061
        END                                                                     SIC1062
        SUBROUTINE EXPAND(II,J,N)
C*************************************************************************
C*                                                                       *
C*      PURPOSE                                                          *
C*         GENERATES POINTER ARRAYS FOR THE FULL MATRIX GIVEN POINTER    *
C*         ARRAYS FOR THE UPPER TRIANGLE OF A SYMETRIC SPARSE MATRIX     *
C*                                                                       *
C*      USAGE                                                            *
C*         CALL EXPAND(II,J,N)                                           *
C*                                                                       *
C*      DESCRIPTION OF PARAMETERS                                        *
C*         II        -  ARRAY CONTAINING THE STARTING LOCATIONS OF       *
C*                      TERMS IN J ASSOCIATED WITH THE ROWS OF THE       *
C*                      UPPER TRIANGULAR MATRIX                          *
C*         J         -  ARRAY CONTAINING THE COLUMN INDICES OF THE       *
C*                      NON-ZERO ELEMENTS IN THE UPPER TRIANGULAR        *
C*                      MATRIX                                           *
C*         N         -  THE NUMBER OF EQUATIONS IN THE SYSTEM (MUST      *
C*                      BE LESS THAN OR EQUAL TO THE DIMENSION OF        *
C*                      II IN THE CALLING PROGRAM)                       *
C*         IORDER    -  NOT USED                                         *
C*         NOUE      -  NOT USED                                         *
C*         IUR       -  ARRAY CONTAINING THE STARTING LOCATIONS OF       *
C*                      TERMS IN IUC ASSOCIATED WITH THE ROWS OF THE     *
C*                      FULL MATRIX ON OUTPUT                            *
C*         IUC       -  ARRAY CONTAINING THE COLUMN INDICES OF THE       *
C*                      NON-ZERO ELEMENTS OF THE FULL MATRIX ON OUTPUT   *
C*         NUMOFF    -  ARRAY CONTAINING THE NUMBER OF NON-ZERO OFF      *
```

```
C*                        DIAGONAL TERMS IN EACH ROW OF THE FULL MATRIX  *
C*                        MUST BE SET BY THE CALLING PROGRAM!            *
C*           ITA       -  NOT USED                                       *
C*                                                                       *
C*        REFERENCES                                                     *
C*           BERRY, R. D. "AN OPTIMAL ORDERING OF ELECTRONIC CIRCUIT     *
C*           EQUATIONS FOR A SPARSE MATRIX SOLUTION" IEEE TRANSACTIONS   *
C*           ON CIRCUIT THEORY VOL CT-18 NO. 1 JANUARY 1971 P. 40-50     *
C*                                                                       *
C***************************************************************************
      COMMON /SORT/IORDER(201),NODE(201),IUR(201),IUC(1613),NUMOFF(201),
     2ITA(201)
      DIMENSION II(1),J(1)
      IUR(1)=1
      ITA(1)=1
      DO 10 I=1,N
      IP1=I+1
      IUR(IP1)=IUR(I)+NUMOFF(I)
   10 ITA(IP1)=IUR(IP1)
      IF(IUR(N+1).GT.1613)WRITE(0,12)IUR(N+1)
   12 FORMAT(26H0ARRAY IUC OVERRUN IN EXPAND/1X,I5,1X,14HCELLS REQUIRED)
      NM1=N-1
      DO 30 I=1,NM1
      JFST=II(I)
      JLST=II(I+1)-1
      IF(JFST.GT.JLST)GO TO 30
      ISUB=ITA(I)
      DO 20 JC=JFST,JLST
      JJC=J(JC)
      IUC(ISUB)=JJC
      JSUB=ITA(JJC)
      IUC(JSUB)=I
      ISUB=ISUB+1
   20 ITA(JJC)=ITA(JJC)+1
   30 CONTINUE
      RETURN
      END
      SUBROUTINE ORDER(II,JJ,N)
C***************************************************************************
C*                                                                       *
C*        PURPOSE                                                        *
C*           DETERMINES A REORDERING OF THE UNKNOWNS IN A SPARSE         *
C*           MATRIX EQUATION SUCH THAT THE NUMBER OF NON-ZERO TERMS      *
C*           CREATED BY AN L-U TYPE DECOMPOSITION IS REDUCED. THE        *
C*           SPARSE MATRIX MUST HAVE A SYMMETRIC STRUCTURE. I.E. IF      *
C*           C(I,J) .NE. 0 THEN C(J,I) .NE. 0 IT IS NOT NECESSARY        *
C*           THAT C(I,J)=C(J,I)                                          *
C*                                                                       *
C*        USAGE                                                          *
C*           CALL ORDER(II,J,N)                                          *
```

```
C*                                                                      *
C*      DESCRIPTION OF PARAMETERS                                       *
C*         II      -                                                    *
C*         II      - -  ARRAY CONTAINING THE STARTING INDICES FOR THE   *
C*                      ROWS OF THE UPPER TRIANGLE OF THE REORDERED     *
C*                      MATRIX ON EXIT. II(J)=K IMPLIES THAT J(K)       *
C*                      AND U(K) CONTAIN THE COLUMN INDEX AND VALUE     *
C*                      RESPECTIVELY OF THE FIRST OFF DIAGONAL TERM     *
C*                      IN ROW J OF THE UPPER TRIANGLE                  *
C*         J        -   ARRAY CONTAINING THE COLUMN INDICES OF THE      *
C*                      UPPER TRIANGLE OF THE REORDERED MATRIX ON       *
C*                      EXIT                                            *
C*         N        -   THE NUMBER OF ROWS (COLUMNS) IN THE MATRIX      *
C*         IORDER   -   ARRAY CONTAINING THE ORIGINAL INDICES OF THE    *
C*                      UNKNOWNS IN THE REORDERED SEQUENCE. IORDER(I)   *
C*                      MUST EQUAL I ON ENTRY                           *
C*         NODE     -   ARRAY COMPLIMENTARY TO IORDER; IF IORDER(J)=K   *
C*                      THEN NODE(K)=J NODE(I) MUST EQUAL I ON ENTRY    *
C*         IUR      -   SAME AS II BUT FOR THE FULL MATRIX, NOT JUST    *
C*                      THE UPPER TRIANGLE (DESTROYED)                  *
C*         IUC      -   SAME AS J BUT FOR THE FULL MATRIX (DESTROYED)   *
C*         NUMOFF   -   ARRAY CONTAINING THE NUMBER OF OFF DIAGONAL     *
C*                      TERMS IN EACH ROW OF THE ORIGINAL FULL MATRIX    *
C*                      ON ENTRY (DESTROYED)                            *
C*         ITA      -   WORK ARRAY USED BY THE ROUTINE                  *
C*                                                                      *
C*      REMARKS                                                         *
C*         ALL ARRAYS IN COMMON EXCEPT ITA MUST BE INITIALIZED          *
C*         BEFORE ENTRY. THE ACTUAL VALUES OF THE MATRIX ELEMENTS ARE   *
C*         NOT USED BY THIS ROUTINE. THE DIAGONAL ELEMENTS AND THE      *
C*         OFF DIAGONAL ELEMENTS ARE STORED IN SEPARATE ARRAYS WITH     *
C*         THE OFF DIAGONAL ELEMENTS INDEXED BY II AND J. IUR(N+1)      *
C*         MUST BE DEFINED BEFORE ENTRY                                 *
C*                                                                      *
C*      SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   *
C*         RENMBR, INSERT                                               *
C*                                                                      *
C*      REFERENCES                                                      *
C*         BERRY, R. D. "AN OPTIMAL ORDERING OF ELECTRONIC CIRCUIT      *
C*         EQUATIONS FOR A SPARSE MATRIX SOLUTION" IEEE TRANSACTIONS    *
C*         ON CIRCUIT THEORY VOL CT-18 NO. 1 JANUARY 1971 P. 40-50      *
C*                                                                      *
C**********************************************************************
      COMMON /SORT/IORDER(201),NODE(201),IUR(201),IUC(1613),NUMOFF(201),
     2ITA(201)
      DIMENSION IFILL(201),II(1),JJ(1)
      ASSIGN 150 TO IRTN
C**********************************************************************
C*                                                                      *
C*      BEGIN PART I                                                    *
```

```
C*        PICK UP ROWS WITH ZERO OR ONE OFF DIAGONAL TERM                *
C*                                                                       *
C***********************************************************************
      LOAD=1
      DO 10 I=1,N
      IR=IORDER(I)
      IF(NUMOFF(IR).LE.1)CALL RENMBR(IR,LOAD,N,IRTN)
   10 CONTINUE
C***********************************************************************
C*                                                                       *
C*        END OF PART I BEGIN PART II                                    *
C*        PICK UP ROWS WHICH WILL NOT INCREASE THE NUMBER OF             *
C*        OFF DIAGONAL TERMS                                             *
C*                                                                       *
C***********************************************************************
      INSRTS=0
   11 LOADED=0
      IRO=LOAD
   12 IR=IORDER(IRO)
      IFILL(IRO)=0
      ICS=IUR(IR+1)-1
      ICT=IUR(IR)
      NUM=0
   15 IC=IUC(ICT)
      IF(NODE(IC).LT.LOAD)GO TO 20
      NUM=NUM+1
      ITA(NUM)=IC
   20 ICT=ICT+1
      IF(ICT.LE.ICS)GO TO 15
      I=1
   25 J=I+1
      IF(J.GT.NUM)GO TO 65
   30 IRT=ITA(I)
      IC=ITA(J)
      ICS=IUR(IRT+1)-1
      ICT=IUR(IRT)
   40 IF(IC.EQ.IUC(ICT))GO TO 50
      ICT=ICT+1
      IF(ICT.LE.ICS)GO TO 40
      IFILL(IRO)=IFILL(IRO)+1
      IF(INSRTS.EQ.1)CALL INSERT(IRT,IC,N)
   50 IF(J.EQ.NUM)GO TO 60
      J=J+1
      GO TO 30
   60 I=I+1
      GO TO 25
   65 IF(INSRTS.NE.1)GO TO 70
      CALL RENMBR(IR,LOAD,N,IRTN)
      INSRTS=0
      GO TO 11
```

```
901      70 IF(IFILL(IRO).NE.0)GO TO 75
902         LOADED=1
903         CALL RENMBR(IR,LOAD,N,IRTN)
904      75 IRO=IRO+1
905         IF(IRO.LE.N)GO TO 12
906         IF(LOADED.NE.0)GO TO 11
907 C*************************************************************************
908 C*                                                                       *
909 C*        END PART II BEGIN PART III                                     *
910 C*        PICK UP ROW WHICH WILL ADD THE FEWEST NON-ZERO OFF DIAGONAL    *
911 C*        TERMS                                                          *
912 C*                                                                       *
913 C*************************************************************************
914         J=LOAD
915      80 ITEST=IORDER(J)
916         K=J+1
917      90 IF(IFILL(K).GE.IFILL(J))GO TO 100
918         IR=IORDER(K)
919         IORDER(K)=ITEST
920         IORDER(J)=IR
921         NODE(IR)=J
922         NODE(ITEST)=K
923         ITEST=IR
924         IR=IFILL(K)
925         IFILL(K)=IFILL(J)
926         IFILL(J)=IR
927     100 K=K+1
928         IF(K.LE.N)GO TO 90
929         IF(IFILL(LOAD).NE.IFILL(J))GO TO 110
930         J=J+1
931         IF(J.LT.N)GO TO 80
932     110 KE=J-1
933         ITEST=IORDER(LOAD)
934         K=LOAD+1
935     120 IF(K.LE.KE)GO TO 130
936         INSRTS=1
937         GO TO 11
938     130 IR=IORDER(K)
939         IF(NUMOFF(IR).LE.NUMOFF(ITEST))GO TO 140
940         IORDER(K)=ITEST
941         IORDER(LOAD)=IR
942         NODE(IR)=LOAD
943         NODE(ITEST)=K
944         ITEST=IR
945     140 K=K+1
946         GO TO 120
947 C*************************************************************************
948 C*                                                                       *
949 C*        END PART III                                                   *
950 C*        REORGANIZE POINTER ARRAYS                                      *
```

```
C*                                                                      *
C************************************************************************
  150 I=1
      IF(IUR(N+1).GT.1613)WRITE(0,152)IUR(N+1)
  152 FORMAT(27HOARRAY IUC OVERRUN IN ORDER/1X,I5,15H CELLS REQUIRED)
      DO 170 LOAD=1,N
      II(LOAD)=I
      IR=IORDER(LOAD)
      ICS=IUR(IR+1)-1
      ICT=IUR(IR)
      IF(ICT.GT.ICS)GO TO 170
      DO 165 IC=ICT,ICS
      IUCIC=IUC(IC)
      IF(IC.EQ.ICS)GO TO 162
      ICP1=IC+1
      DO 160 JC=ICP1,ICS
      IUCJC=IUC(JC)
      IF(NODE(IUCIC).LE.NODE(IUCJC))GO TO 160
      IUC(IC)=IUCJC
      IUC(JC)=IUCIC
      IUCIC=IUCJC
  160 CONTINUE
  162 IF(NODE(IUCIC).LT.LOAD)GO TO 165
      JJ(I)=NODE(IUCIC)
      I=I+1
  165 CONTINUE
  170 CONTINUE
      RETURN
      END
      SUBROUTINE RENMBR(IR,LOAD,N,IRTN)
C************************************************************************
C*                                                                      *
C*     PURPOSE                                                          *
C*        TO RENUMBER ONE UNKNOWN AND UPDATE THE INDEXING ARRAYS        *
C*        IUR AND IUC ACCORDINGLY. CALLED FROM SUBROUTINE ORDER         *
C*                                                                      *
C*     USAGE                                                            *
C*        CALL RENMBR(IR,LOAD,N,IRTN)                                   *
C*                                                                      *
C*     DESCRIPTION OF PARAMETERS                                        *
C*        IR        -  ORIGINAL INDEX OF THE UNKNOWN                    *
C*        LOAD      -  THE NEW INDEX TO BE ASSIGNED TO THE UNKNOWN      *
C*        N         -  THE NUMBER OF UNKNOWNS IN THE SYSTEM OF          *
C*                     EQUATIONS                                        *
C*        IRTN      -  ALTERNATE RETURN ADDRESS                         *
C*                                                                      *
C*     REMARKS                                                          *
C*        ALL ARRAYS IN COMMON ARE THE SAME AS IN SUBROUTINE ORDER       *
C*                                                                      *
C*     REFERENCES
```

```
C*          PERRY, R. O. "AN OPTIMAL ORDERING OF ELECTRONIC CIRCUIT    *
C*          EQUATIONS FOR A SPARSE MATRIX SOLUTION" IEEE TRANSACTIONS  *
C*          ON CIRCUIT THEORY VOL CT-18 NO. 1 JANUARY 1971 P. 40-50    *
C*                                                                     *
C*********************************************************************
      COMMON /SOFT/IORDER(201),NODE(201),IUN(201),IUC(1215),NUMOFF(201),
     2ITA(201)
      LOADIR=0
      NUMFIL=0
   10 IF(NODE(IR).LT.LOAD)GO TO 40
      ITEMP=IORDER(LOAD)
      IROT=NODE(IR)
      IORDER(LOAD)=IR
      NODE(IR)=LOAD
      IORDER(IROT)=ITEMP
      NODE(ITEMP)=IROT
      LOAD=LOAD+1
      IF(LOAD.GT.N)RETURN IRTN
      ICS=IUP(IR+1)-1
      ICT=IUP(IR)
      IF(ICT.GT.ICS)GO TO 40
   20 IR=IUC(ICT)
      IF(NODE(IR).LT.LOAD)GO TO 30
      NUMOFF(IR)=NUMOFF(IR)-1
      IF(NUMOFF(IR).GT.1)GO TO 30
      NUMFIL=NUMFIL+1
      ITA(NUMFIL)=IR
   30 ICT=ICT+1
      IF(ICT.LE.ICS)GO TO 20
   40 LOADIR=LOADIR+1
      IF(LOADIR.GT.NUMFIL)RETURN
      IR=ITA(LOADIR)
      GO TO 10
      END
      SUBROUTINE INSERT(IRT,IC,N)
C*********************************************************************
C*                                                                   *
C*       PURPOSE                                                     *
C*          TO INSERT TERMS INTO THE INDEXING ARRAYS IUP AND IUC     *
C*          WHICH REFER TO NON-ZERO OFF DIAGONAL TERMS WHICH WILL BE *
C*          CREATED BY AN L-U DECOMPOSITION.                         *
C*          CALLED FROM SUBROUTINE ORDER                             *
C*                                                                   *
C*       USAGE                                                       *
C*          CALL INSERT(IRT,IC,N)                                    *
C*                                                                   *
C*       DESCRIPTION OF PARAMETERS                                   *
C*          IRT       -  THE INDEX OF THE ROW INTO WHICH THE TERMS ARE *
C*                       TO BE INSERTED                              *
C*          IC        -  THE COLUMN INDEX WHICH IS TO BE INSERTED    *
```

```
C*          N          -  THE NUMBER OF UNKNOWNS IN THE SYSTEM OF          *
C*                        EQUATIONS                                        *
C*                                                                         *
C*       REMARKS                                                           *
C*          ALL ARRAYS IN COMMON ARE THE SAME AS IN SUBROUTINE ORDER        *
C*                                                                         *
C*       REFERENCES                                                        *
C*          BERRY, R. D. "AN OPTIMAL ORDERING OF ELECTRONIC CIRCUIT        *
C*          EQUATIONS FOR A SPARSE MATRIX SOLUTION" IEEE TRANSACTIONS      *
C*          ON CIRCUIT THEORY VOL CT-16 NO. 1 JANUARY 1971 P. 40-50        *
C*                                                                         *
C***************************************************************************
      COMMON /SOLT/IORDER(201),LODE(201),IUR(201),IUC(1613),NUMOFF(201),
     2ITA(201)
      IDOWN=IUR(N+1)-IUR(IRT+1)
      L=0
   10 L=L+1
      K=IUR(N+1)-L
      IUC(K+1)=IUC(K)
      IF(L.LT.IDOWN)GO TO 10
      NUMOFF(IRT)=NUMOFF(IRT)+1
      IUC(K)=IC
      L=IRT
   20 L=L+1
      IUR(L)=IUR(L)+1
      IF(L.LE.K)GO TO 20
      IDOWN=IUR(N+1)-IUR(IC+1)
      L=0
   30 L=L+1
      K=IUR(N+1)-L
      IUC(K+1)=IUC(K)
      IF(L.LT.IDOWN)GO TO 30
      NUMOFF(IC)=NUMOFF(IC)+1
      IUC(K)=IRT
      L=IC
   40 L=L+1
      IUR(L)=IUR(L)+1
      IF(L.LE.N)GO TO 40
      RETURN
      END
      SUBROUTINE SPSOT1(D,U,II,J,N)
C***************************************************************************
C*                                                                         *
C*       PURPOSE                                                           *
C*          TO TRANSFORM A SYMMETRIC SPARSE MATRIX INTO A SPARSE           *
C*          AUXILIARY MATRIX (IMPLICIT INVERSE)                            *
C*                                                                         *
C*       USAGE                                                             *
C*          CALL SPSOT1(D,U,II,J,N)                                        *
C*                                                                         *
```

```
C*        DESCRIPTION OF PARAMETERS                                      *
C*           D         -  ARRAY CONTAINING THE DIAGONAL OF THE ORIGINAL  *
C*                        MATRIX ON ENTRY AND THE DIAGONAL OF THE        *
C*                        AUXILIARY MATRIX ON EXIT                       *
C*           U         -  ARRAY CONTAINING THE NON-ZERO OFF DIAGONAL     *
C*                        TERMS OF THE UPPER TRIANGLE OF THE ORIGINAL    *
C*                        MATRIX PLUS SPACE FOR NON-ZERO OFF DIAGONAL    *
C*                        TERMS CREATED BY THIS ROUTINE ON ENTRY AND THE *
C*                        NON-ZERO OFF DIAGONAL TERMS OF THE UPPER       *
C*                        TRIANGLE OF THE AUXILIARY MATRIX ON EXIT       *
C*           II        -  ARRAY CONTAINING THE STARTING INDICES IN J AND *
C*                        U OF TERMS ASSOCIATED WITH EACH ROW OF BOTH    *
C*                        MATRICES                                       *
C*           J         -  ARRAY CONTAINING THE COLUMN INDICES OF         *
C*                        CORRESPONDING TERMS IN U                       *
C*           N         -  THE NUMBER OF ROWS (COLUMNS) IN THE MATRIX     *
C*                                                                       *
C*        REMARKS                                                        *
C*           THE NON-ZERO OFF DIAGONAL ELEMENTS OF THE UPPER TRIANGLE    *
C*           ARE STORED BY ROWS. SUBROUTINE ORDER SHOULD BE CALLED       *
C*           BEFORE ENTRY TO IMPROVE THE ORDERING OF THE ELEMENTS        *
C*           AND TO RESERVE SPACE FOR ELEMENTS CREATED BY THIS ROUTINE   *
C*                                                                       *
C*        METHOD                                                         *
C*           "SQUARE ROOT" DECOMPOSITION OF A SYMMETRIC MATRIX           *
C*                                                                       *
C*        REFERENCES                                                     *
C*           BERRY, R. D., "AN OPTIMAL ORDERING OF ELECTRONIC CIRCUIT    *
C*           EQUATIONS FOR A SPARSE MATRIX SOLUTION" IEEE TRANSACTIONS   *
C*           ON CIRCUIT THEORY VOL CT-18 NO. 1 JANUARY 1971 P. 40-50     *
C*           FADDEEV, D. K. AND FADDEEVA, V. N., COMPUTATIONAL           *
C*           METHODS OF LINEAR ALGEBRA, W. H. FREEMAN AND CO., SAN       *
C*           FRANCISCO, 1965, P. 144-147                                 *
C*                                                                       *
C************************************************************************
      COMPLEX D(1),U(1)
      DIMENSION II(1),J(1)
      NM1=N-1
      DO 100 I=1,NM1
      JMIN=II(I)
      JMAX=II(I+1)-1
      D(I)=CSQRT(D(I))
      IF(JMIN.GT.JMAX)GO TO 100
      DO 10 K=JMIN,JMAX
   10 U(K)=U(K)/D(I)
      DO 30 K=JMIN,JMAX
      JJ=J(K)
      D(JJ)=D(JJ)-U(K)*U(K)
      LMIN=K+1
      IF(LMIN.GT.JMAX)GO TO 100
```

```
      M=II(JJ)-1
      MLST=II(JJ+1)-1
      DO 50 L=LMIN,JMAX
   20 M=M+1
      IF(M.GT.MLST)WRITE(0,15)
   15 FORMAT(16HOERROR IN SPSOT1//)
      IF(J(M).NE.J(L))GO TO 20
   50 U(M)=U(M)-U(K)*U(L)
  100 CONTINUE
      S(N)=CSQRT(D(N))
      RETURN
      END
      SUBROUTINE SPSOT2(D,U,S,II,J,N)
C*****************************************************************
C*                                                               *
C*      PURPOSE                                                  *
C*         TO OBTAIN A SOLUTION TO THE SYMMETRIC SPARSE MATRIX   *
C*         EQUATION MX=Y USING THE AUXILIARY MATRIX CALCULATED BY*
C*         SPSW11                                                *
C*                                                               *
C*      USAGE                                                    *
C*         CALL SPSOT2(D,U,S,II,J,N)                             *
C*                                                               *
C*      DESCRIPTION OF PARAMETERS                                *
C*         D         -  ARRAY CONTAINING THE DIAGONAL OF THE     *
C*                      AUXILIARY MATRIX                         *
C*         U         -  ARRAY CONTAINING THE NON-ZERO OFF DIAGONAL*
C*                      TERMS OF THE UPPER TRIANGLE OF THE AUXILIARY*
C*                      MATRIX STORED BY ROWS                    *
C*         S         -  ARRAY CONTAINING THE RIGHT HAND SIDE VECTOR*
C*                      Y ON ENTRY AND THE SOLUTION VECTOR X ON EXIT*
C*         II        -  ARRAY CONTAINING THE STARTING INDICES IN J AND*
C*                      U OF TERMS ASSOCIATED WITH EACH ROW OF THE*
C*                      AUXILIARY MATRIX                         *
C*         J         -  ARRAY CONTAINING THE COLUMN INDICES OF   *
C*                      CORRESPONDING TERMS IN U                 *
C*                                                               *
C*      METHOD                                                   *
C*         "SQUARE ROOT" DECOMPOSITION OF A SYMMETRIC MATRIX     *
C*                                                               *
C*      REFERENCES                                               *
C*         BERRY, R. D. "AN OPTIMAL ORDERING OF ELECTRONIC CIRCUIT*
C*         EQUATIONS FOR A SPARSE MATRIX SOLUTION" IEEE TRANSACTIONS*
C*         ON CIRCUIT THEORY VOL CT-18 NO. 1 JANUARY 1971 P. 40-50*
C*         FADDEEV, D. K. AND FADDEEVA, V. N., COMPUTATIONAL     *
C*         METHODS OF LINEAR ALGEBRA, W. H. FREEMAN AND CO., SAN *
C*         FRANCISCO, 1963, P. 144-147                           *
C*                                                               *
C*****************************************************************
      COMPLEX D(1),U(1),S(1)
```

