(12)

(12)

CHEMICAL
RESEARCH,
DEVELOPMENT &
ENGINEERING
CENTER

CRDEC-TR-416

CHARACTERIZATION OF SPHERES WITH THE SUBMICRON PARTICLE ANALYZER: FEASIBILITY

Jerold R. Bottiger

RESEARCH DIRECTORATE

September 1992

U.S. ARMY
ARMAMENT
MUNITIONS
CHEMICAL COMMAND

Aberdeen Proving Ground, Maryland 21010-5423

Disclaimer

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorizing documents.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	1992 September	Final, 90 Oct - 91 Sep

4. TITLE AND SUBTITLE
Characterization of Spheres with the Submicron Particle Analyzer: Feasibility

5. FUNDING NUMBERS
PR-10162622A552

6. AUTHOR(S)
Bottiger, Jerold R.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
CDR, CRDEC, ATTN: SMCCR-RSP-B, APG, MD 21010-5423

8. PERFORMING ORGANIZATION REPORT NUMBER
CRDEC-TR-416

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

A method is described for inverting light scattering data to find the size parameter (x) and refractive index (n) of small dielectric spheres. The data considered is restricted to that obtainable with the U.S. Army Chemical Research, Development and Engineering Center's Submicron Particle Analyzer (i.e., ratios of intensities detected at a number of fixed directions about the scattering sphere). The inversion process works by comparing measured flux ratios with the same ratios previously calculated over a range of x,n values, and finding those x,n pairs for which measured and calculated ratios are consistently in agreement, to within the experimental uncertainty. Using numerical simulations of measurements and estimating the experimental error to be +/- 10%, we find that about 13 ratio measurements are needed to perform satisfactory inversions.

14. SUBJECT TERMS
Inversion
Light scattering
Aerosol characterization
Aerosol sizing

15. NUMBER OF PAGES
69

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Blank

PREFACE

The work described in this report was authorized under Project No. 10162622A552, Smoke and Obscurants. This work was started in October 1990 and completed in September 1991

The use of trade names or manufacturers' names in this report does not constitute an official endorsement of any commercial products. This report may not be cited for purposes of advertisement.

This report has been approved for release to the public.

Blank

CONTENTS

LIST OF FIGURES

CHARACTERIZATION OF SPHERES WITH THE SUBMICRON PARTICLE ANALYZER: FEASIBILITY

1. INTRODUCTION

The Submicron Particle Analyzer (SPA) is an instrument built by Wyatt Technology Corporation (Santa Barbara, CA) for the U.S. Army Chemical Research, Development and Engineering Center (CRDEC) that is used to study light scattering by aerosol particles.[1,2] The SPA comprises a spherical chamber in the center of which a dilute vertical stream of sampled aerosol particles traverses an intense horizontal laser beam, one particle at a time. Light scattered from each particle is intercepted and measured via 22 optical fibers that are distributed on the surface of the sphere and lead to 22 photomultiplier tubes and associated electronics in a separate instrument rack. The optical fibers, which are terminated on the chamber end with SELFOC gradient index lens, can be deployed among any of 72 ports on the sphere; the same nine port scattering angles are repeated along eight semi-great circles that are 45 degrees apart. The function of the SPA is to gather a set of light scattering data from each aerosol particle, from which physical characteristics of the particles (i.e., size and shape) may be inferred. The aim of our current research with this instrument is to work out the appropriate types of data to be included in the measured sets and to discover the manner in which those data sets may be manipulated to reveal desired particle characteristics.

Light scattering instruments built for sizing spherical particles typically measure scattered intensity in one or two directions and produce a "signal" proportional to either an intensity or a ratio of intensities. These instruments differ according to the scattering and acceptance angles chosen and the spectral nature of the illuminating radiation. In any case, a response curve is calculated that gives the signal level expected as a function of particle diameter; often, a calibration with a few spheres of known size is also required to match calculated and actual signals. There are several well-known problems with this approach. First, it is the nature of scattering by spheres that the response curves are not single valued -- that is, there are several sizes of spheres that can produce any given signal level. This effect may be mitigated, but never completely eliminated, by choices made in the instrument's design. Also, the response curve depends on the refractive index of the sphere; therefore, a different curve and calibration, if necessary, must be available for every different particle material. In practice, this is rarely done. Finally, these instruments provide no way to check whether each scattering particle is actually a sphere. The spherical response curve is simply applied to every particle that scatters light whether it is appropriate or not.

With the SPA and its multiplicity of light scattering channels, we may have an opportunity to design a measurement and analysis technique that will characterize spherical particles with a confidence never before attained. In this report, we will propose a direct inversion technique for processing SPA data, write the necessary computer codes to implement the technique, and then test the procedure using computer-generated, synthetic data to represent

experimental measurements that can be collected with the SPA. We wish to discover whether a useful characterization of spherical particles is feasible given the nature and accuracy of SPA light scattering data.

2. OUTLINE OF THE INVERSION METHOD

In the context of the present problem, an inversion method may be said to be successful when it produces estimates for the size and refractive index of a sphere such that the calculated light scattering properties of that particular sphere agree with the corresponding measured properties. A formal mathematical inverse solution to the MIE equations, to express size parameter (x) and refractive index (n) in terms of scattering intensities, is not possible. Instead, we shall discover acceptable values for x and n by considering, one pair at a time, all possible values of x and n, repeatedly asking if the spheres specified by the x,n pairs scatter light in agreement with the measurements, and noting the ones that do. The phrases "all possible values" and "in agreement with the measurements" require some elaboration.

2.1 N-X Plane.

In principle, there are an infinite number of possible values of x and n, so the range of those variables necessarily must be restricted. A large majority of dielectric materials, both liquids and solids, have refractive indices between 1.3 and 1.8 at visible wavelengths, and so that range is selected. (Absorbing spheres, in contrast with dielectric ones, must be characterized with a third parameter in addition to x and n, and are consequently beyond the scope of this report.) The size parameters considered will be restricted to the range $0<x\leq10$, though with less clear justification; inversion of scattering from small particles should be more reliable owing to their simpler scattering patterns; whereas, a maximum size parameter of 10 -- corresponding in blue light to a physical diameter of about 1.6 μm -- is large enough to include many standard particles available for experiments.

Therefore, as a starting point, we consider spheres represented by their coordinates on the x-n plane in the region $0<x\leq10$ and $1.3\leq n\leq1.8$. The region is divided into a number of much smaller rectangular areas (pixels) of dimensions Δx and Δn, with the aim of letting the sphere described by the central coordinates of each pixel stand for all the spheres represented within that pixel. This scheme succeeds if the relevant scattering properties of central spheres in adjacent pixels differ by less than the expected experimental uncertainty. Clearly, the pixel resolution must be at least as small as the accuracy with which we wish to recover x and n. However, the finer the resolution, the lengthier the inversion calculation, and there is no point in demanding a higher resolution than that which can be supported by the experimental accuracy. None of these factors are known a priori; to get started, we choose, rather arbitrarily, $\Delta x = .05$ and $\Delta n = .005$. This results in an array of 20,200 pixels stacked in 200 columns centered at x = 0.05, 0.10, ... 10.00, and along 101 rows centered at n = 1.300, 1.305, ... 1.800.

Although the pixels represent a 2-dimensional array (the x-n plane), it will be convenient for later purposes to name the pixels with a

single, rather than a double, index. The pixel in the lower left corner (x,n) = (0.05, 1.300) is pixel number 1; that in the lower right corner (10.00, 1.300) is number 200, etc., left to right and up the rows to pixel number 20,200 in the upper right corner (10.00, 1.800)

2.2 Selection of Scattering Properties.

The nature of the experimentally observable quantities, which will be compared to their calculated values for each sphere, must now be specified. As noted in Section 1 of this report, the SPA allows 22 intensities to be recorded for each particle. The laser beam intensity profile is Gaussian; since the exact path of particles through the beam is uncontrollable, the incident beam intensity for any particle is unknown, and the absolute values of the scattered intensities have little meaning. The ratios among the various scattered intensities for each particle are independent of the incident beam and can serve as the light scattering properties for inversion. Since all the detector SELFOC lenses have the same apertures and are equidistant from the scattering particle, we may refer to the observable quantities interchangeably as either intensity ratios or flux ratios.

To distinguish spherical from nonspherical particles, eight detectors, without polarizers, are placed in a ring at a scattering angle of $\theta = 55°$, and the incident beam is prepared in a right circularly polarized state. For spherical particles illuminated in this way, there can be no variation of light scattering with azimuth angle, ' ; uniformity of the eight detector signals confirms particle sphericity. Random experimental noise will preclude exact equality of the eight measurements even for perfect spheres. However, the standard deviation will be small and the average value, owing to the eight times sampling, provides an excellent reference figure for forming ratios. The average intensity measurement of these eight ring detectors will be the denominator in every intensity ratio.

Eight equivalent (scattering independent of ' for spheres) detector ports are available at each of the remaining eight scattering angles: $\theta = 10$, 40, 75, 90, 105, 125, 140, and 170°. At any port, the scattered light may be detected through either a linear polarizer in one of two orientations or without a polarizer at all. There are then 64 ports through which up to 24 different measurements may be made and ratioed to the $\theta = 55°$ measurement to form 24 scattering properties of the particle. Only 14 of the original 22 detectors remain available; therefore, a 14-member subset of the 24 possible ratios must be chosen. The final selection will be discussed in Section 3.4 of this report.

2.3 Pixel Acceptance Criterion.

Preliminary calculations and plots to assess the character of flux ratios over the x-n plane revealed a deficiency in the original plan that divided the plane into pixels with sides $\Delta x = .05$ and $\Delta n = .005$. There are regions that correspond to morphology-dependent resonances[3] of the sphere where the flux ratio grows rapidly to several times its value in neighboring pixels. Figure 1 shows the value of one flux ratio over a line in the x-n plane defined by $n = 1.75$ and $9.3 \leq x \leq 9.6$. This particular ratio happens to

be the intensity received through a horizontal polarizer at θ = 90° to (as always) the intensity received through no polarizer (unpolarized) at θ = 55°. The heavy vertical lines represent pixel boundaries in the x direction, and the dots correspond to the pixel centers, at (9.30,1.75), (9.35, 1.75), ..., (9.60, 1.75).

The amplitude of the ratio can vary by a factor of four or more across a single pixel of this size (Δx = .05) in the vicinity of a resonance; clearly, no single ratio value can adequately represent so large a domain. One obvious cure is to reduce the pixel size; however, judging from the steepest slope in Figure 1, it is estimated that the pixel dimension must be reduced to about Δx = .001, 50 times smaller than present, to ensure that the ratio varies by no more than 5% across the x-dimension of any pixel. A reduction in deltan by a similar factor would also be required, resulting in a network of some 50 million pixels over the area of the x-n plane now covered by about 20 thousand. The inefficiency in designing everywhere for the worst-case condition (up the slope of a resonance) would be colossal; in any event, the enormous demands on computer memory and central processing unit time prohibit this approach.

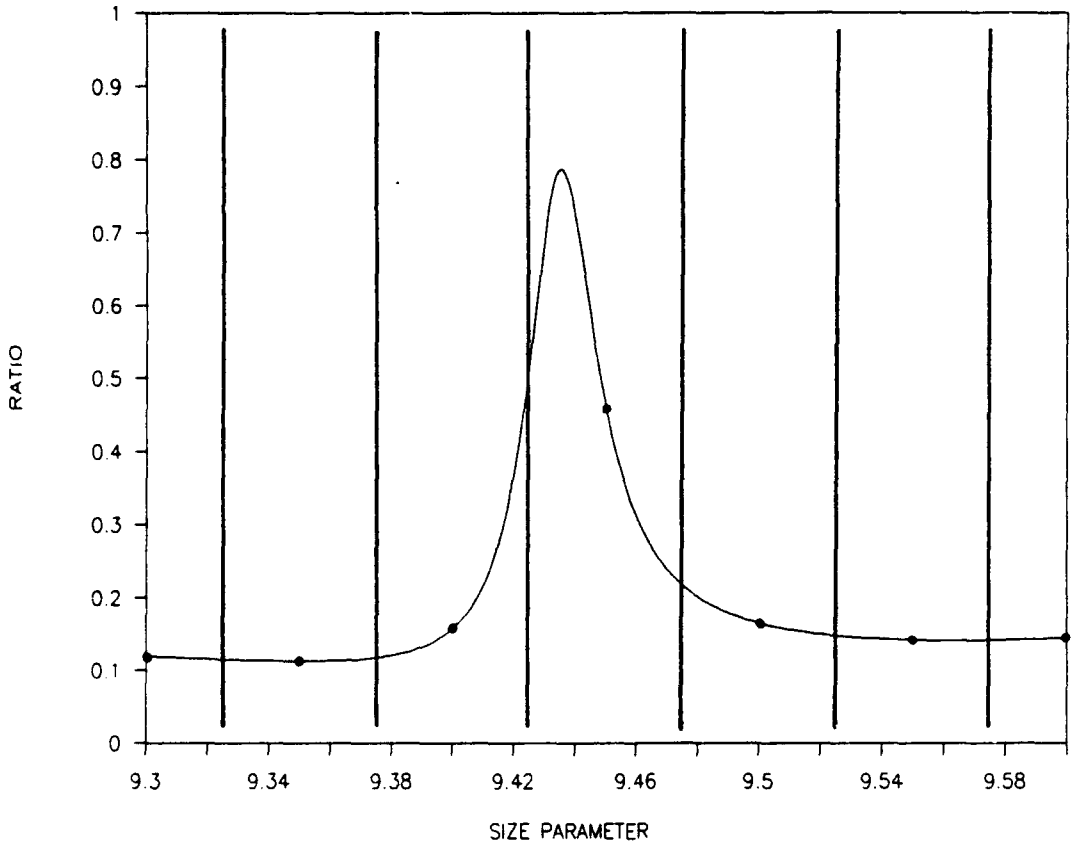


Figure 1. Detailed View of the H090 Flux Ratio Over a Segment of the X,N Plane

Instead, we leave the pixel dimensions unchanged and record for each pixel the minimum and maximum values of the flux ratios over all the pixel, not just the ratio values at the center of the pixel. This doubles the amount of data that must be made available to the inversion program, but that is all. This method will give correct, if not necessarily useful, results with pixels of any dimensions that may be chosen later on.

Each flux ratio determined with the SPA will have an experimental uncertainty associated with it, depending on the absolute magnitude of the measured intensities. Establishing rules for estimating the uncertainty will be an important task in the experimental phase of this inversion project; however, for the present feasibility study, we may simply specify uncertainties as needed. The inversion process may be summarized as follows. Consider a single pixel and a single flux ratio: if the physical sphere's size and refractive index are given by any point (x,n) within the pixel, then the true flux ratio must lie between the calculated minimum and maximum flux ratios. The experimentally determined flux ratio is really a range of ratios from (Measurement - Uncertainty) to (Measurement + Uncertainty). If the calculated and experimental ranges overlap, the pixel may contain the sphere's actual size and refractive index. If there is no overlap, the pixel cannot contain the sphere's parameters. After all the flux ratios have been checked in this manner over a pixel, that pixel may be reported as an inversion solution if agreement occurred for every flux ratio. If at least in principle even one pair of calculated and measured flux ratios failed to overlap, the pixel is to be rejected.

3. CALCULATION OF SCATTERED INTENSITIES

3.1 Scattered Intensity at a Point.

Consider a plane wave incident on an isolated single particle as shown in Figure 2 and ignore for the moment the indicated polarizer. The Stokes vector of the scattered light falling onto an infinitesimally small aperture in the direction θ and located a distance R from the particle is given by[4]

$$\begin{pmatrix} I_s \\ Q_s \\ U_s \\ V_s \end{pmatrix} = \frac{1}{k^2R^2} \begin{pmatrix} S_{11} & S_{12} & S_{13} & S_{14} \\ S_{21} & S_{22} & S_{23} & S_{24} \\ S_{31} & S_{32} & S_{33} & S_{34} \\ S_{41} & S_{42} & S_{43} & S_{44} \end{pmatrix} \begin{pmatrix} I_i \\ Q_i \\ U_i \\ V_i \end{pmatrix} \quad [\text{watt/cm}^2] \qquad (1)$$

where I_i, Q_i, U_i, and V_i are the Stokes parameters of the incident beam and $k = 2\pi/\lambda$. Each Stokes parameter has the dimension of intensity (e.g., watts/cm^2), whereas, the 16 scattering matrix elements S_{ij} are dimensionless and functions of θ.

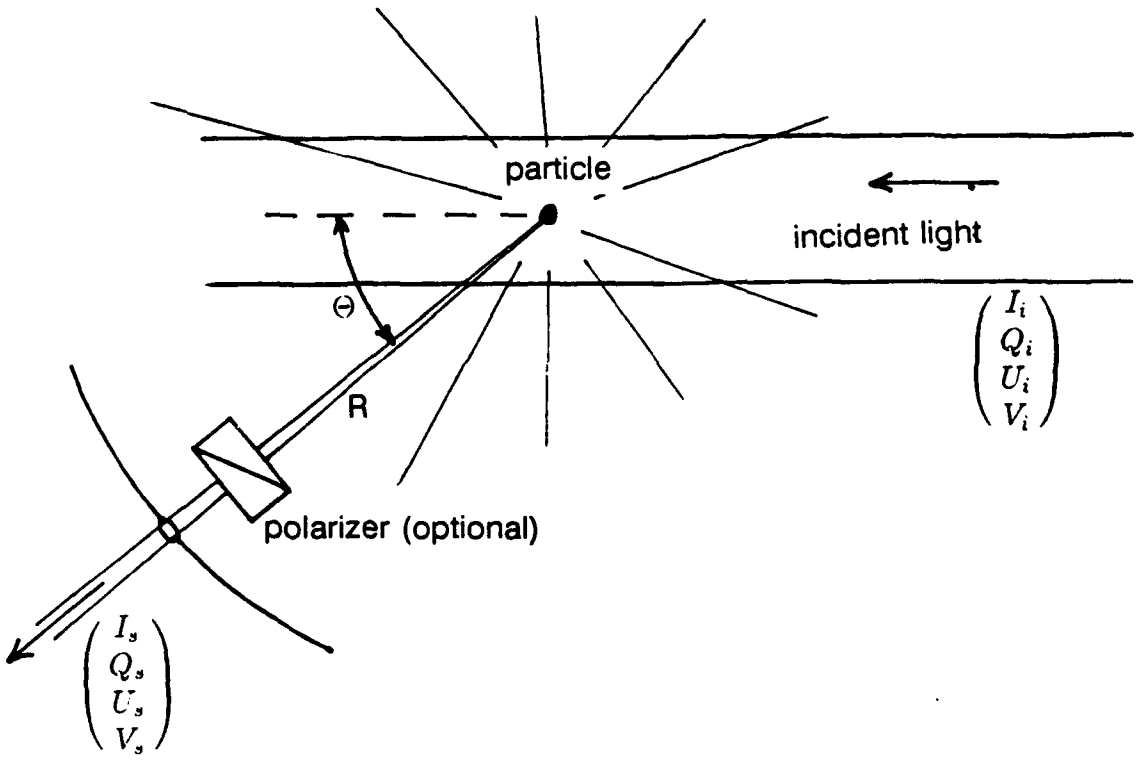


Figure 2. Light Scattering Geometry of the Submicron Particle Analyzer

In the particular case where the particle is spherically symmetric and the incident beam is right circularly polarized and of intensity I_o, equation 1 simplifies to the form

$$\begin{pmatrix} I_s \\ Q_s \\ U_s \\ V_s \end{pmatrix} = \frac{1}{k^2R^2} \begin{pmatrix} S_{11} & S_{12} & 0 & 0 \\ S_{12} & S_{11} & 0 & 0 \\ 0 & 0 & S_{33} & S_{34} \\ 0 & 0 & -S_{34} & S_{33} \end{pmatrix} I_0 \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \qquad (2)$$

The Stokes vector of the scattered light is then

$$\begin{pmatrix} I_s \\ Q_s \\ U_s \\ V_s \end{pmatrix} = \frac{I_0}{k^2 R^2} \begin{pmatrix} S_{11} \\ S_{12} \\ S_{34} \\ S_{33} \end{pmatrix} \tag{3}$$

and the intensity of the scattered light (i.e., the first Stokes parameter) falling upon the aperture is

$$I_s = \frac{I_0}{k^2 R^2} S_{11} \tag{4}$$

Let an ideal linear polarizer be placed in front of the aperture with its transmission axis oriented at an angle ρ with respect to the scattering plane -- the plane containing the aperture and the incident beam. When the polarizer transmission axis is parallel to the scattering plane, $\rho = 0°$. The angle grows positively as the polarizer is rotated counterclockwise, as seen by an observer looking inward through the aperture. With the polarizer inserted, the Stokes vector of scattered light reaching the aperture is given by

$$\begin{pmatrix} I_s \\ Q_s \\ U_s \\ V_s \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & \cos 2\rho & \sin 2\rho & 0 \\ \cos 2\rho & \cos^2 2\rho & \cos 2\rho \sin 2\rho & 0 \\ \sin 2\rho & \cos 2\rho \sin 2\rho & \sin^2 2\rho & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\times \frac{1}{k^2 R^2} \begin{pmatrix} S_{11} & S_{12} & 0 & 0 \\ S_{12} & S_{11} & 0 & 0 \\ 0 & 0 & S_{33} & S_{34} \\ 0 & 0 & -S_{34} & S_{33} \end{pmatrix} I_0 \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{5}$$

After multiplying the factors in equation 5, we find the intensity falling on the small aperture as

$$I_s = \frac{I_0}{2k^2R^2}(S_{11} + S_{12}\cos 2\rho + S_{34}\sin 2\rho) \tag{6}$$

Two polarizer orientations lead to simple results. When the polarizer is oriented horizontally ($\rho = 0°$), the intensity is proportional to $S_{11} + S_{12}$; if it is oriented diagonally ($\rho = 45°$), then the intensity is proportional to $S_{11} + S_{34}$. It is impossible to bring an explicit S_{33} dependence to the scattered light in this configuration; however, the scattering matrix elements for a sphere are related by[5]

$$S_{11}^2 = S_{12}^2 + S_{34}^2 + S_{33}^2 \tag{7}$$

so that, except for its sign, the S_{33} element is implicitly determined when the other three elements are determined.

Letting subscripts H, D, and U represent the cases of horizontal polarizer, diagonal polarizer, and no polarizer, respectively, the following equation summarizes the results of this section and gives the intensity of the scattered light reaching an infinitesimal aperture at scattering angle θ

$$\begin{aligned} IU(\theta) &= \frac{I_0}{2k^2R^2}\left[S_{11}(\theta) + S_{11}(\theta)\right] \\ IH(\theta) &= \frac{I_0}{2k^2R^2}\left[S_{11}(\theta) + S_{12}(\theta)\right] \\ ID(\theta) &= \frac{I_0}{2k^2R^2}\left[S_{11}(\theta) + S_{34}(\theta)\right] \end{aligned} \tag{8}$$

3.2 Correction for Finite Acceptance Angle.

If the intensity of light "I," perpendicularly incident on an aperture of area "A," is constant over its surface, then the flux through the aperture is simply IA. However, if the intensity varies with position on the aperture, an integration is required to find the total flux.

Figure 3 shows a circular aperture of radius r (representing the clear opening of one of the SELFOC lens collecting scattered light in the SPA) located a distance R away from the scatterer in the direction θ_o. The intensity of the scattered light reaching the aperture varies with θ (only), as is suggested by the figure's shading.

