sizes Documentation

Release 2014.03.10

Pete R. Jemian

Sep 27, 2017

Contents

author Pete R. Jemian

email jemian@anl.gov

URL http://prjemian.github.io/sizes

git https://github.com/prjemian/sizes

Caution: Python version is not complete.

The Python version of the *sizes* program (for size distribution analysis of small-angle scattering data using the maximum entropy method of Skilling and Bryan) **is under construction.** Not all features of the C and FORTRAN versions of the code are yet available.

The documentation for this version is rudimentary, as well.

The manual for the FORTRAN version is available: `guide.pdf` as well as a previous version of the FORTRAN documentation in the section here titled: *USER GUIDE FOR THE MAXIMUM ENTROPY SAS ANALYSIS PROGRAM MaxSas.for*.

Contents:

CHAPTER 1

Table of Contents

Contents:

About sizes

License

This software is licensed using the Argonne OPEN SOURCE LICENSE, see `LICENSE` file for details

Change History

This describes user-visible changes between the versions.

This program is a translation from the C which is from the FORTRAN which is from the BASIC version.

Version 2013-05

initial translation from C code

Credits:

- P.R. Jemian, Argonne National Laboratory, USA

Used in Irena package

An IgorPro translation of this code appears in the *Irena* software package, authored by Jan Ilavsky. [cite publication history here]

Credits:

- Jan Ilavsky, Argonne National Laboratory, USA

- P.R. Jemian, Argonne National Laboratory, USA

C translation

The FORTRAN code was translated into C ca. 1990.

Credits:

- P.R. Jemian, Northwestern University, USA

Previously

This progam was written in BASIC by GJ Daniell and later translated into FORTRAN and adapted for SANS analysis. It has been further modified by AJ Allen to allow use with a choice of particle form factors for different shapes. It was then modified by PR Jemian to allow portability between the Digital Equipment Corporation VAX and Apple Macintosh computers.

Credits:

- G.J. Daniell, Dept. of Physics, Southampton University, UK
- J.A. Potton, UKAEA Harwell Laboratory, UK
- I.D. Culverwell, UKAEA Harwell Laboratory, UK
- G.P. Clarke, UKAEA Harwell Laboratory, UK
- A.J. Allen, UKAEA Harwell Laboratory, UK

maximum entropy calculation engine

The 1984 Skilling and Bryan paper is available online[1] through the SAO/NASA ADS Astronomy Abstract Service:

main source code module: *sizes*

USER GUIDE FOR THE MAXIMUM ENTROPY SAS ANALYSIS PROGRAM MaxSas.for

author Pete R. Jemian, Northwestern University, 7 February 1990

Substantial portions of this document are from the documentation supplied with the code MAXE (documentation by Ian Culverwell, UKAEA-Harwell, 23 February 1987) and with its modification called MAXE2 (modifications by Andrew Allen, UKAEA-Harwell, 19 July 1989).

Contents:

[1] http://adsabs.harvard.edu/abs/1984MNRAS.211..111S

INTRODUCTION

This program provides an accurate and reliable analysis of small-angle scattering data. Using an iterative approach it calculates the size distribution of scatterers of a specified form, with the maximum entropy, whose scattering pattern is consistent with the data. The consistency test is the ChiSquared statistic. A full description of the program's methodology and of its rigorous validation can be found in [Culverwell, 1986] and [Potton, 1988a].

The MaxSas code is an interactive program, and users must be prepared to answer the questions that the program will ask. The questions are grouped into two major sections: input data selection and scattering model selection. The input data selection is by file name and Q-vector range. The scattering model selection is by form factor, diameter range, and number of histogram bins. It may be necessary to run this program several times, with slight changes in user-adjustable parameters, in order to obtain a satisfactory analysis.

The running of the program will be demonstrated by example, using synthetic scattering data calculated from a hypothetical bimodal particle size distribution (ref 3). This distribution consists of a narrow Gaussian peak centered at 80 Angstroms with a standard deviation of 20 A, together with a broader secondary Gaussian peak, half the size of the primary one, centered at 200 A with a width of 60 A. In order that they can become familiar with the questions that the program will ask them, prospective users are advised to run the program using this data set, reproducing verbatim the example responses quoted in this guide.