```
      DIMENSION II(1),J(1)
      NM1=N-1
      DO 35 I=1,NM1
      JMIN=II(I)
      JMAX=II(I+1)-1
      S(I)=S(I)/D(I)
      IF(JMIN.GT.JMAX)GO TO 35
      DO 30 K=JMIN,JMAX
      JJ=J(K)
   30 S(JJ)=S(JJ)-S(I)*D(K)
   35 CONTINUE
      S(N)=S(N)/(D(N)*D(N))
      DO 45 K=1,NM1
      L=N-K
      JMIN=II(L)
      JMAX=II(L+1)-1
      IF(JMIN.GT.JMAX)GO TO 45
      DO 40 M=JMIN,JMAX
      IC=J(M)
   40 S(L)=S(L)-D(M)*S(IC)
   45 S(L)=S(L)/D(L)
      RETURN
      END
      SUBROUTINE RANDU(IX,IY,YFL)
C*****************************************************************************
C*                                                                           *
C*      PURPOSE                                                              *
C*         COMPUTES UNIFORMLY DISTRIBUTED RANDOM REAL NUMBERS BETWEEN        *
C*         0.0 AND 1.0 AND RANDOM INTEGERS BETWEEN ZERO AND                  *
C*         2**23. EACH ENTRY USES AS INPUT AN INTEGER RANDOM NUMBER          *
C*         AND PRODUCES A NEW INTEGER AND REAL RANDOM NUMBER.                *
C*                                                                           *
C*      USAGE                                                                *
C*         CALL RANDU(IX,IY,YFL)                                             *
C*                                                                           *
C*      DESCRIPTION OF PARAMETERS                                            *
C*         IX - FOR THE FIRST ENTRY IT IS MUST CONTAIN ANY ODD INTEGER       *
C*              NUMBER WITH SEVEN OR LESS DIGITS. AFTER THE FIRST            *
C*              ENTRY, IX SHOULD BE THE PREVIOUS VALUE OF IY COMPUTED        *
C*              BY THIS SUBROUTINE.                                          *
C*         IY - A RESULTANT INTEGER RANDOM NUMBER REQUIRED FOR THE           *
C*              NEXT ENTRY TO THIS SUBROUTINE. THE RANGE OF THIS             *
C*              NUMBER IS BETWEEN ZERO AND 2**23.                            *
C*         YFL- THE RESULTANT UNIFORMLY DISTRIBUTED, FLOATING POINT,         *
C*              RANDOM NUMBER IN THE RANGE 0.0 TO 1.0.                       *
C*                                                                           *
C*      REMARKS                                                              *
C*         THIS SUBROUTINE IS SPECIFIC TO THE DATACRAFT 6024.                *
C*         16645=2**14+2**6+2**2+1=5 MOD 8                                   *
C*         838e607=2**23-1 IS THE LARGEST INTEGER THE DC 6024                *
```

```
C*          CAN STORE                                                          *
C*          0.1152093E-6=1.0/8368607                                           *
C*                                                                             *
C*       REFERENCES                                                            *
C*          SYSTEM/360 SCIENTIFIC SUBROUTINE PACKAGE P. 77                     *
C*          MALLAREN AND MARSAGLIA, JACM 12, P. 83-89                          *
C*                                                                             *
C*******************************************************************************
      IY=IX*16807
      IF(IY)5,6,6
    5 IY=IY+2147483647
    6 YFL=IY
      YFL=YFL*.1152093E-6
      RETURN
      END
      SUBROUTINE GAUSS(IX,S,AM,V)                                        GAUS0000
C*******************************************************************************GAUS0001
C*                                                                             *GAUS0002
C*       PURPOSE                                                               *GAUS0003
C*          COMPUTES A NORMALLY DISTRIBUTED RANDOM NUMBER WITH A GIVEN         *GAUS0004
C*          MEAN AND STANDARD DEVIATION.                                       *GAUS0005
C*                                                                             *GAUS0006
C*       USAGE                                                                 *GAUS0007
C*          CALL GAUSS(IX,S,AM,V)                                              *GAUS0008
C*                                                                             *GAUS0009
C*       DESCRIPTION OF PARAMETERS                                             *GAUS0010
C*          IX - IX MUST CONTAIN AN ODD INTEGER NUMBER WITH SEVEN OR           *GAUS0011
C*               LESS DIGITS ON THE FIRST ENTRY TO GAUSS. THEREAFTER           *GAUS0012
C*               IT WILL CONTAIN A UNIFORMLY DISTRIBUTED INTEGER RANDOM        *GAUS0013
C*               NUMBER GENERATED BY THE SUBROUTINE FOR USE ON THE NEXT        *GAUS0014
C*               ENTRY TO THE SUBROUTINE.                                      *GAUS0015
C*          S  - THE DESIRED STANDARD DEVIATION OF THE NORMAL                  *GAUS0016
C*               DISTRIBUTION                                                  *GAUS0017
C*          AM - THE DESIRED MEAN OF THE NORMAL DISTRIBUTION                   *GAUS0018
C*          V  - THE VALUE OF THE COMPUTED NORMAL RANDOM VARIABLE              *GAUS0019
C*                                                                             *GAUS0020
C*       REMARKS                                                               *GAUS0021
C*          THIS SUBROUTINE USES RANDU WHICH IS MACHINE SPECIFIC               *GAUS0022
C*                                                                             *GAUS0023
C*       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                         *GAUS0024
C*          RANDU                                                              *GAUS0025
C*                                                                             *GAUS0026
C*       METHOD                                                                *GAUS0027
C*          USES 12 UNIFORM RANDOM NUMBERS TO COMPUTE NORMAL RANDOM            *GAUS0028
C*          NUMBERS BY THE CENTRAL LIMIT THEOREM. THE RESULT IS THEN           *GAUS0029
C*          ADJUSTED TO MATCH THE GIVEN MEAN AND STANDARD DEVIATION.           *GAUS0030
C*          THE UNIFORM RANDOM NUMBERS COMPUTED WITHIN THE SUBROUTINE          *GAUS0031
C*          ARE FOUND BY THE POWER RESIDUE METHOD.                             *GAUS0032
C*                                                                             *GAUS0033
C*       REFERENCES                                                            *GAUS0034
```

```
1301 C*          SYSTEM/360 SCIENTIFIC SUBROUTINE PACKAGE P. 77              *GAUS0035
1302 C*************************************************************************GAUS0036
1303       A=0.0                                                              GAUS0037
1304       DO 50 I=1,12                                                       GAUS0038
1305       CALL RANDU(IX,IY,Y)                                                GAUS0039
1306       IX=IY                                                              GAUS0040
1307    50 A=A+Y                                                              GAUS0041
1308       V=(A-6.0)*S+AM                                                     GAUS0042
1309       RETURN                                                             GAUS0043
1310       END                                                                GAUS0044
```

APPENDIX F
ITERATION COMPUTER PROGRAM (SOR)

A. Far Zone Mutual Impedance Between Moderate and Distantly Separated Sinusoidal Dipoles

Calculation time for filling in mutual impedance elements of A has been improved considerably using a far zone approximation for mutuals between general skew dipoles for dipole spacing $\geq 1\lambda$. Figure VI-1 shows the complete bistatic pattern for a 100 dipole array (8 dip/λ^3) using impedance calculations with two criteria for the far zone approximation; $\geq 1\lambda$, $\geq 1.5\lambda$. Also included is the pattern obtained without using the far zone approximations (exact). Agreement is quite good over these patterns and Table VI-1 gives additional data on backscatter, 360° bistatic average and computation times for this same array using the various methods including sparsed matrix solution using the 10% rule. Figure VI-2 compares bistatic patterns for the full and sparsed matrix calculations. The 10% rule resulted in approximately 90% zeroes in the A matrix. Calculations using the far zone approximation for N = 1000 resulted in an order of magnitude (1/10) savings in time to compute A. The predicted time without the approximation was 10-12 hours, whereas the actual time using far zone mutuals was $\sim$ 1 hour. Any errors that occur, due to this far zone approximation, are not likely to affect the scattering and scintillation statistics.

These simplified mutual impedances are computed based on the far zone electric fields of the two sinusoidal dipoles. Consider dipole #1 located in Fig.VI-3a to have far zone electric field given by

$$\text{(VI-1)} \qquad \bar{E}(r,\theta,\phi) \simeq -\frac{jk_o^2\eta_o}{4\pi}\left\{\frac{e^{-jk_o r}}{k_o r}\left(1-\frac{j}{k_o r}-\frac{1}{k_o^2 r^2}\right)\bar{F}_1(\theta_1,\phi_1)+2j\,\frac{e^{-jk_o r}}{k_o^2 r^2}\left(1-\frac{j}{k_o r}\right)G_1(\theta_1,\phi_1)\,\hat{r}\right\}$$

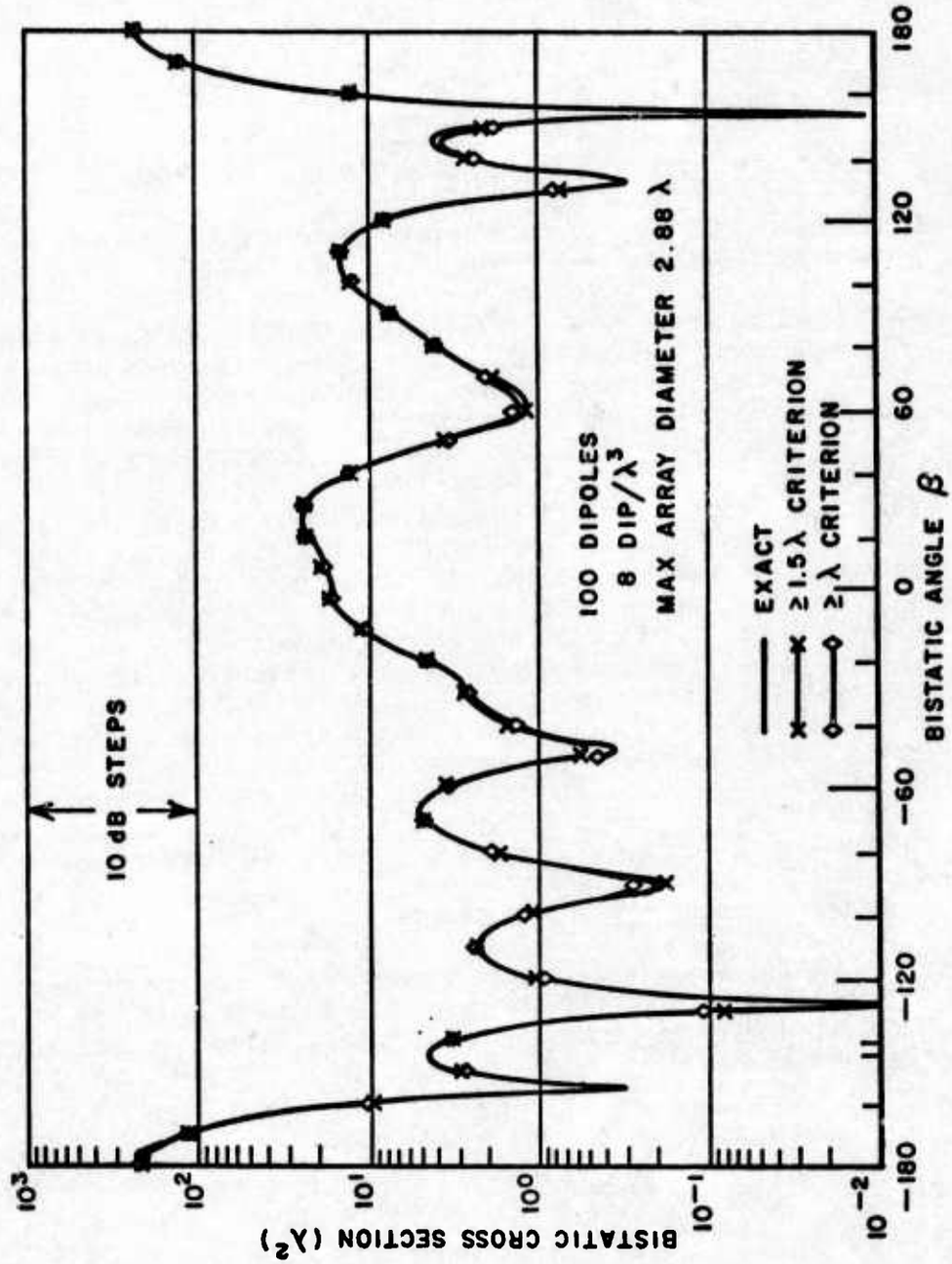


Figure VI-1. Bistatic scatter cross section for 100 dipole array (8 dip/λ^3) comparing exact and far zone A matrix calculations.

TABLE VI-1

DATA COMPARISON FOR DIFFERENT MUTUAL IMPEDANCE CALCULATIONS FOR 100 DIPOLE ARRAY (8 dip/λ^3).

All cross section results obtained using CROUT type (SQROT) solutions.

Calculations	Backscatter σ	360° Bistatic Average	Computation Time for Z
Far Zone* $\geq 1\lambda$	$16.4\lambda^2$	$21.3\lambda^2$	88 sec
Far Zone* $\geq 1.5\lambda$	16.6	21.5	167
Exact**	16.4	21.4	607
Sparsed*** (10%)	9.07	26.4	607

* Mutuals between dipoles spaced $\geq 1\lambda$ (1.5λ) computed using far zone approximation.

** Mutuals between all dipoles computed using closed form expressions of Richmond (ESL Tech Report #2708-4, 1969).

*** Mutuals falling below 10% of diagonal (a_{ii}) are set to zero.

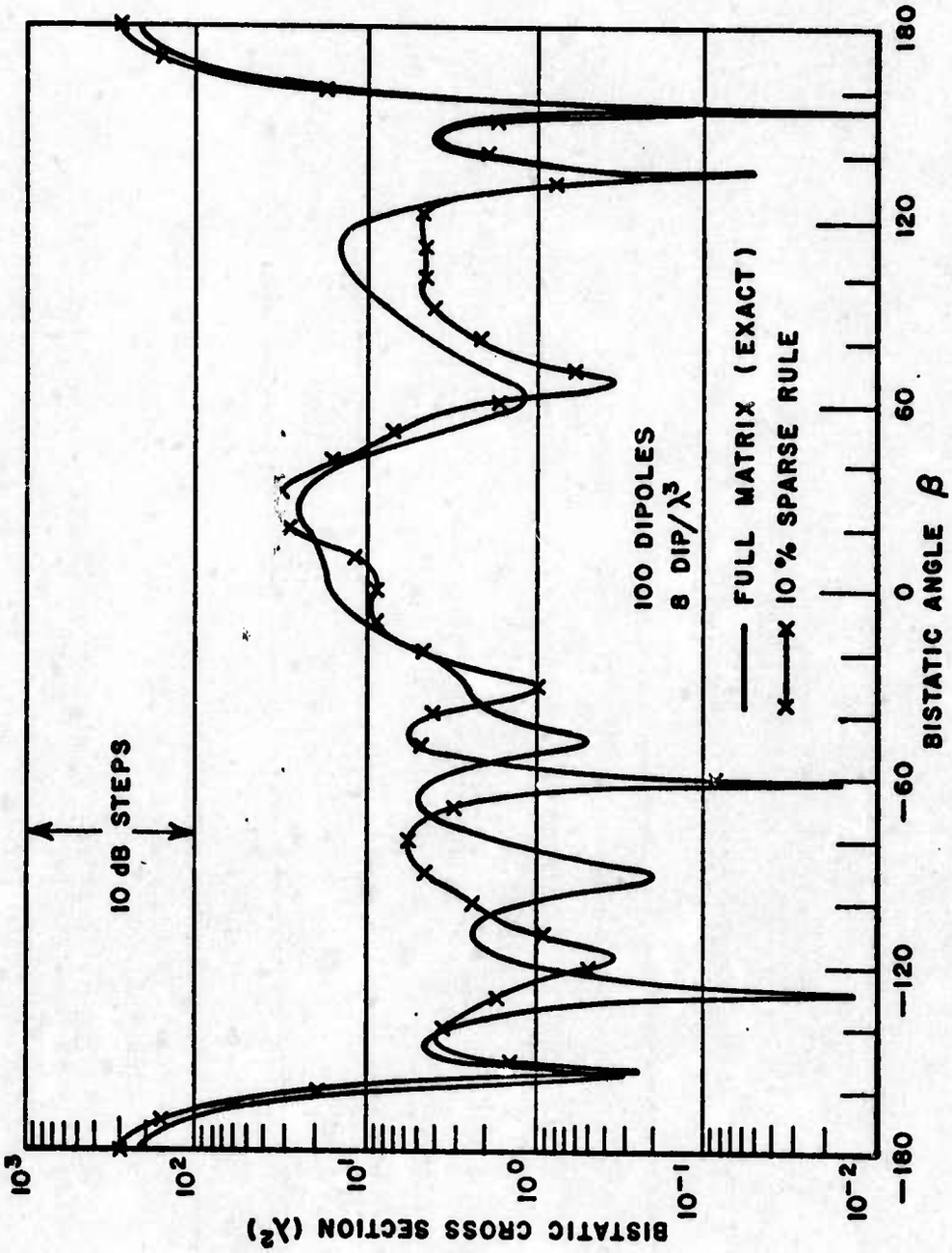
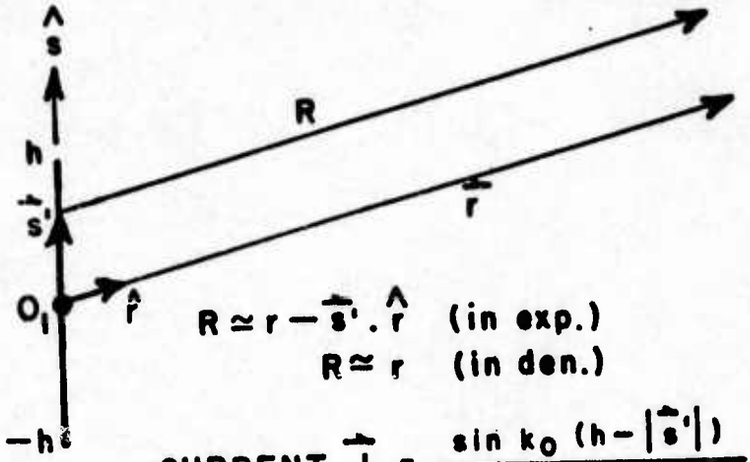


Figure VI-2. Bistatic scatter cross section for 100 dipole array (8 dip/λ^3) comparing full (exact) A matrix and sparse A matrix using 10% sparsing rule, i.e., $a_{ij} = 0$ if $|a_{ij}| < .1\ |a_{ii}|$, i = 1, 2, ..., N.

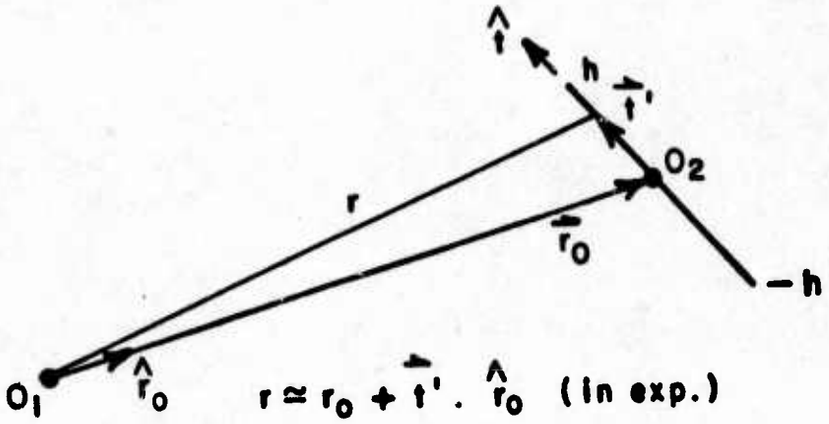


$R \simeq r - \vec{s}' \cdot \hat{r}$ (in exp.)

$R \simeq r$ (in den.)

CURRENT $\vec{J}_1 = \dfrac{\sin k_0 (h - |\vec{s}'|)}{\sin k_0 h} \hat{s}, \quad -h \le s' \le h$

(a) DIPOLE #1 CENTERED AT O_1



$r \simeq r_0 + \vec{t}' \cdot \hat{r}_0$ (in exp.)

$r \simeq r_0$ (in den)

CURRENT $\vec{J}_2 = \dfrac{\sin k_0 (h - |t'|)}{\sin k_0 h} \hat{t}, \quad -h \le t' \le h$

(b) DIPOLE #2 CENTERED AT O_2

Figure VI-3. Sinusoidal dipoles and far zone approximations.

where the field pattern functions are given by the usual radiation integrals, i.e.,

$$\text{(VI-2)} \qquad \bar{F}_1(\theta,\phi) = \int_1 [\bar{J}_1 - (\hat{r} \cdot \bar{J}_1)\,\hat{r}]\, e^{jk_o \bar{s}' \cdot \hat{r}}\, ds'$$

and

$$\text{(VI-3)} \qquad G_1(\theta,\phi) = \int_1 (\hat{r} \cdot \bar{J}_1)\, e^{jk_o \bar{s}' \cdot \hat{r}}\, ds' \quad .$$

Recall from Chapter II the mutual impedance between two dipoles is defined by the following formula,

$$\text{(VI-4)} \qquad a_{12} = - \int_2 \bar{J}_2 \cdot \bar{E}_1\, dt' \quad .$$

Consider dipole #2 in Fig.VI-3(b) to be located by $\bar{r}_o$ with respect to O_1. Then, for moderate to large r_o, Eq. (H-4) simplifies to

$$\text{(VI-5)} \qquad a_{12} \approx \frac{j k_o^2 \eta_o}{4\pi} \left\{ \frac{e^{-jk_o r_o}}{k_o r_o} \left(1 - \frac{j}{k_o r_o} - \frac{1}{k_o^2 r_o^2} \right) \right.$$

$$\bar{F}_1(\theta_1,\phi_1) \cdot \bar{F}_2(\theta_2,\phi_2) + 2j \frac{e^{-jk_o r_o}}{k_o^2 r_o^2} \left(1 - \frac{j}{k_o r_o} \right)$$

$$\left. G_1(\theta_1,\phi_1)\, G_2(\theta_2,\phi_2) \right\}$$

where pattern functions of dipole #2 are

$$\text{(VI-6)} \qquad \bar{F}_2(\theta_2,\phi_2) = \int_2 \bar{J}_2\, e^{-jk_o \bar{t}' \cdot \hat{r}_o}\, dt'$$

and

$$(VI\text{-}7) \qquad G_2(\theta_2,\phi_2) = \hat{r}\cdot\int_2 \bar{J}_2\, e^{-jk_o\bar{t}'\cdot\hat{r}_o}\, dt' \quad .$$

The (θ_1, ϕ_1) angular variations are measured with respect to origin O_1 (center dipole #1) and similarly, the (θ_2, ϕ_2) angles are measured with respect to origin O_2 (center dipole #2).

B. Computer Programs

Note that the "INCLUDE" statement and "CALL ASSIGN" statement in the following computer programs are special commands implemented on the ESL computer. Their functions are explained in the following paragraphs.

Nine logical unit numbers (0-8) are available for use with Fortran. The nine logical unit numbers correspond to nine locations in the I/O device assignment table of the Fortran library I/O system. Before I/O operations can proceed each device or disk file to be used by a Fortran program must either have been assigned to a logical unit number by the user's program or have been assigned by default. The device assignment table is initialized to the default assignments according to the table. Logical unit numbers are referenced in READ and WRITE statements, to specify the device or file on which the read or write operation is to be performed.

Examples:

```
READ (8,-)x,y,z
WRITE(0)(a(i),i=1,100)
WRITE(6,40)a2,zL3
```

Assuming the default assignments are used the first example specifies that three variables, x,y, and z be read from the terminal keyboard in free format. When input from the terminal keyboard is specified the terminal bell is rung to indicate that the READ statement has been executed and the user is expected to supply the input data. The input data should be followed by a carriage return. In the second example the WRITE statement specifies that 100 words be written unformatted (binary format) on the magnetic tape unit 0. The WRITE statement of the third example specifies that formatted output be performed to the temporary output file named .OUT.

Any of the system I/O devices with the exception of the card reader (.CDR) and the line printer (.LPT), or any named disk file can be assigned to any logical unit number from 0 to 7 by means of an OPEN statement or a CALL ASSIGN statement. The user's terminal is permanently assigned to logical unit 8 and cannot be altered by the Fortran program. Logical unit 8 is unavailable for assignment. The format for CALL ASSIGN is:

```
CALL ASSIGN(FILE,USER,LU,$N)
```

where, FILE is a literal constant of from one to six characters (see "Literal Constants" section 2-3 of the Fortran manual) or a floating point variable name or an array name containing a one to six character file or device name. USER is a literal constant of from one to six characters in length, a floating point variable name or array name containing a one to six character user name or floating point zero (0.0).LU is a logical unit number from 0 through 7. N is an optional Fortran statement label number. FILE and USER, if less than six characters, must be filled with trailing spaces to make six characters. If the names are four or more characters in length this is done automatically. If not the names should be extended to be at least four characters in length by adding trailing spaces.

If the array or floating point variable is used for a name, data may be assigned to it using a READ statement with an A format or by means of a literal constant in a DATA statement. If floating point zero is used as the USER calling paramenter the user name under which the program is being executed is assumed. If the FILE calling parameter specifies the name of a non-disk device the USER parameter is not used but must be present. A floating point zero may be used. Devices .LPT and .CDR are not available for assignment. The function of the CALL ASSIGN statement is to cause the specified name (file and user) to be placed in the Fortran library I/O device assignment table in the location corresponding to the specified logical unit number thereby assigning that name to the logical unit number. If the file or device which was previously assigned as the specified logical unit has been engaged in I/O activity and has not been closed it will be automatically closed by call assign before a new assignment is made. If the parameter $N is present, control will be returned to the statement having the label N if an error occurs. An error will be indicated if a non-disk device has been reserved by another user (i.e., the device is busy).

Default logical unit assignments

Logical Unit	Device Number	Description Assignment
0	.MT0	magnetic tape drive 0
1	.DSK	temporary (disk) scratch file
2	.MT1	magnetic tape drive 1
3	.LU3	temporary (disk) scratch file
4	.LU4	temporary (disk) scratch file
5	.IN	temporary disk input file
6	.OUT	temporary disk output file
7	.LU7	temporary (disk) scratch file
8	terminal only --	not reassignable

The INCLUDE statement provides the user with a means of specifying, as part of a Fortran program, that one or more binary object files be loaded along with the program containing the include statement during thyload operation. The INCLUDE statement is used primarily to load subroutines called by Fortran programs or subprograms but not contained in either of the Fortran libraries.

Use of the INCLUDE statement provides the following benefits:

1) Frequently used subroutines (not contained in the Fortran libraries) need not be edited in source language form into each program which calls them. This saves both editing time and file space. File space is conserved because only one compiled binary object version of a subroutine need be kept in the file system even though several programs may call it.

2) Useful subprograms may be easily shared among a number of users since any file in any user's directory may be specified in an INCLUDE statement.

3) A subroutine which is used by many programs may be altered and recompiled without necessitating the recompilation of any of the calling program. (This, of course, also holds for a single calling program).

The format for the INCLUDE statement is:

INCLUDE NAME1,USER1;NAME2,USER2;...;NAMEN,USERN

(1) File Name: TESBGF,SYSAC

Main Program TESBGF computes and stores elements of impedance matrix (A) and sets up SOI submatrices. Include statements (lines 5-7) are as follows:

PAPER,SYSAC	CALL PAPER advances output paper on LU-6 with 1H1 format.
ZGS,SYSAC	subroutine used by ZSKEWF subroutine
CLDB,SYSAC	SUBROUTINE UNICLD
ADEXB,SYSAC	contains LDG load-go-execute subroutine
STGETB,SYSAC	contains SUBROUTINES STOR, CLSTOR and SUBFUNCTION GET
ZSKEWB,SYSAC	SUBROUTINE ZSKEWF

Dimensioned variables:

IJ	integer array denoting m_i for N SOI submatrices
X,Y,Z	coordinates of dipole i
XX,YY,ZZ	coordinates of dipole j
ZR	one row of impedance matrix (A)
XR	one row of impedance matrix separating ZR into real and imaginary parts (See Equivalence statement, line 12.)
ZDIA	diagonal block of impedance matrix - only 1 x 1 for 2 segment model

Assigned files (lines 22-25):

ZZAT	LU 3 contains subscripts of SOI submatrices (Binary format).
ZIND	LU 5 saves iteration method code (0 = J, 1 = SOR, 2 = SOI) and OM (omega) relaxation factor if SOR is used (formated data).
ZOUT	LU 6 contains error messages (lines 149 or 152) when TESBGF defaults to CALL EXIT or contains bistatic angle, increment and iteration start information (formated data) for TESBG4.
DATA	LU 1 saves number of dipoles (NOD), number of segments (NOS), array density (DEN), influence coefficient (CF), aspect (TH,PHI), time (IT), max submatrix size (ICK) and max broadside mode voltage (EBS) (binary format).

Input parameters are as follows:

HAFBIS	one-half bistatic pattern sector (degrees); if <0, bistatic pattern is not computed.
DPHZ	bistatic angle increment (degrees)
ISTART	iteration starting step (k).
NOD	number of dipoles in array
NOS	number of segments per dipole
DEN	density of array (dip/λ^3)
METH	iteration method code (0 = J, 1 = SOR, 2 = SOI)
OM	relaxation factor (SOR)
CF	influence coefficient (SOI)
TH,PHI	aspect angle in degrees (θ_o,ϕ_o)

Variable names used:

NOSP	number of coordinate points per dipole

NOSM	number of modes per dipole
N	total number of modes
AK	dipole radius
HK	dipole half-length
DK	segment length
RS	overall radius of array (R_0)
IP	random number generator starting number (IBM-SSP RANDU); 7 digit odd number preferable.

Lines 77 through 139 compute positions of array dipoles (UNICLD, lines 80 and 102) and mutual impedances are then computed by ZSKEWF in line 121. Note, RHO is center to center distance between i^{th} and j^{th} dipoles and lines 113 to 120 prescribe type of impedance calculation is to be used, e.g., INT < 0 specifies "far zone" calculations (see Appendix H), INT = 0 requires exact "closed form" integration and INT = 4 chooses Simpson's Rule 4-point integration. Lines 128 to 132 apply SOI influence criterion to generate SOI submatrices. Impedance calculations are only necessary for the upper triangular elements of the impedance matrix (A is symmetric). However, the CALL STOR (line 137) packs and stores full rows of matrix on disk storage. The symmetric lower triangular elements of the A matrix are read into the ZR array (for the i^{th} row) in lines 89 to 97. SUBFUNCTION GET (line 97) retrieves and unpacks previously stored data from the disk to fill in the i^{th} row for $j < i$. The CALL CLSTOR (line 142) permanently closes all "packed" storage disk files. Once this is done, files can only be read using the GET subfunction contained in binary file GETB,SYSAC found in main program TESBG4,SYSAC. Line 157 automatically executes TESBG4,SYSAC (BIGCO,SYSAC) which solves the system of equations via the prescribed iterative method.

(2) File Name: TESBG1,SYSAC

Main program TESBG1 is identical to TESBGF,SYSAC except it does not use "far zone" calculations for mutual impedances, but instead uses only closed form and Simpson's four point integration.

(3) File Name: TESBG4,SYSAC

Main program TESBG4 solves simultaneous system using either J, SOR or SOI iteration. Compiled (binary) version of TESBG4 must be under file name BIGCO,SYSAC and is executed by either TESBGF, TESBG1 or TESBG5. All input data for TESBG4 are available on disk files ZZAT, ZINP and DATA. Parameters are identical to

definitions given for TESBGF, except for ITE which corresponds to iteration method code (METH).

Assigned files (lines 21-29):

PLOT	LU ϕ will contain bistatic cross section pattern output data (formated data).
ZZAT	LU 3 same as for TESBGF
ZADT	LU 4 sample of output which can be read without removing program from "background" running mode (formated data).
ZINP	LU 5 same as for TESBGF
DATA	LU 1 same as for TESBGF
ZOUT	LU 6 contains output data from iteration (formated data).

All iterations are performed between lines 93 to 222. CALL EZFFD sets up excitation column (b) and lines 98-129 solve N subsystems using the SOI submatrices and SUBROUTINE SQROT (Cholesky). Lines 130-156 compute the residuals for SOI (line 148) or solve system via J or SOR (lines 150-153). Lines 157-159 save "latest" solution information in the event the iteration is stopped and restarted with ISTART $\neq$ 1. Lines 160-175 compute residual 1-norm $||r^{(k)}||_1$ and lines 176 to 196 compute bistatic cross section pattern SIGG (λ^2). Bistatic pattern is written into file name PLOT. Line 197 computes bistatic pattern average (SIG) over sector and line 198 computes normalized average residual $\epsilon^{(k)}$. Lines 199-204 compute total scatter cross section σ_T(TSC) via Forward Scatter Theorem. Lines 205-219 write and rewrite output data in following form:

IPEP	iteration k
SIG	bistatic average $\langle\sigma\rangle$
A	$\epsilon^{(k)}$ norm
TSC	total scatter cross section σ_T
ETH	forward scattered electric field (complex)
CI(N)	dipole mode current on N^{th} dipole (complex)

Reading and writing into LU 4 and 5 allows the latest accumulation of output data to be observed without removing program from background. Final output is accumulated in file ZOUT and is closed by escaping (ESC) program. Note, σ backscatter data are available only in PLOT as the "center" data point in the bistatic pattern.

(4) File Name: TESBG5,SYSAC

Main program TESBG5 is a utility program to be used to execute TESBG4 (BIGCO) when a new impedance matrix is not required. TESBG5 has two modes of operation. Both modes request input data (same as TESBGF) in line 18, then PAUSE (line 19). A transfer directly to TESBG4 is affected at this point, by pressing ESCAPE (ESC); however, file ZINP must already have the prescribed method code written on line 1 (and relaxation factor on line 2) in any format. This mode is especially useful when changing bistatic pattern cut or restarting SOR with a different relaxation factor. The second mode is initiated by pressing RETURN after the PAUSE. The "old data" will be displayed and a request for new parameters will occur. This mode can be used to change all parameters including SOI matrices; however, it is most used for changing only the desired aspect angle.

(5) File Name: UNICLD, SYSAC

SUBROUTINE UNICLD computes position and orientation of one dipole-at-a-time. The calling parameters are as follows:

IX	random number initialization on entry and next random number in sequence on return to main program.
RS	random array max radius
HK	dipole half-length
DK	segment length
NOSP	number of coordinate points per dipole
X,Y,Z	dipole coordinates returned to main program

Subroutine call to RANDU uses following I/O parameters:

IX	random number initialization
IY	next random number
A1	uniformly distributed random variable in range $0 \leq A1 \leq 1$.

(6) File Name: ZSKEWF,SYSAC

SUBROUTINE ZSKEWF computes mutual impedance between two general skew dipoles. Calling parameters are as follows:

XA,YA,ZA XB,YB,ZB XC,YC,ZC	three coordinate points of dipole i
X1,Y1,Z1 X2,Y2,Z2 X3,Y3,Z3	three coordinate points of dipole j
INT	type of calculation; INT < 0 = far zone approx, INT = 0 closed form integrals, INT = 4 Simpson's integration (4 pt.).
CDK	Cos (DK)
SDK	Sin (DK)
D	dipole half-length
R	center-to-center spacing between i^{th} and j^{th} dipoles.
Z12	mutual impedance returned to main program.

Lines 5-31 calculate far zone approximation and lines 32-50 calculate "exact" values using SUBROUTINE ZGS, a standard Richmond subroutine for calculating mutual impedances between two general skew monopoles.

(7) File Name: EZFFD,SYSAC

SUBROUTINE EZFFD computes far zone electric fields scattered from random array assuming one ampere of current flows on each dipole. Calling parameters are as follows:

X,Y,Z	array dipole coordinates (dummy variables)
ET,EP	$\hat{\theta}$ polarized and $\hat{\phi}$ polarized electric fields (dummy variables)
ETT	$\hat{\theta}$ polarized electric field (N dimensional) returned to main program.
NOD	same as TESBGF

NOSM	same as TESBGF
NOSP	same as TESBGF
IP	same as TESBGF
RS	same as TESBGF
HK	same as TESBGF
DK	same as TESBGF
CTH	Cos (θ) scattering angle
STH	Sin (θ) scattering angle
PHI	ϕ scattering angle (degrees).

Subroutine regenerates random array with UNICLD and calculates the far zone electric field of each dipole using SUBROUTINE ZFFD, a standard Richmond routine for computing the far zone electric field of a single skew dipole located near the origin.

(8) File Name: STGETS,SYSAC

SUBROUTINES STOR, CLSTOR and SUBFUNCTION GET are listed here in assembler programming language. The assembled version of this program must be included in TESBGF and TESBG1 under file name STGETB,SYSAC.

(9) File Name: GETS,SYSAC

SUBFUNCTION GET is listed here in assembler language. The assembled version of this program must be included in TESBG4 under file name GETB,SYSAC.

```
C  THIS PROGRAM COMPUTES MUTUAL IMPEDANCE MATRIX FOR N ELEMENT
C  RANDOM ARRAY OF RESONANT DIPOLES AND PROCESSES THE MATRIX
C  FOR USE IN TESDOF WHICH COMPUTES THE RADAR CROSS SECTION BY
C  JACOBI, SOR AND SOI ITERATION METHODS.....
      INCLUDE PAPER.SYSAC(ZGS.SYSAC
      INCLUDE CLOD.SYSAC(AUEXB.SYSAC
      INCLUDE SIGFTN.SYSAC(ZSHEWR.SYSAC
      INTEGER IJ(115)
      DIMENSION X(5),XX(5),Y(5),YY(5),Z(5),ZZ(5)
      COMPLEX ZR(1200)
      DIMENSION XR(2400)
      EQUIVALENCE (ZR(1),XR(1))
      COMPLEX CJI,P11,P12,P21,P22
      COMPLEX ECC,ZDIA(5,5)
      DATA PI,TP,ETA/3.14159,6.283185,376.727/
      DATA CJI/(0.0,-.3068BF-2)/
      CALL ESC(31000)
      WRITE(6,7)
    7 FORMAT(1X,'HALF BISTATIC ANG, INCR & ISTART =')
      READ(8,-)HAFBIS,DPHZ,ISTART
      CALL FERK (0)
      CALL ASSIGN(6HZZAT,6HSYSVC(,6)
      CALL ASSIGN(6HZIMP,6HSYSVC(,5)
      CALL ASSIGN(6HZOUT,6HSYSVC(,6)
      CALL ASSIGN(6HDATA,6HSYSVC(,1)
      IDM=115
      WRITE(6,250)
  250 FORMAT('READ NOD,NOS,DEN,METH =')
      READ(8,-)NOD,NS,DEN,METH
      WRITE(6,2)NOD,NS,DEN
      WRITE(6,-)METH
      IF(METH.NE.1)GO TO 4
      WRITE(6,5)
    5 FORMAT(' READ OMEGA =')
      READ(8,-)OM
      WRITE(5,-)OM
    2 FORMAT(1X,'NEL=',I4,2X,'NSRC=',I3,2X,'ELE DENSITY=',1E9.3)
    4 WRITE(6,9)
    9 FORMAT('READ Z THRESHOLD =')
      READ(8,-)CF
      WRITE(6,3)CF
    3 FORMAT(1X,'CF=',1E9.3)
   13 NOS=NS
      WRITE(6,20)
   20 FORMAT(' READ TH, PHI =')
      READ(8,-)TH,PHI
      WRITE(6,884)TH,PHI
  884 FORMAT('TH, PHI =',2E9.3)
      CALL CLASSN
      THR=TH*.017453
      PHR=PHI*.017453
      CAI=SIN(THR)*COS(PHR)
      CBI=SIN(THR)*SIN(PHR)
      CGI=COS(THR)
      CALL GETCP(IT)
```

```
      NOSP=NOS+1
      NOSM=NOS-1
      NM=NOD*NOS
      NP=NOD*NOSP
      NMD=NOSP-NOS/2
      HL=.2567
      TP=TP/10
      N=NOD*NOSM
      ZOS=4600*TP*TP
      AL=HL/100
      AK=AL*TP
      HK=HL*TP
      DK=2*HK/NOS
      CDK=COS(DK)
      SDK=SIN(DK)
      ANOS=NOD/DEN
      RS=.62035*ANOS**(1/3.)*TP
      IP=7305111
      NI=NOS
      ICK=0
      IX=IP
      DO 80 I=1,N
      IF(NI.LT.NOS)GO TO 52
      IY=IX
      CALL UNICLO(IX,RS,HK,DK,NOSP,X,Y,Z)
      NI=1
50 I1=NI
      I2=NI+1
      I3=NI+2
      NJ=NOS
      IDI=(I-1)/NOSM+1
      IN=0
      IF(I.EQ.1)GO TO 60
      ISL=I-1
      DO 40 INZ=1,ISL
      KP=I+N*(INZ-1)
      KI=KP+KP-1
      JP=INZ+INZ-1
      DO 40 IM=0,1
      JP=JP+IP
      KI=KI+IM
40 YR(JP)=GLT(KI)
60 DO 80 J=1,N
      IF(J.LT.I)GO TO 84
      IDJ=(J-1)/NOSM+1
      IF(NJ.LT.NOS)GO TO 53
      CALL UNICLO(IY,RS,HK,DK,NOSP,XX,YY,ZZ)
      NJ=1
53 J1=NJ
      J2=NJ+1
      J3=NJ+2
      INT=0
      IDIA=I-NOSM*(IDI-1)
      JDIA=J-NOSM*(IDJ-1)
      IF(IDI.NE.IDJ)GO TO 51
```

```
      IF(IDI.NE.1)GO TO 76
      GO TO 56
   51 IJT=-1
      RHO=SQRT((XY(NMD)-X(NMD))*(XX(NMD)-X(NMD))+(YY(NMD)-Y(NMD))*
     *(YY(NMD)-Y(NMD))+(ZZ(NMD)-Z(NMD))*(ZZ(NMD)-Z(NMD)))
      IF(RHO.GT.CK)GO TO 54
      INT=0
      GO TO 56
   54 IF(RHO.GT.TP)GO TO 56
      INT=4
   56 CALL ZSKLWF(X(I1),Y(I1),Z(I1),X(I2),Y(I2),Z(I2),X(I3),Y(I3),
     *Z(I3),YY(I1),YY(I1),ZZ(I1),YY(I2),YY(I2),ZZ(I2),XX(I3),YY(I3)
     *,ZZ(I3),INT,CDK,SDK,CK,RHO,ZR(J))
      IF(IFI.EQ.IDJ)ZDIA(IDIA,JDIA)=ZR(J)
      ZDM=CABS(ZDIA(IDIA,IDIA))
      GO TO 84
   76 ZR(J)=ZDIA(IDIA,JDIA)
   84 IF(CABS(ZR(J)).LT.CF*ZDM)GO TO 88
      IN=IN+1
      IF(IN.GT.ICK)ICK=IN
      IF(ICK.GT.IDM)GO TO 88
      IJ(IN)=J
   80 NJ=NJ+1
      IF(ICK.GT.IDM)GO TO 81
      WRITE(8)IN,(IJ(JK),JK=1,IN)
   81 IR=N+2
      CALL STOR(ZR(1),IR,IERR)
      IF(IERR.NE.0)GO TO 14
   82 NI=NI+1
      CLOSE 8
      CALL CLSTOR
      CALL GETCP(JT)
      IT=(JT-IT)/100
      EBS=.99267496
      WRITE(1)NOD,NS,DEN,CF,TH,PHI,IT,ICK,EBS
      IF(ICK.GT.IDM.AND.METH.EQ.P)GO TO 15
      GO TO 998
   14 WRITE(6,411)
  411 FORMAT(1X,'OVERFLOW RANGE IN SUBZ')
      GO TO 1000
   15 WRITE(6,412)ICK
  412 FORMAT('MAX DIM EXCEEDED ICK=',I4)
      GO TO 1000
  998 CLOSE 6
  999 CALL ASSIGN(4HZOUT,6HSYSVC(,6)
      WRITE(6,-)HAFBIS,DPHZ,ISTART
      CALL LOG(5HBIGCO,5HSYSAC)
 1000 CALL EXIT
      END
```

```
C  THIS PROGRAM COMPUTES MUTUAL IMPEDANCE MATRIX FOR N ELEMENT
C  RANDOM ARRAY OF RESONANT DIPOLES AND PROCESSES THE MATRIX
C  FOR USE IN TESBCL WHICH COMPUTES THE RADAR CROSS SECTION BY
C  JACOBI, SOR AND SOI ITERATION METHODS.....
      OPTIONS 32K
      INCLUDE PAPER,SYSAC;IZGS,SYSAC
      INCLUDE CLOK,SYSAC;AUFXB,SYSAC
      INCLUDE STGETR,SYSAC
      DIMENSION XC(13340)
      COMMON XC
      INTEGER IS(1151),INJ(1200),IJJ(115)
      DIMENSION X(5),XX(5),Y(5),YY(5),Z(5),ZZ(5)
      COMPLEX ZR(1200)
      DIMENSION XR(2400)
      EQUIVALENCE (ZR(1),XR(1))
      COMPLEX CC(6670)
      EQUIVALENCE (XC(1),CC(1))
      COMPLEX CJI,P11,P12,P21,P22
      COMPLEX ECC,ZDTA(5,5)
      DATA PI,TP,ETA/3.14159,6.28318,376.727/
      DATA CJI/(0.0,-.530888E-2)/
      CALL ESC(%1000)
      WRITE(*,7)
    7 FORMAT(1X,'HALF BISTATIC ANG, INCR & ISTART =')
      READ(8,-)NAFBIS,DPHZ,ISTART
      CALL FERR (0)
      CALL ASSIGN(4HFZAT,6HSYSVC1,5)
      CALL ASSIGN(4HFZINP,6HSYSVC1,5)
      CALL ASSIGN(4HFZOUT,6HSYSVC1,6)
      CALL ASSIGN(4HDATA,6HSYSVC1,1)
      IUM=115
      WRITE(*,250)
  250 FORMAT('READ NOD,NOS,DEN,METH =')
      READ(8,-)NOD,NS,DEN,METH
      WRITE(6,2)NOD,NS,DEN
      WRITE(5,-)METH
      IF(METH.NE.1)GO TO 4
      WRITE(*,5)
    5 FORMAT(' READ OMEGA =')
      READ(8,-)OM
      WRITE(5,-)OM
    9 FORMAT(1X,'ELF=',1I4,2X,'WSFG=',1I2,2X,'ELE DENSITY=',1E9.3)
    4 WRITE(*,9)
    9 FORMAT('READ Z THRESHOLD =')
      READ(8,-)CF
      WRITE(*,8)CF
    8 FORMAT(1X,'C=',1E9.3)
   13 NUS=NS
      WRITE(*,20)
   20 FORMAT(' READ TH, PHI =')
      READ(8,-)TH,PHI
      WRITE(*,884)TH,PHI
  884 FORMAT('TH, PHI =',2E9.3)
      CALL DFASSN
      THR=TH*.017453
```

```
      PHR=PHI*.017453
      CAI=SIN(THR)*COS(PHR)
      CBI=SIN(THR)*SIN(PHR)
      CCI=COS(THR)
      CALL GETCOF(IT)
      NOSP=NOS+1
      NOSM=NOS-1
      NM=NOD*NOS
      NP=NOD*NOSP
      NAD=NOSP*NOS/2
      HL=.2567
      PK=TP*TP/100
      N=NOD*NOSM
      ZOS=3600*TP*TP
      AL=HL/100
      AK=AL*TP
      HK=HL*TP
      DK=2*HK/NOS
      CDK=COS(DK)
      SDK=SIN(DK)
      ANOS=NOD/DLN
      PS=.62035*ANOS**(1/3.)*TP
      IP=7387111
      NI=NOS
      ICK=0
      IX=IP
      DO 82 I=1,N
      IF(NI.LT.NOS)GO TO 52
      YY=IX
      CALL UNICLD(IX,RS,HK,DK,NOSP,X,Y,Z)
      N1=1
   59 I1=NI
      I2=NI+1
      I3=NI+2
      NJ=NOS
      IOI=(I-1)/NOSM+1
      INJ(I)=0
      IF(I.EQ.1)GO TO 50
      ISL=I-1
      DO 40 INZ=1,ISL
      KP=I+N-(INZ-1)
      KI=KP+KP-1
      JP=INZ+INZ-1
      DO 40 IM=0,1
      JP=JP+IM
      KI=KI+IM
   40 XR(JP)=GET(KI)
   50 DO 80 J=1,N
      IF(J.LT.I)GO TO 84
      IOJ=(J-1)/NOSM+1
      IF(NJ.LT.NOS)GO TO 53
      CALL UNICLD(IY,RS,HK,DK,NOSP,XX,YY,ZZ)
      NJ=1
   53 J1=NJ
      J2=NJ+1
```

```
      J3=NJ+2
      INT=0
      IDIA=I-NOSM*(IDI-1)
      JDIA=J-NOSM*(IDJ-1)
      IF(IDI.NE.IDJ)GO TO 51
      IF(IDI.NE.1)GO TO 76
      GO TO 56
   51 INT=4
      DO 55 MI=I1,I3
      DO 55 MJ=J1,J3
      RHOS=(XX(MJ)-X(MI))*(XX(MJ)-X(MI))+(YY(MJ)-Y(MI))*(YY(MJ)-
     &Y(MI))+(ZZ(MJ)-Z(MI))*(ZZ(MJ)-Z(MI))
      IF(RHOS.GT.CK)GO TO 55
      INT=0
      GO TO 56
   54 CONTINUE
   56 ZR(J)=(0.0,0.0)
      CALL ZGS(X(I1),Y(I1),Z(I1),X(I2),Y(I2),Z(I2),XX(J1),YY(J1)
     &,ZZ(J1),XX(J2),YY(J2),ZZ(J2),AK,DK,COK,SOK,DK,SOK,INT,P11,P12,
     &P21,P22)
      ZR(J)=ZR(J)+P22
      CALL ZGS(X(I1),Y(I1),Z(I1),X(I2),Y(I2),Z(I2),XX(J2),YY(J2)
     &,ZZ(J2),XX(J3),YY(J3),ZZ(J3),AK,DK,COK,SOK,DK,SOK,INT,P11,P12,
     &P21,P22)
      ZR(J)=ZR(J)+P21
      CALL ZGS(X(I2),Y(I2),Z(I2),X(I3),Y(I3),Z(I3),XX(J1),YY(J1)
     &,ZZ(J1),XX(J2),YY(J2),ZZ(J2),AK,DK,COK,SOK,DK,SOK,INT,P11,P12,
     &P21,P22)
      ZR(J)=ZR(J)+P12
      CALL ZGS(X(I2),Y(I2),Z(I2),X(I4),Y(I3),Z(I3),XX(J2),YY(J2)
     &,ZZ(J2),XX(J3),YY(J3),ZZ(J3),AK,DK,COK,SOK,DK,SOK,INT,P11,P12,
     &P21,P22)
      ZR(J)=ZR(J)+P11
      IF(IDI.EQ.IDJ)ZDIA(IDIA,JDIA)=ZR(J)
      ZDM=CABS(ZDIA(IDIA,IDIA))
      GO TO 84
   76 ZR(J)=ZDIA(IDIA,JDIA)
   84 IF(CABS(ZR(J)).LT.CF*ZDM)GO TO 80
      INJ(I)=INJ(I)+1
      IN=INJ(I)
      IF(IN.GT.IDM)GO TO 15
      IF(IN.GT.ICK)ICK=IN
      IJ(IN)=J
   80 NJ=NJ+1
      WRITE(5)IN,(IJ(JK),JK=1,IN)
      IS=N*2
      CALL STOR(ZR(1),IS,IERR)
      IF(IERR.NE.0)GO TO 14
   69 NI=NI+1
      CLOSE 5
      CALL CLSTOR
      CALL GETCP(JT)
      IT=(JT-IT)/100
      EPS=.2**2674156
      WRITE(1)NOD,NB,DEN,CF,TH,PHI,IT,ICK,EPS
```