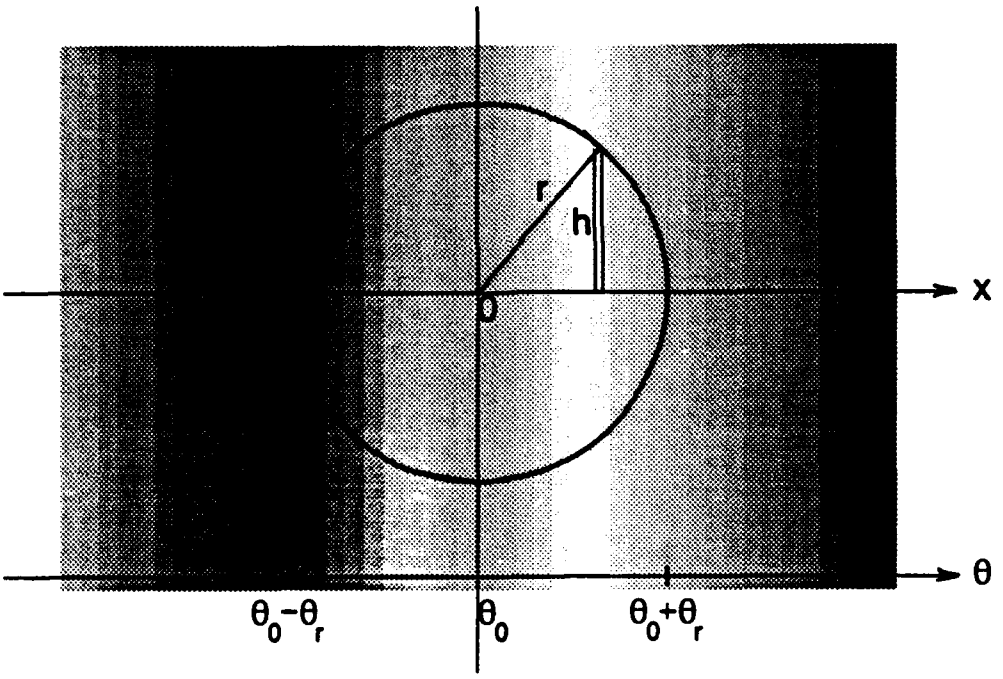


Figure 3. Scattered Light of Varying Intensity Falling on a SELFOC Fiber-Optic Lens

Since $x = R(\theta - \theta_0)$ and $dx = R\,d\theta$, the total flux through the circular aperture is given by

$$F = \int_{\theta_0-\theta_r}^{\theta_0+\theta_r} I(\theta)\, 2h(\theta)\, R\, d\theta \tag{9}$$

The height h satisfies

$$r^2 = x^2 + h^2 = R^2(\theta - \theta_0)^2 + h^2 \tag{10}$$

and so

$$F = \int_{\theta_0-\theta_r}^{\theta_0+\theta_r} 2R^2\, I(\theta) \sqrt{\theta_r{}^2 - (\theta - \theta_0)^2}\ \ d\theta \tag{11}$$

If the expressions for $I(\theta)$ (equation 8) are substituted into equation 11, the resulting integral cannot be evaluated analytically owing to the complexity of the S_{ij}. A tractable approximation to $I(\theta)$ can be derived by evaluating $I(\theta)$ at a set of discrete angles in the vicinity of the aperture and interpolating linearly between the calculated values. The SELFOC lenses have a clear diameter of 1.8 mm and are located 93 mm from the center of the SPA chamber; therefore, $\theta_r = .9/93 = .00968$ radians, or 0.55°. The computer program (described Section 3.3) can calculate the S_{ij} at intervals of one degree or greater, and the lenses are centered on integral degree values. Figure 4 shows the situation approximately to scale.

From Figure 4, we see that $I(\theta)$ to the left of θ_0 is given by the straight line whose equation is

$$I(\theta) = \frac{I(\theta_0) - I(\theta_0 - d)}{d}\Big[\theta - (\theta_0 - d)\Big] + I(\theta_0 - d) \tag{12}$$

while for $\theta \geq \theta_o$

$$I(\theta) = \frac{I(\theta_0 + d) - I(\theta_0)}{d}\left[\theta - (\theta_0)\right] + I(\theta_0) \tag{13}$$

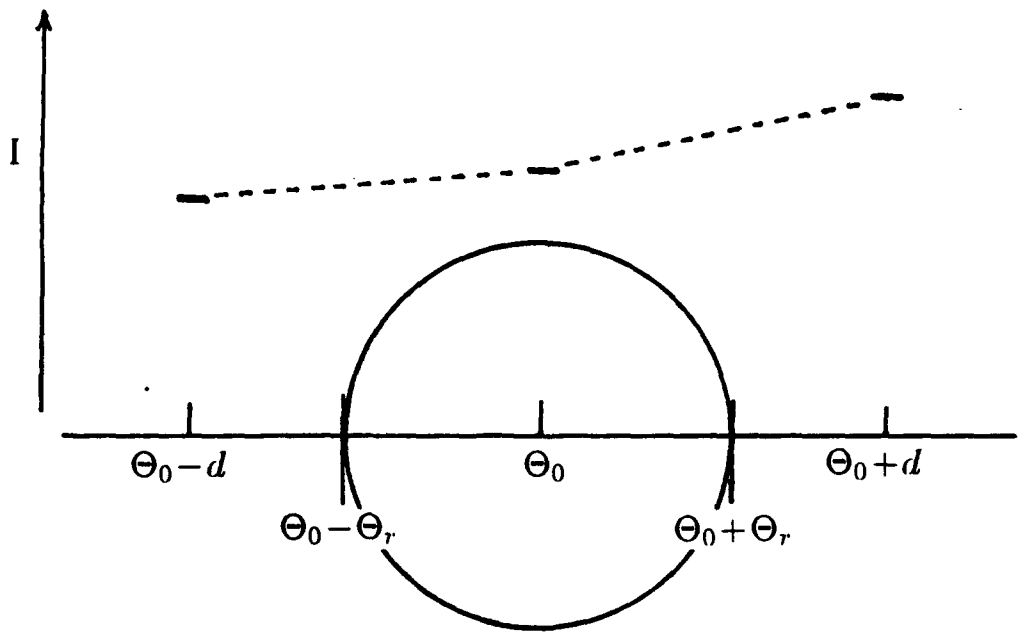


Figure 4. Calculation, at One Degree Intervals, of Light Scattering in the Vicinity of a SELFOC Lens

Substituting equations 12 and 13 into equation 11 gives the total flux through the aperture as

$$F = 2R^2 \int_{\theta_0-\theta_r}^{\theta_0} \sqrt{\theta_r^2-(\theta-\theta_0)^2}\left\{\frac{I(\theta_0)-I(\theta_0-d)}{d}\left[\theta-(\theta_0-d)\right]+I(\theta_0-d)\right\}d\theta$$

$$+ 2R^2 \int_{\theta_0}^{\theta_0+\theta_r} \sqrt{\theta_r^2-(\theta-\theta_0)^2}\left\{\frac{I(\theta_0+d)-I(\theta_0)}{d}\left[\theta-(\theta_0)\right]+I(\theta_0)\right\}d\theta \qquad (14)$$

Making a change of variable to $\phi = \theta - \theta_0$ simplifies the expression to

$$F = 2R^2\left\{\frac{I(\theta_0)-I(\theta_0-d)}{d}\int_{-\theta_r}^{0}\sqrt{\theta_r^2-\phi^2}\,(\phi+d)\,d\phi \;+\; I(\theta_0-d)\int_{-\theta_r}^{0}\sqrt{\theta_r^2-\phi^2}\,d\phi\right\}$$

$$+ 2R^2\left\{\frac{I(\theta_0+d)-I(\theta_0)}{d}\int_{0}^{\theta_r}\sqrt{\theta_r^2-\phi^2}\,\phi\,d\phi \;+\; I(\theta_0)\int_{0}^{\theta_r}\sqrt{\theta_r^2-\phi^2}\,d\phi\right\} \qquad (15)$$

The integrals in equation 15 are readily evaluated; after some algebra, the flux equation reduces to

$$F = \pi\theta_r^2 R^2\left[\frac{2\theta_r}{3\pi d}I(\theta_0-d) \;+\; \left(1-\frac{4\theta_r}{3\pi d}I(\theta_0)\right) \;+\; \frac{2\theta_r}{3\pi d}I(\theta_0+d)\right] \qquad (16)$$

This equation takes the simple form

$$F = \pi \theta_r^2 R^2 I(\theta_0) \tag{17}$$

as expected when θ_r goes to zero or when all three intensities are set equal to $I(\theta_0)$.

When the numerical values for θ_r and d are substituted (.55 and 1°), we find

$$F = \pi \theta_r^2 R^2 \left[.1175\, I(\theta_0 - 1°) + .7650\, I(\theta_0) + .1175\, I(\theta_0 + 1°) \right] \tag{18}$$

which gives the proportions for combining the intensity at the nominal detector angle with the intensities calculated one degree before and after that angle to correct for the finite acceptance angle of the SELFOC lenses. The correction makes little difference for spheres in the size range of interest because the intensity is quite a slow varying function of scattering angle for these small particles; however, for completeness and safety, the correction will be included in the intensity calculations that follow.

3.3 Mie Computer Programs.

The expression of the quantities S_{ij} (scattering matrix elements in equations 2 through 8) in terms of the size and refractive index of spheres is the subject of Mie theory, which is discussed in detail in the standard texts of light scattering.[4,6,7,8] The Mie equations are quite complex and are evaluated by computer to yield up numerical values for the scattering matrix elements.

The computer program adapted for this project was distributed several years ago by Peter Barber, now at Clarkson University (Pottsdam, NY), and is named, appropriately, MIE. MIE calculates various light scattering functions for homogeneous spheres and is based on the original program by Dave,[9] later expanded upon by Wiscombe.[10] MIE requests as input the size parameter and real and imaginary parts of the sphere's refractive index and the desired angular increment (1, 2, 3, 5, and 10° are allowed) at which calculations are to be performed. The output is written to a file named SPHERE.DAT, and comprises a few lines repeating the input data and displaying calculated efficiencies, followed by seven columns of data headed with the following titles: ANGLE, M SUB 2, M SUB 1, S SUB 21, D SUB 21, INTENSITY, and POLAR.

The scattering ANGLEs run from 0 to 180 °, inclusively, in steps of the specified increment. The next four columns relate to the scattering matrix elements as

$$\begin{aligned} \frac{1}{2}(\text{M SUB 2} + \text{M SUB 1}) &= S_{11} \\ \frac{1}{2}(\text{M SUB 2} - \text{M SUB 1}) &= S_{12} \\ \text{S SUB 21} &= S_{33} \\ \text{D SUB 21} &= S_{34} \end{aligned} \tag{19}$$

The sixth and seventh columns are useful combinations of previous ones, namely

$$\begin{aligned} \text{INTENSITY} &= \frac{1}{2}\,(\text{M SUB 2} + \text{M SUB 1}) \qquad (= S_{11}) \\ \text{POLAR} &= \frac{(\text{M SUB 2} - \text{M SUB 1})}{(\text{M SUB 2} + \text{M SUB 1})} \qquad \left(= \frac{S_{12}}{\text{INTENSITY}}\right) \end{aligned} \tag{20}$$

The heart of Barber's MIE program[3] is the subroutine SMIE, which does the actual evaluation of the quantities M SUB 2, etc. The input parameters, or functions derived from them, are passed to SMIE, which then returns a 3-dimensional array called ELTRMX(I,J,K) to the calling program.

The index I (= 1,4) specifies the type of scattering function in each element of the array, according to

$$\begin{aligned} 1 &\sim \text{M SUB 2} \\ 2 &\sim \text{M SUB 1} \\ 3 &\sim \text{S SUB 21} \\ 4 &\sim \text{D SUB 21} \end{aligned} \tag{21}$$

The indices J (= 1,(90/D)+1) and K (= 1,2), together, specify the scattering angle appropriate to each element of the array, given D, the increment between scattering angles. The relations between θ, D, J, and K are

$$\theta = \mathrm{D}(\mathrm{J}-1) \qquad \text{when } \mathrm{K}=1, \quad (0 \le \theta \le 90^\circ)$$

$$\text{and} \qquad \theta = 180^\circ - \mathrm{D}(\mathrm{J}-1) \qquad \text{when } \mathrm{K}=2, \quad (91^\circ \le \theta \le 180^\circ) \tag{22}$$

This subroutine SMIE is used, unaltered, in my program HILO.F for calculating the data sets of flux ratios.

3.4 Calculation of the Flux Ratio Data Sets: HILO.F.

All elements have now been assembled to write a program to calculate the minimum and maximum values of each flux ratio over each pixel in the x-n plane. Looking ahead to the results of the calculation, when a 3-D surface plot is made of any of these flux ratios over the x-n plane (Figures 5 and 6 of the main body of this report and Appendix B), a landscape is revealed of nearly parallel valleys and sharply rimmed ridges. There are no pits or peaks in the flux ratio values -- apparent spikes along the tops of ridges are artifacts of the plotting program -- therefore, in virtually every case, the extreme of the flux ratio for a pixel will occur on the perimeter of the pixel. Occasionally, when an absolute valley bottom or ridge crest is attained on the plane, the slope at that point is very gentle in at least one direction; therefore, there is no significant difference between that interior minimum (or maximum) value and the minimum (or maximum) around the pixel's perimeter.

So, instead of calculating flux ratios at a high density of x,n points over the pixels, it suffices to perform the calculations only on the perimeter around each pixel. More time is saved by noting that each interior pixel edge is common to two pixels and, in most cases, only two edges per pixel need be calculated. By avoiding calculations at points internal to the pixel, we save computation time by a factor of nearly NSAMPS/2, where NSAMPS is the number of sampling points along one edge of a pixel. In the program HILO.F, NSAMPS is set equal to 6 in the left half of the plane and equal to 20 in the more steeply corrugated right half of the plane.

The source code for HILO.F is listed in Appendix A and is reasonably well self documented; an overview of its operation follows. The main program HILO selects pixels in order from left to right, one row at a time, beginning with the bottom row (n = 1.300), and on each pixel selects which of the four edges to evaluate. HILO calls subroutine SIDE once for each edge and specifies NSAMPS. SIDE calculates the values of x and n at the

sampling sites along the edge, and at each sampling site calls the subroutine SMIE. SMIE returns (to SIDE) the array ELTRMX, which holds scattering functions as a function of scattering angle (every one degree) for the site. SIDE then calls subroutine UDHINT, which computes the three intensities, IH, ID, and IU, for each of the nine SPA scattering angles (correcting for detector acceptance angle) and writes them to one column of the array IO. When IO is finally filled, it contains 27 rows and NSAMPS columns of intensities.

The intensities in IO are converted by SIDE to flux ratios by dividing each element in every row by the corresponding element in the row representing the reference intensity, namely the intensity scattered at 55° through no polarizer. Another array, MINMAX, is filled with values taken from IO; the smallest of the NSAMPS values in the kth row of IO is put into the first column of the kth row of MINMAX and the largest value of the same row goes in the second column. At the end, MINMAX is returned to HILO and contains the minimum and maximum values of each flux ratio over whichever pixel edge HILO originally requested.

Back in HILO, the contents of MINMAX are entered in a much larger 3-D array called ACCUM, which accumulates this edge data for all of the 200 pixels in the row. HILO keeps track of which edge data can be copied from an earlier calculation and which must be computed freshly. When ACCUM is filled, the four edges of each pixel are compared and the very smallest and largest flux ratio values for each pixel are appended to the output files. HILO then moves up to the next row of pixels and the process begins again.

Each output file comprises 20,200 lines, one for each pixel and in pixel order. A single real number specifying the minimum (or maximum) value attained by a particular flux ratio on the corresponding pixel is on each line. The files are named by describing the numerator flux with four characters. The first letter, one of H, D, or U indicates that the light either passed through a horizontally oriented polarizer, a diagonally oriented polarizer, or no polarizer, respectively, while the last three digits indicate the scattering angle. The extensions ".min" and ".max" are added to specify files of minimum and maximum flux ratios. There are 54 data files in all -- minimums and maximums for each of 3 polarizations at 9 scattering angles -- but of course two files, U055.MIN and U055.MAX, contain only 1.0's.

HILO was compiled and run on a Stardent minicomputer (Ardent Computer Corporation, Dobbs Ferry, NY) and required just over 5 hr to complete. The data sets were transferred to a personal computer, where the graphing program "Surfer" was used to prepare plots for visualizing the flux ratios. Figures 5 and 6 show typical data sets; the surfaces of minimum (Figure 5) and maximum (Figure 6) values of the flux ratio U090 were plotted over the x-n plane in both surface and contour formats. The true surface of U090 lies on or above the surface of U090.MIN and on or below the surface of U090.MAX. The regularly spaced spikes along mountain crests, especially evident in Figure 5, are plotting artifacts, which result because the sampling and plotting mesh of pixels is too coarse to represent the true knife-edge ridges. Actually, there are no points of relative maxima of U090 over the plotted part of the x-n plane, and only two relative minima I can find --

shallow valley bottoms indicated by closed contour loops at about (x,n) = (6.6,1.5) and (8.3,1.4).

Plots of flux ratio maximum values, such as those in Figure 6, present a better picture of the true flux ratio than do plots of flux ratio minimum values. One can imagine that the upper plot of Figure 6 represents an opaque paint, covering - perhaps too thick at places - the flux ratio surface.

Surface plots of the maximum values of all the flux ratios are cataloged in Appendix B of this report. At scattering angles of 10 and 170°, there are few differences among the H, D, and U flux ratios (except for a factor of 2), which was expected. At 0 and 180°, the distinctions vanish altogether because $S_{12} = S_{34} = 0$ (equation 8). Therefore, we would deploy detectors to measure, at the most, one of the three polarizations at those two extreme angles. Polarization-dependent differences among the flux ratios at more central scattering angles are much more apparent.

Graphs in Appendix B of this report allow us to envision the nature of the light scattering flux ratios, but they offer little guidance in selecting the optimum ratios to measure with the 14 available detectors. The problem is that contour lines for all the flux ratios follow the same patterns, which lay in similar directions along lines of n,x = constant. There are no data sets whose contours run perpendicular to the others, which, because of their orthogonality, would be especially important to include among the 14 measurements. Instead, we settled on data sets according to the following experimentally pragmatic criteria. Too much background light is picked up by detectors at 10 and 170°; until that can be corrected by redesigning the scattering chamber, no data will be collected at those angles. Generally, more light is available without a polarizer than through one; because higher intensity implies better signal to noise ratio, we take all the remaining unpolarized data sets (e.g., U040, U075, U090, U105, U125, and U140). At most scattering angles, the scattering through a horizontal polarizer seems to differ more from the unpolarized scattering than the scattering through a diagonal polarizer; therefore, we collect horizontal data sets at the same six scattering angles. This would leave room for two diagonally polarized data sets, which we take at 40 and 90°.

4. TESTING THE INVERSION METHOD

4.1 Program INVERT.

The FORTRAN program "INVERT" was written to explore and test the inversion procedure. INVERT first reads in a number of files, which includes the computed min/max values over the x-n plane for the 14 selected flux ratios, a row of experimental detector calibration coefficients (used in this study to apply controlled errors to the synthetic input data), and the synthetic input data [an N by 24 array of numbers (EXPDAT??.TST) generated in a separate program (DATGEN) and simulating SPA measurements on a run of N particles]. When inverting real data, the experimental uncertainty associated with each flux ratio measurement will be individually determined, based on the

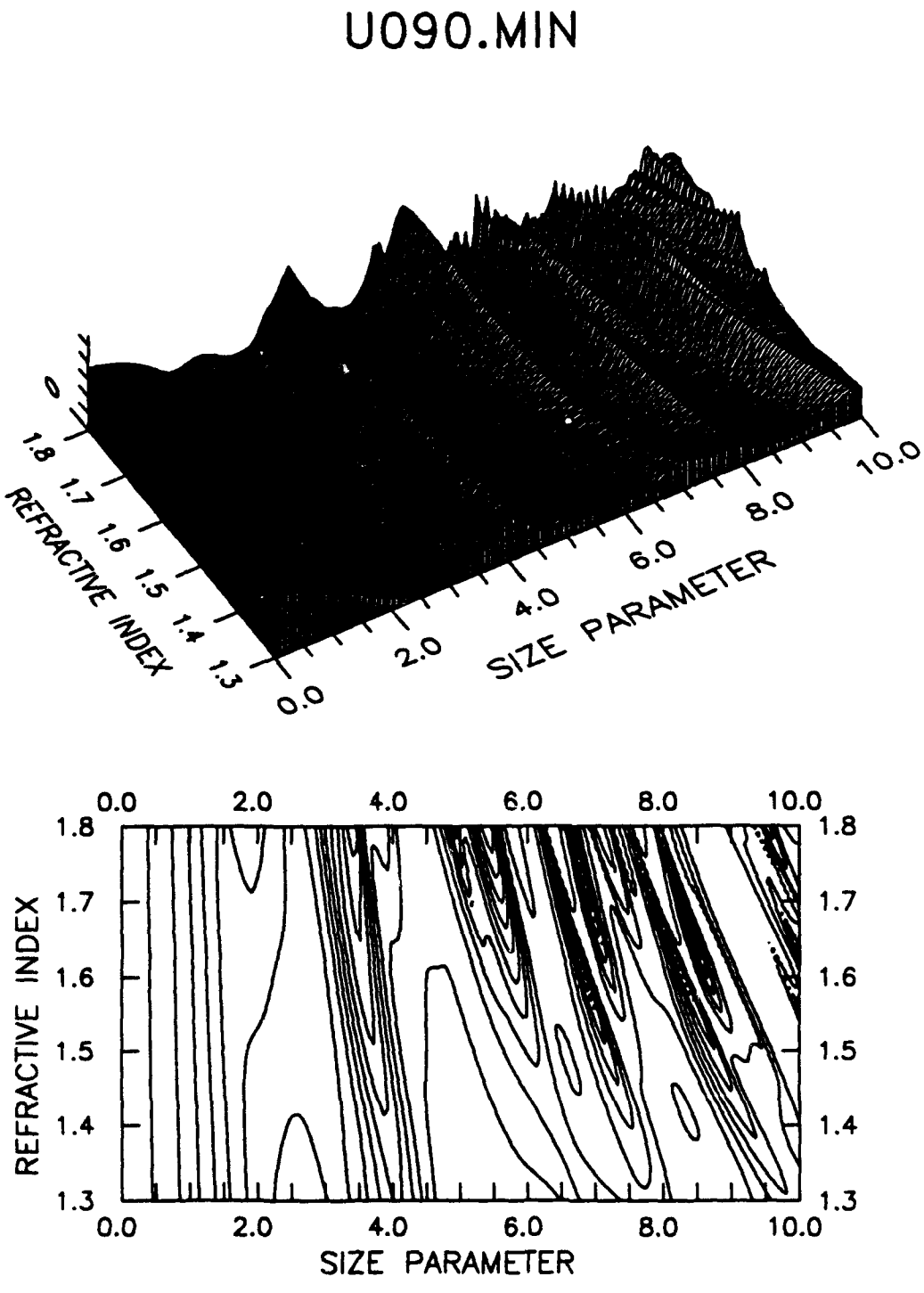


Figure 5. Surface of Pixel Minimum Values of the Flux Ratio U090 Over the X,N Plane