INPUT DATA FORMAT

The input data file should be in the same format as in the supplied synthetic scattering data (file: BIMODAL.SAS). Formally the data should exist in the file as ordered triples of scattering vector (in 1/A), intensity and estimated error of intensity. Typically, the data will be in three columns, separated by "white space" of spaces and/or tabs. The intensity may be in any units as the program will ask for a conversion factor between these units and 1/cm units. The numbers are read as floating point numbers, hence 0.0015, 1.5E-3, 0.15E-2, and 15E-4 will be identical. However, do not be tempted to force double precision input (such as 1.5D-3) because this may provoke a nasty error condition.

The name of the data set should any standard filename for the operating system on which the program is run. A typical name would be BIMODAL.SAS (the synthetic scattering described earlier) which describes bimodal SAS data. If the data file resides in a directory other than the default, then you will have to specify that as part of the file name. Example file names, including a "full path description" follow for the more popular operating systems:

Digital Equipment Corporation VAX running VMS:

```
DISK$MPD_USERS:[CULVERWEL.MAXE]BIMODAL.SAS
```

Apple Macintosh:

```
Hard Drive:MaxSas Folder:test case:BIMODAL.SAS
```

MS-DOS computer (such as the IBM-PC):

```
C:\MAXSAS\TEST\BIMODAL.SAS
```

OUTPUT FORMAT

One output file is created by the program - a data file whose name is user supplied (e.g. BIMODAL.MAX).

The first few lines of output of a typical output file are as follows:

```
Results of maximum entropy analysis of SAS
   version 3.1 (PRJ)                , edited:7 February 1990
```

```
input file:    Bimodal.Sas
output file:   Bimodal.Out
-----------------------------------------

N(D) dD is number of particles/cm**3
   whose size is between D and D + dD

       D, A      V(D)*N(D), 1/A          N(D), 1/A/cm^3          dD, A
       ----      --------------          --------------          -----
      10.00         8.81630E-15             1.68379E+07            10.0000
      20.00         2.16369E-14             5.16543E+06            10.0000
      30.00         6.36171E-14             4.49999E+06            10.0000
```

The four columns are separated by TABs and also spaces.

term	description
D	dimension of scatterer in Angstroms
V(D)*N(D)	volume-weighted differential number distribution
N(D)	differential number distribution
dD	width of the bin in Angstroms in terms of “D”

V(D)*N(D) is the distribution solved by the program. N(D) is the distribution most often reported by other analysis techniques (TEM, mercury porosimetry, etc.). N(D) is calculated from the second one by dividing by the particle volume. This table continues as:

```
     380.00         8.11590E-07             2.82480E+10            10.0000
     390.00         5.74524E-07             1.84976E+10            10.0000
     400.00         3.71052E-07             1.10728E+10            10.0000




     Q 1/A           I 1/cm         ^I 1/cm         dI 1/cm               z
     -----           ------         -------         -------            ----
7.4936E-03       2.1200E+00      2.2330E+00      1.6700E-01       -0.676923
9.9975E-03       1.9000E+00      1.9596E+00      1.0200E-01       -0.584780
1.2501E-02       1.7840E+00      1.6605E+00      6.6900E-02        1.845438
```

These columns are also separated by a combination of spaces and tabs

term	description
Q	input Q-vector in 1/Angstrom units
I	input intensity in 1/cm units
^I	intensity calculated from the distribution above
dI	input error in 1/cm, scaled by the error scaling factor
z	standardized residual, z = (I - ^I)/dI

```
 9.2452E-02       4.3770E-03      4.2886E-03      1.6700E-04        0.529397
 9.5008E-02       3.4510E-03      3.7178E-03      1.5000E-04       -1.778752
 9.7564E-02       3.5330E-03      3.2077E-03      1.3500E-04        2.409922
 9.9937E-02       2.9560E-03      2.7793E-03      1.2300E-04        1.436465


Input data: Bimodal.Sas
Contrast =       1.0000000 x 10^28 m^-4.
 spheroid: D x D x D*   1.00000000000000
Data conversion factor to 1/cm =  1.00000E+00
```

After all the intensities have been printed, the same summary appears in the output file as appeared on the screen shown above. Only the first few lines are shown here.