```
166         GO TO 990
167      14 WRITE(6,411)
168     411 FORMAT(1X,'OVERFLOW RANGE IN SUFZ')
169         GO TO 1000
170      15 WRITE(6,412)ICK
171     412 FORMAT('MAX DIM EXCEEDED ICK=',I4)
172         GO TO 1000
173     99A CLOSE *
174     999 CALL ASSIGN(4HZOUT,6HSYSVC1,6)
175         WRITE(6,*)HAFPTS,OPNZ,ISTART
176         CALL LOG(6HUIGCO,6HSYSAC)
177    1000 CALL EXIT
178         END
```

```
C  THIS PROGRAM COMPUTES SCATTER CROSS SECTION OF N ELEMENT RANDOM
C  ARRAY USING JACOBY, SOR & SOI ITERATION.  MUTUAL IMPEDANCE
C  MATRIX MUST BE PREPROCESSED BY TESBOP, TESBO1 or TESBO5.....
      OPTIONS 32K
      INCLUDE EZFED.SYSAC(SOPUTB.SYSAC
      INCLUDE CLUB.SYSAC(ZFED.SYSAC
      INCLUDE PAPER.SYSAC(GETB.SYSAC
      DIMENSION XC(13440)
      COMPLEX CC(6670),CCI(115)
      INTEGER IJ(115)
      DIMENSION X(5),XX(5),Y(5),YY(5),Z(5),ZZ(5)
      COMPLEX CI(1200),CJ(1200),ETT(1200),ZR(1200)
      DIMENSION XR(2400)
      EQUIVALENCE (ZR(1),XR(1)),(CC(1),XC(1))
      COMMON CC
      COMPLEX CJI,ETT,ETP,EPP,EPT,ETH,EPH,ET,EP
      DATA PI,TP,ETA/3.14159,6.28318,376.727/
      DATA CJI/(0.0,-.530888E-2)/
      CALL ESC(1998)
      CALL ERRP (0)
      CALL ASSIGN(4HZOUT,6HSYSVC(,6)
      READ(6,-)HAFBIS,DPHZ,ISTART
      RISTAT=HAFBIS*2
      CLOSE 6
      CALL ASSIGN(4HPLOT,6HSYSVC(,0)
      CALL ASSIGN(4HZZAT,6HSYSVC(,3)
      CALL ASSIGN(4HZADT,6HSYSVC(,4)
      CALL ASSIGN(4HZIMP,6HSYSVC(,5)
      CALL ASSIGN(4HDATA,6HSYSVC(,1)
      READ(1)NOD,NS,DEN,CF,TH,PHI,IT,ICK,EBS
      IF(ISTART.EQ.1)CLOSE 1
      WRITE(6,5)NOD,NS,DEN
    5 FORMAT(1X,'NELE=',1I4,2X,'NSFG=',1I2,2X,'ELE DENSITY=',1E9.3)
      WRITE(6,10)
   10 FORMAT(' J=0, SOR=1, SOI=2  , ')
      READ(5,-)ITE
      IF(ITE.EQ.0)WRITE(6,2)
    2 FORMAT(1X,'JACOBI')
      IF(ITE.NE.2)GO TO 12
      WRITE(6,3)
    3 FORMAT(1X,'SOI')
   12 IF(ITE.NE.1)GO TO 15
      WRITE(6,14)
   14 FORMAT(' OMEGA= ')
      READ(5,-)OMEGA
      WRITE(6,4)OMEGA
    4 FORMAT(1X,'SOR',2X,'OMEGA=',1E9.3)
   15 NOS=NS
      REWIND 5
      WRITE(6,884)TH,PHI
  884 FORMAT('TH, PHI =',2E9.3)
      WRITE(6,8)IT
    8 FORMAT(' TIME TO COMP Z =',1I12,'SEC')
      WRITE(6,11)CF,ICK
   11 FORMAT(1X,'C=',1E9.3,2X,'MAX DIM=',1I5)
```

```
      CALL CLASSN
      THR=TH*.017453
      CALL GETCP(ITT)
      NOSP=NOS+1
      NOSM=NOS-1
      NM=NOD*NOS
      NP=NOD*NOSP
      NMD=NOSP*NOS/2
      HL=.2567
      CK=TP*TP/100
      N=NOD*NOSM
      AL=HL/100
      AK=AL*TP
      HK=HL*TP
      DK=2*HK/NOS
      COK=COS(DK)
      SDK=SIN(DK)
      ANOS=NOD/OEN
      RS=.2845*ANOS**(1/3.)*TP
C     EBS=.2926741*6
      EBS=0
C  RANDOM ARRAY #1
      IP=2165413
C  RANDOM ARRAY #2
C     IP=7599111
      NI=NOS
  102 IT2=1
      CTH=COS(THR)
      STH=SIN(THR)
      IF(ISTART.EQ.1)GO TO 121
      READ(1)ISTART,(CJ(JK),CI(JK),JK=1,N)
      ISTART=ISTART+1
      CLOSE 1
      GO TO 125
  121 DO 124 IPK=1,N
  124 CI(IPK)=(0.0,0.0)
  125 IBK=1
      DO 943 IPEP=ISTART,1000
      A=0
      PHZ=PHI
      CALL EZFFD(X,Y,Z,ET,EP,ETT,NOD,NOSM,NOSP,IP,RS,HK,CK,CTH,STH,PHZ)
      IF(ITT.NE.2)GO TO 131
      IF(IPEP.NE.1)GO TO 123
      DO 122 I=1,N
  122 CJ(I)=-ETT(I)*CJI
  123 DO 130 L=1,N
      READ(3)IN,(IJ(JK),JK=1,IN)
      DO 680 I=1,IN
      IDI=IJ(I)
      CCI(I)=CJ(IDI)
      DO 680 J=I,IN
      IDJ=IJ(J)
      IDF=IDJ+N*(IDI-1)
      KI=(I-1)*IN-(I*I-I)/2+J
      IOF=IOF+IDF-1
```

```
111         KI=K1+KJ-1
112         DO 680 IM=0,1
113         IOF=IOF+IM
114         KI=K1+1M
115   680 XC(KI)=GLT(IOF)
116         CALL SGROT(CC,CCI,0,1,IN)
117         IF(IN.EQ.N)GO TO 12A
118         DO 685 I=1,IN
119         IF(IJ(I).NE.L)AO TO 685
120         ET=CCI(I)
121         GO TO 686
122   685 CONTINUE
123   686 CI(L)=CI(L)+ET
124   130 CONTINUE
125         REWIND 4
126         GO TO 131
127   12A DO 129 I=1,N
128   129 CI(I)=CCI(I)
129   131 CONTINUE
130         DO 138 II=1,N
131         I=II
132         IF(IPK.LT.0)I=N-II+1
133         EP=(0.0,0.0)
134         DO 136 J=1,N
135         KI=J+N*(I-1)
136         JP=J+J-1
137         KI=KI+KI-1
138         DO 135 IM=0,1
139         JP=JP+1M
140         KI=KI+1M
141   135 XR(JP)=GLT(KI)
142         IF(ITE.NE.2.AND.I.EQ.J)GO TO 136
143   139 EP=EP+ZR(J)*CI(J)
144   136 CONTINUE
145         ET=ETT(I)+CJT
146         ETTM=CABS(ET)
147         IF(ETTM.GT.EBS)EBS=ETTM
148   137 CJ(I)=-EP-ET
149         IF(ITE.EQ.2)GO TO 138
150         CJ(I)=CJ(I)/ZR(I)
151         IF(ITE.NE.1)GO TO 138
152         CI(I)=CI(I)+OMEGA*(CJ(I)-CI(I))
153   138 CONTINUE
154         IF(ITE.NE.0)GO TO 141
155         DO 142 I=1,N
156   142 CI(I)=CJ(I)
157   141 WRITE(1)NOD,NS,DEN,CF,TH,PHI,IT,ICK,EBS
158         WRITE(1)IPEP,(CJ(JK),CI(JK),JK=1,N)
159         CLOSE 1
160         IBK=-IBK
161         DO 160 I=1,N
162         EP=(0.0,0.0)
163         ET=ETT(I)*CJT
164         DO 156 J=1,N
165         KI=J+N*(I-1)
```

```
      JP=J*J-1
      KI=KI*KI-1
      DO 155 IM=0,1
      JP=JP+IM
      KI=KI+IM
155   XR(JP)=GET(KI)
154   EP=EP*ZR(J)*CI(J)
      A=CABS(EP*ET)
      A=A+A
160   CONTINUE
      IF(HAFBIS)162,164,164
162   PHZ=PHI
      DPHZ=0
      INUM=1
      GO TO 166
164   INUM=HAFBIS/DPHZ
      PHZ=PHI-DPHZ*INUM
      INUM=INUM*2+1
166   SIG=0
      DO 158 ITP=1,INUM
      IF(HAFBIS)168,167,167
167   CALL EZFFD(X,Y,Z,ET,EP,ETT,NOD,NOSM,NOSP,IP,RS,HK,DK,CTH,STH,PHZ)
168   ETH=(0.0,0.0)
      DO 150 I=1,N
150   ETH=ETH+CI(I)*ETT(I)
      ETM=CABS(ETH)
      SIGG=4*PI*ETM*ETM
      SIG=SIG+SIGG
      WRITE(6,1)IPEP,PHZ,SIGG
    1 FORMAT(1X,I4,2X,1F8.2,2X,1E9.3)
158   PHZ=PHZ+DPHZ
      SIG=SIG/INUM
      A=A/M/EOS
      PHZ=PHI+180
      CALL EZFFD(X,Y,Z,ET,EP,ETT,NOD,NOSM,NOSP,IP,RS,HK,DK,CTH,STH,PHZ)
      ETH=(0.0,0.0)
      DO 170 I=1,N
170   ETH=ETH+CI(I)*ETT(I)
      TSC=-2*AIMAG(ETH)
      IF(IPEP.EQ.1)GO TO 248
      IOUT=IPEP-1
      DO 247 IOS=1,IOUT
      READ(4,300)IFC,SIGP,AA,TS,ET,EP
247   WRITE(5)IFC,SIGP,AA,TS,ET,EP
      REWIND 4
248   WRITE(6,300)IPEP,SIG,A,TSC,ETH,CI(N)
      WRITE(5)IPEP,SIG,A,TSC,ETH,CI(N)
      WRITE(6,300)IPEP,SIG,A,TSC,ETH,CI(N)
      REWIND 5
      DO 249 IOS=1,IPEP
      READ(5)IFC,SIGP,AA,TS,ET,EP
249   WRITE(4,300)IFC,SIGP,AA,TS,ET,EP
      REWIND 5
      REWIND 4
300   FORMAT(1X,3I4,1X,1E10.4,1X,1F10.4,1X,1E10.4,1X,4E10.4)
```

```
      IF(IN.EQ.N)GO TO 998
 998 CONTINUE
 99A CALL GETCLP(JT)
      JT=(JT-ITT)/100
      WRITE(6,7)JT
   7 FORMAT(/,'1FR TIME =',I12,'SEC')
      IF(HAFDIS)804,802,802
 802 WRITE(6,9)DISTAT,DPHZ
   9 FORMAT(1X,'DISTATIC AVE OVER',1F8.2,'DEG IN',1F8.2,'DEG INCR')
 804 CALL PAPER
      CALL EXIT
      END
```

```
C  THIS PROGRAM COMPUTES NEW SOI FOR A GIVEN Z MATRIX AND/OR
C  SETS UP NEW INCIDENCE ASPECT ANGLE...
C
      INCLUDE PAPER,SYSAC(GETB,SYSAC
      INCLUDE AUEXP,SYSAC
      INTEGER IJ(115)
      DIMENSION X(5),YX(5),Y(5),YY(5),Z(5),ZZ(5)
      COMPLEX ZH(1200)
      DIMENSION XH(2000)
      EQUIVALENCE (ZH(1),XH(1))
      COMPLEX COI,P11,P12,P21,P22
      COMPLEX ECC,ZDTA(5,5)
      DATA PI,TP,ETA/3.14159,6.28318,376.727/
      DATA COI/(0.0,-.5308E-2)/
      CALL ESC(1979)
      WRITE(1,7)
    7 FORMAT(1X,'HALF BISTATIC ANG, INCR & ISTART =')
      READ(8,*)HAFBTS,DPHZ,ISTART
      PAUSE
      CALL ERRH (0)
      CALL ASSIGN(4HZZAT,6HSYSVC(1,3)
      CALL ASSIGN(4HZINP,6HSYSVC(1,5)
      CALL ASSIGN(4HZOUT,6HSYSVC(1,6)
      CALL ASSIGN(4HDATA,6HSYSVC(1,1)
      IDM=115
      WRITE(2,8)
    8 FORMAT(1X,'OLD DATA')
      READ(1)NOD,NS,DEN,CF,TH,PHI,TT,ICK
      WRITE(6,2)NOD,NS,DEN
      WRITE(6,3)CF
      WRITE(6,688)TH,PHI
      WRITE(6,14)IT,ICK
   14 FORMAT(1X,'TIME=',I16,2X,'MAX DIM=',I18)
      REWIND 1
      WRITE(6,16)
   16 FORMAT(1X,'NEW ASPECT ONLY? 0=YES 1=NO')
      READ(8,*)NASP
      WRITE(2,250)
  250 FORMAT(///,'READ NOD,NOS,DEN,METH =')
      READ(8,*)NOD,NS,DEN,METH
      WRITE(6,2)NOD,NS,DEN
      WRITE(6,*)METH
      IF(METH.NE.1)GO TO 4
      WRITE(2,5)
    5 FORMAT(' READ OMEGA =')
      READ(8,*)OM
      WRITE(6,*)OM
    2 FORMAT(1X,'NELF=',I14,2X,'NSFG=',I12,2X,'ELE DENSITY=',1E9.3)
    4 WRITE(2,9)
    9 FORMAT('READ Z THRESHOLD =')
      READ(8,*)CF
      WRITE(6,3)CF
    3 FORMAT(1X,'C=',1E9.3)
   13 NOS=NS
      WRITE(2,20)
```

```
   20 FORMAT(' READ TH, PHI ')
      READ(5,*)TH,PHI
      WRITE(6,884)TH,PHI
  884 FORMAT(' TH, PHI ',2E9.3)
      CALL ESC(31H08)
      IF(NASP.EQ.0)GO TO 9B
      CALL DEASSN
      NOSM=NOS-1
      M=NOD*NDSP
      ICK=0
      DO 82 I=1,N
      IN=0
      DO 65 J=1,N
      KI=J+N*(I-1)
      JP=J+J-1
      MI=KI+M*I-1
      DO 65 IM=0,1
      JP=JP+IM
      KI=KI+IM
   65 XH(JP)=GLT(MI)
      ZOM=CABS(ZH(I))*CF
      DO 80 J=1,N
      IF(CABS(ZP(J)).LT.ZOM)GO TO 80
      IN=IN+1
      IF(IN.GT.ICK)ICK=IN
      IF(ICK.GT.IDM)GO TO 80
      IJ(IN)=J
   80 NJ=NJ+1
      IF(ICK.GT.IDM)GO TO 82
      WRITE(5)IN,(IJ(JK),JK=1,IN)
   82 NI=NI+1
      CLOSE 5
   98 WRITE(1)POD,NS,DEV,CF,TH,PHI,IT,ICK,ERS
      IF(ICK.GT.IDM.AND.*ETH.EQ.2)GO TO 15
      GO TO 998
   15 WRITE(6,912)ICK
  912 FORMAT(' AX DIM EXCEEDED ICK=',I4)
      GO TO 1000
  998 CLOSE 4
  999 CALL ASSIGN(4HZOUT,6HSYSVC1,6)
      WRITE(4,*)MAFRTS,DPHZ,ISTART
      CALL LOG(5HRIGHO,6HSYSAC)
 1000 CALL EXIT
      END
```

```
C  THIS PROGRAM COMPUTES UNIFORM DENSITY SPHERICAL CLOUD DISTRIBUTION...
C
      SUBROUTINE UNICLD(IX,RS,HK,DK,NOSP,X,Y,Z)
      DIMENSION X(1),Y(1),Z(1)
      TP=2*3.14159
      CALL RANDU(IX,IY,A1)
      IX=IY
      PPH=A1*TP
      CALL RANDU(IX,IY,A1)
      IX=IY
      COST=2*A1-1
      SINT=SQRT(1-COST*COST)
      CA=SINT*COS(PPH)
      CB=SINT*SIN(PPH)
      CC=COST
      CALL RANDU(IX,IY,A1)
      IX=IY
      RC=A1**(RS*RS*RS)
      RC=RC**(1/3.)
      XX=RC*CA
      YY=RC*CB
      ZZ=RC*CC
      CALL RANDU(IX,IY,A1)
      IX=IY
      PPH=A1*TP
      CALL RANDU(IX,IY,A1)
      IX=IY
      COST=2*A1-1
      SINT=SQRT(1-COST*COST)
      CA=SINT*COS(PPH)
      CB=SINT*SIN(PPH)
      CC=COST
      S=HK
      DO 45 I=1,NOSP
      X(I)=XX+S*CA
      Y(I)=YY+S*CB
      Z(I)=ZZ+S*CC
   45 S=S+DK
      RETURN
      END
C
C
      SUBROUTINE RANDU(IX,IY,A1)
      IY=IX*16645
      IF(IY)5,6,6
    5 IY=IY+2308607+1
    6 A1=IY
      A1=A1*.1192093E-6
      RETURN
      END
```

```
      SUBROUTINE ZSKFWF(XA,YA,ZA,XB,YB,ZB,XC,YC,ZC,X1,Y1,Z1,X2,Y2,
     1Z2,X3,Y3,Z3,INT,CDK,SDK,D,R,Z12)
      COMPLEX EJKR,Z12,P11,P12,P21,P22
      IF(INT.GE.0)GO TO 50
      CAT=(X2-X1)/D
      CBT=(Y2-Y1)/D
      CGT=(Z2-Z1)/D
      CAS=(XB-XA)/D
      CBS=(YB-YA)/D
      CGS=(ZB-ZA)/D
      CAR=(X3-XB)/R
      CBR=(Y3-YB)/R
      CGR=(Z3-ZB)/R
      SDT=CAS*CAT+CBS*CBT+CGS*CGT
      CTH1=CAS*CAR+CBS*CBR+CGS*CGR
      CTH2=-(CAT*CAR+CBT*CBR+CGT*CGR)
      SS1=1.-CTH1*CTH1
      SS2=1.-CTH2*CTH2
      SDKS=SDK*SDK
      F1=COS(D*CTH1)
      F1=CDK-C1
      F2=CDK-COS(D*CTH2)
      R2=R*R
      EJKR=CMPLX(COS(P),-SIN(R))
      DR2=D*D/R2
      FR=D*(1.+DR2*SS1)*SIN(D*CTH1)+2.*DR2*C1*CTH1
      AR=ABS(SS1*SS2)
      IF(AR.GT..001)GO TO 30
      FR=60.*CTH2*FR*D/(R2*SDK)
      Z12=FR*EJKR
      GO TO 60
   30 FR=-CTH2*FR/R
      FT=F1*(SDT*CTH1*CTH2)/SS1
      CST=120.*F2/(R*SDKS*SS2)
      Z12=FR*F1*CMPLX(1./R,1.-1./R2)
      Z12=CST*EJKR*Z12
      GO TO 60
   50 AK=D/50
      CALL ZGS(XA,YA,ZA,XB,YB,ZB,X1,Y1,Z1,X2,Y2,Z2
     2,AK,D,CDK,SDK,D,SDK,INT,P11,P12,P21,P22)
      Z12=P22
      CALL ZGS(XA,YA,ZA,XB,YB,ZB,X2,Y2,Z2,X3,Y3,Z3
     2,AK,D,CDK,SDK,D,SDK,INT,P11,P12,P21,P22)
      Z12=Z12+P21
      CALL ZGS(XB,YB,ZB,XC,YC,ZC,X1,Y1,Z1,X2,Y2,Z2
     2,AK,D,CDK,SDK,D,SDK,INT,P11,P12,P21,P22)
      Z12=Z12+P12
      CALL ZGS(XB,YB,ZB,XC,YC,ZC,X2,Y2,Z2,X3,Y3,Z3
     2,AK,D,CDK,SDK,D,SDK,INT,P11,P12,P21,P22)
      Z12=Z12+P11
   60 RETURN
      END
```

```
      SUBROUTINE EZFFD(X,Y,Z,ET,EP,ETT,NOD,NOSM,NOSP,IP,RS,HK,DK,
     XCTH,STH,PHI)
      DIMENSION X(1),Y(1),Z(1)
      COMPLEX ET,EP,ETT(1)
      PHR=PHI*.017453
      CPH=COS(PHR)
      SPH=SIN(PHR)
      IX=IP
      L=1
      INET=0
      DO 125 JFL=1,NOD
      CALL OBJCLO(IY,RS,HK,DK,NOSP,X,Y,Z)
      ISO=1
      DO 120 I=1,NOSM
      J=ISO
      JFF=J+1
      CALL ZFFD(X(J),Y(J),Z(J),X(JFF),Y(JFF),Z(JFF),DK,CTH,STH,
     XCPH,SPH,ET,EP)
      ETT(L)=ET
      L=L+1
  120 ISO=ISO+1
  125 CONTINUE
      RETURN
      END
```

```
 1 * CALL STOR(Y,NX,NOVRFL)
 2 * CALL CLSTOR
 3 * Y=GET(N)
 4          XDEF   STOR,STOR
 5          XDEF   CLSTOR,CLS
 6          XDEF   GET,GET
 7 STOR     TEM    OPN
 8          BNZ    ST1
 9          TLO    =BUF1
10          TKM    PNTR
11          AOK    1024
12          TKM    PMAX
13          TMA    ES
14          TAM    FILE+1
15          TMA    VC
16          TAM    FILE+3
17          TZM    NBLKW1
18          TOA    1
19          TAM    NBLKW2
20 ST1      GAP    1
21          TIM    SRCEAD
22          GAP    1
23          TMK*   DACU/I
24          GAP    1
25          TJM    SAVJ
26          TIM    OVRAD
27          TZM*   OVRAD
28          NKK
29          BNN    RETN
30 STLOOP   TZM    ROUND
31          TMD*   SRCEAD
32          RAD    5
33          REE
34          LAD    5
35          BNO    SQ1
36          AUM    ROUND
37          AME    =#2000#000
38 SQ1      AUM    SRCEAD
39          TMA*   SRCEAD
40          AMB    ROUND
41          BOD    OVERFL
42          COB    #17
43          BOP    OVERFL
44          COB    #A60
45          BON    UNDRFL
46          RRD    5
47 SQSTR    TEM*   PNTR
48          AUM    PNTR
49          AUM    SRCEAD
50          TMA    PMAX
51          CMA    PNTR
52          BOP    ST2
53          BSL    BLKOUT
54 ST2      BWK    STLOOP
55 RETN     BUC*   SAVJ
```

```
 56 OVERFL   AUM*   OVRAD
 57          CZE
 58          BON    *+3
 59          TME    =-36777777
 60          BUC    SUSTR
 61          TME    =-37000000
 62          BUC    SUSTR
 63 UNDRFL   TZE
 64          BUC    SUSTR
 65 **********************
 66 CLS      TJ4    SAVJ
 67          TLO    WBUF1
 68          CMK    PNTR              BUF EMPTY?
 69          BOZ    *+2               YES
 70          BSL    BLKOUT            OUTPUT LAST BLOCK
 71          TZM    OPN
 72          BUC*   SAVJ
 73 **********************
 74 GET      GAP    1
 75          TJ4    SAVJ
 76          TMA*   UACU/I
 77          SOA    1
 78          BON    ERR
 79          ESA
 80          OVM    =512
 81          CMA    NBLKW1
 82          BNZ    GET2
 83          TLO    WBUF1
 84          AEK
 85          BUC    GET1
 86 GET2     CMA    NBLKW2
 87          BNZ    GET3
 88          TLO    WBUF2
 89          AEK
 90          BUC    GET1
 91 GET3     TLO    BUF
 92          AEK
 93          CMA    NBLK
 94          BOZ    GET1
 95          TAM    NBLK
 96          OMA    =BG
 97          TAM    BLOCK
 98          TMA    NBLK
 99          RLA    1
100          ESA
101          OVM    =79
102          AME    ES
103          TEM    FILEGT+1
104          AMA    VC
105          COB    ",*
106          BON    *+5
107          AOA    1
108          COB    "1*
109          BON    *+2
110          AOA    1
```

```
111             TAM   FILFGT+3
112             BLU   $OPEN
113             DAC   ID
114             DAC   FILFGT
115             BNZ   ERR
116             BLU   $SETCRM
117             DAC   ID
118             DAC   BLOCK
119             BNZ   ERR
120             BLU   $IN
121             DAC   ID
122             DAC   BUF
123             BNZ   ERR
124             BLU   $WAIT
125             DAC   ID
126             BLU   $CLOSE
127             DAC   ID
128 RET1        TMI*  UAC0/K
129             BNZ   *+3
130             TOA   *201
131             BUC   UNS01
132             TZA
133             LRD   5
134             QBB   B4
135             BOZ   UNS01
136             OMA   =B7P6R*
137 UNS01       TDX
138             BUC*  SAVJ
139 ********.*****************
140 BLKOUT      ***                   OUTPUT B-BLOCK FILE
141             TKM   SAVK
142             BLU   $OPEN
143             DAC   ID
144             DAC   FILE
145             BNZ   ERR
146             BLU   $DELETE
147             DAC   ID
148             BNZ   NFU
149             BLU   $OPEN
150             DAC   ID
151             DAC   FILE
152             BNZ   ERR
153             BLU   $OUT
154             DAC   ID
155             DAC   WBUF1
156             BNZ   ERR
157             BLU   $WAIT
158             DAC   ID
159             BLU   $OUT
160             DAC   ID
161             DAC   WBUF2
162             BNZ   ERR
163             BLU   $WAIT
164             DAC   ID
165             BLU   $CLOSE
```