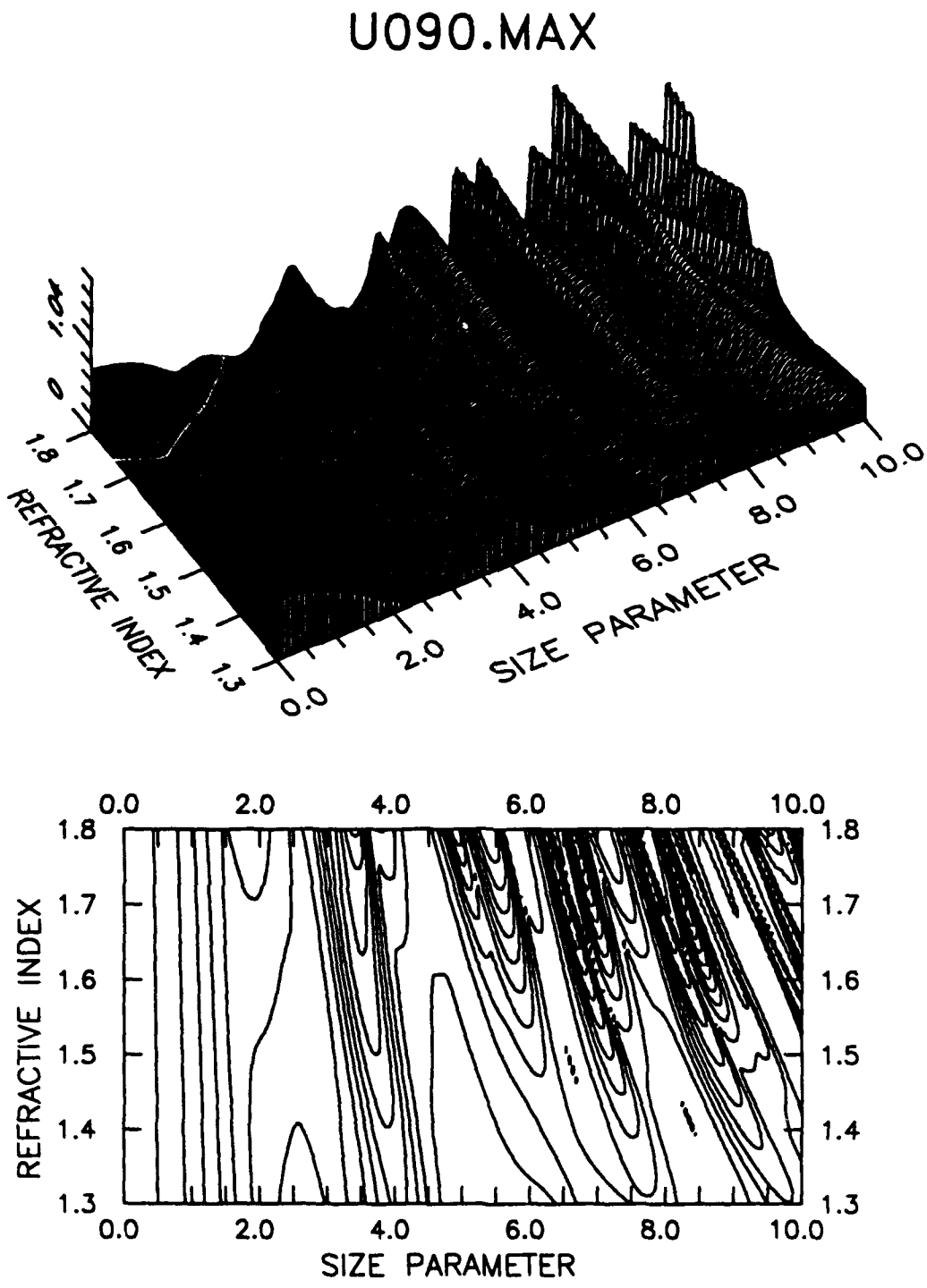


Figure 6. Surface of Pixel Maximum Values of the Flux Ratio U090 Over the X,N Plane

measured absolute scattering intensity. However, for this feasibility study, we have assigned a number of constant uncertainties to the flux ratios to observe their effect.

Taking one particle at a time (i.e., one row of EXPDAT??.TST), INVERT computes the 55°-ring average intensity and the 14 flux ratios, and then expands the 14 flux ratios into measured flux ratio ranges (nominal value plus and minus the assigned uncertainty). On each pixel, INVERT must check for agreement (overlap) between the particle's "measured" range of ratio values and the previously calculated range of ratio values for each of the 14 flux ratios. This is accomplished with an algorithm (described in the following paragraph) that is much faster than literally asking 20,200 times for each of 14 flux ratios if an overlap occurs.

The 28 files of minimum and maximum flux ratios read in by INVERT are not exactly the files outputted by HILO -- U040.MIN and others. HILO files are first acted on by the program ORDERALL, which sorts each file in ascending order of flux ratio and simultaneously writes for each file an ancillary integer file that relates the original pixel order to the newly sorted order. The file U040.MIN, written by HILO, is renamed ORD_U040.MIN when its data are sorted, and the corresponding integer file is named NDX_U040.MIN; other files are analogously named. The files ORD*.* and NDX*.* (a total of 56) are read by INVERT during its initialization.

The method of determining if overlap occurs between a pixel's calculated range and the experimental range for a flux ratio and the motive for sorting the data files into ascending order are illustrated in Figure 7. In the upper half of Figure 7, the short horizontal bars represent the calculated minimum values of one of the 14 flux ratios; the shaded patches above the bars are reminders that the calculated flux ratio range is at or above the minimum value. There are 20,200 short horizontal bars, but only a few are shown in Figure 7. The pixel numbers, written above the horizontal axis, indicate which pixels on the x-n plane the horizontal bars correspond to (corresponding ORD_ and NDX_ files list, in order from left to right, the indicated flux ratio values and pixel numbers, respectively). The lower half of Figure 7 is analogous to the upper half but represents the maximum calculated values of the same flux ratio.

The measured value of the same flux ratio for one particular particle is indicated in Figure 7 by a horizontal band whose top edge has the flux ratio value (Measurement + Uncertainty) and whose bottom edge is at (Measurement - Uncertainty). This horizontal band is the same in both the minima and maxima halves of the figure. Notice that the bars are calculated values and fixed in place for all time; a different horizontal band is superimposed for each measured particle. Which pixels have an overlap between the measurement band and the calculated ranges? Both plots divide naturally into three regions of answers. In the upper plot, the pixels from the leftmost one out to the marker, A, may or may not overlap, depending on the pixels' maximum values (information not available is in the upper plot). Between markers A and B, the calculated ranges must overlap with the measured range because the minima are embedded in the measured range. Beyond marker B,

no pixels may overlap because the minimum calculated value exceeds the greatest value allowed by measurement.

Similarly, in the lower part of the Figure 7 where calculated maxima are plotted, pixels from No. 1 to marker A cannot overlap, pixels from A to B must overlap, and pixels to the right of marker B might overlap with the measurement band, depending on how low the calculated minima are. A bit of reflection shows that if a pixel is in the "cannot overlap" region of one plot, it must be in the "might overlap" region of the other. Therefore, the aggregate of all the "cannot overlap" pixels from both plots includes, exactly once, every pixel that does not overlap with the measured flux ratio range. The remaining pixels then do overlap (i.e., agree with the measured data for this particular flux ratio). So to find the pixels that do not overlap, we must find point B of the upper plot and point A of the lower plot. More precisely, we must search the file of ordered minimum values to find the number of the first entry that is larger than (Measurement + Uncertainty). For example, if the 12,313th entry were the first entry to exceed (Measurement + Uncertainty), then lines 12,313 through 20,200 of the corresponding NDX_ file would contain the pixel numbers of pixels, which do not agree with the data. We then search the file of ordered maximum calculated values to find the last entry that is less than (Measurement - Uncertainty) -- perhaps entry number 4002. Then, lines 1 through 4,002 of its corresponding NDX_ file contain all the remaining pixels that do not satisfy the measurement for this particular flux ratio. The appropriate ranges of the two NDX_ files can be copied to an array that keeps track of how many times each pixel fails to agree with flux ratio measurements for each particle.

The reason for doing all this is that finding the points A and B is computationally very fast; a simple sequence of binary decisions homes right in on them. For example, to find marker B in the array of minima values, we start at entry No. 10,100 (the midpoint between 1 and 20,200) and ask if the entry value is greater than or less than the target value of (Measurement + Uncertainty). Everything to the left or right of 10,100 (depending on the answer) can be eliminated. We then go to the midpoint of the surviving range and again compare the value found there to the target value, which results in eliminating half that range. After only 15 repetitions (2**15 = 32768 > 20200), we find the desired point in the array. Similarly, 15 comparisons between (Measurement - Uncertainty) and entries in the file of maxima values find the critical point there, point A, and, as a result, the remainder of the nonoverlapping pixels. Thirty yes/no questions asked on the sets of sorted data provide exactly the same information as 20,200 pairs of yes/no questions asked on the original data in a brute force manner.

A separate 20,200 element array accumulates for each pixel the number of times (from 0 to 14) that the calculated and measured flux ratios failed to overlap for each particle. Ideally, only pixels with a score of zero (never failed to overlap) should be counted as solutions for the particle in question. However, because actual measurements of flux ratios may occasionally be in error by more than our best estimate of the experimental uncertainty, it may be desirable to admit solutions that do not necessarily satisfy all 14 of the available measurements. In this investigation, we wanted to see how the number of false returns grew as we pared the number of

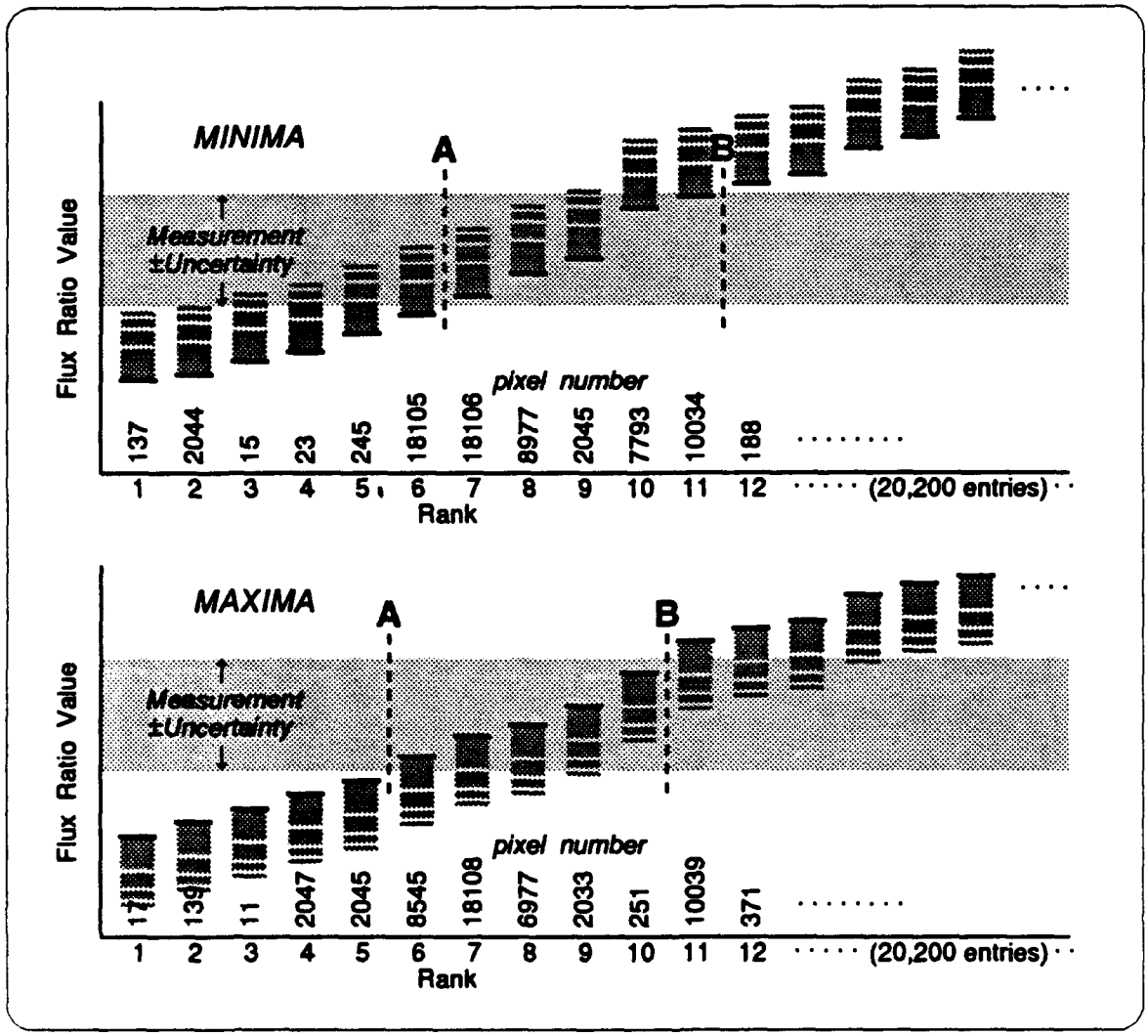


Figure 7. Comparison, for One Flux Ratio, of Calculated (Individual Bars) and Measured (Horizontal Band) Ratio Values

ratios with which agreement was required, and how the domain of solutions varied with different levels of experimental uncertainty. A few of the many tests done are shown in Figures 8-15.

Figures 8-15 are drawn by the page formatting program "PageGarden," by Bloc Publishing Corporation (Coral Gables, FL). The actual output file written by INVERT is a set of statements that instruct PageGarden where to draw pixels in the x-n plane and how dark to shade them, which depends on the number of overlaps. Simple changes to INVERT can alter the information related by the pixel print density.

4.2 Results of the Inversion Trials.

Four pairs of x,n coordinates (indicated by crosses in Figure 8) were selected and used to calculate the four rows of scattering measurements that would be recorded by the SPA instrument sampling the corresponding four spheres. These data were then entered to INVERT and the uncertainty level was set to +/- 0.3%. In Figure 8, the solidly shaded pixels are those on which flux ratios formed from the entered as "measured" data agree with previously calculated minimum and maximum flux ratios for all 14 cases. Partially shaded pixels agreed with 12 or 13 of the flux ratio "measurements," and open (nonshaded) pixels agreed with 10 or 11 of the flux ratio "measurements." An uncertainty of only 0.3% is completely unrealistic. Figure 8 only confirms that INVERT returns the correct solution when given essentially perfect data. However, notice that some spurious returns are already appearing in the upper right of the Figure 8.

Figure 9 shows the return when the stated experimental uncertainty is increased to +/- 3.0%. We believe 3% is about the upper limit for the SPA accuracy; under the most favorable conditions, that accuracy might be approached by one or two of the detectors. There are more returns further from the true solutions, but still the pattern is quite compact.

An accuracy of +/- 10% is more typical of the expected SPA performance. This is the uncertainty level assumed in Figure 10, which shows results for the same four spheres. We still see returns that are tightly clustered, except when as few as 10 agreements (overlaps) are accepted.

At +/- 30% uncertainty (Figure 11), the number of returns with only 10-11 agreements is becoming quite large. Even worse than the high number of these returns is the way they are distributed in disconnected patches all over the x-n plane; there is no hint of where the right answer might lie. The pattern looks much better for 12-13 agreements and is quite good for 14 agreements. We expect 30% to be near the lower limit of SPA measurement accuracy.

We have looked at many plots such as those in Figures 8-11, including plots in which the input data was corrupted with random errors, although always within the limits set by the assumed experimental uncertainty. There is surprisingly little difference if the data is actually distorted or not; the nature of the returns is pretty much completely established by the level of experimental uncertainty allowed. Taking 10% as an average

uncertainty value for the SPA, we concluded that requiring 13 or more agreements of a pixel to return it as a possible inversion solution should produce useful results.

Figures 12 through 15 show the inversion result for 12 spheres, with uncertainty levels of 0.3, 3.0, 10.0, and 30.0% assumed in the measurements. A pixel is shaded if it agrees with at least 13 of the 14 measurements. The outcome is encouraging. We see mostly compact connected patches of returns whose size parameter spread is about 0.3, corresponding to roughly 0.05 μm for blue light. The refractive index spread is not so useful (about 0.1), but the product nx is very accurately determined for each particle.

5. CONCLUSION

We conclude that it is feasible to characterize small dielectric spheres with data measured by the Submicron Particle Analyzer (SPA). The data sets available are not as independent as one could wish for; each additional measurement contributes relatively little to the process of winnowing away unsuitable (x,n) pairs so that about a dozen measurements are required. Although the method is inefficient, it does apparently work. Matters could be improved by including information not directly related to the flux ratios, such as the number of relative minima in the angular scattering from 0 to 180°. These measurements are suggested by Quist and Wyatt in their paper,[11] which describes an inversion method very similar to this one; however, the SPA does not currently support any types of measurements other than flux ratios.

In the future, we will undertake to demonstrate the inversion method using actual experimental data taken with the SPA. The first step must be an accurate assessment of the instrument's error characteristics as functions of received intensity. If we are successful in the inversion of experimental data, we will attempt to extend the method to spheres of larger size and/or made of absorbing materials. At the same time, we believe improvements can be made to the inversion code itself. It may be possible to replace each of the ordered minimum and maximum calculated flux ratio files with a fairly simple equation that can be solved for the critical points in Figure 7, thereby greatly reducing the size of computer random access memory required to run the inversion program.

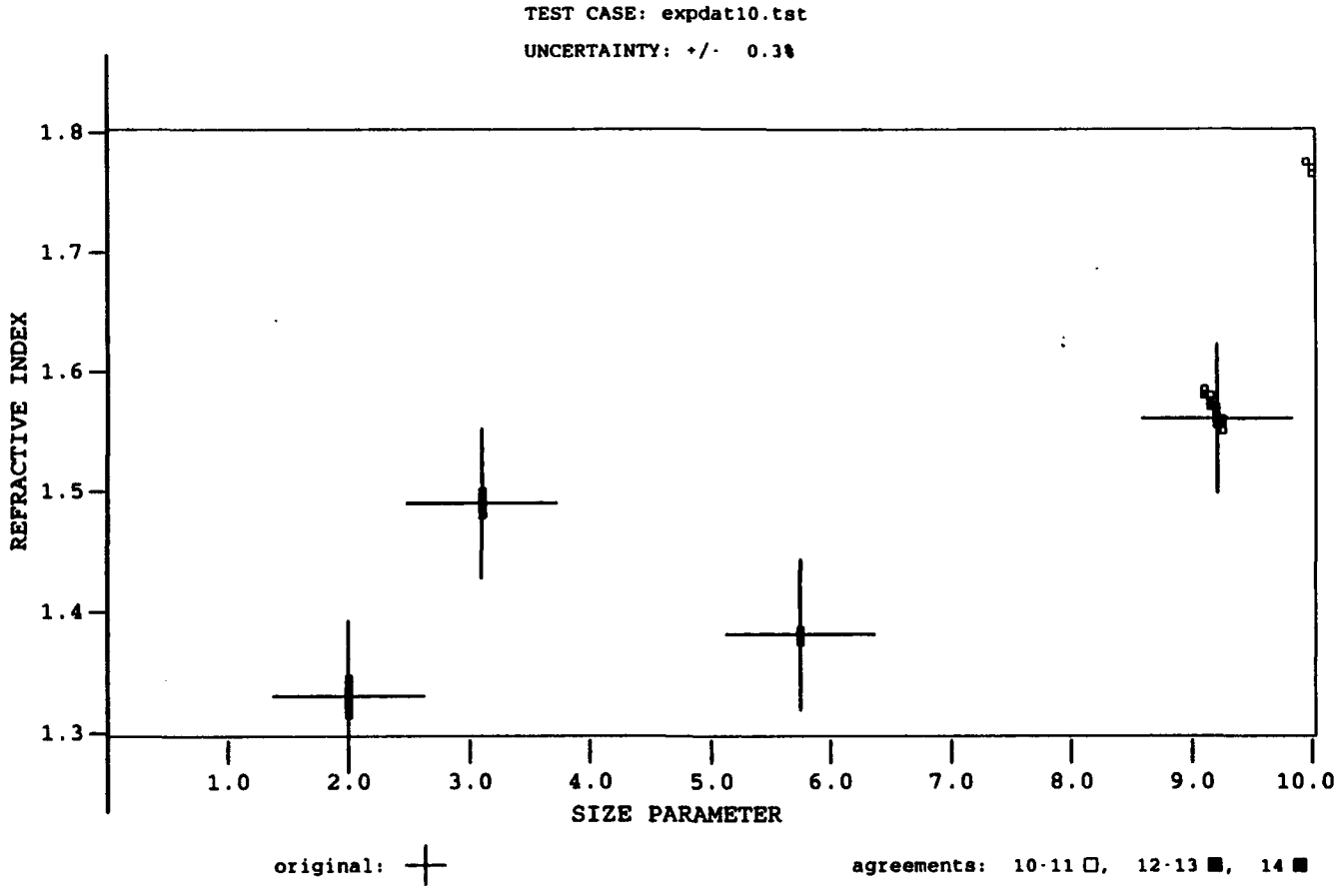


Figure 8. Inversion Results for Four Spheres, Assuming =/- 0.3% Data Uncertainty

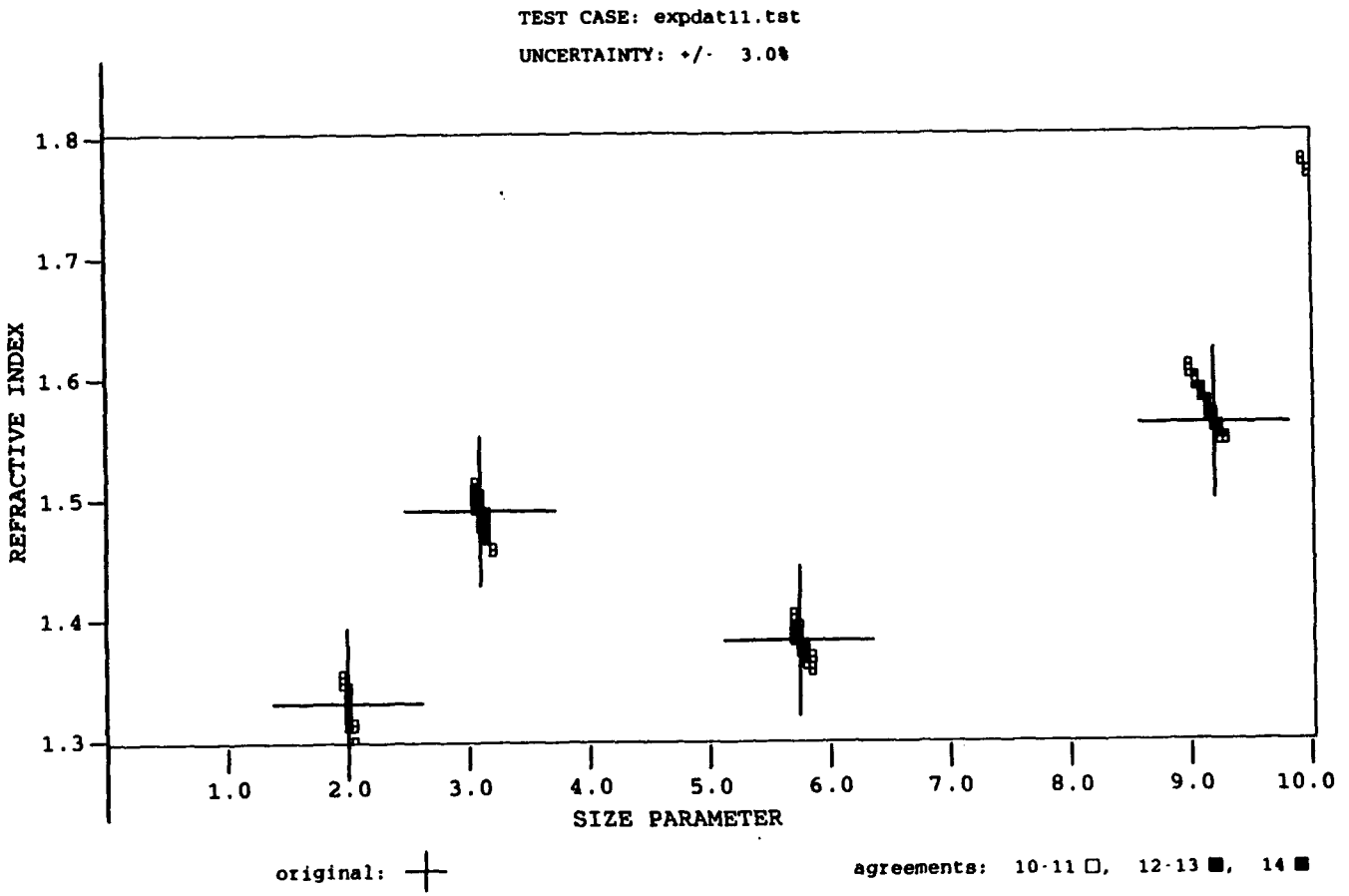


Figure 9. Inversion Results for Four Spheres, Assuming =/- 3.0% Data Uncertainty

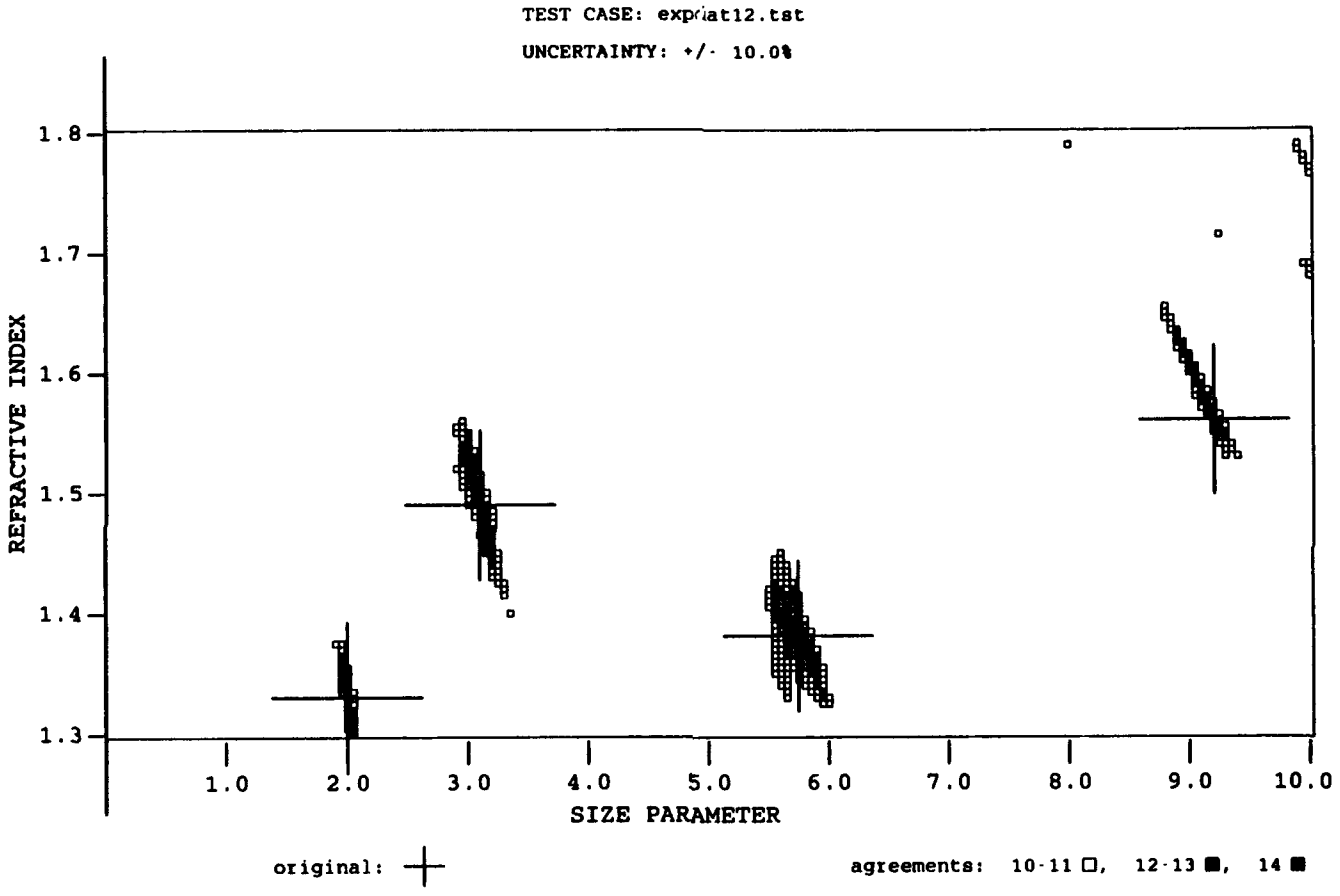


Figure 10. Inversion Results for Four Spheres, Assuming =/- 10.0% Data Uncertainty

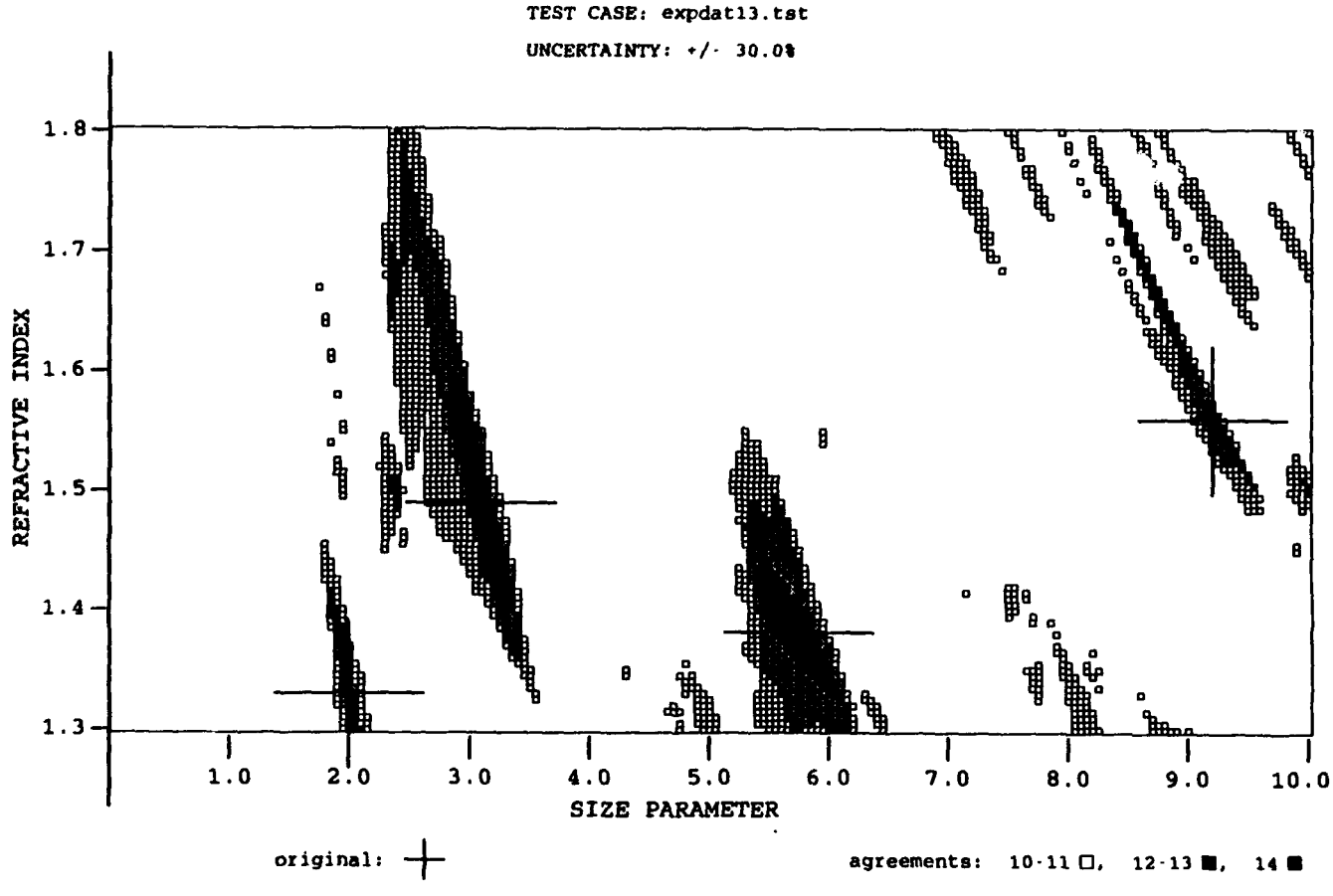


Figure 11. Inversion Results for Twelve Spheres, Assuming =/- 30.0% Data Uncertainty

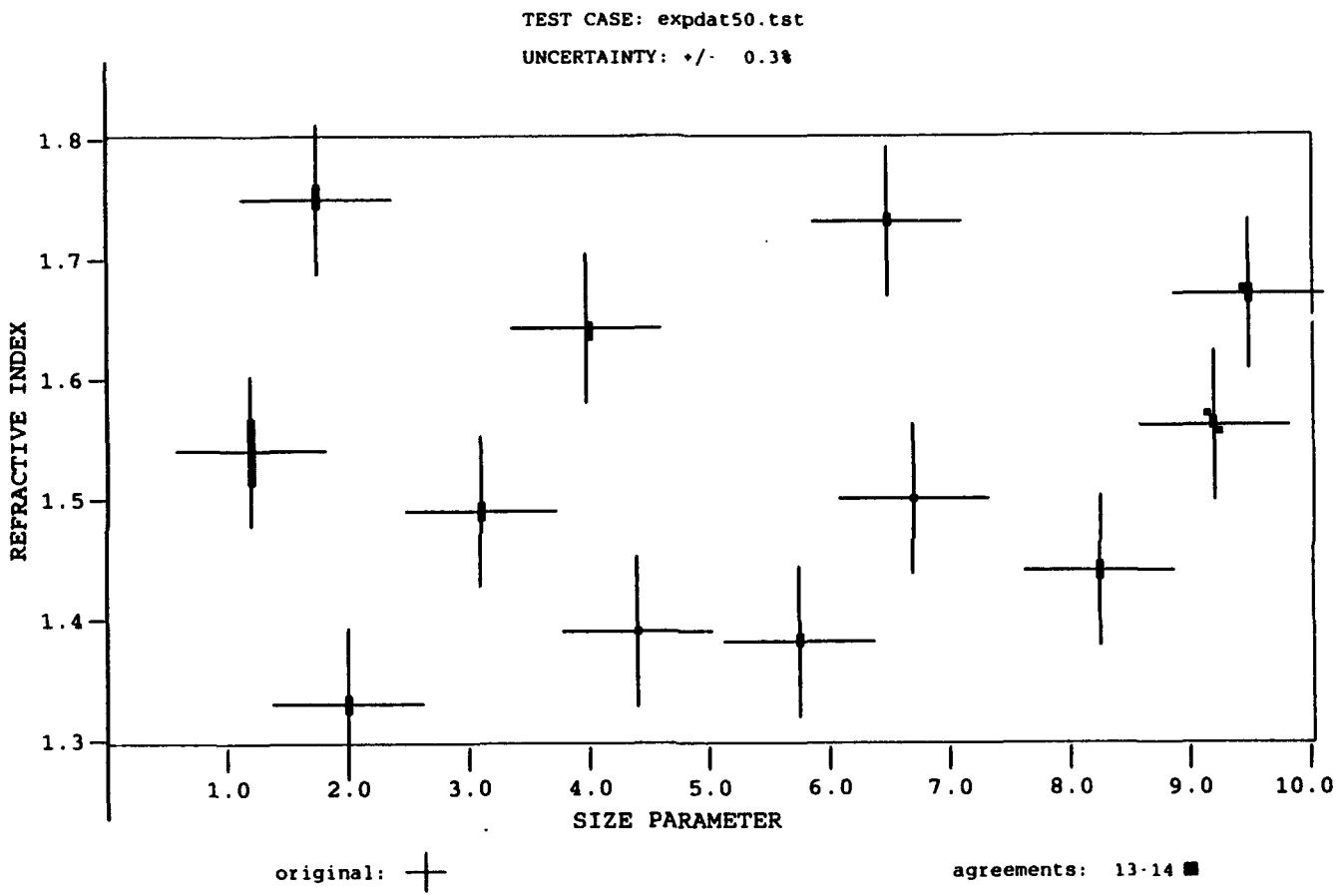


Figure 12. Inversion Results for Twelve Spheres, Assuming =/- 0.3% Data Uncertainty

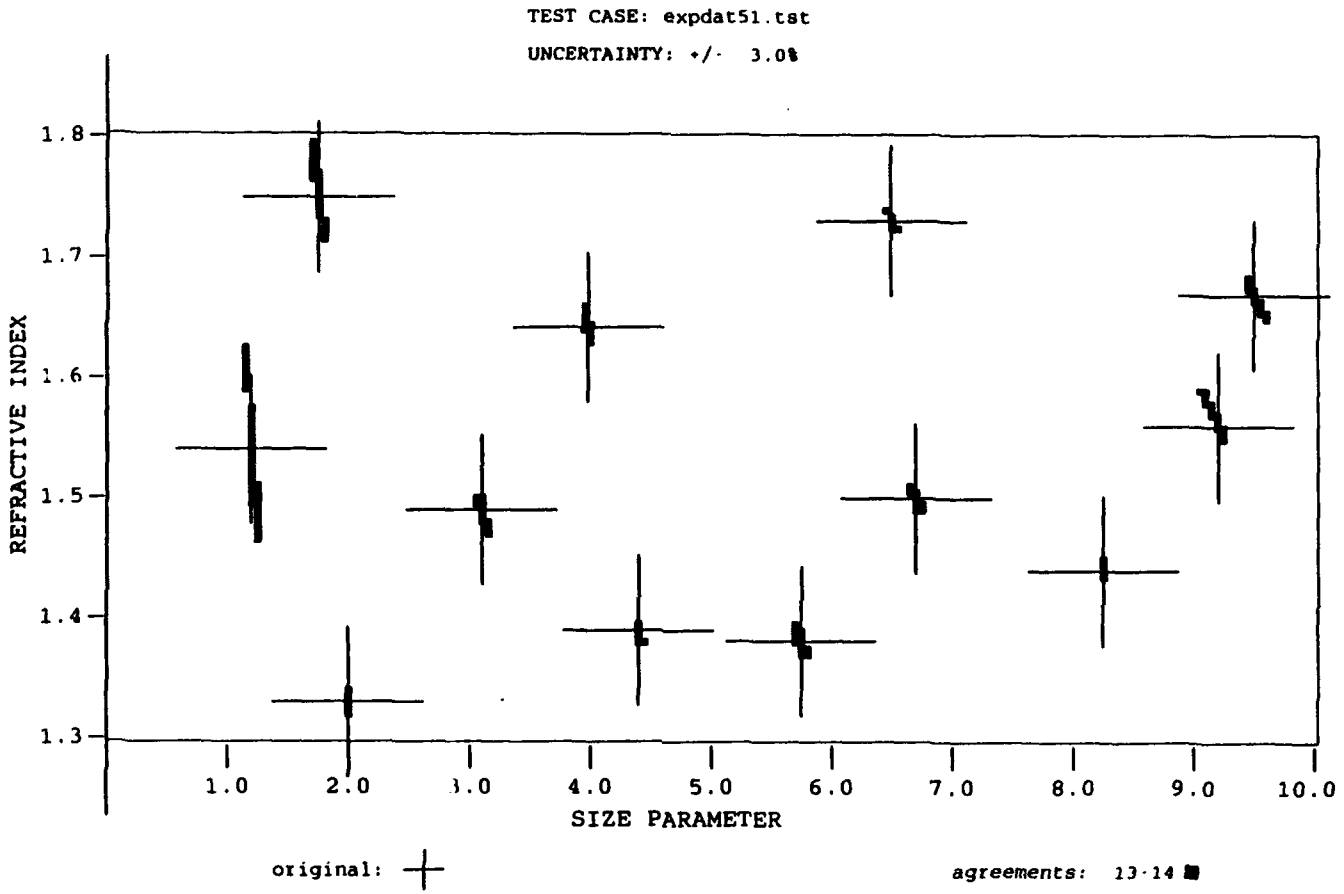


Figure 13. Inversion Results for Twelve Spheres, Assuming =/- 3.0% Data Uncertainty

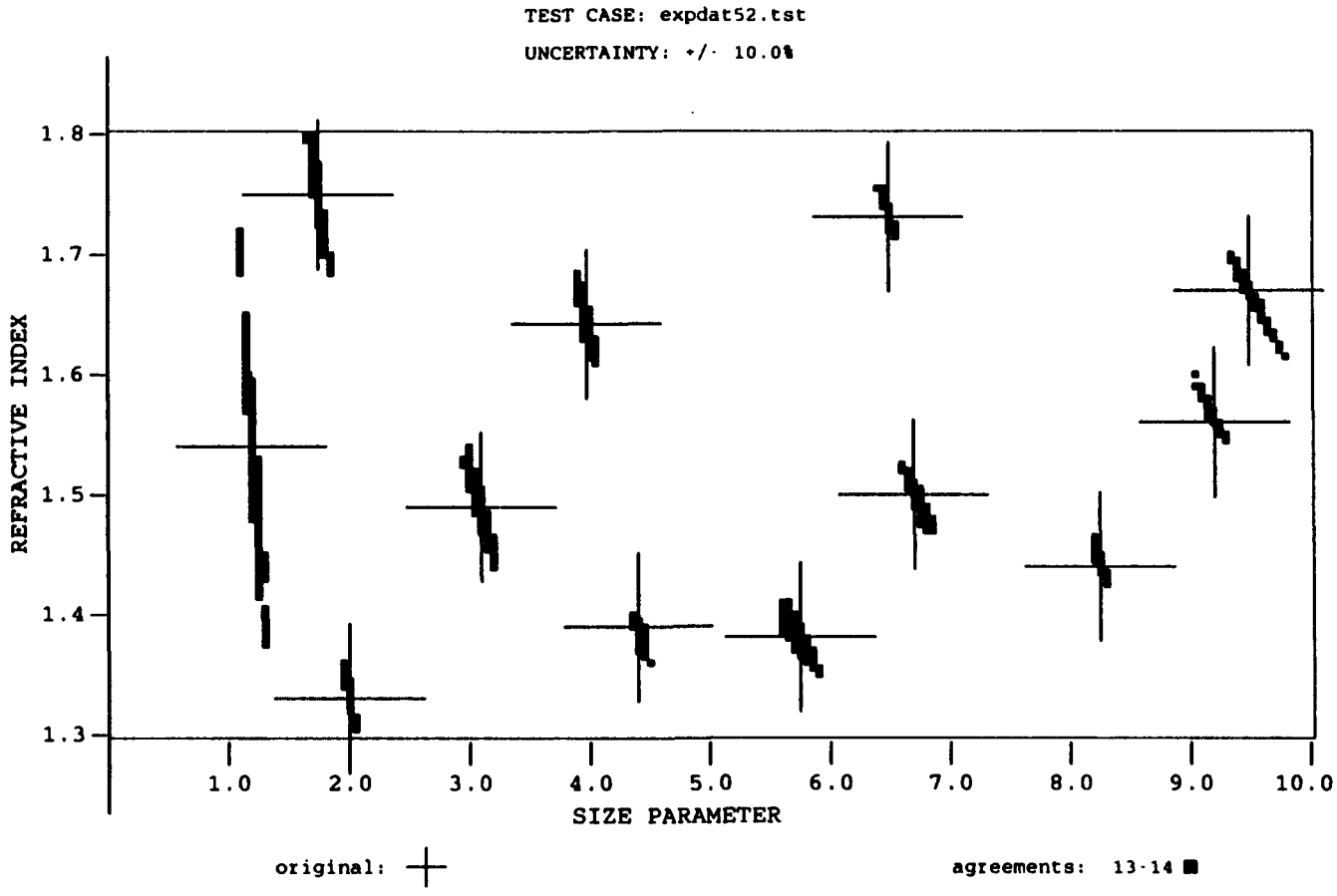


Figure 14. Inversion Results for Twelve Spheres, Assuming =/- 10.0% Data Uncertainty

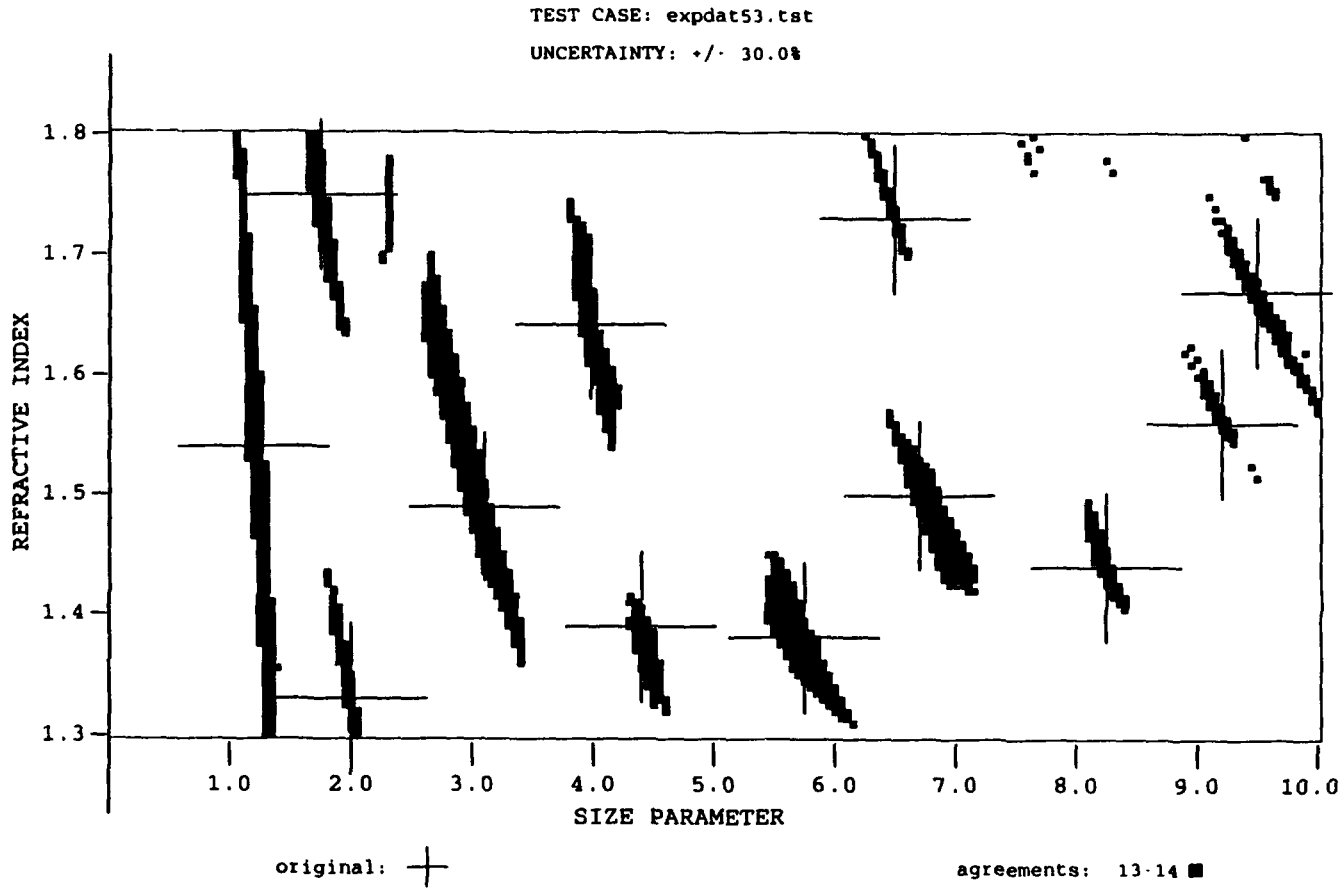


Figure 15. *Inversion Results for Twelve Spheres, Assuming =/- 30.0% Data Uncertainty*

LITERATURE CITED

1. Wyatt, P.J., and Jackson, C., Submicron Particle Analyzer, CRDEC-CR-067, U.S. Army Chemical Research, Development and Engineering Center, Aberdeen Proving Ground, MD, April 1990, UNCLASSIFIED Report (AD A223 558).