```
166           DAC   IO
167           AUM   FILE+1
168           TMA   ES
169           AOA   79
170           CMA   FILE+1
171           ROP   BLKOUT1
172           TMA   ES
173           TAM   FILE+1
174           AUM   FILE+3
175           TOB   ",*
176           CMB   FILE+3
177           BOZ   SKIP
178           TOR   "1*
179           CMB   FILE+3
180           BNZ   *+2
181 SKIP      AUM   FILE+3
182 BLKOUT1   AUM   2
183           DAC   NBLKW1
184           AOM   2
185           DAC   NBLKW2
186           TLO   BUF1
187           TKM   PNTR
188           TMK   SAVK
189           BUC*  BLKOUT
190 **********************
191 NMU       BLU   $CRLF
192           BLU   $WPK
193           DAC   =18
194           DAC   ="NO MORE USER NAMES"
195           BUC   ERRCL
196 **********************
197 ERR       BLU   $CRLF
198           BLU   $WPK
199           DAC   =17
200           DAC   ="STOR OR GET ERROR"
201 ERRCL     BLU   $CLOSE
202           DAC   IO
203           TZM   OPN
204           RLL   $EXIT
205 **********************
206 OVRAD     ***
207 ROUND     ***
208 RAVJ      ***
209 OPN       ***
210 PNTR      ***
211 SAVK      ***
212 PMAX      ***
213 BUF       EQIV  '177000
214 IO        DATA  7
215 FILE      DATA  "FILES SYS    " (CHANGE SYS TO SYS1)
216 FILE1     DATA  "FILES SYS    " (CHANGE SYS TO SYS1)
217 SKCEAD    ***
218 DAC0/I    DAC   0,I
219 DAC0/K    DAC   0,K
220 NBLK      DATA  -1
```

```
221 BLOCK   ***
222 VC      DATA  *25441442       =VC= (CHANGE TO *2544:442 =VC=)
223 FS      DATA  *21251442       =FS=
224 RBLKN1  ***
225 NBLKN2  ***
226 RBUF1   BOAT  512(0)
227 WBUF2   BOAT  512(0)
228         END
```

```
 1 * Y=GET(N)
 2         XDEF  GET,GET
 3 GET     GAP   1
 4         TJM   SAVJ
 5         TMA=  DACO/I
 6         SOA   1
 7         BON   ERR
 8         ESA
 9         OVM   =512
10         TEK
11         CMA   NBLK
12         BOZ   GET1
13         TAM   NBLK
14         OMA   =80
15         TAM   BLOCK
16         TMA   NBLK
17         RLA   1
18         ESA
19         OVM   =79
20         AME   ES
21         TEM   FILE+1
22         AMA   VC
23         COM   ".*"
24         BON   *+5
25         AOA   1
26         COB   "!*"
27         BON   *+2
28         AOA   1
29         TAM   FILE+3
30         BLU   $OPEN
31         DAC   ID
32         DAC   FILE
33         BNZ   ERR
34         BLU   $SETCOM
35         DAC   ID
36         DAC   BLOCK
37         BNZ   ERR
38         BLU   $IN
39         DAC   ID
40         DAC   BUF
41         BNZ   ERR
42         BLU   $WAIT
43         DAC   ID
44         BLU   $CLOSE
45         DAC   ID
46 GET1    TME*  BUF/K
47         BNZ   *+3
48         TOA   '201
49         BUC   UNSG1
50         TZA
51         LRD   5
52         QBB   B4
53         BOZ   UNSG1
54         OMA   =B7B6B5
55 UNSG1   TOX
```