2. Wyatt, P.J. et al., "Aerosol Particle Analyzer," Appl. Optic. Vol. 27, pp 217-221 (1988).

3. Barber, P.W., Owen, J.F., and Chang, R.K. "Resonant Scatter for Characterization of Aissymmetric Dielectric Objects," IEEE Trans Antennas Propag. AP-30, pp 168-172 (1982).

4. Bohren, C.F., and Huffman, D.R., Absorption and Scattering of Light by Small Particles, John Wiley and Sons, Incorporated, New York, NY, 1983.

5. Fry, E.S., and Kattawar, G.W., "Relationships Between Elements of the Stokes Matrix," Appl. Opt. Vol. 20, pp 2811-2814 (1981).

6. Van de Hulst, H.C., Light Scattering by Small Particles, John Wiley and Sons, Incorporated, New York, NY, 1957.

7. Stratton, J., Electromagnetic Theory, McGraw-Hill, New York, NY, 1941.

8. Kerker, M., The Scattering of Light and Other Electromagnetic Radiation, Academic Press, Incorporated, New York, NY, 1969.

9. Dave, J.V., Subroutines for Computing the Parameters of the Electromagnetic Radiation Scattered by a Sphere, Report 320-3237, IBM Scientific Center, Palo Alto, CA, 1968, UNCLASSIFIED Report.

10. Wiscombe, W.J., "Improved MIE Scattering Algorithms," Appl. Opt. Vol. 19, pp 1505-1509 (1980).

11. Quist, Q.M., and Wyatt, P.J., "Empirical Solution to the Inverse-Scattering Problem by the Optical Strip-Map Technique," J. Opt. Soc. Am. Vol. 2 (11), pp 1979-1985 (1985).

Blank

APPENDIX A
HILO.F LISTING

PROGRAM **HILO.F**
VERSION **Tue Sep 10 09:18:02 1991**

LINE #	SOURCE CODE	PAGE 1

```
      PROGRAM HILO
C        NO INPUT DATA.   A REGION OF THE X-N PLANE, 0<X<=10 AND
C     1.3<=N<=1.8, IS DIVIDED INTO RECTANGLES (PIXELS) WITH DELTA X =
C     .05 AND DELTA N = .005 : 200 COLUMNS AND 101 ROWS.   HILO
C     CALCULATES THE MINIMUM AND MAXIMUM VALUES OF EACH OF 27 FLUX
C     RATIOS FOR EACH PIXEL.  OUTPUT FILENAMES SPECIFY FLUX NUMERATOR
C     POLARIZATION AND ANGLE; DENOMINATOR ALWAYS U055.  ONLY X,N PAIRS
C     ON THE PERIMETERS OF PIXELS CONSIDERED.

      INTEGER NROW, NSAMPS, NSQR, DIRECT, K, I
      REAL MINMAX(27,2), ACCUM(8,200,27)
      CHARACTER*1 DUMMY
      CHARACTER FILNAM(54)*8

      FILNAM(1) =  'U010.MIN'
      FILNAM(2) =  'U010.MAX'
      FILNAM(3) =  'H010.MIN'
      FILNAM(4) =  'H010.MAX'
      FILNAM(5) =  'D010.MIN'
      FILNAM(6) =  'D010.MAX'
      FILNAM(7) =  'U040.MIN'
      FILNAM(8) =  'U040.MAX'
      FILNAM(9) =  'H040.MIN'
      FILNAM(10) = 'H040.MAX'
      FILNAM(11) = 'D040.MIN'
      FILNAM(12) = 'D040.MAX'
      FILNAM(13) = 'U055.MIN'
      FILNAM(14) = 'U055.MAX'
      FILNAM(15) = 'H055.MIN'
      FILNAM(16) = 'H055.MAX'
      FILNAM(17) = 'D055.MIN'
      FILNAM(18) = 'D055.MAX'
      FILNAM(19) = 'U075.MIN'
      FILNAM(20) = 'U075.MAX'
      FILNAM(21) = 'H075.MIN'
      FILNAM(22) = 'H075.MAX'
      FILNAM(23) = 'D075.MIN'
      FILNAM(24) = 'D075.MAX'
      FILNAM(25) = 'U090.MIN'
      FILNAM(26) = 'U09C.MAX'
      FILNAM(27) = 'H090.MIN'
      FILNAM(28) = 'H090.MAX'
      FILNAM(29) = 'D090.MIN'
      FILNAM(30) = 'D090.MAX'
      FILNAM(31) = 'U105.MIN'
      FILNAM(32) = 'U105.MAX'
      FILNAM(33) = 'H105.MIN'
      FILNAM(34) = 'H105.MAX'
      FILNAM(35) = 'D105.MIN'
      FILNAM(36) = 'D105.MAX'
      FILNAM(37) = 'U125.MIN'
      FILNAM(38) = 'U125.MAX'
      FILNAM(39) = 'H125.MIN'
      FILNAM(40) = 'H125.MAX'
      FILNAM(41) = 'D125.MIN'
      FILNAM(42) = 'D125.MAX'
      FILNAM(43) = 'U140.MIN'
      FILNAM(44) = 'U140.MAX'
      FILNAM(45) = 'H140.MIN'
      FILNAM(46) = 'H140.MAX'
```

PROGRAM HILO.F
VERSION Tue Sep 10 09:18:02 1991


```
      FILNAM(47) = 'D140.MIN'
      FILNAM(48) = 'D140.MAX'
      FILNAM(49) = 'U170.MIN'
      FILNAM(50) = 'U170.MAX'
      FILNAM(51) = 'H170.MIN'
      FILNAM(52) = 'H170.MAX'
      FILNAM(53) = 'D170.MIN'
      FILNAM(54) = 'D170.MAX'

C     THE FIRST ROW (N=1.300) AND THE REMAINING 100 ROWS
C     ARE TREATED SPEARATELY

C       ************* FIRST ROW ********************

C     GET MINS AND MAXS ON ALL FOUR SIDES (DIRECTions) OF FIRST SQUARE
      NROW = 1
      NSQR = 1
C     WHENEVER X < 5, HAVE "SIDE" LOOK AT 6 POINTS ALONG EACH EDGE OF
C     THE PERIMETER
      NSAMPS = 6
      DO 20 DIRECT=1,4
        CALL SIDE(DIRECT, NROW, NSQR, NSAMPS, MINMAX)
          DO 22 K=1,27
             ACCUM(2*DIRECT-1,NSQR,K) = MINMAX(K,1)
             ACCUM(2*DIRECT,NSQR,K) = MINMAX(K,2)
   22     CONTINUE
   20 CONTINUE

C      REMAINING 199 SQUARES IN THIS FIRST ROW REQUIRE THE CALLING OF
C     "SIDE" FOR EACH OF THREE DIRECTions

      DO 30 NSQR=2,100
C        WRITE(*,400) NSQR
C 400    FORMAT(1H ,'WORKING ON ROW 1 SQUARE ',I3)

C      WEST SIDE OF CURRENT SQUARE = EAST SIDE OF PREVIOUS SQUARE:
       DO 32 K=1,27
        ACCUM(7,NSQR,K) = ACCUM(3,NSQR-1,K)
        ACCUM(8,NSQR,K) = ACCUM(4,NSQR-1,K)
   32  CONTINUE
        DO 34 DIRECT=1,3
         CALL SIDE(DIRECT, NROW, NSQR, NSAMPS, MINMAX)
          DO 36 K=1,27
           ACCUM(2*DIRECT-1,NSQR,K) = MINMAX(K,1)
           ACCUM(2*DIRECT,NSQR,K) = MINMAX(K,2)
   36     CONTINUE
   34   CONTINUE
   30 CONTINUE

C     FOR THE RIGHT HALF OF THE ROW (X>5) INCREASE NSAMPS TO 20
C     AND PROCEED AS BEFORE
      NSAMPS = 20

      DO 40 NSQR=101,200
C        WRITE(*,401) NSQR
C 401    FORMAT(1H ,'WORKING ON ROW 1 SQUARE ',I3)

C      WEST SIDE OF CURRENT SQUARE = EAST SIDE OF PREVIOUS SQUARE:
       DO 42 K=1,27
        ACCUM(7,NSQR,K) = ACCUM(3,NSQR-1,K)
```

PROGRAM HILO.F
VERSION Tue Sep 10 09:18:02 1991


```
      ACCUM(8,NSQR,K) = ACCUM(4,NSQR-1,K)
42  CONTINUE
      DO 44 DIRECT=1,3
       CALL SIDE(DIRECT, NROW, NSQR, NSAMPS, MINMAX)
        DO 46 K=1,27
         ACCUM(2*DIRECT-1,NSQR,K) = MINMAX(K,1)
         ACCUM(2*DIRECT,NSQR,K) = MINMAX(K,2)
46     CONTINUE
44   CONTINUE
40 CONTINUE

C     NOW THE ARRAY ACCUM(8,200,27) IS FILLED WITH DATA FROM THE ENTIRE
C     FIRST ROW.  FOR EACH PIXEL IN THE ROW, FIND THE SMALLEST (OF THE
C     FOUR) MINIMUM AND THE LARGEST (OF FOUR) MAXIMUM.
C     OPEN FILES (2 AT A TIME) AND WRITE OUT THE ABSOLUTE MIN AND MAX
C     FLUX RATIOS FOR EACH PIXEL.

      DO 50 K=1,27
       OPEN(UNIT=3, FILE=FILNAM(2*K-1), ACCESS='SEQUENTIAL',
     &      FORM='FORMATTED', STATUS='NEW')
       OPEN(UNIT=4, FILE=FILNAM(2*K), ACCESS='SEQUENTIAL',
     &      FORM='FORMATTED', STATUS='NEW')
        DO 52 NSQR=1,200
          WRITE (3,'(E10.4)') MIN(ACCUM(1,NSQR,K),ACCUM(3,NSQR,K),
     &                            ACCUM(5,NSQR,K),ACCUM(7,NSQR,K))
52    CONTINUE
      DO 54 NSQR=1,200
        WRITE (4,'(E10.4)') MAX(ACCUM(2,NSQR,K),ACCUM(4,NSQR,K),
     &                            ACCUM(6,NSQR,K),ACCUM(8,NSQR,K))
54    CONTINUE
     CLOSE(3)
     CLOSE(4)
50 CONTINUE

C     ************** THIS ENDS THE FIRST ROW **********************
C     *************************************************************

C     THE REMAINING 100 ROWS ARE ALL HANDLED THE SAME SO WE HAVE ONE
C     GIANT LOOP FROM HERE TO THE END OF THE PROGRAM:

      DO 300 NROW=2,101

C     FOR WHATEVER THE CURRENT ROW IS, PUT THE NORTH MIN AND MAX VALUES
C     OF THE PREVIOUS ROW INTO THE SOUTH MIN AND MAX VALUES OF THE
C     CURRENT ROW FOR EACH OF THE 27 PAGES:

      DO 60 K=1,27
      DO 62 NSQR=1,200
      ACCUM(5,NSQR,K) = ACCUM(1,NSQR,K)
      ACCUM(6,NSQR,K) = ACCUM(2,NSQR,K)
62  CONTINUE
60  CONTINUE

C       THE FIRST SQUARE MUST BE TREATED SEPARATELY FROM THE REST:
     NSAMPS=6
     NSQR=1

     CALL SIDE(1,NROW,NSQR,NSAMPS,MINMAX)
     DO 70 K=1,27
     ACCUM(1,1,K) = MINMAX(K,1)
```

PROGRAM HILO.F
VERSION Tue Sep 10 09:18:02 1991


```
      ACCUM(2,1,K) = MINMAX(K,2)
   70 CONTINUE

      CALL SIDE(2,NROW,NSQR,NSAMPS,MINMAX)
      DO 72 K=1,27
      ACCUM(3,1,K) = MINMAX(K,1)
      ACCUM(4,1,K) = MINMAX(K,2)
   72 CONTINUE

      CALL SIDE(4,NROW,NSQR,NSAMPS,MINMAX)
      DO 74 K=1,27
      ACCUM(7,1,K) = MINMAX(K,1)
      ACCUM(8,1,K) = MINMAX(K,2)
   74 CONTINUE

C      THIS COMPLETES THE FIRST SQUARE ONLY.  THE REMAINDER OF THE
C      ROW IS DONE IN TWO PARTS; THE LEFT HALF AT LOW RESOLUTION
C      (NSAMPS=6) AND THE RIGHT HALF AT HIGH RESOLUTION (NSAMPS=20):
C      "SIDE" MUST NOW ONLY BE CALLED TWICE FOR EACH PIXEL, SINCE EACH
C      PIXEL'S BOTTOM AND LEFT EDGES HAVE ALREADY BEEN CONSIDERED (AND
C      REMEMBERED IN "ACCUM") AS PARTS OF PREVIOUSLY EVALUATED ADJACENT
C      PIXELS.

C        ****************** LEFT HALF OF ROW ****************

      NSAMPS = 6
      DO 80 NSQR=2,100

C     WRITE(*,402) NROW, NSQR
C 402 FORMAT(1H ,'WORKING ON ROW ',I3,2X,'SQUARE ',I3)

C     GET CURRENT SQUARE'S WEST SIDE FROM PREVIOUS SQUARE'S EAST SIDE:
      DO 82 K=1,27
      ACCUM(7,NSQR,K) = ACCUM(3,NSQR-1,K)
      ACCUM(8,NSQR,K) = ACCUM(4,NSQR-1,K)
   82 CONTINUE

        DO 84 DIRECT=1,2
        CALL SIDE(DIRECT,NROW,NSQR,NSAMPS,MINMAX)
            DO 86 K=1,27
            ACCUM(2*DIRECT-1,NSQR,K) = MINMAX(K,1)
            ACCUM(2*DIRECT,NSQR,K)   = MINMAX(K,2)
   86       CONTINUE
   84   CONTINUE
   80 CONTINUE

C        ************* RIGHT HALF OF ROW **********************

      NSAMPS = 20
      DO 90 NSQR=101,200

C     WRITE(*,403) NROW, NSQR
C 403 FORMAT(1H ,'WORKING ON ROW ',I3,2X,'SQUARE ',I3)

C     GET CURRENT SQUARE'S WEST SIDE FROM PREVIOUS SQUARE'S EAST SIDE:
      DO 92 K=1,27
      ACCUM(7,NSQR,K) = ACCUM(3,NSQR-1,K)
      ACCUM(8,NSQR,K) = ACCUM(4,NSQR-1,K)
   92 CONTINUE

```

PROGRAM **HILO.F**
VERSION **Tue Sep 10 09:18:02 1991**

LINE # SOURCE CODE PAGE 5

```
      DO 94 DIRECT=1,2
      CALL SIDE(DIRECT,NROW,NSQR,NSAMPS,MINMAX)
          DO 96 K=1,27
          ACCUM(2*DIRECT-1,NSQR,K) = MINMAX(K,1)
          ACCUM(2*DIRECT,NSQR,K)   = MINMAX(K,2)
 96       CONTINUE
 94    CONTINUE
 90 CONTINUE


C     THE 27 PAGES OF ACCUM ARE ALL FILLED NOW FOR THE CURRENT ROW.
C     WRITE RESULTS OUT TO FILES, APPENDING EACH TIME A FILE IS
C     RE-OPENED.  (THERE ARE TOO MANY FILES -54- TO HAVE THEM ALL
C     OPENED AT ONCE.  WE MUST OPEN AND CLOSE AS NEEDED FOR EACH
C     ROW.)   We are still inside the nrow loop

      DO 100 K=1,27
       OPEN(UNIT=3, FILE=FILNAM(2*K-1), ACCESS='SEQUENTIAL',
     &      FORM='FORMATTED', STATUS='OLD')
       OPEN(UNIT=4, FILE=FILNAM(2*K), ACCESS='SEQUENTIAL',
     &      FORM='FORMATTED', STATUS='OLD')

C     IN ORDER TO APPEND DATA TO AN 'OLD' FILE, REOPEN FILE AND
C     READ A DUMMY CHARACTER REPEATEDLY TO PLACE THE FILE POINTER TO
C     JUST PAST EOF, THEN BACKSPACE TO 'ERASE' EOF MARKER, LEAVING THE
C     POINTER POINTING TO WHERE EOF USED TO BE.  START APPENDING NEW
C     DATA AT THAT LOCATION.

      DO 103 I=1,21000
 103    READ(3,'(A)',END=105) DUMMY
 105    BACKSPACE(3)
        DO 102 NSQR=1,200
          WRITE (3,'(E10.4)') MIN(ACCUM(1,NSQR,K),ACCUM(3,NSQR,K),
     &                            ACCUM(5,NSQR,K),ACCUM(7,NSQR,K))
 102    CONTINUE

      DO 107 I=1,21000
 107    READ(4,'(A)',END=109) DUMMY
 109    BACKSPACE(4)
        DO 104 NSQR=1,200
          WRITE (4,'(E10.4)') MAX(ACCUM(2,NSQR,K),ACCUM(4,NSQR,K),
     &                            ACCUM(6,NSQR,K),ACCUM(8,NSQR,K))
 104    CONTINUE
       CLOSE(3)
       CLOSE(4)
 100 CONTINUE
C     FINISHED APPENDING TO MIN & MAX DATA FILES FOR THIS ROW

 300 CONTINUE
      STOP
      END



      SUBROUTINE SIDE (DIRECT, NROW, NSQR, NSAMPS, MINMAX)
C       GIVEN THE LOCATION OF A PIXEL (VIA NROW & NSQR), WHICH OF
C     THE FOUR EDGES TO EVALUALTE (VIA DIRECT), AND THE NUMBER
C     OF EVALUATION POINTS ALONG THAT EDGE (VIA NSAMPS), RETURNS
C     2-D ARRAY MINMAX, WHOSE ROWS CORRESPOND TO THE 27 FLUX RATIOS,
C     AND WHICH GIVES IN COL 1 THE MIN VALUE OF THE FLUX RATIO
```

PROGRAM HILO.F
VERSION Tue Sep 10 09:18:02 1991

LINE # SOURCE CODE PAGE 6

```
C      ALONG GIVEN EDGE AND IN COL 2 THE MAX VALUE.

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION THETD(91), ELTRMX(4,91,2)
      INTEGER NROW, NSQR, NSAMPS, SAMP, NDLT, NANG, DIRECT, JX, NN
      REAL MINMAX(27,2), IO(28,20), IU, IH, ID

      NDLT=1
      RFI=0.0
      JX=90/NDLT + 1
      NANG=0
      DO 10 J=1,JX
        THETD(J) = DBLE(NANG)
        NANG = NANG + NDLT
   10 CONTINUE

      DELTAX = .05/NSAMPS
      DELTAN = .005/NSAMPS
      CENTRN = 1.300 + .005*(NROW -1)
      CENTRX = 0.05 * NSQR

C     AT EACH SAMPLING POINT ALONG EDGE, CALC X AND N
      DO 20 SAMP=1,NSAMPS
        IF (DIRECT .EQ. 1) THEN
           RFR = CENTRN + .0025
           X = CENTRX -.025 -0.5*DELTAX + SAMP*DELTAX
        ELSE IF (DIRECT .EQ. 2) THEN
           RFR = CENTRN - .0025 -.5*DELTAN + SAMP*DELTAN
           X = CENTRX +.025
        ELSE IF (DIRECT .EQ. 3) THEN
           RFR = CENTRN - .0025
           X = CENTRX -.025 -0.5*DELTAX + SAMP*DELTAX
        ELSE IF (DIRECT .EQ. 4) THEN
           RFR = CENTRN - .0025 -.5*DELTAN + SAMP*DELTAN
           X = CENTRX -.025
        ENDIF

C     GET SCATT FUNCTIONS VS ANGLE AT SAMPLING POINT, VIA ELTRMX
       CALL SMIE (X, RFR, RFI, QSCAT, QEXT, CTBRQS, THETD, ELTRMX, JX)

C     FORM UNPOL, HORIZ POL, AND DIAG POL INTENSITIES AT SPA SCATTERING
C     ANGLES FROM ELTRMX RETURNS, CORRECTING FOR DETECTOR ACCEPTANCE
C     ANGLE.  PUT THEM INTO ARRAY IO (27+1 ROWS, ONE FOR EACH FLUX
C     RATIO PLUS THE REFERENCE INTENDSITY (U055) REPEATED IN LAST ROW),
C     AND ONE COL FOR EACH SAMPLING POINT (6 OR 20).
        CALL UDHINT(11,1,IU,IH,ID,ELTRMX)
        IO(1,SAMP) = IU
        IO(2,SAMP) = IH
        IO(3,SAMP) = ID

        CALL UDHINT(41,1,IU,IH,ID,ELTRMX)
        IO(4,SAMP) = IU
        IO(5,SAMP) = IH
        IO(6,SAMP) = ID

        CALL UDHINT(56,1,IU,IH,ID,ELTRMX)
        IO(7,SAMP) = IU
        IO(8,SAMP) = IH
        IO(9,SAMP) = ID

```

PROGRAM HILO.F
VERSION Tue Sep 10 09:18:02 1991


```
      CALL UDHINT(76,1,IU,IH,ID,ELTRMX)
      IO(10,SAMP) = IU
      IO(11,SAMP) = IH
      IO(12,SAMP) = ID

          IU = 0.05825*(ELTRMX(2,90,1)+ELTRMX(1,90,1))
     1        +0.3835*(ELTRMX(2,91,1)+ELTRMX(1,91,1))
     2        +0.05825*(ELTRMX(2,90,2)+ELTRMX(1,90,2))
          IH = 0.05825*ELTRMX(1,90,1)
     1        +0.3835*ELTRMX(1,91,1)
     2        +0.05825*ELTRMX(1,90,2)
          ID = 0.029125*(ELTRMX(2,90,1)+ELTRMX(1,90,1))
     1        +0.05825*ELTRMX(4,90,1)
     2        +0.19175*(ELTRMX(2,91,1)+ELTRMX(1,91,1))
     3        +0.3835*ELTRMX(4,91,1)
     4        +0.029125*(ELTRMX(2,90,2)+ELTRMX(1,90,2))
     5        +0.05825*ELTRMX(4,90,2)
       IO(13,SAMP) = IU
       IO(14,SAMP) = IH
       IO(15,SAMP) = ID

      CALL UDHINT(76,2,IU,IH,ID,ELTRMX)
      IO(16,SAMP) = IU
      IO(17,SAMP) = IH
      IO(18,SAMP) = ID

      CALL UDHINT(56,2,IU,IH,ID,ELTRMX)
      IO(19,SAMP) = IU
      IO(20,SAMP) = IH
      IO(21,SAMP) = ID

      CALL UDHINT(41,2,IU,IH,ID,ELTRMX)
      IO(22,SAMP) = IU
      IO(23,SAMP) = IH
      IO(24,SAMP) = ID

      CALL UDHINT(11,2,IU,IH,ID,ELTRMX)
      IO(25,SAMP) = IU
      IO(26,SAMP) = IH
      IO(27,SAMP) = ID

      IO(28,SAMP) = IO(7,SAMP)
   20 CONTINUE

C     THE ARRAY IO IS NOW FILLED UP WITH INTENSITIES.
C     NEXT DIVIDE BY IU055 (IN 28TH ROW) TO GET FLUX RATIOS IN IO

      DO 30 IROW=1,27
      DO 31 ICOL=1,NSAMPS
      IO(IROW,ICOL) = IO(IROW,ICOL) / IO(28,ICOL)
   31 CONTINUE
   30 CONTINUE

C     FILL THE ARRAY MINMAX WITH THE MINIMUM AND MAXIMUM VALUES
C     FOR EACH FLUX RATIO

      DO 34 K=1,27
        MINMAX(K,1) = IO(K,1)
        MINMAX(K,2) = IO(K,1)
          DO 35 SAMP=2,NSAMPS
```

PROGRAM **HILO.F**
VERSION **Tue Sep 10 09:18:02 1991**


```
421            MINMAX(K,1) = MIN(IO(K,SAMP),MINMAX(K,1))
422            MINMAX(K,2) = MAX(IO(K,SAMP),MINMAX(K,2))
423       35   CONTINUE
424
425   C         MINMAX(K,1) NOW CONTAINS THE SMALLEST OF THE NSAMPS VALUES
426   C      OF THE KTH FLUX RATIO
427   C         MINMAX(K,2) NOW CONTAINS THE LARGEST OF THE NSAMPS VALUES
428   C      OF THE KTH FLUX RATIO
429
430       34 CONTINUE
431          RETURN
432          END
433
434
435
436          SUBROUTINE SMIE(X,RFR,RFI,QSCAT,QEXT,CTBRQS,THETD,ELTRMX,JX)
437   C       STRAIGHT FROM PETER BARBER'S PROGRAM
438   C       COMMENT C'S ADDED NEAR END TO SUPPRESS PRINTING NO. PARTIAL WAVES
439          IMPLICIT DOUBLE PRECISION (A-H,O-Z)
440          DIMENSION T(4),WFNR(2),WFNI(2),ELTRMX(4,91
441         1,2),PI(3,91),TAU(3,91),THETD(91),COSTH(91),SINTH2(91)
442          DIMENSION ACAPR(1050),ACAPI(1050)
443          DTR = .017453292500
444          C = RFR**2+RFI**2
445          RRFR = RFR/C
446          RRFI = RFI/C
447          RX = 1.0D0/X
448          RRFXR = RRFR*RX
449          RRFXI = RRFI*RX
450          NMX2 = IFIX(SNGL(X+4.05D0*X**.33333333D0+2.0D0))
451          IF(NMX2.LE.1050) GO TO 6
452          WRITE(*,2)
453        2 FORMAT( 'KA IS TOO LARGE')
454          STOP
455        6 NMX1 = IFIX(SNGL(1.1D0*X*DSQRT(C)))
456          IF(NMX1.LT.150) NMX1 = 150
457          CNAR = 0.0D0
458          CNAI = 0.0D0
459          DO 7 N = NMX1,NMX2,-1
460          CN = DBLE(FLOAT(N+1))
461          CNR = CN*RRFXR
462          CNI = CN*RRFXI
463          DR = CNR+CNAR
464          DI = CNI+CNAI
465          D = DR**2+DI**2
466          CNAR = CNR-DR/D
467          CNAI = CNI+DI/D
468        7 CONTINUE
469          ACAPR(NMX2) = CNAR
470          ACAPI(NMX2) = CNAI
471          NMM = NMX2-1
472          DO 8 N = NMM,1,-1
473          CN = DBLE(FLOAT(N+1))
474          CNR = CN*RRFXR
475          CNI = CN*RRFXI
476          DR = CNR+ACAPR(N+1)
477          DI = CNI+ACAPI(N+1)
478          D = DR**2+DI**2
479          ACAPR(N) = CNR-DR/D
480          ACAPI(N) = CNI+DI/D
```

```
    8 CONTINUE
      DO 9 J = 1,JX
      TH = DTR*THETD(J)
      COSTH(J) = DCOS(TH)
      SINTH2(J) = DSIN(TH)**2
      PI(1,J) = 0.0D0
      PI(2,J) = 1.0D0
      TAU(1,J) = 0.0D0
      TAU(2,J) = COSTH(J)
    9 CONTINUE
      WM1R = DCOS(X)
      WM1I = -DSIN(X)
      WFNR(1) = -WM1I
      WFNI(1) = WM1R
      WFNR(2) = WFNR(1)*RX-WM1R
      WFNI(2) = WFNI(1)*RX-WM1I
      TC1R = ACAPR(1)*RRFR-ACAPI(1)*RRFI+RX
      TC1I = ACAPI(1)*RRFR+ACAPR(1)*RRFI
      TC2R = ACAPR(1)*RFR+ACAPI(1)*RFI+RX
      TC2I = ACAPI(1)*RFR-ACAPR(1)*RFI
      A = TC1R*WFNR(2)-WFNR(1)
      B = TC1I*WFNR(2)
      C = TC1R*WFNR(2)-TC1I*WFNI(2)-WFNR(1)
      D = TC1I*WFNR(2)+TC1R*WFNI(2)-WFNI(1)
      CD2 = C**2+D**2
      FNAR = (A*C+B*D)/CD2
      FNAI = (B*C-A*D)/CD2
      A = TC2R*WFNR(2)-WFNR(1)
      B = TC2I*WFNR(2)
      C = TC2R*WFNR(2)-TC2I*WFNI(2)-WFNR(1)
      D = TC2I*WFNR(2)+TC2R*WFNI(2)-WFNI(1)
      CD2 = C**2+D**2
      FNBR = (A*C+B*D)/CD2
      FNBI = (B*C-A*D)/CD2
      FNAPR = FNAR
      FNAPI = FNAI
      FNBPR = FNBR
      FNBPI = FNBI
      QSCAT = 3.0D0*(FNAR**2+FNAI**2+FNBR**2+FNBI**2)
      QEXT = 3.0D0*(FNAR+FNBR)
      CTBRQS = 0.0D0
      FNAR = 1.5D0*FNAR
      FNAI = 1.5D0*FNAI
      FNBR = 1.5D0*FNBR
      FNBI = 1.5D0*FNBI
      DO  10 J = 1,JX
      ELTRMX(1,J,1) = FNAR*PI(2,J)+FNBR*TAU(2,J)
      ELTRMX(2,J,1) = FNAI*PI(2,J)+FNBI*TAU(2,J)
      ELTRMX(3,J,1) = FNBR*PI(2,J)+FNAR*TAU(2,J)
      ELTRMX(4,J,1) = FNBI*PI(2,J)+FNAI*TAU(2,J)
      ELTRMX(1,J,2) = FNAR*PI(2,J)-FNBR*TAU(2,J)
      ELTRMX(2,J,2) = FNAI*PI(2,J)-FNBI*TAU(2,J)
      ELTRMX(3,J,2) = FNBR*PI(2,J)-FNAR*TAU(2,J)
      ELTRMX(4,J,2) = FNBI*PI(2,J)-FNAI*TAU(2,J)
   10 CONTINUE
      N = 2
      DO 15 I = 1,NMX2
      FN = DBLE(FLOAT(N))
      T(1) = 2.0D0*FN-1.0D0
      T(2) = FN-1.0D0
```

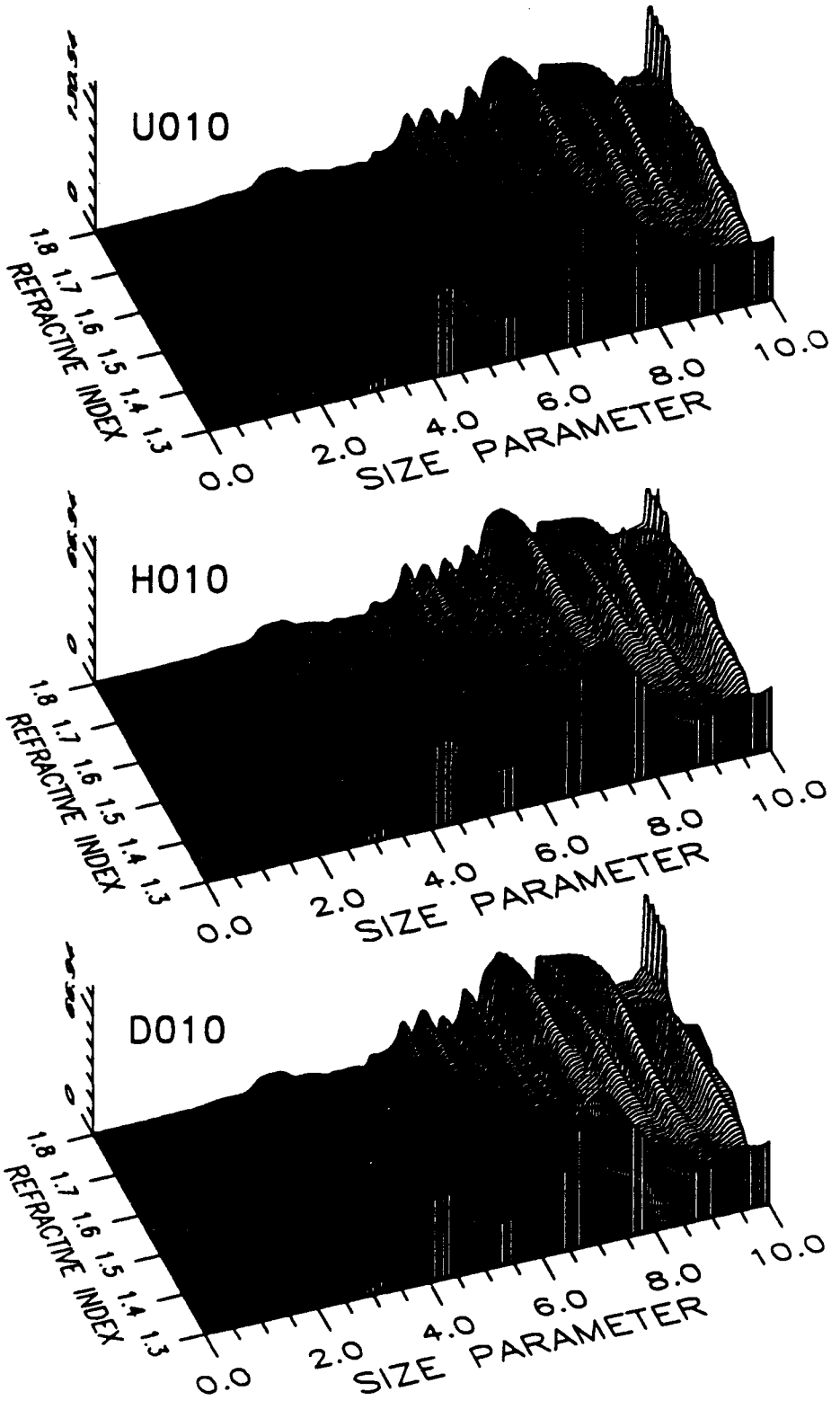
PROGRAM **HILO.F**
VERSION **Tue Sep 10 09:18:02 1991**

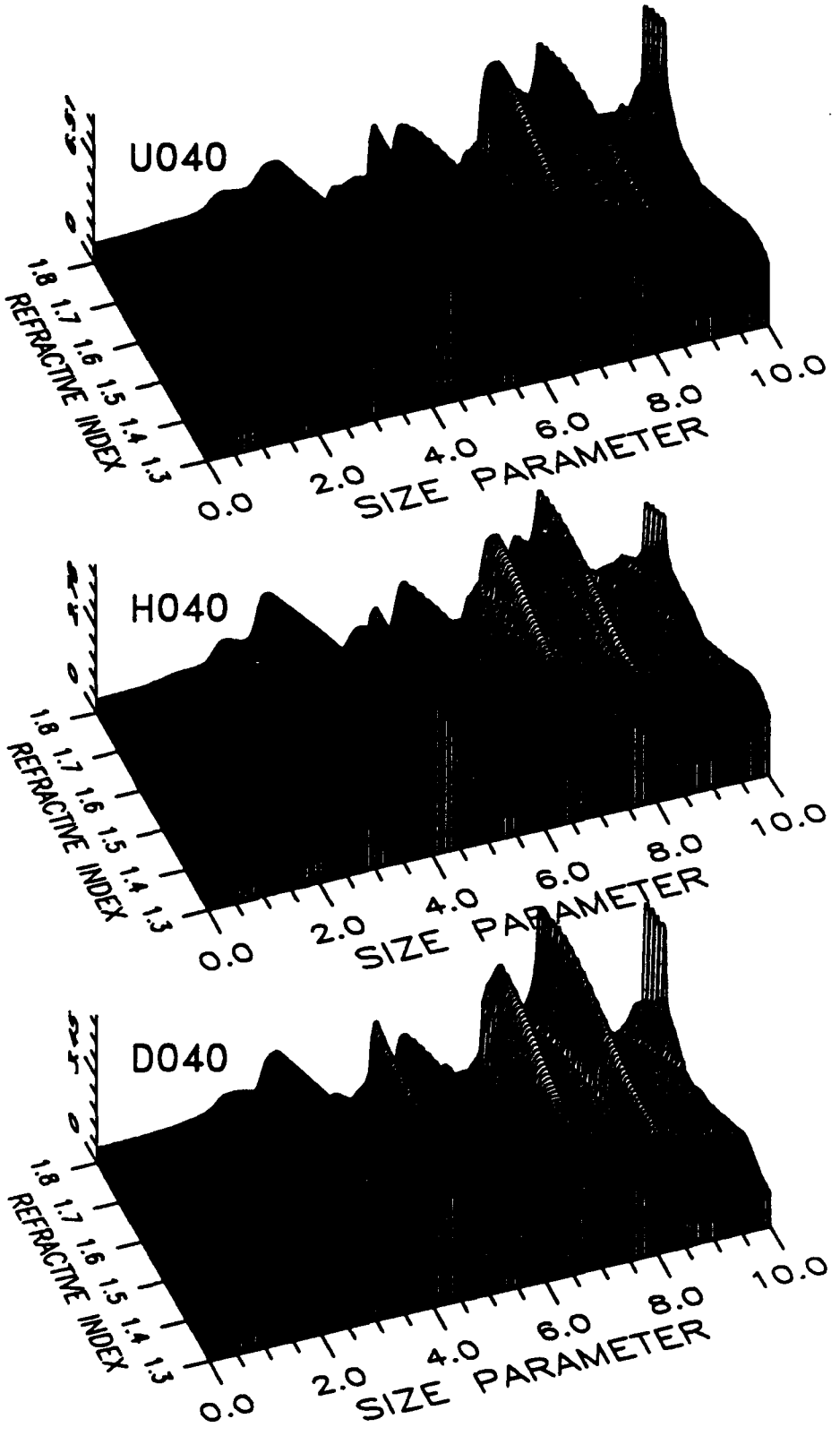

```
      T(3) = 2.0D0*FN+1.0D0
      DO 11 J = 1,JX
      PI(3,J) = (T(1)*PI(2,J)*COSTH(J)-FN*PI(1,J))/T(2)
      TAU(3,J) = COSTH(J)*(PI(3,J)-PI(1,J))-T(1)*SINTH2(J)*PI(2,J)+TAU(1
     1,J)
   11 CONTINUE
      WM1R = WFNR(1)
      WM1I = WFNI(1)
      WFNR(1) = WFNR(2)
      WFNI(1) = WFNI(2)
      WFNR(2) = T(1)*RX*WFNR(1)-WM1R
      WFNI(2) = T(1)*RX*WFNI(1)-WM1I
      TC1R = ACAPR(N)*RRFR-ACAPI(N)*RRFI+FN*RX
      TC1I = ACAPI(N)*RRFR+ACAPR(N)*RRFI
      TC2R = ACAPR(N)*RFR+ACAPI(N)*RFI+FN*RX
      TC2I = ACAPI(N)*RFR-ACAPR(N)*RFI
      A = TC1R*WFNR(2)-WFNR(1)
      B = TC1I*WFNR(2)
      C = TC1R*WFNR(2)-TC1I*WFNI(2)-WFNR(1)
      D = TC1I*WFNR(2)+TC1R*WFNI(2)-WFNI(1)
      CD2 = C**2+D**2
      FNAR = (A*C+B*D)/CD2
      FNAI = (B*C-A*D)/CD2
      A = TC2R*WFNR(2)-WFNR(1)
      B = TC2I*WFNR(2)
      C = TC2R*WFNR(2)-TC2I*WFNI(2)-WFNR(1)
      D = TC2I*WFNR(2)+TC2R*WFNI(2)-WFNI(1)
      CD2 = C**2+D**2
      FNBR = (A*C+B*D)/CD2
      FNBI = (B*C-A*D)/CD2
      T(4) = T(1)/(FN*T(2))
      T(2) = (T(2)*(FN+1.0D0))/FN
      CTBRQS = CTBRQS+T(2)*(FNAPR*FNAR+FNAPI*FNAI+FNBPR*FNBR+FNBPI*FNBI)
     1+T(4)*(FNAPR*FNBPR+FNAPI*FNBPI)
      T(4) = FNAR**2+FNAI**2+FNBR**2+FNBI**2
      QSCAT = QSCAT+T(3)*T(4)
      QEXT = QEXT+T(3)*(FNAR+FNBR)
      T(2) = FN*(FN+1.0D0)
      T(1) = T(3)/T(2)
      K = (N/2)*2
      DO 13 J = 1,JX
      ELTRMX(1,J,1) = ELTRMX(1,J,1)+T(1)*(FNAR*PI(3,J)+FNBR*TAU(3,J))
      ELTRMX(2,J,1) = ELTRMX(2,J,1)+T(1)*(FNAI*PI(3,J)+FNBI*TAU(3,J))
      ELTRMX(3,J,1) = ELTRMX(3,J,1)+T(1)*(FNBR*PI(3,J)+FNAR*TAU(3,J))
      ELTRMX(4,J,1) = ELTRMX(4,J,1)+T(1)*(FNBI*PI(3,J)+FNAI*TAU(3,J))
      IF(K.EQ.N) GO TO 12
      ELTRMX(1,J,2) = ELTRMX(1,J,2)+T(1)*(FNAR*PI(3,J)-FNBR*TAU(3,J))
      ELTRMX(2,J,2) = ELTRMX(2,J,2)+T(1)*(FNAI*PI(3,J)-FNBI*TAU(3,J))
      ELTRMX(3,J,2) = ELTRMX(3,J,2)+T(1)*(FNBR*PI(3,J)-FNAR*TAU(3,J))
      ELTRMX(4,J,2) = ELTRMX(4,J,2)+T(1)*(FNBI*PI(3,J)-FNAI*TAU(3,J))
      GO TO 13
   12 ELTRMX(1,J,2) = ELTRMX(1,J,2)+T(1)*(-FNAR*PI(3,J)+FNBR*TAU(3,J))
      ELTRMX(2,J,2) = ELTRMX(2,J,2)+T(1)*(-FNAI*PI(3,J)+FNBI*TAU(3,J))
      ELTRMX(3,J,2) = ELTRMX(3,J,2)+T(1)*(-FNBR*PI(3,J)+FNAR*TAU(3,J))
      ELTRMX(4,J,2) = ELTRMX(4,J,2)+T(1)*(-FNBI*PI(3,J)+FNAI*TAU(3,J))
   13 CONTINUE
      IF(T(4).LT.1.0D-14) GO TO 20
      N = N+1
      DO 14 J = 1,JX
      PI(1,J) = PI(2,J)
```