```
56          BUC*  SAVJ
57 ********************
58 ERR      BLU   $CRLF
59          BLU   SWPR
60          DAC   =9
61          DAC   ="GET ERROR"
62          BLU   $CLOSE
63          DAC   ID
64          BLL   $EXIT
65 ********************
66 SAVJ     ***
67 BUF      EQIV  '177000
68 ID       DATA  7
69 FILE     DATA  "FILE  SYS    " (CHANGE SYS TO SYS)
70 DAC0/1   DAC   0,I
71 BUF/K    DAC   BUF,K
72 APLK     DATA  -1
73 BLOCK    ***
74 VC       DATA  '25441442       =VC" (CHANGE TO '25441442 =VC")
75 ES       DATA  '21251442       =ES"
76          END
```

APPENDIX G
SPLIT-GATE AND LEADING EDGE TRACKER PERFORMANCE ON SHORT PULSE ECHOES

A. INTRODUCTION

This report considers the tracking behavior of split-gate and leading edge range trackers against short pulse target returns with several peaks. The objective is to learn where these trackers tend to track on such waveforms.

Section II of this report discusses the characteristics of split-gate and leading edge trackers, and explains the effects of lockup. Section III of the report describes a computer simulation used to observe tracker behavior in typical missile attack situations. Section IV shows some typical curves of tracking performance.

B. DISCUSSION OF TRACKER PERFORMANCE

The performance of a range tracker with an extended target echo depends on several factors, the two most important being the shape of the waveform and the design of the tracker. In this report, we consider two types of range trackers, a split-gate tracker and a leading edge tracker.

A split-gate tracker multiplies the received echo pulse by a pair of gates. The video waveform during each gate is integrated and the two integrator outputs are subtracted. The resulting difference voltage is proportional to the gate offset from the center of the pulse. This difference voltage is used to correct the gate position for the next pulse.

A leading edge tracker first differentiates the received waveform and then tracks the result with a split gate tracker. Only the positive side of the differentiated waveform is used by the split gate tracker. Negative values of the derivative are excluded.

In the absence of thermal noise, a split gate tracker may track at any point in the echo waveform where there is equal area under each gate. In a waveform with multiple peaks, there may be several such points, depending on the gate width and the specific waveform. For example, if the echo waveform is as shown in Figure 1, and the gate width is as indicated,

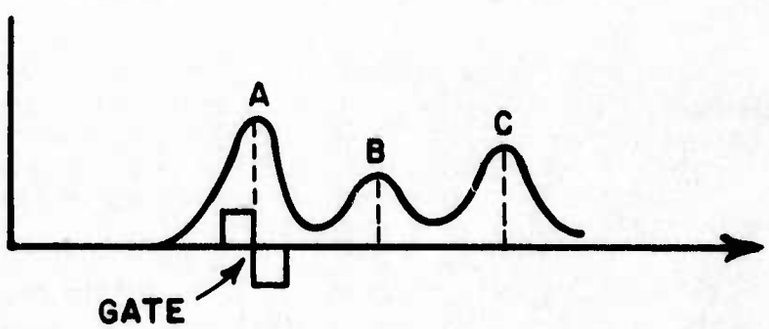


Figure G-1. An extended target echo.

there will be at least three points where the gate can track, points A,B and C. Which point is actually tracked in a given situation depends on the method used to lock up the tracker, as discussed below.

Similarly, a leading edge tracker will track at any point where the differentiated video waveform has equal area under each gate. Figure G-2 shows a possible video waveform v(t) and its derivative v'(t). (The regions where v'(t) is negative are not included in the differentiated version).

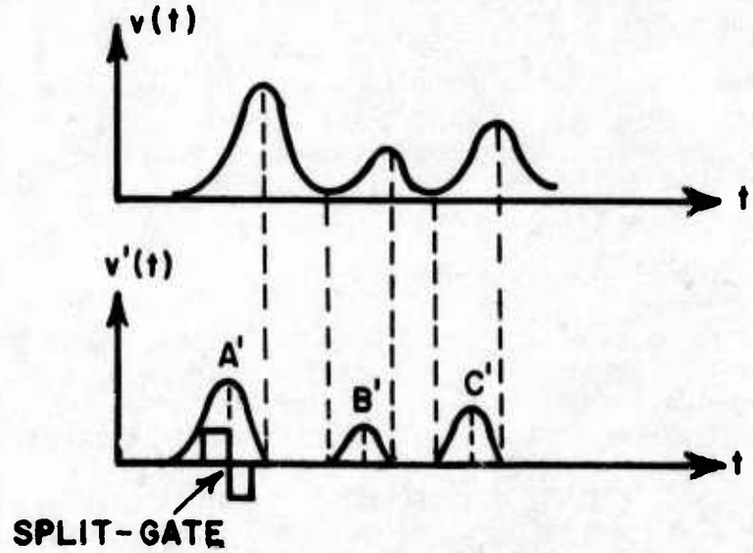


Figure G-2. A differential target echo.

When v'(t) is tracked by a split gate tracker, the waveform in Figure G-2 will have three points where the tracker can track, A', B' and C'. Again, the actual point chosen by the tracker depends on how the tracker locks up.

The most common method for locking up a split gate tracker uses two different gates, a wide gate and a narrow gate. The wide gate is much wider than the target echo width and is used for locking up the tracker. The narrow gate is matched to the target echo width and is used for tracking. In the search mode, the range is tracked with the wide gate until successive samples of the error voltage indicate that the wide gate is centered on the echo pulse. When this happens, the tracker switches to the narrow gate, which tracks with greater accuracy because it admits less noise than the wide gate.

With this method of lockup, the wide gate tends to align itself with the point which divides the area under the entire echo pulse in half. When the tracker switches to the narrow gate, the tracker then moves to the local peak in the waveform nearest this point. In an echo waveform with several peaks, the particular peak chosen by the tracker depends on the relative strengths and locations of the peaks.

A second method of locking up a split gate tracker is to slew the gate position across the range interval where the target is located. The two gate outputs are summed and compared with a threshold. When the gate enters the echo pulse, the sum output will rise and cross the threshold.

At this point, the loop is changed to a tracking mode, and the difference output is used to time the gate.

With this method of lockup, the tracker locks on the first part of the waveform it encounters where the early and late gates have equal output. Normally, one slews from short range out towards longer range. In this case, the tracker locks on the earliest part of the waveform where the gate outputs are equal, i.e., on the first pulse of the extended echo.

Thermal noise in the tracking loop complicates the situation because it produces tracking jitter. If the tracker is tracking a low amplitude peak, the tracking jitter may be large enough to make the loop drop out of lock. If this happens, the loop will then try to relock. Where it winds up depends on the lockup procedure used. A wide gate technique will return the tracker to the point which divides the area of the total echo waveform in half. A slewing technique moves the gate to the next peak of the waveform in the direction slewed.

Typically, the short pulse return from an aircraft contains several peaks with different amplitudes. The shape of the waveform is highly dependent on viewing angle. The number of peaks, their locations, widths and amplitudes all change with viewing angle. During a missile attack against an aircraft, the echo waveform seen by the missile radar changes continuously.

It is obvious that the part of the echo tracked by the split gate or leading edge tracker depends on the shape of the waveform. With wide gate-narrow gate lockup technique, some waveforms will cause the tracker to track the earliest peak and other waveforms will cause it to track a later peak. Since the shape of the echo from an aircraft target changes rapidly with aspect angle, and since thermal noise also causes the loop to unlock at random times and then relock, it is impossible to generalize about where a tracker will track.

For a missile attacking an aircraft, the evolution of the echo waveform with time during the track is difficult to predict unless all parameters of the attack situation are taken into account. For example, the performance of the range tracker depends on the angle tracker, because the angle tracker affects the trajectory of the missile and hence the target look angle. Range tracker performance is also dependent on aircraft and missile dynamics (acceleration rates, turning rates, etc.), since these also affect the trajectories and hence the look angle. Of course, the signal-to-noise ratio is dependent on target range, so tracking loop jitter changes as the missile closes on the aircraft.

For these reasons, the most realistic way to determine how a tracker behaves is to simulate a missile attack with the important dynamic variables included and observe the tracker performance. Such a simulation was developed and is described in the next section.

C. THE SIMULATION

A Fortran program has been written to simulate the important aspects of the missile attack situation. The missile is fired from the ground at the

aircraft; the aircraft trajectory is controlled by the computer operator. Realistic velocities, turning rates, etc., are programmed for both the aircraft and missile (see details below). The missile has a monopulse angle tracker, and homes on a predicted intercept point continuously updated during the attack. The echo waveform of the aircraft is simulated as the sum of three gaussian pulses whose relative strengths and time delays depend on the aircraft viewing angle. The aircraft can drop a chaff scatterer on command, modeled as a single scatterer contributing an additional gaussian pulse to the target echo. The chaff scatterer decelerates instantly to zero velocity when dropped from the aircraft. The Fortran program used for this simulation is given below. Some of the detailed characteristics are described below.

The simulation details can be broken down into six areas -- airplane maneuvering, missile maneuvering, range tracker characteristics, angle tracker characteristics, radar properties of the aircraft, and intercept point prediction. We briefly describe the assumptions below.

Airplane Maneuvering

The airplane is maneuvered by the computer operator. The program allows a maximum turning rate of 4.5°/second in steps of 0.6° per second. There is no interrelation in θ and ϕ maneuvering. The velocity of the plane can be set from 100 meters/second to 410 meters/second in steps of 10 meters/second with no acceleration restrictions. These figures give a turning radius of about 0.5 km at full speed.

Missile Maneuvering

The missile speed is controlled automatically. The velocity is 50 meters/second at launch and the missile accelerates at 0.1 meter/second/second. These numbers result in a typical impact velocity of about 60 meters/second, since most encounters take approximately 100 seconds. The missile has a maximum turning rate of 11.5°/second, controlled by the angle tracker described below.

Range Tracker Characteristics

Two types of range trackers are used, a split-gate and a leading edge tracker. The leading edge tracker operates by differentiating the received video waveform, excluding negative values of the derivative, and tracking the resultant waveform with a split-gate tracker, as discussed earlier.

The range tracker uses two gate widths to lock up. A wide gate is used for initial acquisition and a narrow gate for tracking. The narrow gate width is 10 meters and the wide gate width is 100 meters.

Angle Tracker Characteristics

The angle tracker is a conventional 4-channel monopulse tracker that derives pointing error information in both angular coordinates. The sum beamwidth is 20°.

Radar Properties of the Target

The target echo waveform is modelled as a sum of three gaussian shaped pulses along the aircraft separated 10 meters apart. Short pulse radar returns from scale models under controlled conditions appear to have such a structure. From front to back, the three pulses have relative strengths of 5, 3 and 4 units of voltage.

Intercept Point Prediction

A predicted path is computed for the target. During the track, if the target is found to deviate from the predicted path by more than 5 meters in range and 1° in angle, the predictor readjusts the path calculation. This window area is to allow for noise and overshoot in the gates. The prediction is based on the assumption of a straight line path.

D. TYPICAL RESULTS

Figures G-3 through G-16 show typical plots of range tracking error for a missile attacking an aircraft in straight level flight. The curves show the range tracking error as a function of the target range, with the smallest range at the left side of the curve. Thus, time runs from right to left on the curve, since the missile starts at large range and then closes to zero range. Ten second increments are marked with x's on the graphs.

Figure G-3 shows a typical curve of tracking error for a leading edge tracker. At large range, the range tracker is using the wide gate. At the point marked "0" on the curve, the tracker converts to the narrow gate. It is seen that the sinusoidal wandering stops at this point, and the tracker performance improves.

It is seen that after the tracker switches to the narrow gate, the tracking error persists at about +12 meters. On these curves, range is measured from the center pulse in the (three pulse) echo waveform, so this residual range error means that the tracker is tracking the leading pulse in the echo waveform.

Figures G4-7 show several additional runs, all with the same leading edge tracker. Figures G-3 through G-7 are all run under the same conditions, except for aircraft velocity. In Figure G-3, the aircraft velocity is 140 meters/second, in Figure G-4 it is 180 m/s, in Figure G-5 it is 220 m/s, in Figure G-6 it is 260 m/s and in Figure G-7 it is 300 m/s. We note that at 180 m/s (Figure F-4), the steady-state error is approximately -7 meters; in this case,, the tracker is tracking the rear pulse. At 220 m/s (Figure G-5), the tracker tracks the center pulse. (The residual error is approximately +2 meters -- it is not zero because the leading edge tracker tracks the front edge of the center pulse). In Figure G-5, the tracker switches to the narrow tracking gate at a range of 4.85 km, but then goes back to the wide gate at 4.5 km, and finally returns to the narrow gate at 3.2 km. This case illustrates how the tracker returns to the wide gate if the narrow gate drops out of lock due to range jitter. Figures G-6 and G-7 (aircraft velocities of 260 m/s and 300 m/s), the range tracker again tracks the leading pulse.

Figures G-8 through G-15 show a similar set of runs using a split gate tracker instead of a leading edge tracker. All curves are run under the same conditions, except for different aircraft velocities. The velocity spans the range 100 meters/second to 410 meters/second, as labelled on the individual curves. In this set of runs, it is seen that the split-gate tracker tracked any of the three pulses, depending on the aircraft velocity.

Observe the expanded scale used in the split gate tracker results. The reason is that the split gate tracker tended to lock on to the first pulse it observed with little overshoot. It shifted to the narrower gate almost immediately. Tracking jitter is also higher for the split gate tracker. This is most likely caused by the larger width of the pulse being tracked. Compare pulses A and A' Figures G-1 and G-2. These pulse widths tend to correlate with the jitter magnitudes observed, approximately 1-1/2 meters peak to peak for the leading edge tracker and four meters peak to peak for the split gate tracker. Another factor may be the additional separation or isolation of the pulses being tracked for the leading edge tracker. This would also tend to reduce the jitter for the leading edge tracker.

E. CONCLUSIONS

In general, a split-gate tracker may track on any peak in a waveform where there is equal area under each gate. In a waveform with several peaks, the particular peak chosen depends on the lockup method used and the evolution of the waveform shape during the lockup procedure. Moreover, the lockup has a statistical behavior because of the thermal noise in the loops. Thus, with a given echo waveform, the tracker may sometimes lock on one peak and sometimes on another peak.

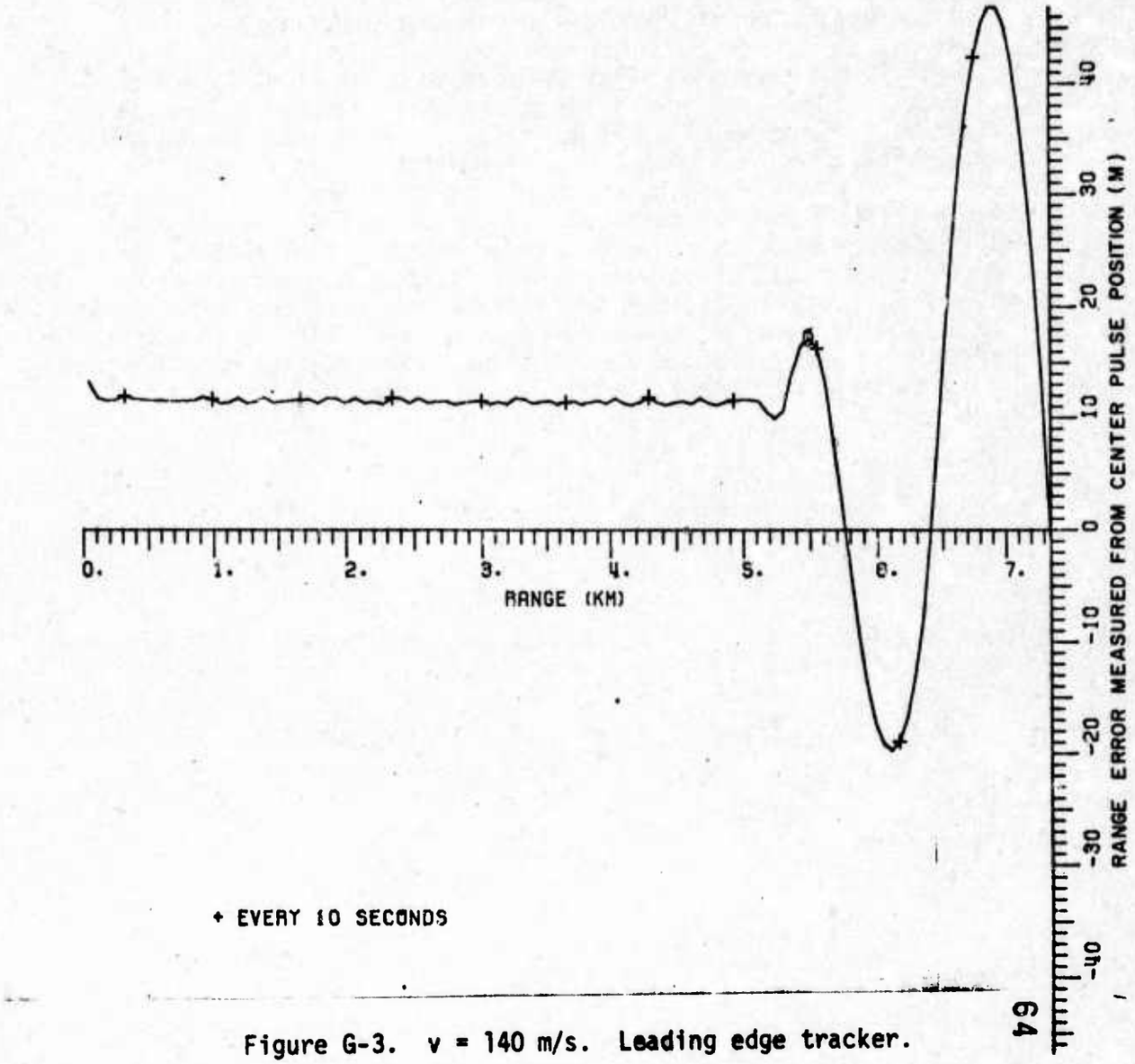
A leading edge tracker operates by first differentiating the received video waveform and then tracking that waveform with a split-gate tracker. Hence the same comments apply to a leading edge tracker.

Usually, if a particular peak in an echo waveform (or its derivative) is predominant, the split-gate tracker will settle on that peak. However, during a typical missle attack trajectory, the strongest peak in an aircraft echo will sometimes be from the front of the aircraft and sometimes from the rear. Hence it is not possible to draw any general conclusions about which part of an aircraft is tracked. Moreover, the tracker does not always track the strongest peak. The examples shown in Figures G-4 and G-5 are cases where the tracker tracks the intermediate peak and the weakest peak.

In this study, we assumed the short pulse response for an aircraft in the form of three Gaussian pulses of different magnitudes. The simulation of the reflections from the scattering centers as a Gaussian pulse is probably reasonable in that the precise shape of

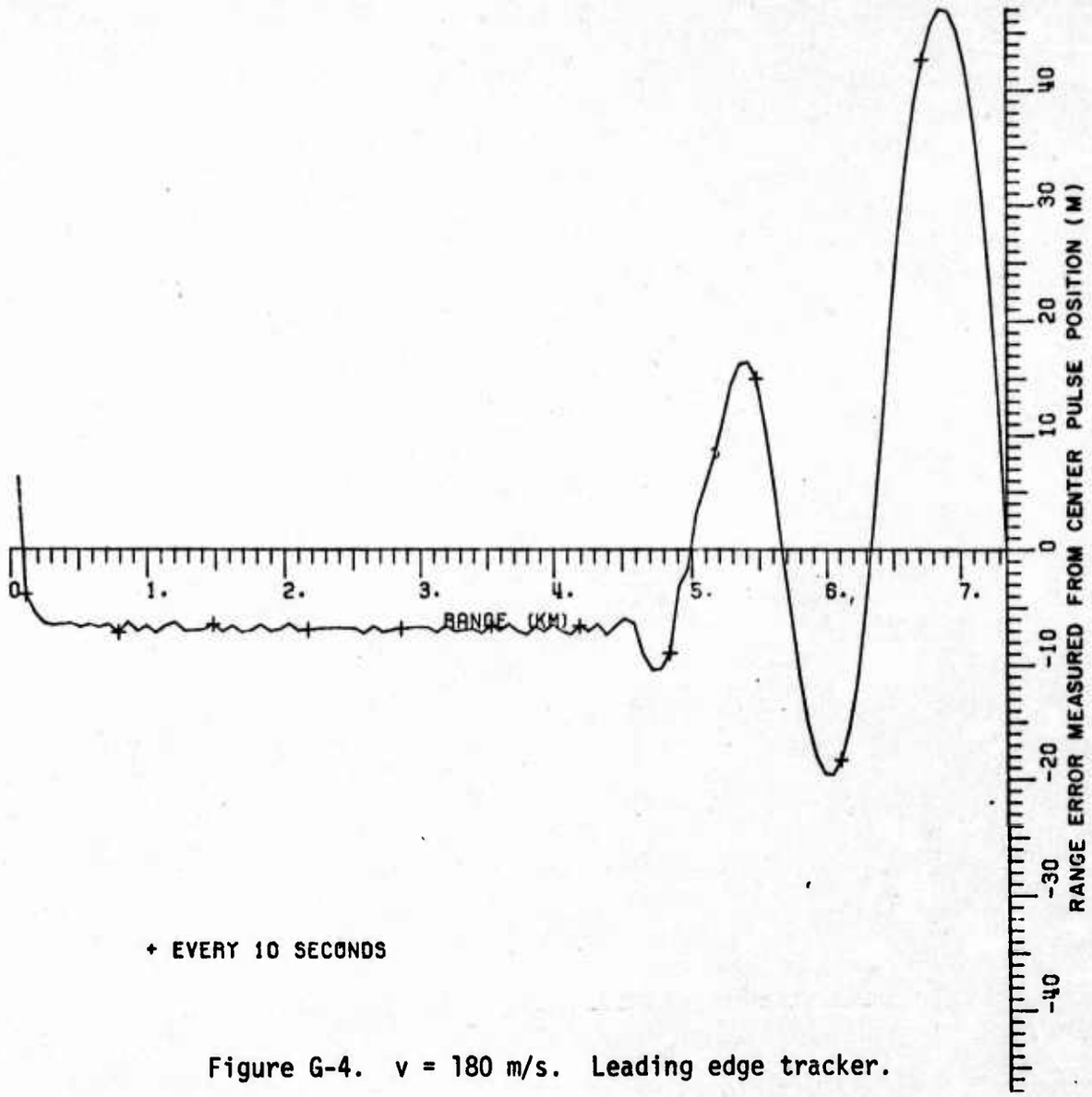
the individual pulses would not materially alter the results. The important feature is the pulse position and duration, their relative maximum values, the number and spacing of the received pulses. Of particular importance is the manner in which these pulse properties change as the relative orientation of the radar and the aircraft changes in flight. These parameters were not available to us at the time of this study. Any future effort of this type should be prefaced by a study that would generate this specific data.

It is observed that since the precise pulse shape is not critical, it may be practical to generate these pulse returns in a relatively simple way. It is generally conceded that present GTD capabilities are such that the scattered fields can be predicted with reasonable accuracy and costs for all parts of the aircraft except the jet intake and jet exhausts. It should be practical to obtain the appropriate pulse properties experimentally from these critical scatterers using a short pulse radar. Used in conjunction with a directional antenna to isolate the intake response from other scatterers as the radar is moved around an actual aircraft, this type of measurement would provide the additional data required to evaluate the performance of the range trackers.



64

Figure G-3. v = 140 m/s. Leading edge tracker.



+ EVERY 10 SECONDS

Figure G-4. v = 180 m/s. Leading edge tracker.

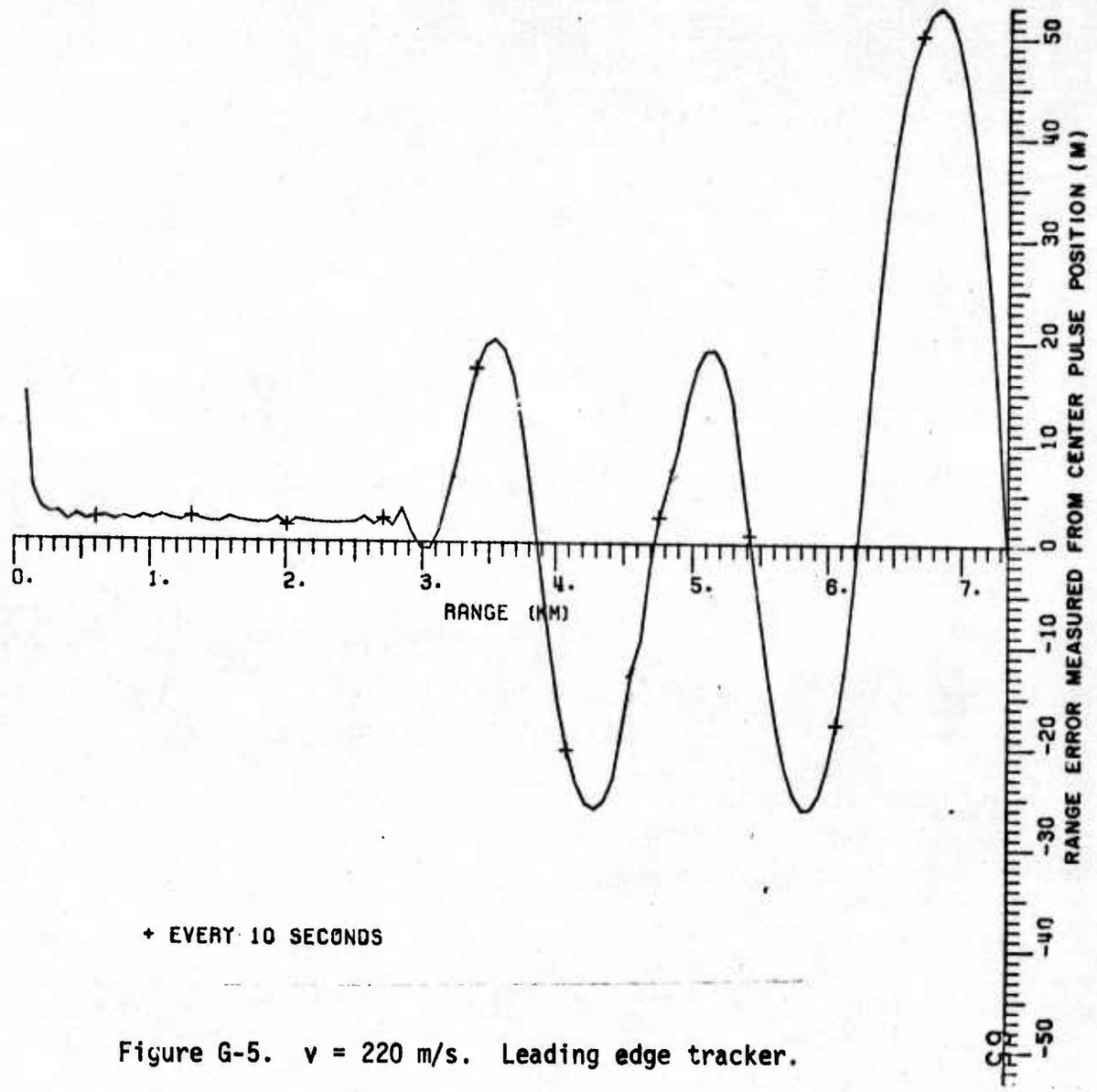


Figure G-5. v = 220 m/s. Leading edge tracker.

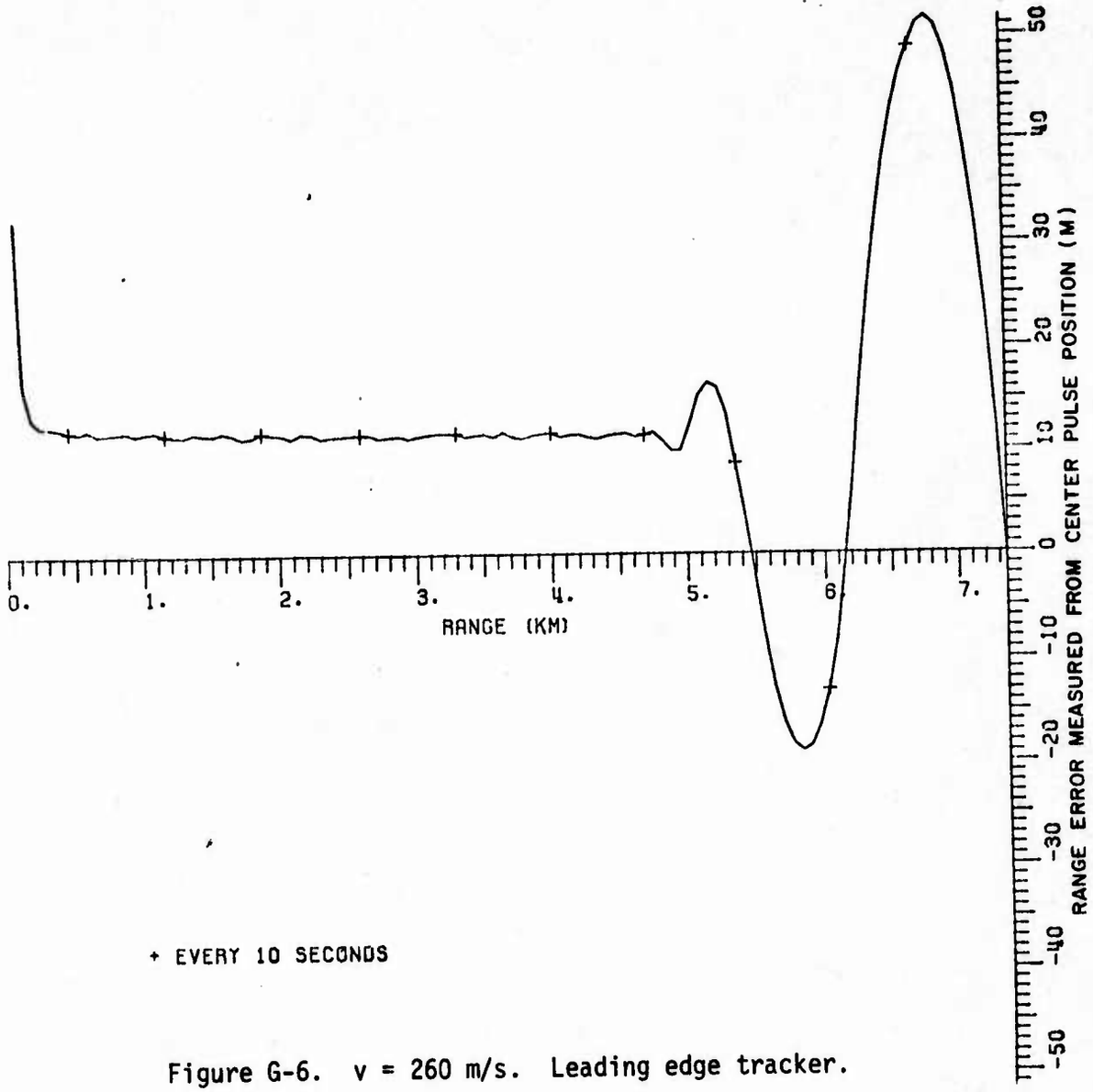


Figure G-6. v = 260 m/s. Leading edge tracker.

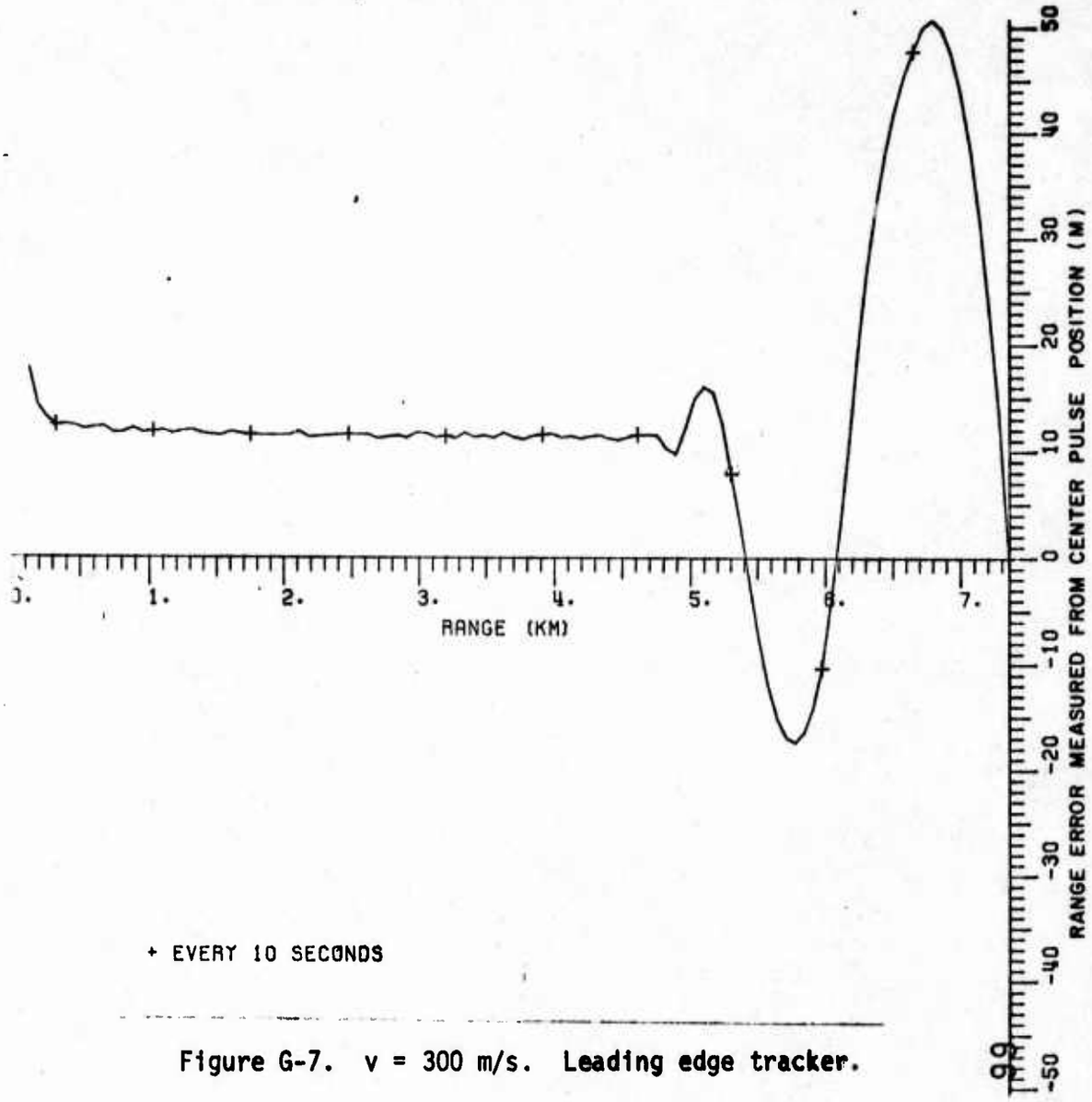


Figure G-7. v = 300 m/s. Leading edge tracker.

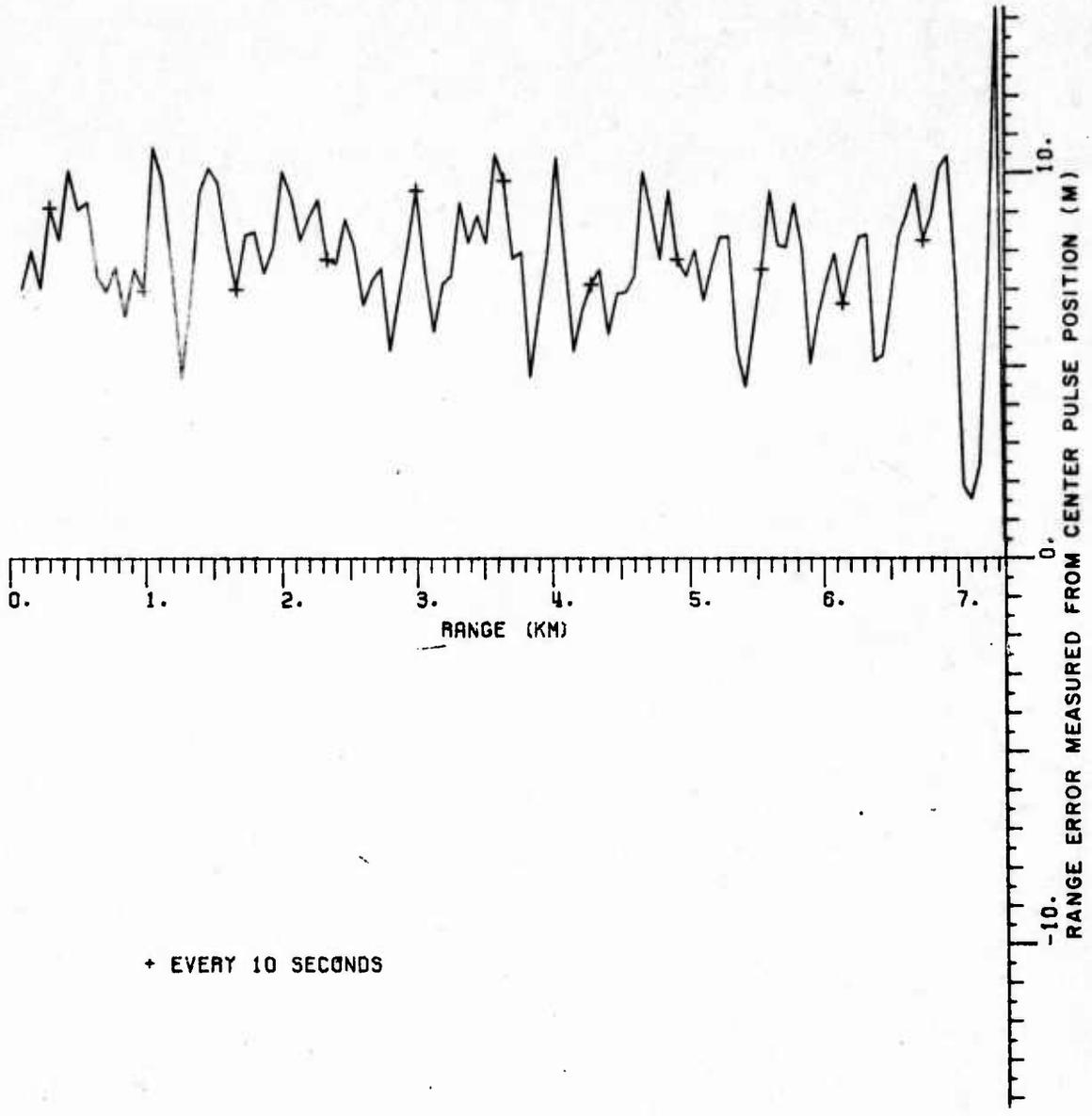


Figure G-8. v = 100 m/s. Split-gate tracker.

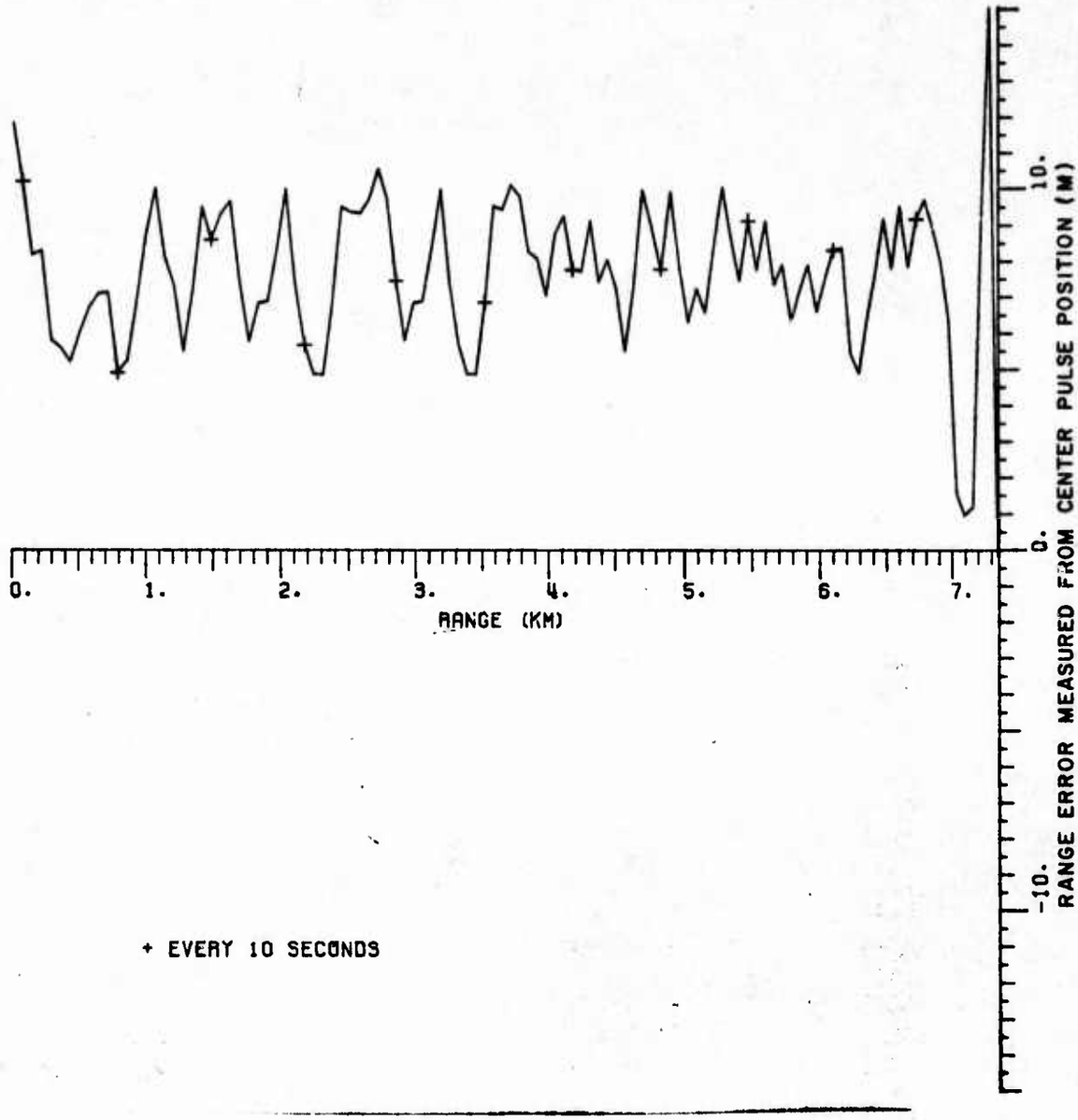


Figure G-9. v = 180 m/s. Split-gate tracker.

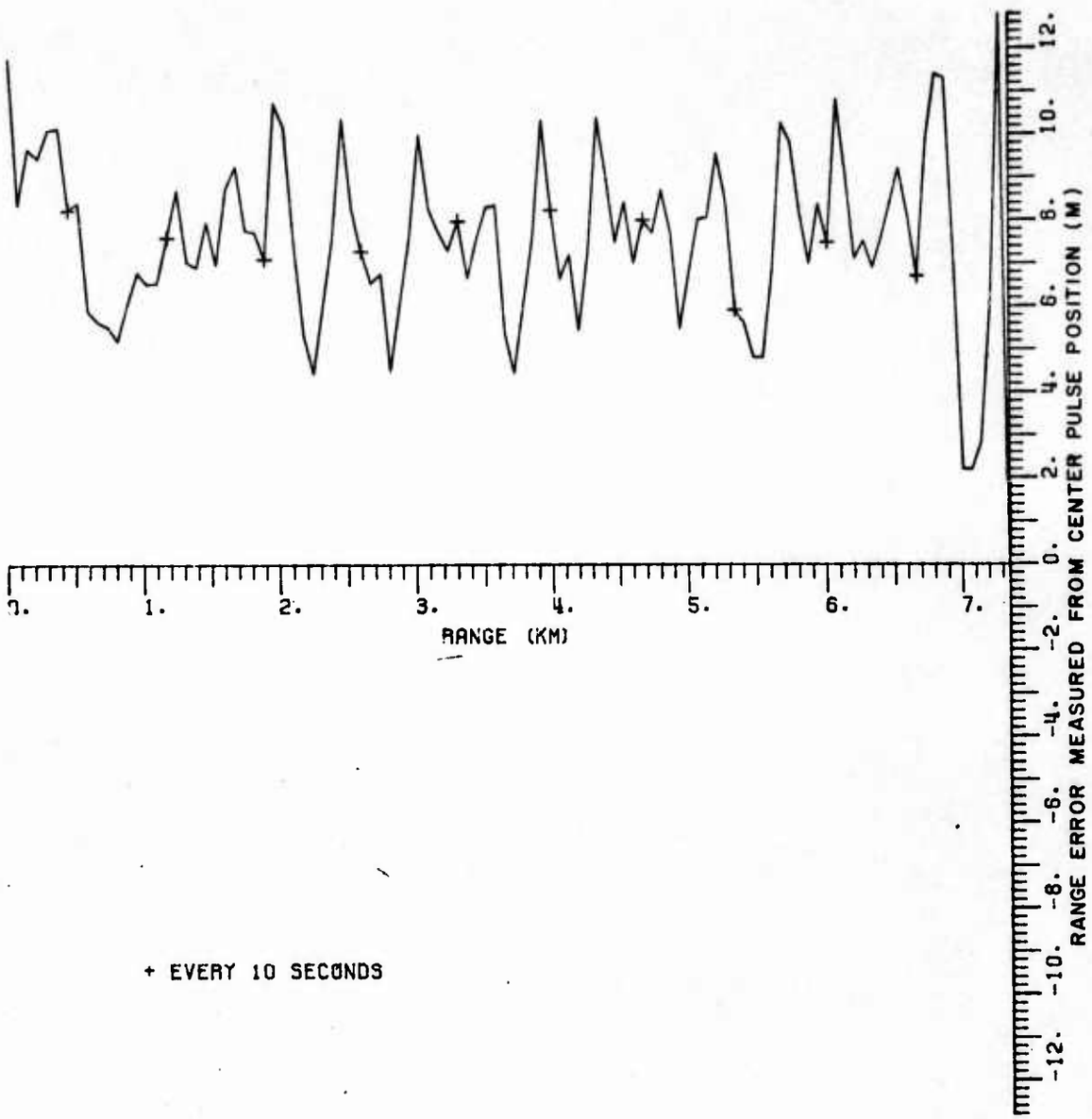


Figure G-10. v = 200 m/s. Split-gate tracker.

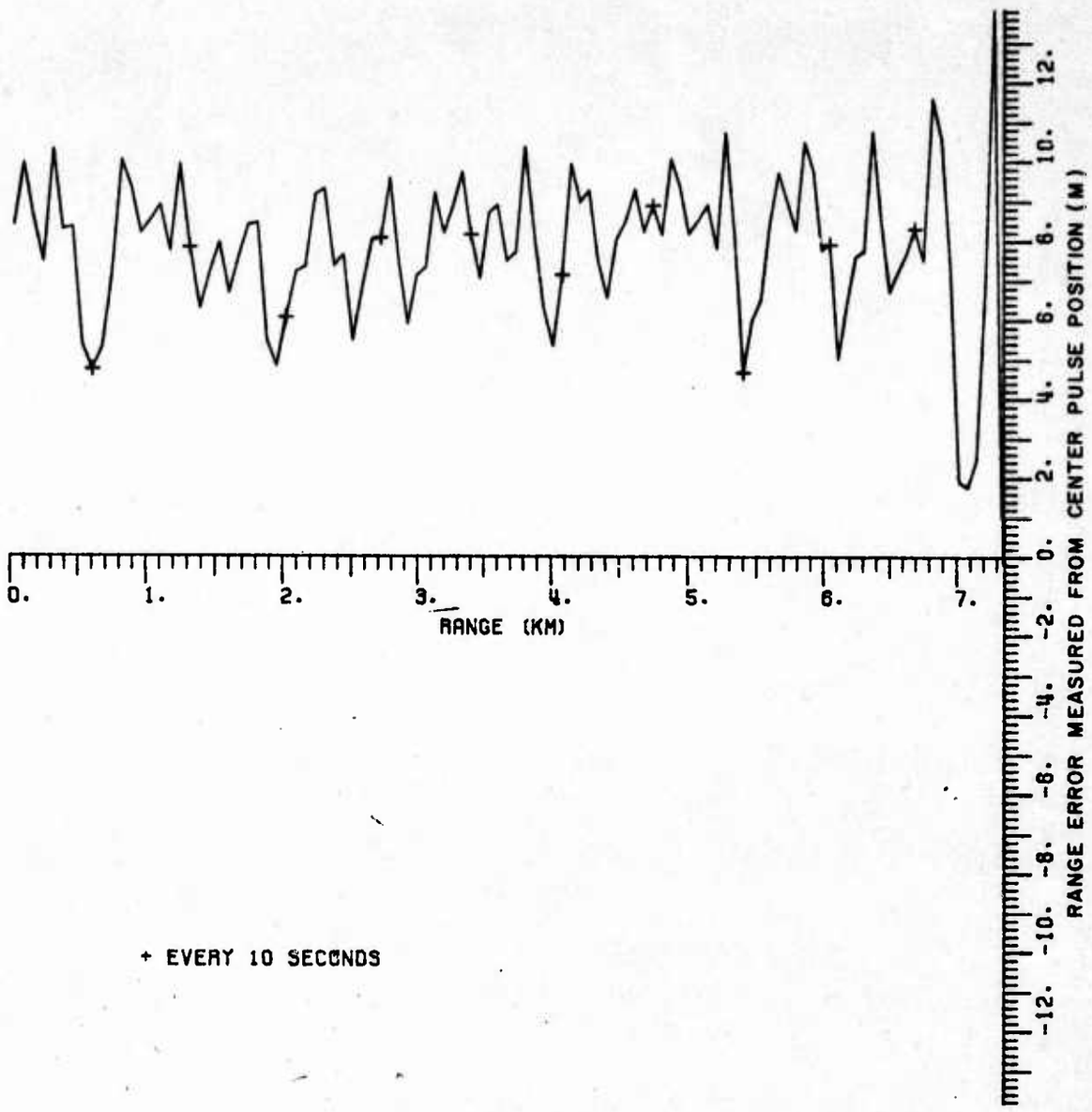


Figure G-11. v = 200 m/s. Split-gate tracker.

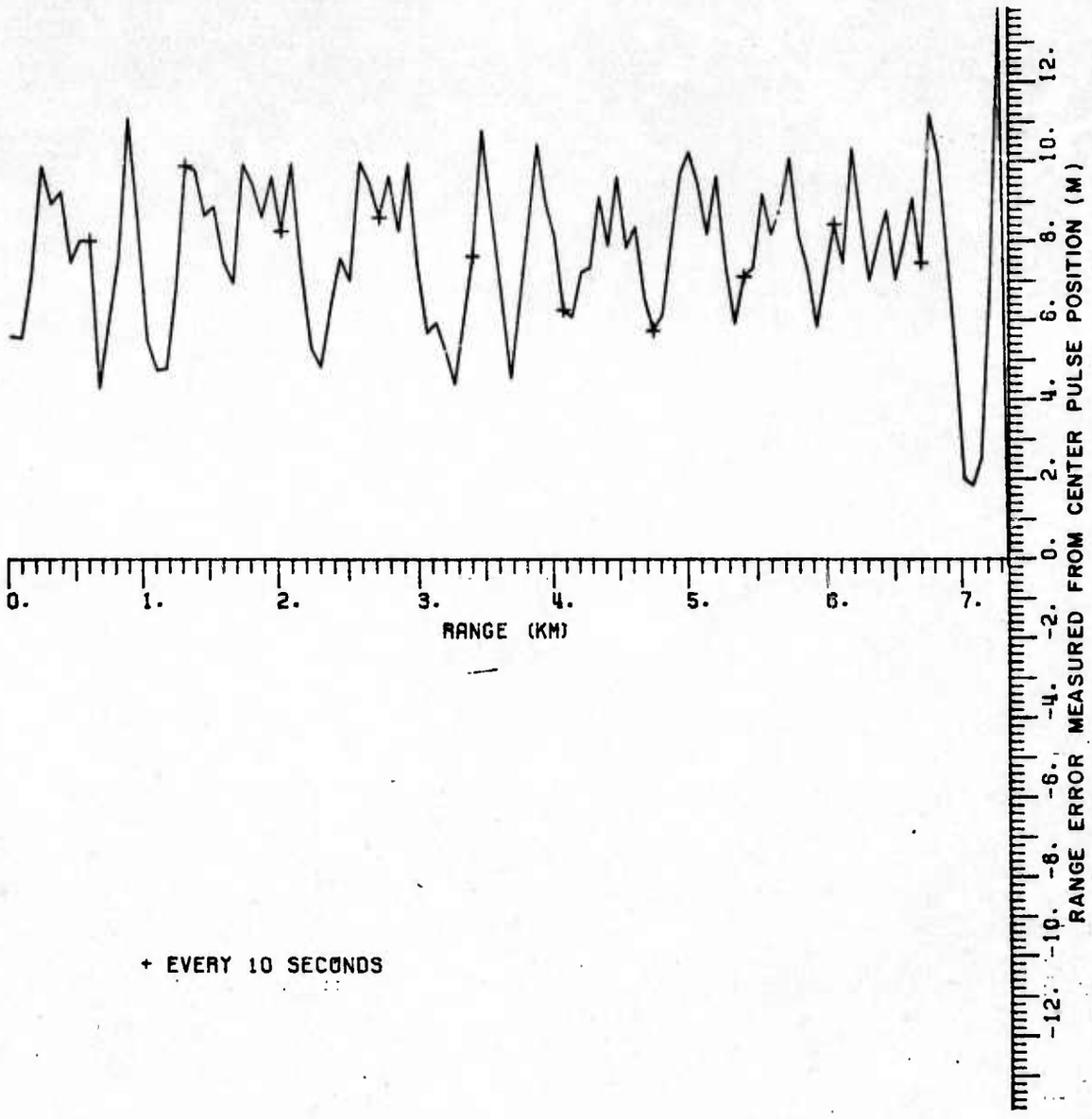


Figure G-12. v = 260 m/s. Split-gate tracker.

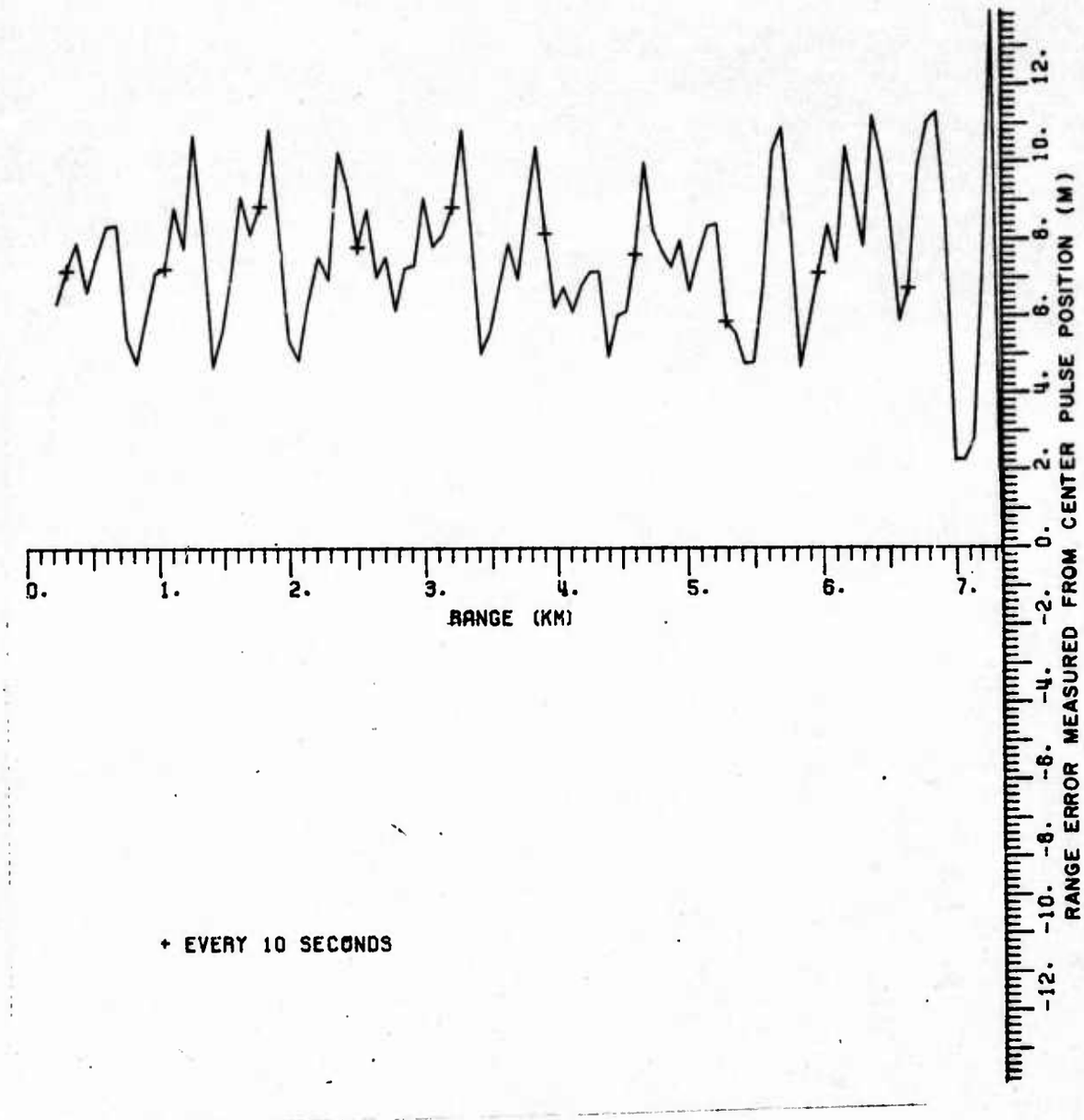


Figure G-13. v = 340 m/s. Split-gate tracker.

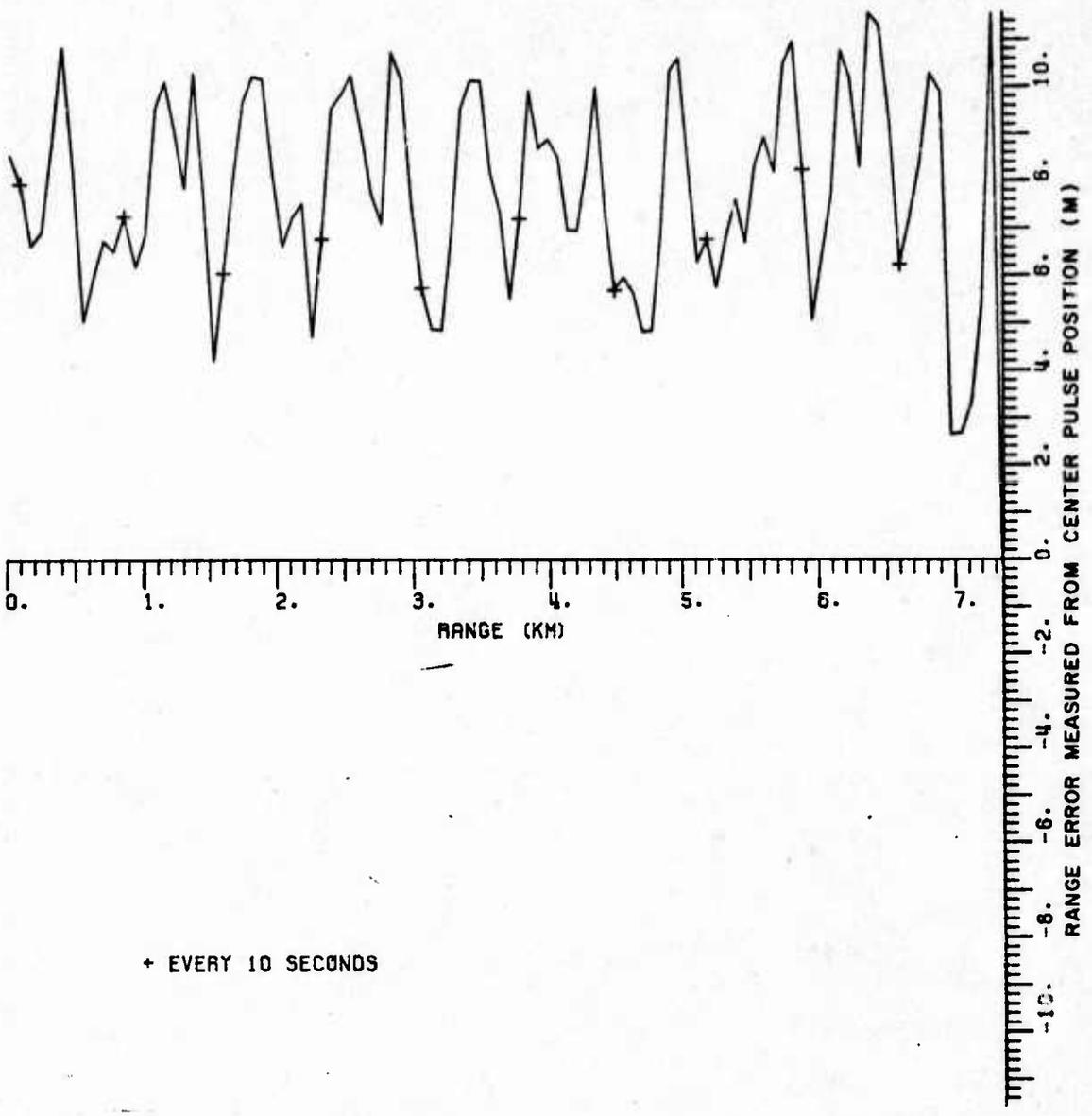


Figure G-14. v = 380 m/s. Split-gate tracker.

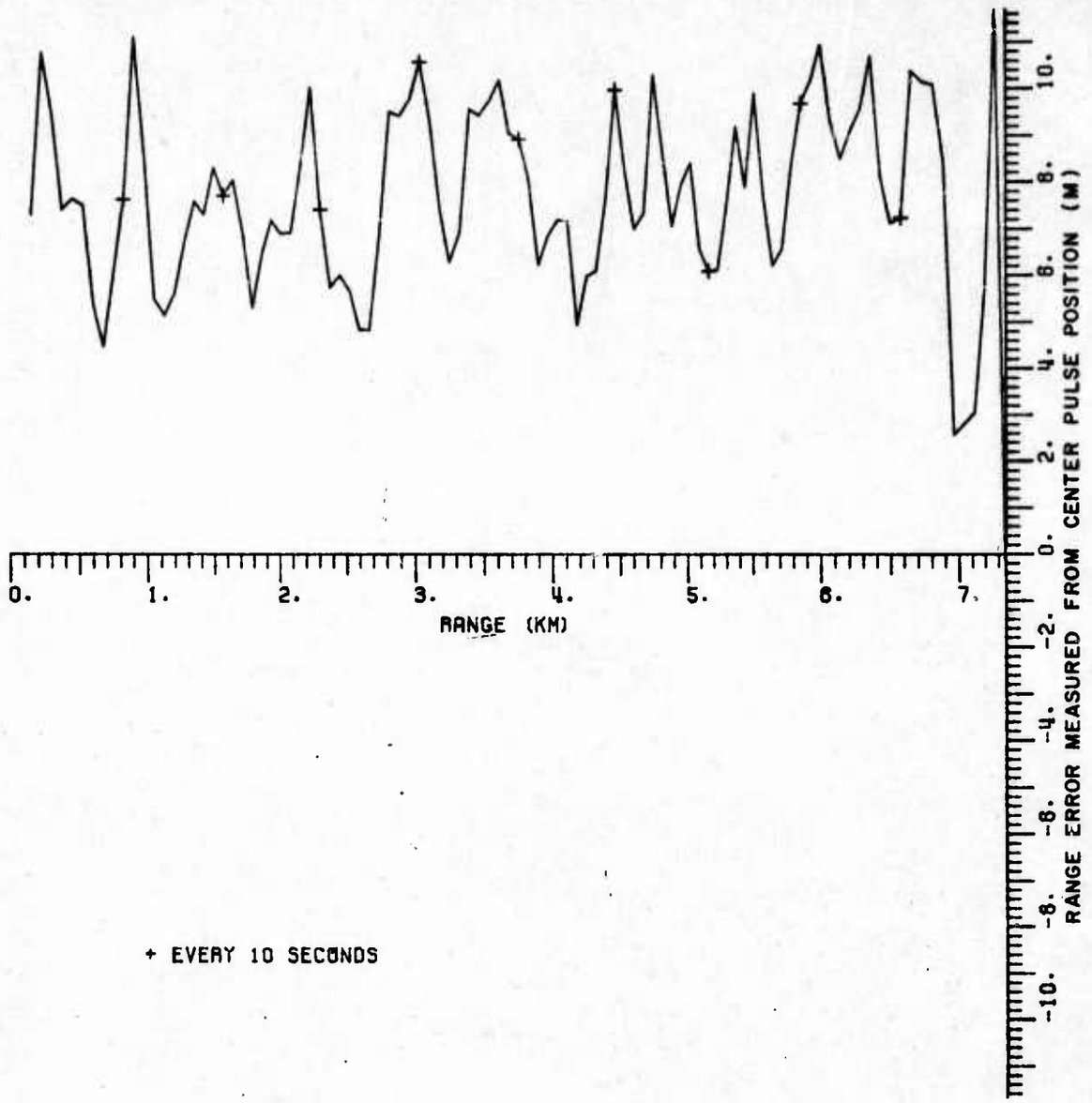


Figure G-15. v = 410 m/s. Split-gate tracker.

F. COMPUTER PROGRAM