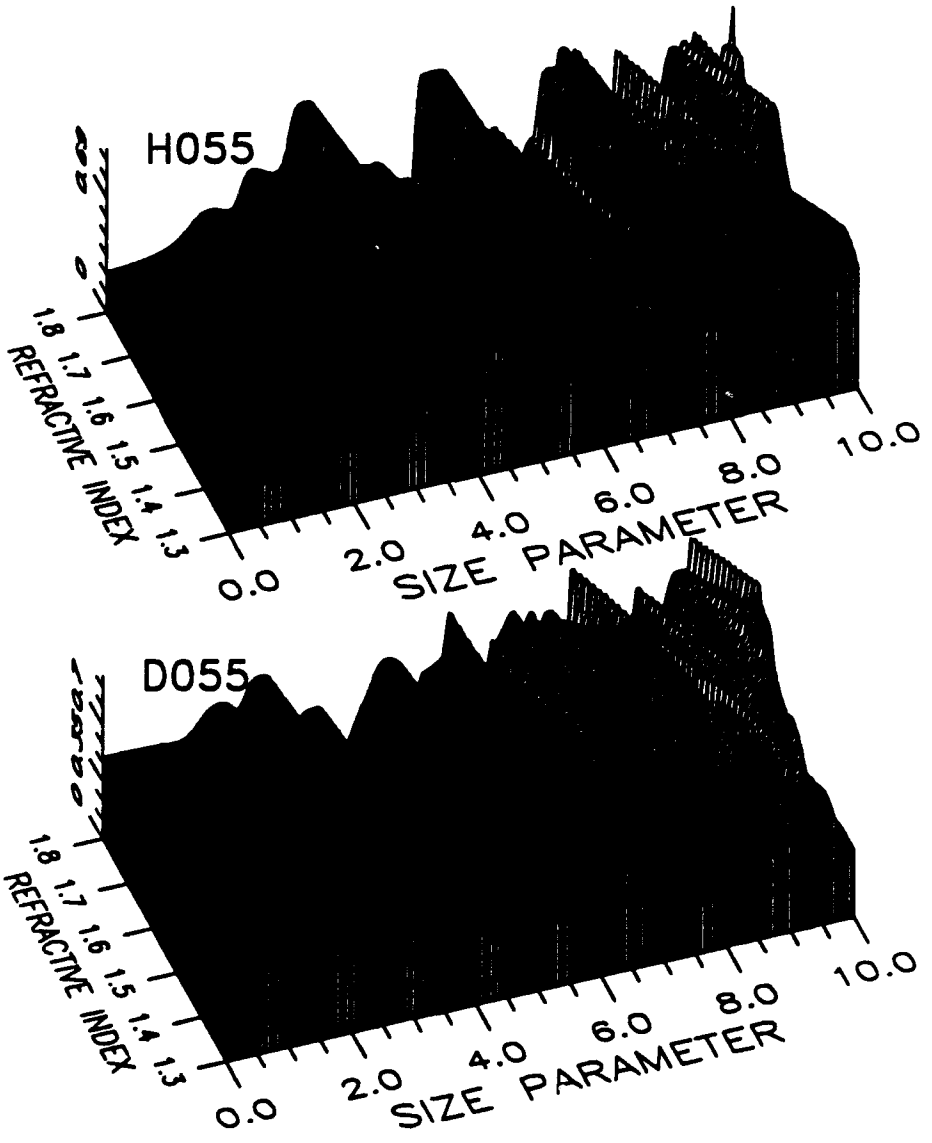
```
      PI(2,J) = PI(3,J)
      TAU(1,J) = TAU(2,J)
      TAU(2,J) = TAU(3,J)
   14 CONTINUE
      FNAPR = FNAR
      FNAPI = FNAI
      FNBPR = FNBR
      FNBPI = FNBI
   15 CONTINUE
      QSCAT = 0.0D0
      QEXT = 0.0D0
      RETURN
   20 DO 30 J = 1,JX
      DO 30 K = 1,2
      DO 25 I = 1,4
      T(I) = ELTRMX(I,J,K)
   25 CONTINUE
      ELTRMX(2,J,K) = T(1)**2+T(2)**2
      ELTRMX(1,J,K) = T(3)**2+T(4)**2
      ELTRMX(3,J,K) = T(1)*T(3)+T(2)*T(4)
      ELTRMX(4,J,K) = T(2)*T(3)-T(4)*T(1)
   30 CONTINUE
      C = 2.0D0*RX**2
      QSCAT = C*QSCAT
      QEXT = C*QEXT
      CTBRQS = 2.0D0*CTBRQS*C
C     WRITE(6,200) N
C 200 FORMAT(I6)
      RETURN
      END

      SUBROUTINE UDHINT(JJ,KK,IU,IH,ID,ELTRMX)
C       GIVEN INTEGERS WHICH TOGETHER SPECIFY A PARTICULAR ONE OF THE
C       NINE SPA SCATTERING ANGLES (JJ & KK), AND THE COMPLETE ARRAY
C       ELTRMX, RETURNS COMBINATIONS OF THE SCATTERING MATRIX ELEMENTS
C       FOR THE GIVEN ANGLE, WITH CORRECTION FOR FINITE DETECTOR
C       ACCEPTANCE ANGLE ACCOMPLISHED BY ADDING IN BITS FROM SCATTERING
C       ANGLES ONE DEGREE HIGHER AND LOWER.
      REAL IU,IH,ID
      INTEGER JJ,KK
      DOUBLE PRECISION ELTRMX(4,91,2)

            IU = 0.05825*(ELTRMX(2,JJ-1,KK)+ELTRMX(1,JJ-1,KK))
     1           +0.3835*(ELTRMX(2,JJ,KK)+ELTRMX(1,JJ,KK))
     2           +0.05825*(ELTRMX(2,JJ+1,KK)+ELTRMX(1,JJ+1,KK))
            IH = 0.05825*ELTRMX(1,JJ-1,KK)
     1           +0.3835*ELTRMX(1,JJ,KK)
     2           +0.05825*ELTRMX(1,JJ+1,KK)
            ID = 0.029125*(ELTRMX(2,JJ-1,KK)+ELTRMX(1,JJ-1,KK))
     1           +0.05825*ELTRMX(4,JJ-1,KK)
     2           +0.19175*(ELTRMX(2,JJ,KK)+ELTRMX(1,JJ,KK))
     3           +0.3835*ELTRMX(4,JJ,KK)
     4           +0.029125*(ELTRMX(2,JJ+1,KK)+ELTRMX(1,JJ+1,KK))
     5           +0.05825*ELTRMX(4,JJ+1,KK)

      RETURN
      END
```

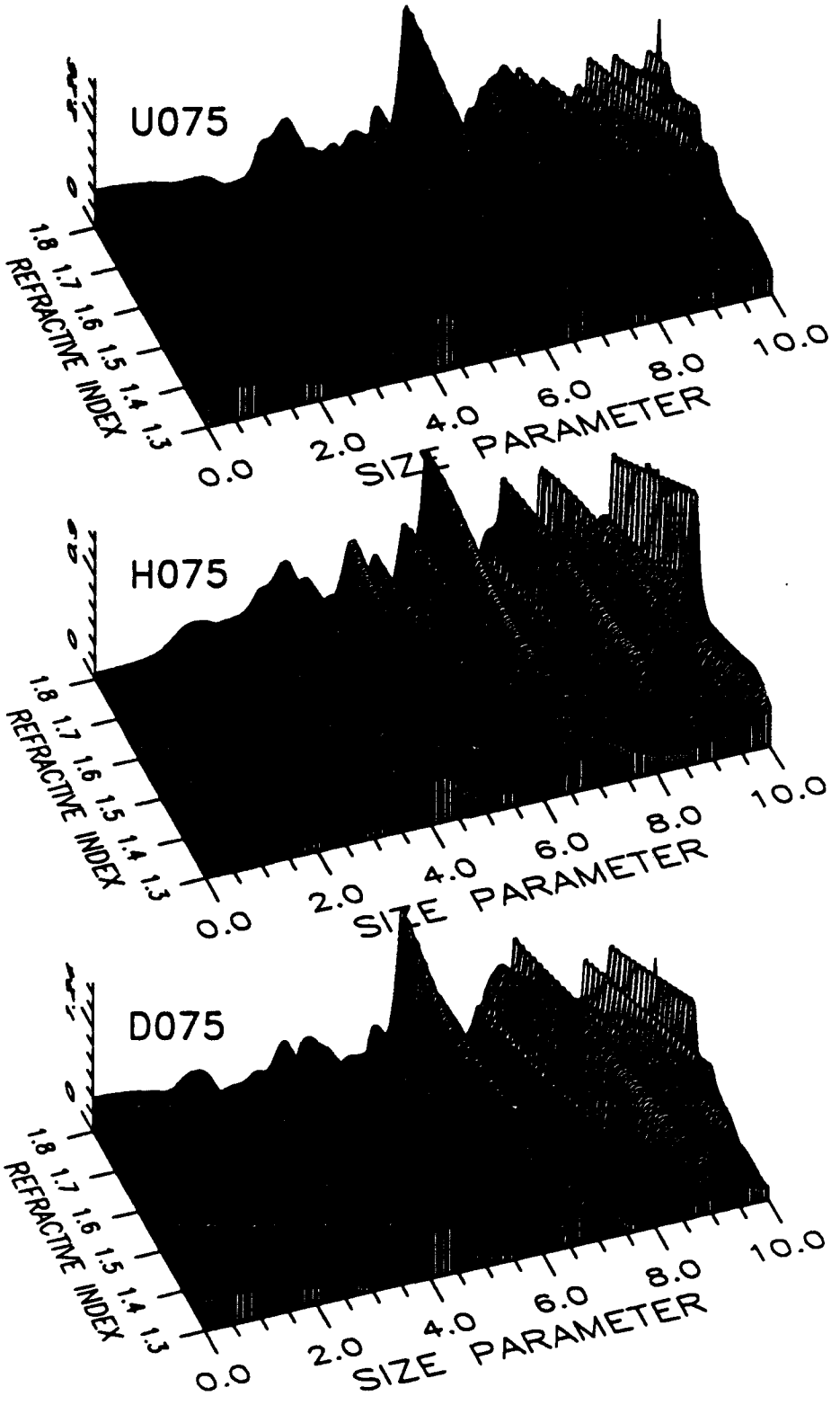
Blank

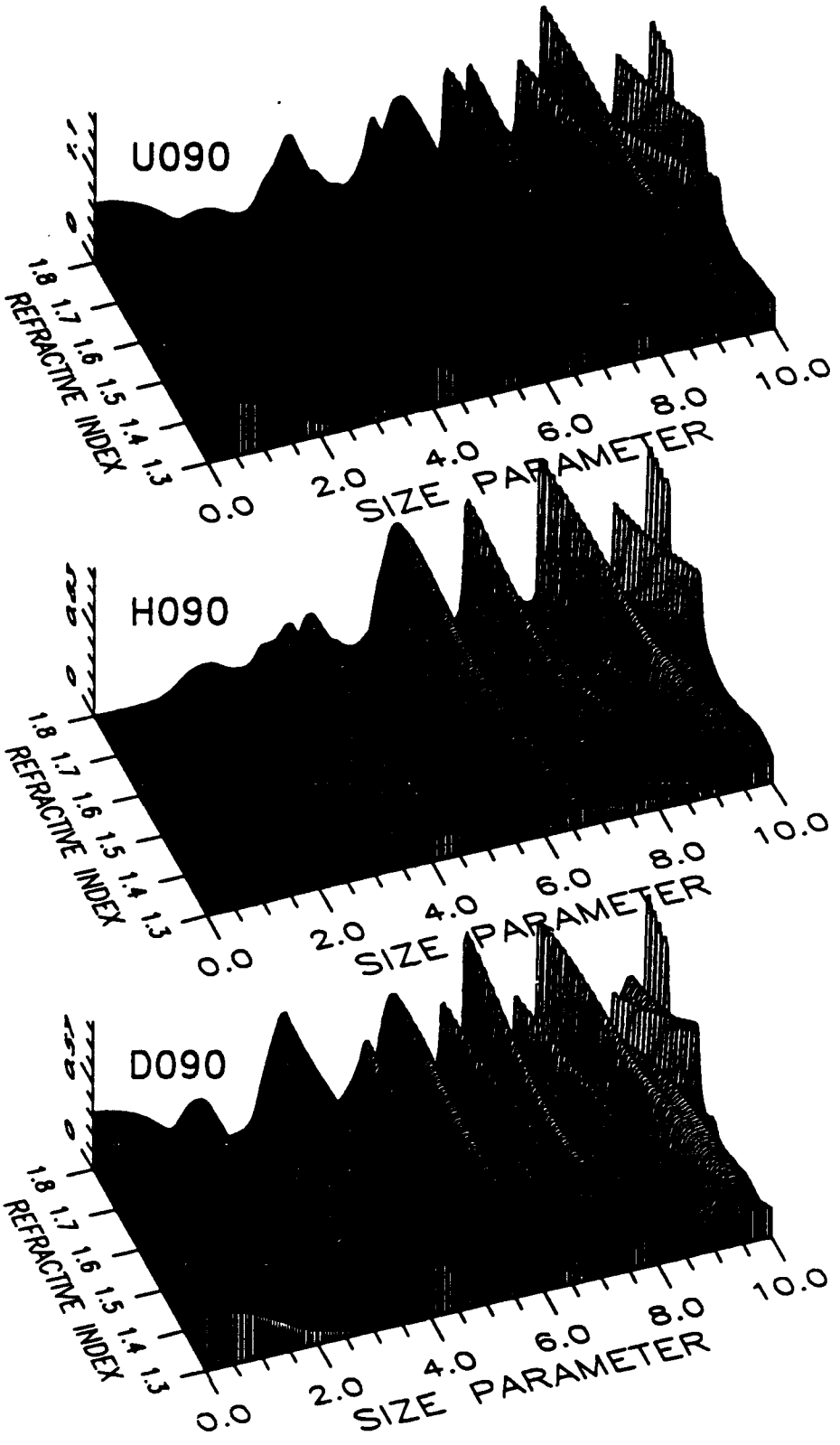
APPENDIX B
CALCULATED RATIO MAPS

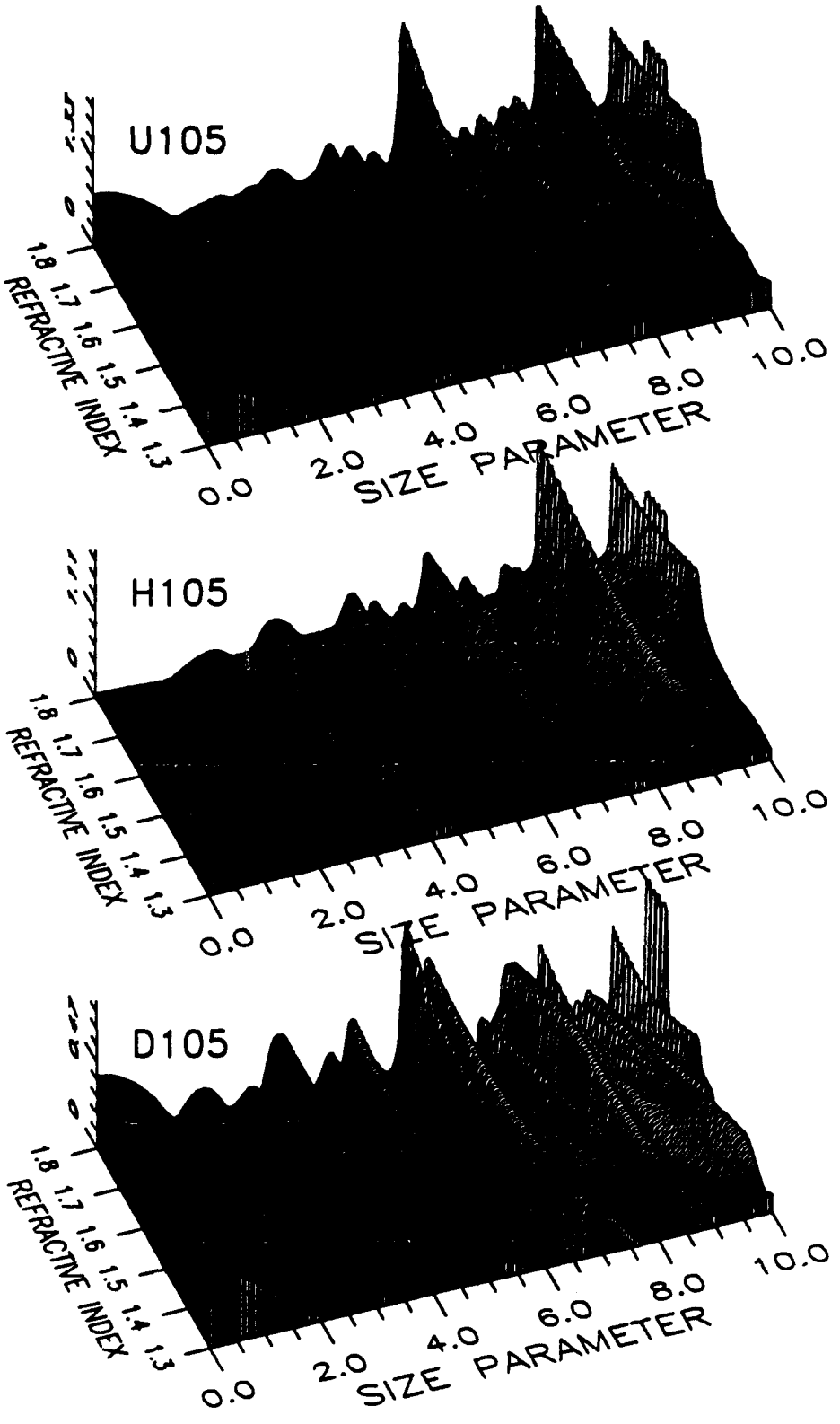


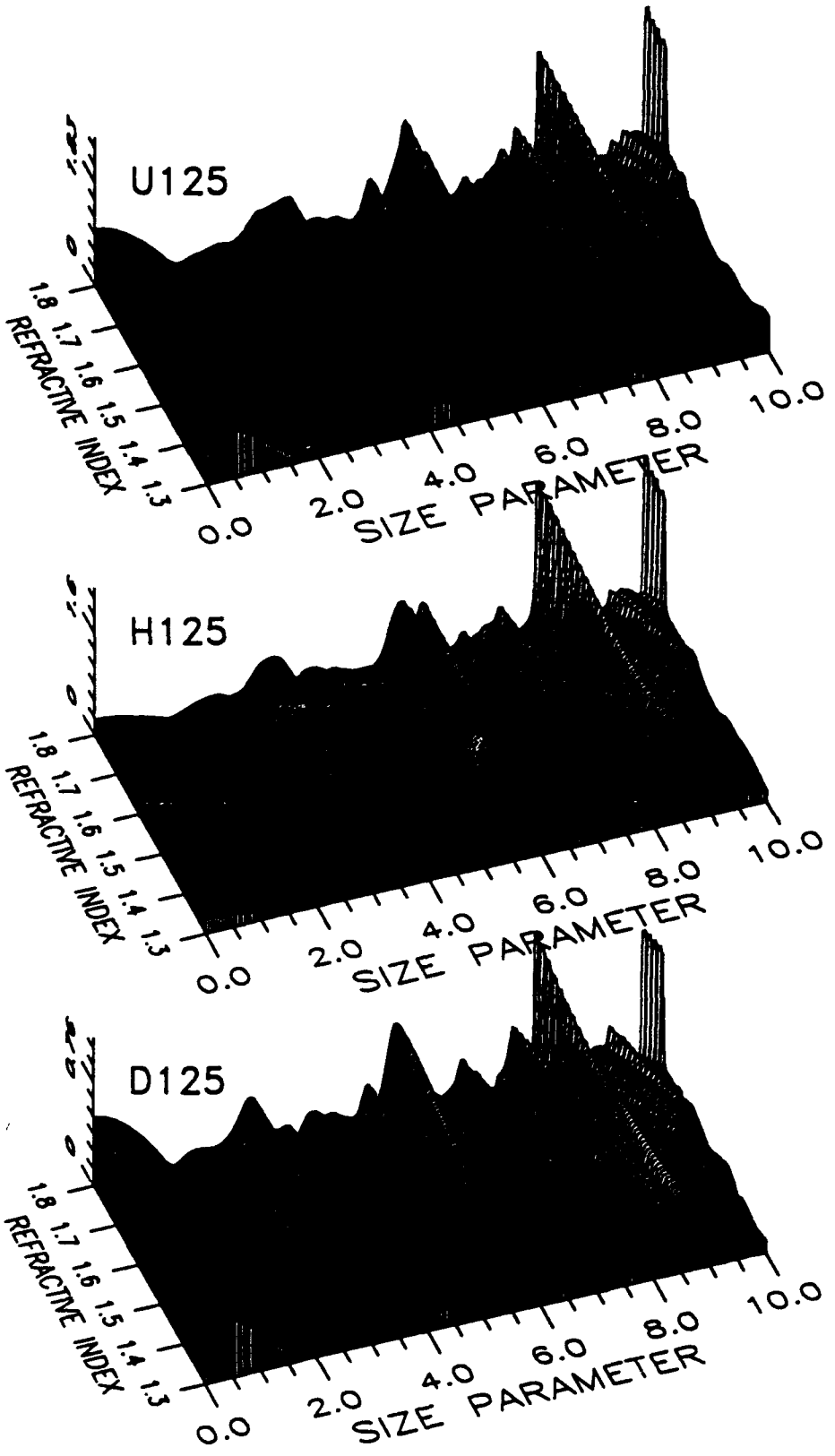

U040
H040
D040
REFRACTIVE INDEX
1.8 1.7 1.6 1.5 1.4 1.3
0.0 2.0 4.0 6.0 8.0 10.0
SIZE PARAMETER

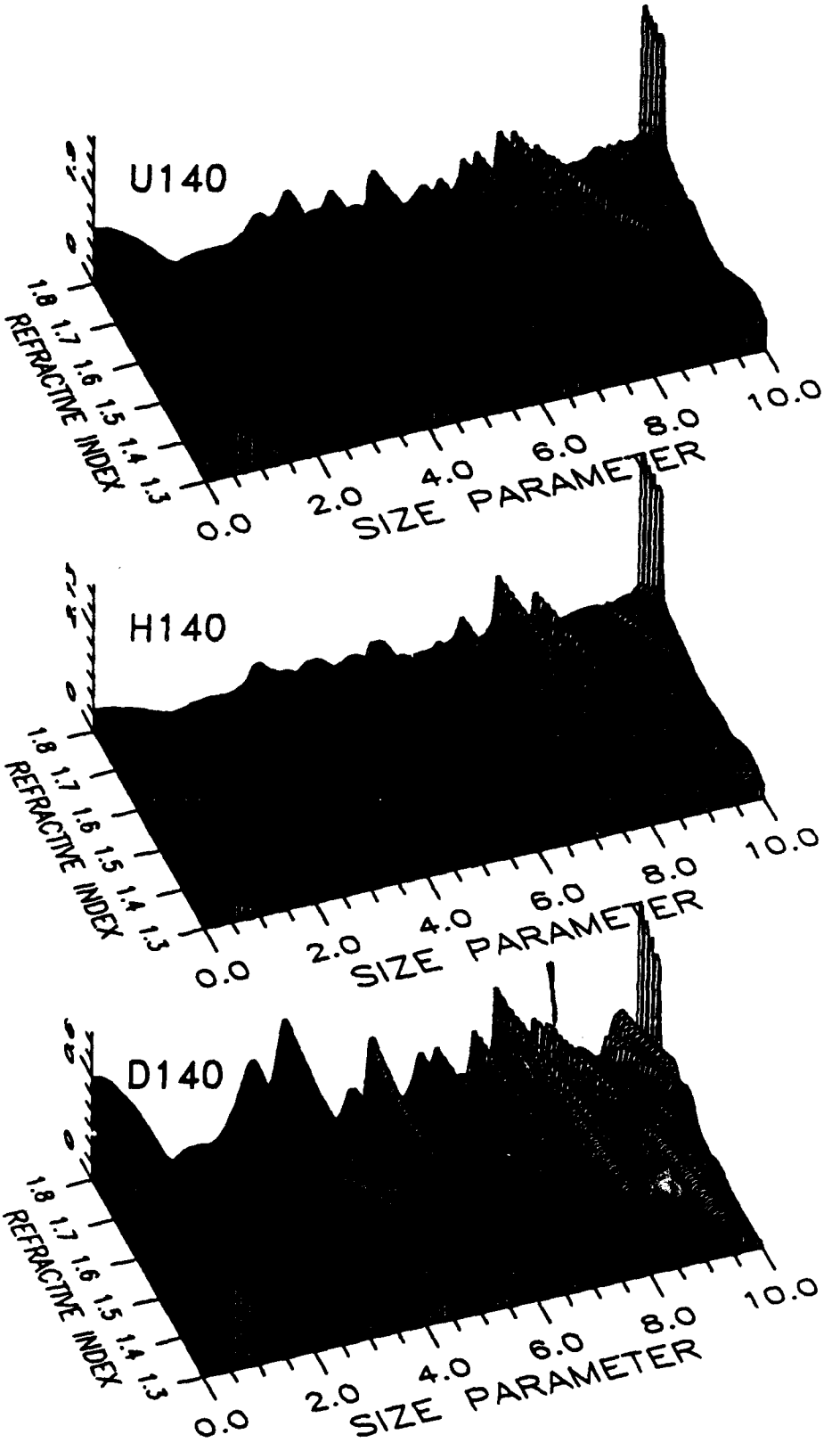

H055
REFRACTIVE INDEX
1.8 1.7 1.6 1.5 1.4 1.3
0.0 2.0 4.0 6.0 8.0 10.0
SIZE PARAMETER
D055
REFRACTIVE INDEX
1.8 1.7 1.6 1.5 1.4 1.3
0.0 2.0 4.0 6.0 8.0 10.0
SIZE PARAMETER

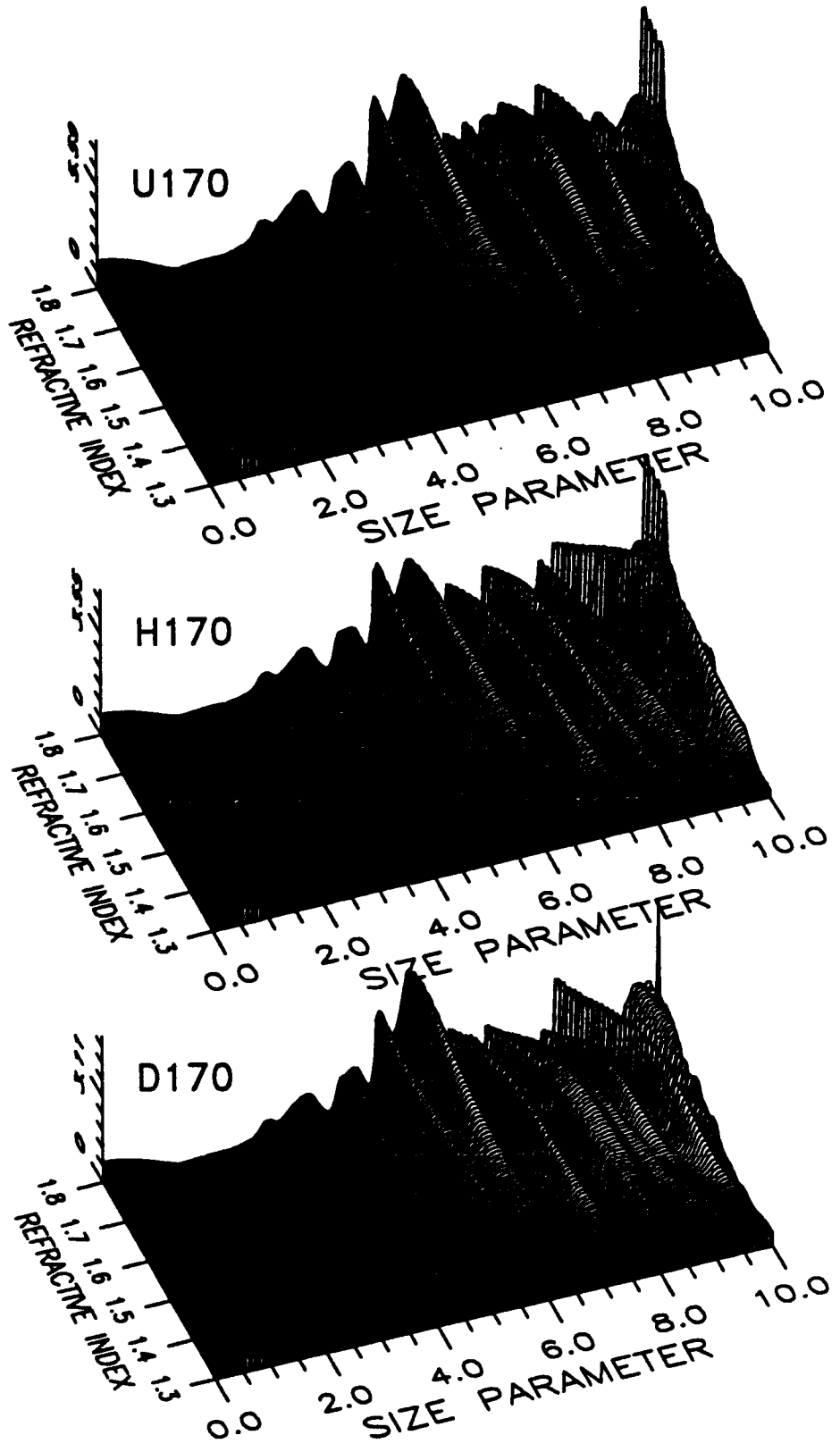

U075
H075
D075
REFRACTIVE INDEX
1.8 1.7 1.6 1.5 1.4 1.3
0.0 2.0 4.0 6.0 8.0 10.0
SIZE PARAMETER


U090
REFRACTIVE INDEX
1.8 1.7 1.6 1.5 1.4 1.3
0.0 2.0 4.0 6.0 8.0 10.0
SIZE PARAMETER
H090
REFRACTIVE INDEX
1.8 1.7 1.6 1.5 1.4 1.3
0.0 2.0 4.0 6.0 8.0 10.0
SIZE PARAMETER
D090
REFRACTIVE INDEX
1.8 1.7 1.6 1.5 1.4 1.3
0.0 2.0 4.0 6.0 8.0 10.0
SIZE PARAMETER


U105
H105
D105
REFRACTIVE INDEX
1.8 1.7 1.6 1.5 1.4 1.3
0.0 2.0 4.0 6.0 8.0 10.0
SIZE PARAMETER


U125
H125
D125
REFRACTIVE INDEX
1.8 1.7 1.6 1.5 1.4 1.3
0.0 2.0 4.0 6.0 8.0 10.0
SIZE PARAMETER


U140
H140
D140
REFRACTIVE INDEX
1.8 1.7 1.6 1.5 1.4 1.3
0.0 2.0 4.0 6.0 8.0 10.0
SIZE PARAMETER


U170
REFRACTIVE INDEX
1.8 1.7 1.6 1.5 1.4 1.3
0.0 2.0 4.0 6.0 8.0 10.0
SIZE PARAMETER
H170
REFRACTIVE INDEX
1.8 1.7 1.6 1.5 1.4 1.3
0.0 2.0 4.0 6.0 8.0 10.0
SIZE PARAMETER
D170
REFRACTIVE INDEX
1.8 1.7 1.6 1.5 1.4 1.3
0.0 2.0 4.0 6.0 8.0 10.0
SIZE PARAMETER

Blank

APPENDIX C
INVERT.F LISTING

PROGRAM **INVERT.F**
VERSION **Thu Jan 16 12:05:06 1992**

LINE #	SOURCE CODE	PAGE 1

```
      PROGRAM INVERT
      REAL ORDMIN(14,20200), ORDMAX(14,20200), EXPDAT(1000,25)
      REAL CALDAT(25), AVG, STD, SPX, SPHLIM, UNCAVG, UNCFIX
      REAL FLUX(14), UNC(14), HIVAL(14), LOVAL(14)
      INTEGER NDXMIN(14,20200), NDXMAX(14,20200), NPARTS
      INTEGER I, J, K, NUMCHN, HITS(20200), MISSES(20200)
      INTEGER MISLIM, ITRY, ILO, IHI, COUNT, DENS
      CHARACTER*12 ORNFIL(14), ORXFIL(14), NDNFIL(14), NDXFIL(14)
      CHARACTER*24 EXPNAM, OUTNAM, CALNAM, OLDNAM
      CHARACTER*7 DATE

C        Below, left side expressions are character variables used
C        only in do loops to read in the previously calculated data.
C        Right side strings are the file names containing the data.


      ORNFIL(1) = 'ORD_U040.MIN'
      ORXFIL(1) = 'ORD_U040.MAX'
      NDNFIL(1) = 'NDX_U040.MIN'
      NDXFIL(1) = 'NDX_U040.MAX'

      ORNFIL(2) = 'ORD_U075.MIN'
      ORXFIL(2) = 'ORD_U075.MAX'
      NDNFIL(2) = 'NDX_U075.MIN'
      NDXFIL(2) = 'NDX_U075.MAX'

      ORNFIL(3) = 'ORD_U090.MIN'
      ORXFIL(3) = 'ORD_U090.MAX'
      NDNFIL(3) = 'NDX_U090.MIN'
      NDXFIL(3) = 'NDX_U090.MAX'

      ORNFIL(4) = 'ORD_U105.MIN'
      ORXFIL(4) = 'ORD_U105.MAX'
      NDNFIL(4) = 'NDX_U105.MIN'
      NDXFIL(4) = 'NDX_U105.MAX'

      ORNFIL(5) = 'ORD_U125.MIN'
      ORXFIL(5) = 'ORD_U125.MAX'
      NDNFIL(5) = 'NDX_U125.MIN'
      NDXFIL(5) = 'NDX_U125.MAX'

      ORNFIL(6) = 'ORD_U140.MIN'
      ORXFIL(6) = 'ORD_U140.MAX'
      NDNFIL(6) = 'NDX_U140.MIN'
      NDXFIL(6) = 'NDX_U140.MAX'

      ORNFIL(7) = 'ORD_H040.MIN'
      ORXFIL(7) = 'ORD_H040.MAX'
      NDNFIL(7) = 'NDX_H040.MIN'
      NDXFIL(7) = 'NDX_H040.MAX'

      ORNFIL(8) = 'ORD_H075.MIN'
      ORXFIL(8) = 'ORD_H075.MAX'
      NDNFIL(8) = 'NDX_H075.MIN'
      NDXFIL(8) = 'NDX_H075.MAX'

      ORNFIL(9) = 'ORD_H090.MIN'
      ORXFIL(9) = 'ORD_H090.MAX'
      NDNFIL(9) = 'NDX_H090.MIN'
      NDXFIL(9) = 'NDX_H090.MAX'
```

PROGRAM INVERT.F
VERSION Thu Jan 16 12:05:06 1992


```
      ORNFIL(10) = 'ORD_H105.MIN'
      ORXFIL(10) = 'ORD_H105.MAX'
      NDNFIL(10) = 'NDX_H105.MIN'
      NDXFIL(10) = 'NDX_H105.MAX'

      ORNFIL(11) = 'ORD_H125.MIN'
      ORXFIL(11) = 'ORD_H125.MAX'
      NDNFIL(11) = 'NDX_H125.MIN'
      NDXFIL(11) = 'NDX_H125.MAX'

      ORNFIL(12) = 'ORD_H140.MIN'
      ORXFIL(12) = 'ORD_H140.MAX'
      NDNFIL(12) = 'NDX_H140.MIN'
      NDXFIL(12) = 'NDX_H140.MAX'

      ORNFIL(13) = 'ORD_D040.MIN'
      ORXFIL(13) = 'ORD_D040.MAX'
      NDNFIL(13) = 'NDX_D040.MIN'
      NDXFIL(13) = 'NDX_D040.MAX'

      ORNFIL(14) = 'ORD_D090.MIN'
      ORXFIL(14) = 'ORD_D090.MAX'
      NDNFIL(14) = 'NDX_D090.MIN'
      NDXFIL(14) = 'NDX_D090.MAX'


C       Fill the array ORDMIN(14,20200) with previously calculated data
      DO 10 I=1,14
      OPEN (UNIT=3, FILE=ORNFIL(I),ACCESS='SEQUENTIAL',
     &      FORM='FORMATTED', STATUS='OLD')
      READ(3,*) (ORDMIN(I,J), J=1,20200)
   10 CONTINUE
      CLOSE (3)

C       Fill the array ORDMAX(14,20200) with previously calculated data
      DO 20 I=1,14
      OPEN (UNIT=3, FILE=ORXFIL(I),ACCESS='SEQUENTIAL',
     &      FORM='FORMATTED', STATUS='OLD')
      READ(3,*) (ORDMAX(I,J), J=1,20200)
   20 CONTINUE
      CLOSE (3)

C       Fill the array NDXMIN(14,20200) with previously calculated data
      DO 30 I=1,14
      OPEN (UNIT=3, FILE=NDNFIL(I),ACCESS='SEQUENTIAL',
     &      FORM='FORMATTED', STATUS='OLD')
      READ(3,*) (NDXMIN(I,J), J=1,20200)
   30 CONTINUE
      CLOSE (3)

C       Fill the array NDXMAX(14,20200) with previously calculated data
      DO 40 I=1,14
      OPEN (UNIT=3, FILE=NDXFIL(I),ACCESS='SEQUENTIAL',
     &      FORM='FORMATTED', STATUS='OLD')
      READ(3,*) (NDXMAX(I,J), J=1,20200)
   40 CONTINUE
      CLOSE (3)
C       The previously calculated data is now all set up in RAM.  Read
C       in the experimental data from a group of ~500 particles, ie a
```

PROGRAM **INVERT.F**

VERSION **Thu Jan 16 12:05:06 1992**


```
C        file outputted from TRANS.EXE such as EI118A.PRN.  The length
C        of the filename cannot be known ahead of time, so it must be
C        determined during runtime.  The maximum length of the filename
C        should not exceed 24 characters; I will also assume that the
C        number of particles represented by the data does not exceed
C        1000.

   50 WRITE (*,*) ' ENTER PATH/FILE NAME OF DATA SET TO PROCESS'
      WRITE (*,*) ' EXAMPLES:  II83.PRN  or C:\WYATT\VI107B.PRN'
      WRITE (*,*) ' or enter x to quit program'
      READ (*,'(A)') EXPNAM
        IF ((EXPNAM(1:1) .EQ.'X') .OR. (EXPNAM(1:1) .EQ. 'x')) THEN
        GO TO 270
        END IF
      OPEN (UNIT=3, FILE=EXPNAM, ACCESS='SEQUENTIAL',
     &      FORM='FORMATTED', STATUS='OLD')


C        Pause here to construct OUTNAM, the name of the output file
C        which will be written at the end of this entire program.  It
C        will, by the construction of its name, go to whatever sub-
C        directory the data set (EXPNAM) came from.  In effect we will
C        just substitute .MAP for .PRN in the string held by EXPNAM.
         I=1
   60    I=I+1
         IF (I .GE. 22) GO TO 50
         IF (EXPNAM(I:I) .NE. '.') GO TO 60
         OUTNAM = EXPNAM(1:I) // 'MAP'
C           The "." is in the position corresponding to the current
C           value of I.  The total length of OUTNAM (and EXPNAM) is
C           I+3 characters.
C                              Back to reading in the data:

      DO 70 I=1,1000
      READ (3,*,END=80) (EXPDAT(I,J), J=1,25)
   70 CONTINUE
C        Control should jump from the do-loop to statemen 80 when EOF
C        is reached, presumably before I=1000.  When it exits the
C        current value of I -1    should be the
C        number of rows of data, ie., the number of particles in the
C        data set (nparts).
   80 CLOSE(3)
      NPARTS = I-1

C       temporary
      WRITE (*,*) ' ENTER THE AMOUNT OF UNCERTAINTY TO BE ASSOCIATED'
      WRITE (*,*) ' WITH THIS THESE DATA.  Enter as a decimal number'
      WRITE (*,*) ' from 0 to 1.0  ie, +/- 25% entered as 0.25'
      READ (*,*) UNCFIX
      DO 79 I=1,14
      UNC(I) = UNCFIX
   79 CONTINUE



C      Somewhere on disk I must have a file of correction factors
C      previously determined in a calibration proceedure and applicable
C      to the experimental data which is here being analyzed.  Those
C      factors must be read in and multiplied against the corresponding
C      columns of EXPDAT.  Assume the format of the calibration file is
```

PROGRAM **INVERT.F**

VERSION **Thu Jan 16 12:05:06 1992**


```
C
C                    23Feb91      EI107A.PRN
C                    0            1
C                    1            1.0762
C                    2            0.8997
C                    .            .
C                    .            .
C                    22           1.677
C                    23           0.944
C
C      First line gives date of calibration and the data set (of
C      spheres) used to perform the calibration.  Thereafter the left
C      column gives the fiber optic channel number and the right column
C      the corresponding multiplicative correction factor.  The
C      channels labeled 0 and 15 are the bundles, and we'll just set
C      the correction factor to 1.   25 lines in all.

      WRITE (*,*) 'ENTER THE PATH/FILE NAME OF THE APPLICABLE'
      WRITE (*,*) 'CALIBRATION FILE.  Enter L to just re-use the Latest'
      WRITE (*,*) 'filename used this session.'
      READ (*,'(A)') CALNAM
      IF ((CALNAM(1:1) .EQ. 'L') .OR. (CALNAM(1:1) .EQ. 'l')) GO TO 85
      OLDNAM = CALNAM
   85 CALNAM = OLDNAM
      OPEN (UNIT=3, FILE=CALNAM, ACCESS='SEQUENTIAL',
     &      FORM='FORMATTED', STATUS='OLD')
      READ (3,'(A)') DATE
      CALDAT(1) = 1.0
      DO 90 K=2,25
      READ(3,*) NUMCHN, CALDAT(K)
   90 CONTINUE
      CLOSE (3)

C       Now make the calibration correction:
      DO 100 I=1, NPARTS
           DO 110, K=1, 25
           EXPDAT(I,K) = EXPDAT(I,K)*CALDAT(K)
  110      CONTINUE
  100 CONTINUE

C     Now the array EXPDAT contains the corrected data values, ready
C     for use.


C      Initialize the group_accumulator [HITS(20200)] to all zeros.
C      This is the only time it will be zeroed; its status at the end
C      of the program is the output we're looking for.  Also initialize
C      the particle_accumulator [MISSES(20200)] for the first time; it
C      will be reset to zero again at the end of each particle do-loop.

      DO 120 J=1,20200
      HITS(J) = 0
      MISSES(J) = 0
  120 CONTINUE

C      Now begins the particle do-loop; ie, we look at particles one at
C      a time (the Ith particle)

      DO 200 I=1, NPARTS

```

PROGRAM INVERT.F
VERSION Thu Jan 16 12:05:06 1992


```
241       AVG = (EXPDAT(I,13) + EXPDAT(I,14) + EXPDAT(I,15) + EXPDAT(I,16)
242      &     + EXPDAT(I,22) + EXPDAT(I,23) + EXPDAT(I,24) + EXPDAT(I,25))
243      &       /8.0
244       STD =SQRT(((EXPDAT(I,13)-AVG)**2 + (EXPDAT(I,14)-AVG)**2
245      &         + (EXPDAT(I,15)-AVG)**2 + (EXPDAT(I,16)-AVG)**2
246      &         + (EXPDAT(I,22)-AVG)**2 + (EXPDAT(I,23)-AVG)**2
247      &         + (EXPDAT(I,24)-AVG)**2 + (EXPDAT(I,25)-AVG)**2)/8.0)
248       SPX = 1.0 - (STD/(2.6458*AVG))
249
250
251 C      If this particle isn't a good sphere, stop right now
252       SPHLIM = 0.9
253       IF (SPX .LT. SPHLIM) GO TO 200
254
255 C      The uncertainty to be associated with each data point (in
256 C      EXPDAT) depends upon the magnitude of the data, generally
257 C      getting worse as the intensity dims.  I don't yet know how to
258 C      calculate the uncertainty, but I will assume that it will be by
259 C      exactly the same formula for all channels.  Give the observables
260 C      numbers (1-14) for identification.  See page 14 hand notes.
261
262 C     UNCAVG = f(avg) not yet known
263
264       FLUX(1) = EXPDAT(I,11)
265       FLUX(2) = EXPDAT(I,8)
266       FLUX(3) = EXPDAT(I,9)
267       FLUX(4) = EXPDAT(I,20)
268       FLUX(5) = EXPDAT(I,5)
269       FLUX(6) = EXPDAT(I,6)
270       FLUX(7) = EXPDAT(I,10)
271       FLUX(8) = EXPDAT(I,7)
272       FLUX(9) = EXPDAT(I,19)
273       FLUX(10) = EXPDAT(I,21)
274       FLUX(11) = EXPDAT(I,3)
275       FLUX(12) = EXPDAT(I,4)
276       FLUX(13) = EXPDAT(I,12)
277       FLUX(14) = EXPDAT(I,18)
278
279         DO 124 J=1,14
280 C       UNC(J) = function of flux(j) still unknown
281         HIVAL(J) = (FLUX(J)/AVG) * (1.0 + UNC(J))
282         LOVAL(J) = (FLUX(J)/AVG) * (1.0 - UNC(J))
283
284   124   CONTINUE
285
286 C        Now we do a do-loop over the observables, writing to MISSES
287 C        the result of each observable trial
288
289           DO 150 K=1,14
290             ILO = 1
291             IHI = 20200
292             ITRY = 10100
293             DO 130 COUNT=1,15
294              IF (ORDMIN(K,ITRY) .LT. HIVAL(K)) THEN
295              ILO = ITRY
296              ITRY = INT((ITRY+IHI)/2)
297              ELSE
298              IHI = ITRY
299              ITRY = INT((ITRY+ILO)/2)
300              END IF
```

```
  130       CONTINUE


C      Now ITRY is the element of ORDMIN whose value equals HIVAL(K)
            DO 132 L=ITRY,20200
            MISSES(NDXMIN(K,L)) = MISSES(NDXMIN(K,L)) + 1
  132       CONTINUE

C      Repeat the above for ORDMAX:
            ILO = 1
            IHI = 20200
            ITRY = 10100
            DO 134 COUNT=1,15
             IF (ORDMAX(K,ITRY) .LT. LOVAL(K)) THEN
             ILO = ITRY
             ITRY = INT((ITRY+IHI)/2)
             ELSE
             IHI = ITRY
             ITRY = INT((ITRY+ILO)/2)
             END IF
  134       CONTINUE

C      This time at the end of the loop ITRY is the element of ORDMAX
C      whose value equals (closely enough) LOVAL.  The elements of
C      NDXMAX from 1 up to ITRY contain the numbers (ie, names) of some
C      more missed pixels.  Increment MISSES accordingly.
            DO 136 L=1,ITRY
            MISSES(NDXMAX(K,L)) = MISSES(NDXMAX(K,L)) + 1
  136       CONTINUE
C      This completes the incrementing of MISSES for this one (kth)
C      observable.  Loop back for the next observable.
  150     CONTINUE

C      The array MISSES is now filled, for this one particle.  Each of
C      its 20200 elements contains an integer from 0 to 14, which
C      counts the number of times the corresponding pixel FAILED to be
C      in agreement with the data.
C
C      Next step for this particle is to increment the array HITS.  For
C      testing purposes, will tolerate three different levels of
C      hitting:  eg. no misses, 1-2 misses, and 3-4 misses.
C      (Later, define only two levels of hitting: zero or 1 misses
C      and 2 or more misses)
C
C      Whenever HITS is incremented, reset MISSES to zeros.
C
      DO 160 J=1,20200
        IF (MISSES(J) .LE. 1) THEN
        HITS(J) = HITS(J) + 1
        END IF
        MISSES(J) = 0
  160 CONTINUE

C      That does it for this particle, go back for the next one.
  200 CONTINUE

C      Now all the particles have been analyzed, and HITS contains the
C      distribution over all particles (ie, each element of HITS
C      contains an integer which is the number of particles whose r and
C      n could have been the same as the pixel's r and n).  Output the
```

PROGRAM INVERT.F
VERSION Thu Jan 16 12:05:06 1992

LINE # SOURCE CODE PAGE 7

```
C       array HITS - but sensibly; if there are no hits in a pixel, just
C       skip that pixel, don't write out a zero.  DENS is a parameter
C       that will control how darkly a pixel square is printed under
C       PageGarden.  First include EXPNAM and UNCFIX on picture

      OPEN (UNIT=4, FILE=OUTNAM, ACCESS='SEQUENTIAL',
     &      FORM='FORMATTED', STATUS='NEW')

      WRITE(4,*) ' disp 4.5" 1.5" "TEST CASE:" '
      WRITE(4,201) EXPNAM
  201 FORMAT (' dispc " ',A,'"')
      WRITE(4,*) ' disp 4.5" 1.75" "UNCERTAINTY: +/- " '
      WRITE(4,202) 100.0*UNCFIX
  202 FORMAT (' dispc "',F4.1,'%"')

      DO 220 J=1,20200
      IF (HITS(J) .EQ. 0) THEN
      GO TO 220
      ELSE
      DENS = 60
      END IF

      NPOSX = 525 + 12*MOD((J-1),200)
      NPOSY = 1870 - 12*INT((J-1)/200)

      WRITE(4,250) NPOSX, NPOSY, DENS
  250 FORMAT(1X , 'area ',I4,'d ',I4,'d',' 12d 12d ',I3,/,
     &       1X , 'box +0 +0 12d 12d 1d')

  220 CONTINUE
      CLOSE(4)

C       Go back for another data set.
      GO TO 50
  270 STOP
      END
```