```
      INCLUDE DELAY,LIB1
      INCLUDE POPSUB,LIB1
      INCLUDE MISSB,2000
      LOGICAL LOUT,ICHAFF,LAMS
      DIMENSION CPA(3),V(3)
      DIMENSION PLANE(5),ROCKET(5),CHAFF(3,100),IBUFF(510)
      DIMENSION SPLITA(3),DAMP(3),DR(3)
      EXTERNAL CRETLT
      COMMON SIZE,PLANE,ROCKET,CHAFF,NCHAFF,LOUT,RANGE,THETA,PHI
      COMMON RANGES,THETAS,PHIS,SNR,EARLY,ALATE,ADJUST,XINTER,YINTER
      COMMON IMODE
      DATA SPLANE,WPLANE/.01,4000./
      DATA SCHAFF,WCHAFF/.04,150./
      DATA RMIN/.1/
      DATA DR1/.15/
      DATA SPLITA/-.01,-.0015,-.01/
      DATA TURNS,TURNS1/.01,.01/
      DATA STDGND/100./
      DATA BNOISE/1./
      DATA WEAGT/.05/
      DATA WMAX/.5/
      DATA BASEV/.01/
      DATA DV/.001/
      DATA ROCKV/.05/
      DATA ROCKV2/.05/
      DATA ICHAFF/.FALSE./
      DATA DR/.85,.003,7./
      DATA THETAN/.01/
      DATA PHIN/.01/
      DATA RANGEN/1./
      DATA TSPLIT/.008/
      DATA PSPLIT/.008/
      DATA RANGN1/1./
      DATA DAMP/-.01,-.0015,-.01/
      DATA CHAMIN/.03/
      DATA TDAMP,PDAMP/.007,.007/
      DATA WNEW/.2/
      DATA VROCK1/.0001/
      DATA SWMODE/.3/
      DATA PLANEA/.001/
      DATA WNEW/.01/
C     DDTT IS THE TIME INTERVAL, ASSUMED TO BE ONE SECOND
      CALL CRTON
      CALL CRTPTS(IBUFF,510,4)
S     DDTT=1.
      RANGE=0.
      RANGES=0.
      THETA=0.
      THETAS=0.
      PHI=0.
      PHIS=0.
      TPP=0.
      TMP=0.
      XINTER=0.
      YINTER=0.
```

```
      CALL FERR(2)
      *CHAFF=0
      IRNG=4
      IANG=1
      LOUT=.FALSE.
      PLANE(3)=5.
      PLANE(4)=0.
      PLANE(5)=0.
      ROCKET(1)=5.
      ROCKET(2)=2.
      ROCKET(3)=0.
      ROCKET(4)=-160.
      ROCKET(5)=42.
      RANGES=-1.
      ROCKET(4)=ROCKET(4)*3.1415926/180.
      ROCKET(5)=ROCKET(5)*3.1415926/180.
      PLANE(4)=PLANE(4)*3.1415926/180.
      PLANE(5)=PLANE(5)*3.1415926/180.
      TURNS1=ABS(TURNS1)
      WRITE(8,9)
 9    FORMAT('USE CANNED DATA? (T/F)')
      READ(8,*)LANS
      IF(LANS) GO TO 101
      CALL ESC($12)
      WRITE(8,10)
10    FORMAT('ENTER OPTION VALUES'/' 1=R**2 2=SPLIT GATE 3=DELAY LINE',
     *' 4=THRESHOLD'/' 1=COS**2 '/
     *'T=DUMP TO TAPE F=NO DUMP')
12    READ(8,*)IRNG,IANG,LOUT
15    WRITE(8,20)
20    FORMAT('ENTER PLANE ELEVATION,HEADING,DIVE ANGLE')
      READ(8,*)PLANE(3),PLANE(4),PLANE(5)
      PLANE(4)=PLANE(4)*3.1415926/180.
      PLANE(5)=3.1415926*PLANE(5)/180.
      PLANE(1)=0.
      PLANE(2)=0.
      WRITE(8,30)
30    FORMAT('ENTER MISSILE POSITION (X,Y,Z),HEADING,DIVE ANGLE')
      READ(8,*)ROCKET
      ROCKET(4)=ROCKET(4)*3.1415926/180.
      ROCKET(5)=ROCKET(5)*3.1415926/180.
      CALL CALANG(-ROCKET(1),-ROCKET(2),PLANE(3)-ROCKET(3)
     *,RANGE,THETA,PHI)
      RANGES=RANGE
      CALL OUTPUT
      WRITE(8,40)
40    FORMAT('OK? (T/F)')
      READ(8,*)LANS
      IF(.NOT.LANS)GO TO 15
      WRITE(8,50)
50    FORMAT('ENTER TRACKING INFO. (R,THETA,PHI,DR) ')
      READ(8,*)RANGES,THETAS,PHIS,RANGEV
101   THETAS=THETAS*3.1415926/180.
      PHIS=PHIS*3.1415926/180.
      TS=0
```

```
111         DO 55 I=1,5
112         CALL POPSW(5-I,J)
113  55     IS=IS*2+2-J
114         PLANEV=BASEV+DV*IS
115         IF(RANGES.GT.0)GO TO 60
116         CALL CALANG(-ROCKET(1),-ROCKET(2),PLANE(3)-ROCKET(3),
117        *  RANGES,THETAS,PHIS)
118         RANGEV=PLANEV*COS(PLANE(4)-THETAS)*COS(PLANE(5)-PHIS)-
119        *   ROCKV*COS(ROCKET(4)-THETAS)*COS(ROCKET(5)-PHIS)
120         RANGES=RANGES-RANGEV
121  60     THETAP=THETAS
122         PHIP = PHIS
123         NCHAFF=0
124         TMISS=4
125         THETAL=0.
126         PHIL=0.
127         THETAV=0.
128         PHIV=0.
129         XB=0.
130         YB=0.
131         ADJUST=0.
132         IMODE=1
133         CALL ESC(1000)
134  C      MAIN LOOP.   TRACKING SEGMENT
135  100    DO 250 II=1,250
136  C        PRIME ANGLE VALUES
137         CALL CALANG(-ROCKET(1),-ROCKET(2),PLANE(3)-ROCKET(3)
138        *,RANGE,THETA,PHI)
139         DIS2=RANGE*RANGE
140         GO TO(110,120,130,140),IRNG
141  110    SIGNAL=SINGNO/AMAX1(DIS2,RMIN)
142         SIGNAL=SIGNAL*(1.+COS(THETA-ROCKET(4)))**2*(1.+COS(PHI-ROCKET(5)))**2
143         RANGES=RANGE
144         ANOISE=BNOISE+RETUR1(1,RANGE-DR1,RANGE+DR1)
145         SNR=(SIGNAL+ANOISE)/ANOISE
146         RANGES=RANGES+RANF(RANGN1)/SNR
147         GO TO 150
148  120    RANGES=RANGES+RANGEV
149         IRNG1=IRNG/2
150         EARLY=RETURN(IRNG1,RANGES-DR(IMODE),RANGES)+
151        *   RETUR1(IRNG1,RANGES-DR(IMODE),RANGES)
152         ALATE=RETURN(IRNG1,RANGES,RANGES+DR(IMODE))+
153        *   RETUR1(IRNG1,RANGES,RANGES+DR(IMODE))
154         SIGNAL=RETURN(1,RANGES-DR(IMODE),RANGES+DR(IMODE))
155         ANOISE=BNOISE+RETUR1(1,RANGES-DR(IMODE),RANGES+DR(IMODE))
156  125    FORMAT(2F10.4)
157         SNR=(SIGNAL+ANOISE)/ANOISE
158         EARLY=ABS(EARLY+RANF(RANGEN)/SNR)
159         ALATE=ABS(ALATE+RANF(RANGEN)/SNR)
160         IF(SIGNAL.LE.0.)GO TO 150
161         ADJUST=(EARLY-ALATE)/(EARLY+ALATE)
162         IF(II.EQ.1)ADJUSL=ADJUST
163         RANGEV=RANGEV+SPLITA(IRNG1)*ADJUST+DAMP(IRNG1)*(ADJUST-ADJUSL)
164         IF(ABS(ADJUST)+ABS(ADJUST-ADJUSL).GT.SWMODE)GO TO 120
165         IF(IMODE.EQ.2.OR.II.LT.5)GO TO 120
```

```
166         TMODE=2
167         WRITE(6,127)
168 127     FORMAT(' TARGET ACQUIRED')
169 125     ADJUSL=ADJUST
170         GO TO 150
171 130     STOP RANGE
172 140     GO TO 120
173 150     THETAS=THETAS+THETAV
174         PHIS=PHIV+ATAN(COS(THETAV)*TAN(PHIS))
175         GO TO(155,160,170,180),IANG
176 155     CONTINUE
177         THETAS=THETA+RANF(THETAN)/SNR
178         PHIS=PHI+RANF(PHIN)/SNR
179         WEIGHT=SIGNAL*WFACT
180         IF(WEIGHT.GT.WMAX)WEIGHT=WMAX
181         GO TO 200
182 160     PH=0.
183         PL=0.
184         TH=0.
185         TL=0.
186         CALL ANGLE2(-ROCKET(1),-ROCKET(2),PLANE(3)-ROCKET(3),
187        *SPLANE,WPLANE,PH,PL,TH,TL)
188         IF(NCHAFF.EQ.0)GO TO 165
189         DO 162 I=1,NCHAFF
190 162     CALL ANGLE2(CHAFF(1,I)-ROCKET(1),CHAFF(2,I)-ROCKET(2),
191        *CHAFF(3,I)-ROCKET(3),SCHAFF,WCHAFF,PH,PL,TH,TL)
192 165     CONTINUE
193         TH=ABS(TH+RANF(THETAN)/SNR)
194         TL=ABS(TL+RANF(THETAN)/SNR)
195         PH=ABS(PH+RANF(PHIN)/SNR)
196         PL=ABS(PL+RANF(PHIN)/SNR)
197         ATHETA=(TH-TL)/(TH+TL)
198         APHI=(PH-PL)/(PH+PL)
199         IF(II.NE.1)GO TO 168
200         THETAL=ATHETA
201         PHIL=APHI
202 168     ATHETA=ATHETA*TSPLIT+(ATHETA-THETAL)*TDAMP
203         APHI=APHI*PSPLIT+(APHI-PHIL)*PDAMP
204         THETAV=THETAV+ATHETA/RANGES
205         PHIV=PHIV+APHI/RANGES
206         IF(ABS(PHIS).LT.1.5707969)GO TO 200
207         THETAS=THETAS+3.1415926
208         IF(PHIS)166,166,167
209 166     PHIS=-3.1415926-PHIS
210         GO TO 200
211 167     PHIS=3.1415926-PHIS
212         GO TO 200
213 170     PH=0.
214         PL=0.
215         TH=0.
216         TL=0.
217         CALL ANGLE2(-ROCKET(1),-ROCKET(2),PLANE(3)-ROCKET(3),SPLANE
218        *,WPLANE,PH,PL,TH,TL)
219         GO TO 165
220 180     STOP ANGLE
```

```
  200  CONTINUE
2200   X2=XB+ROCKET(1)+RANGES*COS(THETAS)*COS(PHIS)
       Y2=YB+ROCKET(2)+RANGES*SIN(THETAS)*COS(PHIS)
       IF(II-5)2201,2202,2203
2201   X0=X2
       Y0=Y2
       GO TO 2220
2202   X1=X0
       Y1=Y0
       VX=(X2-X1)/DDTT
       VY=(Y2-Y1)/DDTT
       VO=SQRT(VX*VX+VY*VY)
       TINTER=RANGES/ROCKV
       TMP=TINTER
       TMP=TINTER
       GO TO 2204
2203   XP=X1+VX*DDTT
       YP=Y1+VY*DDTT
2209   W=WNEW/AMAX1(RANGES,1.)
       X2=(XP+W*X2)/(1+W)
       Y2=(YP+W*Y2)/(1+W)
       VX=(X2-X0)/DDTT/2.
       VY=(Y2-Y0)/DDTT/2.
       VO=SQRT(VX*VX+VY*VY)
2210   TINTER=(TINTER+TMP)/2.-DDTT
2204   XINTER=(XINTER+VX*TINTER+X2)/2.
       YINTER=(YINTER+VY*TINTER+Y2)/2.
       DPM=SQRT((XB+ROCKET(1)-XINTER)**2+(YB+ROCKET(2)-YINTER)**2)
       COS5=COS(ROCKET(5))
       IF(COS5. EQ. 0.)COS5=1.
       TMP=DPM/(ROCKV+(TMP-DDTT)*VROCKI*.5)/COS5
       X0=X1
       Y0=Y1
       X1=X2
       Y1=Y2
C1     WRITE(6,2303)XB,YB,X2,Y2,XP,YP,VX,VY,XINTER,YINTER,VROCK,
C1     *  TMP,TINTER,THETAP,PHIP,PTHETA,PPHI
2303   FORMAT(5('(',F8.5,',',F8.5,')'),F8.5/
       *5('(',F8.5,',',F8.5,')'))
C      SINCE WE NOW HAVE X,Y INTERCEPT THEN, FIND
C      RELATIVE ANGLE FROM ROCKET TO PLANE
       RELIX=XINTER-ROCKET(1)-X2
       RELIY=YINTER-ROCKET(2)-Y2
       RELIZ=RANGES*SIN(PHIS)
       CALL CALANG(RELIX,RELIY,RELIZ,RNG,PTHETA,
       *PPHI)
C1     IF(RNG.GT.,1) GO TO 2212
C1     WRITE(6,2304) TINTER,RNG,ROCKV,PLANE(3),ROCKET(1),
C1     *ROCKET(2),ROCKET(3)
 2304  FORMAT('INTERCEPT  '7F10.5)
 2212  IF((SNR.LE.1.5).OR.(TINTER.LT.4).OR.(TMP.LT.A)) GO TO 2219
2213   IF(II-10) 2220,2230,2231
C      INITIALLY SET TO CALCULATED VALUES
 2230  THETAP=THETAS
       PHIP=PHIS
```

```
276         GO TO 2220
277 C       TEST TO SEE IF GOING TOO FAST
278  2231   TCOR = .6*PRIME(PTHETA-THETAP)
279         IF (ABS(TCOR).LT..2) GO TO 2234
280         IF(TCOR) 2232,2233,2233
281  2232   TCOR = -.2
282         GO TO 2234
283  2233   TCOR = .2
284  2234   THETAP = THETAP+TCOR
285         PHIP=PHIP+.3*PRIME(PPHI-PHIP)
286 C1      WRITE(6,2305) YP,YP,VX,VY,VJ
287 C1      WRITE(6,2307) TINTER,XINTER,YINTER,RNG,PTHETA,PPHI,THETAP
288 C1     *,PHIP
289  2307   FORMAT('TINTER,XINTER,YINTER,RNG,PTHETA,PPHI,THET'
290         **AP,PHIP'/8F10.5)
291  2206   CONTINUE
292         GO TO 2216
293 C       TURN ROCKET TOWARDS PREDICTED TARGET IF COURSE PREDICTED
294 C       IF NEAR TARGET COMPUTE CPA AND CHECK IF TIME TO EXPLODE
295  2219   V(1)=ROCKV*COS(THETAP)*COS(PHIP)-PLANEX
296         V(2)=ROCKV*SIN(THETAP)*COS(PHIP)-PLANEY
297         V(3)=ROCKV*SIN(PHIP)-PLANEZ
298 C1      WRITE(6,2321)SNR,TINTER,TMP
299  2321   FORMAT('CHECKING CPA ',3F10.5)
300 C       CALCULATE TIME TO CPA
301         S1=0.
302         S2=0.
303         DO 2218 I=1,3
304         S1=S1+V(I)*(ROCKET(I)-PLANE(I))
305  2218   S2=S2+V(I)*V(I)
306         TIMCPA=-S1/S2
307 C1      WRITE(6,2323)V,S1,S2,TIMCPA,PLANEX,PLANEY,PLANEZ
308  2323   FORMAT('TERMS'12F10.5)
309 C       CHECK IF MISSILE HAS LOST TRACK
310         IF(SNR.LT.1.5)GO TO 2229
311 C       CHECK IF CPA COMING UP
312         IF(TIMCPA.GT.1.0)GO TO 2213
313         IF(TIMCPA.LT.0.)GOTO2215
314 C       CALCULATE POSITION OF CPA
315  2214   S1=0
316         DO 2217 I=1,3
317         CPA(I)=ROCKET(I)-PLANE(I)+V(I)*TIMCPA
318  2217   S1=S1+CPA(I)*CPA(I)
319         S1=SQRT(S1)
320 C       IS CPA CLOSE ENOUGH TO COUNT
321 C1      WRITE(6,2324)S1
322  2324   FORMAT('DIS'F10.5)
323         TINTER=TINTER-TIMCPA
324         WRITE(6,2320)CPA,TIMCPA,S1,TINTER,ROCKV
325  2320   FORMAT('INTERCEPT='7F10.5)
326         CALL LETTER(3,.7,.5,3,.4,12H INTERCEPT  ,ID,ID,12)
327         CALL LETTER(2,.7,.1,.4,6H CPA (,ID,ID,5)
328         CALL NUMBER(2,.7,.7,.,.15,1000.*CPA(1),0,.1)
329         CALL NUMBER(3,.7,.7,.,.15,1000.*CPA(2),0,.1)
330         CALL NUMBER(4,.7,.7,.,.15,1000.*CPA(3),0,.1)
```

```
331         CALL LETTER(5.5,7.,1.4,3H ) ,ID,ID,3)
332         CALL LETTER(2.,6.5,1.4,6H RANGE,ID,ID,6)
333         CALL NUMBER(3.,6.5,.15,1000*S1,0.,1)
334         CALL CRTPLT(0.,0.,999)
335         GO TO 1000
336 C       HERE IF SNR SMALL
337 2229    IF(IMODE.EQ.2)GO TO 2228
338         IF(TIMCPA .GT. 4.)GO TO 2215
339         IF(TIMCPA.LT.1.)GO TO 2214
340         GO TO 2216
341 2215    TMISS=TMISS-1
342 C1      WRITE(6,2322)TMISS,TIMCPA
343 2322    FORMAT('NO GOOD'2F10.5)
344         TIMCPA=0.
345         IF(TMISS)2214,2214,2216
346 2228    IMODE=1
347         WRITE(6,2227)
348 2227    FORMAT('SWITCHING TO ACQUISITION MODE')
349 2216    ROCKET(4)=THETAP
350         ROCKET(5)=PHIP
351 C       OUTPUT INFO TO OPERATOR
352 C       PUT FIRST INTERCEPT MUST BE CONVERTED TO PLANES REF
353 C       FROM ABSOLUTE REF
354 C       SWITCH TO ACQUIRE MODE IF MISSILE LOSSES TRACK
355 2220    XINTER=XINTER-X2
356         YINTER=YINTER-Y2
357         CALL OUTPUT
358         XINTER=XINTER+X2
359         YINTER=YINTER+Y2
360         IS=0
361         DO 205 I=1,5
362         CALL POPSW(5-I,J)
363         IS=IS*2+2-J
364 205     CONTINUE
365         PLANV1=BASEV+DV*IS
366         IF(ABS(PLANV1-PLANEV).GT.PLANEA)GO TO 2051
367         PLANEV=PLANV1
368         GO TO 2053
369 2051    IF(PLANV1.LT.PLANEV)GO TO 2052
370         PLANEV=PLANEV+PLANEA
371         GO TO 2053
372 2052    PLANEV=PLANEV-PLANEA
373 2053    COSP=COS(PLANE(5))
374         PLANEX=PLANEV*COS(PLANE(4))*COSP
375         PLANEY=PLANEV*SIN(PLANE(4))*COSP
376         PLANEZ=PLANEV*SIN(PLANE(5))
377         PLANE(3)=PLANE(3)+PLANEZ
378 C       GET PLANE TURNINGS
379 C       HEADING CHANGE IN 6-9
380 C       RISE ANGLE CHANGE IN 10-13
381         IS=0
382         DO 206 I=1,4
383         CALL POPSW(10-I,J)
384 206     IS=IS*2+2-J
385         PLANE(4)=PLANE(4)+(IS-8)*TURNS
```

```
386         IS=0
387         DO 207 I=1,4
388         CALL POPSW(14-I,J)
389 207     IS=IS*2+2-J
390         PLANE(5)=PLANE(5)+(IS-8)*TURNS1
391 C       PROCESS PLANE LOOPINGS
392         IF(ABS(PLANE(5)).LT.1.57079)GO TO 209
393         PLANE(4)=PLANE(4)+3.14159
394         TURNS1=-TURNS1
395         IF(PLANE(5).LT.0)GO TO 208
396         PLANE(5)=3.1415926-PLANE(5)
397         GO TO 209
398 208     PLANE(5)=-3.1415926-PLANE(5)
399 209     CONTINUE
400 C       MOVE THE PLANE (PLANE IS MOVED BY MOVING EVERYTHING ELSE
401 C       IN THE OPPOSITE DIRECTION)
402         ROCKET(1)=ROCKET(1)-PLANEX
403         ROCKET(2)=ROCKET(2)-PLANEY
404         XB=XB+PLANEX
405         YB=YB+PLANEY
406         DO 210 I=1,NCHAFF
407         CHAFF(1,I)=CHAFF(1,I)-PLANEX
408 210     CHAFF(2,I)=CHAFF(2,I)-PLANEY
409         COSP=COS(ROCKET(5))
410 C       MOVE THE MISSILE IN ITS STRAIGHT LINE PATH
411 C       INCREASE SPEED VS TIME
412         ROCKV=ROCKV2+II*VROCKI
413         ROCKET(1)=ROCKET(1)+ROCKV*COS(ROCKET(4))*COSP
414         ROCKET(2)=ROCKET(2)+ROCKV*SIN(ROCKET(4))*COSP
415         ROCKET(3)=ROCKET(3)+ROCKV*SIN(ROCKET(5))
416 C       DROP A CHAFF CLOUD IF SWITCH 17 IS UP AND WAS DOWN BEFORE
417 C       (ICHAFF INDICATES PREVIOUS STATE OF CHAFF SWITCH)
418         CALL POPSW(17,I)
419         GO TO (220,230),I
420 220     IF(ICHAFF)GO TO 240
421         NCHAFF=NCHAFF+1
422         DO 225 I=1,3
423         CHAFF(I,NCHAFF)=PLANE(I)
424 225     CONTINUE
425         ICHAFF=.TRUE.
426         GO TO 240
427 230     ICHAFF=.FALSE.
428 240     CONTINUE
429 C       CHECK IF MISSILE HAS LOST TRACK
430   250   CONTINUE
431 C       THIS IS THE SECTION TO DECIDE WHAT TO DO WHEN THE PROGRAM
432 C       HAS BEEN TERMINATED
433 1000    II=II-1
434         WRITE(6,1010)II
435 1010    FORMAT(I4/'CONTINUE,CODE(EXIT,CONT,RESTART)')
436         READ(6,*)I
437         IF(I)1020,100,*
438 1020    CALL EXIT
439         END
```

REFERENCES

1. Van de Hulst, H. C., Light Scattering by Small Particles, John Wiley and Sons, Inc., 1957.

2. Glock, F., M. Hammermesh, and M. Phillips, "Return Cross Section from Randomly Oriented Resonant Half-Wavelength Chaff," Radio Research Laboratory, Harvard University, Cambridge, Massachusetts, Tech. Memo 411-TM-427, June 1944.

3. Fink, B., "Radar Countermeasures," Electronics, Vol. 19, pp. 92-97, January 1946.

4. Hessenrer, R. A., Jr., "Scatter Communications with Radar Chaff," Proc. IRE, Vol. 49, No. 3, March 1961, pp. 211-217.

5. Kaiper, G. P., "A Study of Chaff Echoes at 515 MHz," Radio Research Lab., Harvard University, Cambridge, Massachusetts, Rept. 411-73, December 1943.

6. Kelly, E. J., and E. C. Lerner, "A Mathematical Model for the Radar Echo from a Random Collection of Scatterers," Tech. Rept. No. 123, M.I.T. Lincoln Laboratory, 15 June 1956, AD 48613.

7. Wong, J. L., I. S. Reed, and Z. A. Kaprielian, "A Model for Radar Echo from a Collection of Rotating Dipole Scatterers," IEEE Trans. on Aero. and Elect. Syst., Vol. AES-3, No. 2, March 1967.

8. Wong, J. L., I. S. Reed, and Z. A. Kaprielian, "Scattering by Randomly Varying Media with Application to Radar Detection and Communications," Report USCEE 247, University of S. California, L.A., January 1968, AD 835 645.

9. Richmond, J. H., "Computer Analysis of Three Dimensional Wire Antennas," Report 2708-4, 22 December 1969, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract DAAD05-69-C-0031 for Department of the Army, Aberdeen Proving Ground, Maryland.

10. Richmond, J. H., L. Schwab and R. Wickliff, "Scattering Characteristics of Some Thin Wire Chaff Elements," Report 2584-8, 19 February 1970, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract F33615-68-C-1252 for Air Force Avionics Laboratory, Wright-Patterson Air Force Base, Ohio. (AD 865 406)

11. Wickliff, R. G., "Scattering from Some Finite Arrays of Randomly Oriented Coupled Dipole Elements," Report 2584-12, June 1971, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract F33615-68-C-1252 for Air Force Avionics Laboratory, Wright-Patterson Air Force Base, Ohio. (AD 885 859)

12. Garbacz, R. J., R. Wickliff, V. Cable, R. Caldecott and D. Lam, "Advanced Radar Reflector Studies," Report 3401-1, June 1974, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract F33615-72-C-1435 for Wright-Patterson Air Force Base.

13. Rayleigh, Lord (J. W. Strutt), The Theory of Sound, 1877, Dover Publications, pp. 35-42.

14. Borison, S. L., "Probability Density for the Radar Cross Section of One or More Randomly Oriented Dipoles," Group Rept. 1964-33, M.I.T. Lincoln Laboratory, 22 June 1964. AD 442 679.

15. Stantar, R., "Chaff Cloud Signature I Measurement Program," Rept. AFAL-TR-73-57, Hycor, Inc., Woburn, Mass., under USAF Avionics Laboratory Contract F33615-72-C-1606.

16. Spiegel, M. R., Theory of Problems of Statistics, Schaum Publishing Co., New York, 1961, p. 70.

17. Spiegel, M. R., op. cit., p. 29.

18. Spiegel, M. R., op. cit., p. 123.

19. Hoel, P. G., Introduction to Mathematical Statistics, 3rd Ed., John Wiley and Sons.

20. Spiegel, M. R., op. cit., pp. 142-144.

21. Richmond, J. H., "Reaction Theorems and Plane Surface Waves," Engineering Experiment Station, College of Engineering, The Ohio State University, Vol. 28, No. 4, July 1959, Part I.

22. Harrington, R. F., Field Computation by Moment Methods MacMillan Company, New York, 1968.

23. Kennaugh, E. M., "Polarization Properties of Radar Reflections," Report 389-12, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract AF 28(099)-90 for Rome Air Development Center, March 1952.

24. Richmond, J. H., "Radiation and Scattering by Thin-Wire Structures in the Complex Frequency Domain," Report 2902-10, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Grant NGL 36-008-138 for National Aeronautics and Space Administration, July 1973.

25. Carson, J. R., "Reciprocal Theorems in Radio Communication," Proc. of the IRE, Vol. 17, June 1929, pp. 952-956.

26. Richmond, J. H., "Radiation and Scattering by Thin-Wire Structures in the Complex Frequency Domain," Report 2902-10, July 1973, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering;;prepared under Grant No. NGL 36-008-138 for National Aeronautics and Space Administration.

27. Kouyoumjian, R. G., "The Calculation of the Echo Areas of Perfectly Conducting Objects by the Variational Method," Doctoral Thesis, The Ohio State University, Columbus, Ohio, 1953.

28. Wang, N. N., J. H. Richmond, et al., "Sinusoidal Reaction Formulation for Radiation and Scattering from Conduting Surfaces," IEEE Trans. on Antennas and Propagation, Vol. AP-23, No. 3, May 1975, pp. 376-382.

29. Richmond, J. H., "A Wire-Grid Model for Scattering by Conduting Bodies," IEEE Trans. on Antennas and Propagation, Vol. AP-14, No. 4, November 1966, pp. 782-786.

30. Lin, Y. T. and J. H. Richmond, "EM Modeling of Aircraft at Low Frequencies," IEEE Trans. on Antennas and Propagation, Vol. AP-23, No. 1, January 1975, pp. 53-56.

31. Thiele, G. A., "Wire Antennas," in Computer Techniques for Electromagnetics, ed. by R. Mittra, Pergamon Press, New York, 1973, Chapter 2.

32. Richmond, J. H., "Computer Analysis of Three-Dimensional Wire Antennas," Report 2708-4, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract DAAD05-69-C-0031 for Ballistic Research Laboratory, Aberdeen Proving Ground, December 1969.

33. Miller, E. K. and F. J. Deadrick, "Some Computational Aspects of Thin Wire Modeling," Interaction Notes, Note 153, Lawrence Livermore Laboratory, University of California, Livermore, June 19, 1973.

34. Tai, C. T., "Electromagnetic Back-Scattering from Cylindrical Wires," Journal of Applied Physics, Vol. 23, No. 8, August 1952, pp. 909-916.

35. Miller, E. K. and F. J. Deadrick, "Thin Wire Modeling."

36. Richmond, J. H., "Computer Analysis."

37. Ibid, pp. 10-15.

38. Cable, V.P., "Application of Linear Iteration to Electromagnetic Scattering by Random Arrays of Wires," Report 3401-2, August 1975, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract F33615-72-C-1435 for Wright-Patterson Air Force Base, Ohio.

39. Ibid, pp. 10-15.

40. Rose, D. J. and R.A. Willoghly (Ed.), Sparse Matrices and Their Applications, Plenum Press, New York - London, 1972.

41. Tinney, W. F. and J. W. Walker, "Direct Solutions of Sparse Network Equations by Optimally Ordered Triangular Factorization," IRRE Proceedings, Vol. 55, No. 11, November 1967, pp. 1801-1809.

42. Sato, N. And W. F. Tinney, "Techniques for Exploiting the Sparsity of the Network Admittance Matrix," IEEE Trans. on Power Apparatus and Systems, Vol. 82, December 1963, pp. 944-950.

43. Gustavson, F. G. W. Liniger, and R. Willoughbly, "Symbolic Generation of an Optical Crout Algorithm for Sparse Systems of Linear Equations," Journal of the Association for Computing Machinery, vol. 17, no. 1, January 1970, pp. 87-109.

44. Berry, R.D., "An optimal Ordering of Electronic Circuit Equations for a Sparse Matrix Soluiton," IEEE Trans. on Circuit Theory, vol. CT-18, no. 1, January 1971, pp. 40-50.

45. Tewarson, R. P., Sparse Matrices , Academic Press, New York-London, 1973.

46. Fedeev, D. K. and V. N. Faddeeva, Computational Methods of Linear Algebra, W. H. Freeman and Co., San Francisco, 1963, pp. 144-147.

47. Westlake, J. R., A Handbook of Numerical Matrix Inversion and Solution of Linear Equations, John Wiley and Sons, Inc., New York, London, Sydney, pp. 4-15.

48. Varga, R. S., Matrix Iterative Analysis, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1962, pp. 56-57.

49. Varga, Matrix Iterative Analysis pp. 56-57.

50. Ibid, p. 58.

51. Young, D. M., Iterative Solution of Large Linear Systems, Academic Press, New York, 1975, Chapter 11.

53. Ibid.

54. Ibid.

55. Varga, Matrix Iterative Analysis, Chapter 3.

56. Ibid,

57. Young, Iterative Solution, p. 41.

58. Fair, G., "ALLMAT: ATSS-360 FORTRAN IV Subroutine for Eigenvalues and Eigenvectors of a General Complex Matrix," Lewis Research Center, NASA TN O-7032, N71-15833, January 1971.

59. Tai, C. T., "An Iterative Method."

60. Castello, D. and B.A. Munk, "Table of Mutual Impedance of Identical Dipoles in Echolon," Report 2382-1, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract F 33615-67-C-1507 for Air Force Avionics Laboratory, Wright-Patterson Air Force Base, October 1967.

61. Garbacz, R. J., "Introduction of Characteristic Modes for Chaff Applications," Report 2584-6, April 1970, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract F33615-68-C-1252 for Air Force Avionics Laboratory, Wright-Patterson Air Force Base, Ohio. (AD 869 208) p. 38.

62. Potter, E. A. and R. T. Compton, Jr., "Computer Simulation of Aircraft Tracker Problems in the Presence of Chaff," Report 3401-3, September 1975, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract F33615-73-C-1173 for Aeronautical Systems Division/PPMEB, Wright-Patterson Air Force Base, Ohio.

SUPPLEMENTARY

INFORMATION

Errata

AD-B013 005

Page 112 is not available.

DTIC-DDAC
14 Dec 84