The program creates two extra pseudo-particle sizes, one that is "smaller" than the smallest bin in the distribution and one that is "larger" than the largest bin in the distribution. The scattering from these pseudo-particles is approximated at low angles by the Guinier relation and at high angles by the Porod relation. The object of the user is to define the dimension range large enough that neither of these pseudo-particles develops any significant volume fraction. The percentage of the distribution that was assigned to each of these sizes is reported. Provided that both are small compared to the total volume fraction of scatterers (also given in the summary) then the program has detected essentially all of the particles contributing to the observed scattering. If they are not, it may be worth re-running the program with a different diameter range in order to detect this extraneous volume fraction.

EXAMPLE OF THE PROGRAM EXECUTION ON A DEC Vax

The following is an excerpt from the execution of MaxSas as it is analyzing the supplied test distribution BIMODAL.SAS, a bimodal distribution with two Gaussian peaks: one at 80 Angstroms with a sigma of 20 A and the other at 200 A with a sigma of 60 A. The Gaussian at 200 A is half the height of the one at 80 A. For a further discussion of this distribution, see [Culverwell, 1986]. The exact volume fraction of the original distribution was not specified.

In the section to follow, all user responses will be all in upper case where appropriate and will be followed by a {CR}, signifying that the user has pressed the return key. {CR} by itself signifies that the user has accepted the default answer to the question, as shown in <default>. Almost all questions have a default response, the only exception being the output file names which may take any value EXCEPT blank or the same as the input file name. Initially, all the defaults are preset to the proper answers for the supplied test distribution BIMODAL.SAS. If you type in a new value, that will value will become the default the next time that the question is asked.

The example to follow will be interrupted from time to time for explanations. These will be isolated between rows of "====" signs. For further explanations of each question which is asked, see the appropriate section appearing elsewhere in this document.

Now here's the excerpt from the beginning of the run. Remember that all user responses are on a single line and are terminated with a {CR}.

```
$ RUN MaxSas{CR}

Size distributions from SAS data using the maximum entropy criterion
   version: 3.1 (PRJ)                ,   7 February 1990
 Input file? <Quit>
Bimodal.Sas{CR}
 Output file?
Bimodal.Out{CR}
 Minimum q-vector? [1/A] <  1.000000000000000E-008>
{CR}
 Maximum q-vector? [1/A] <   100.000000000000     >
{CR}
 Scattering contrast? [10^28 m^-4] <   1.00000000000000     >
{CR}
 Factor to convert data to 1/cm? <   1.00000000000000     >
{CR}
 Error scaling factor? <   1.00000000000000     >
{CR}
 Background? <  0.000000000000000E+000>
{CR}
 Spheroids: D x D x vD,  Aspect ratio (v)?  <   1.00000000000000     >
{CR}
 Bin step scale? (1=Linear, 2=Log) <           1>
```

```
{CR}
 Number of histogram bins? <          40>
{CR}
 Maximum value of D? [A] <   400.000000000000     >
{CR}
 Minimum value of D? [A] <   10.0000000000000     >
{CR}
 Maximum number of iterations? <          20>
{CR}
 Reading from file: Bimodal.Sas
         38 points were read from the file
         38 points were selected from the data
 Preparation of the GRID function...
 Setting BASE constant at   1.000000000000000E-012
 MaxEnt routine beginning ...
```

To recap what has happened so far, the program was started and the input data file BIMODAL.SAS was specified for analysis. All the default answers appeared to be acceptable so the program read 38 points from the file, of which all 38 points were retained for the analysis. The program informs you that it has set an internal array called BASE to 1.0E-12. BASE is the initial guess for the distribution. If there is a value in the output distribution that is comparable with this number, then that particular histogram bin has no significant information. Consider then that BASE is the "featureless" distribution and rest assured that it is quite flat.

At this point, the program has gotten all the adjustable parameters that it needs and is now proceeding to attempt to solve the problem.

The fitting routine is an iterative one. If there are N histogram bins and P data points then the computation time for each iteration will be on the order of (very approximate) Order(2N+P). (This is why both N and P are bounded.)

All sorts of information will begin to scroll on the screen at an alarming rate as the Maximimum Entropy routine, or MaxEnt, (see [Skilling, 1984] for details on what's happening) begins to extract statistically significant information from the SAS data. With each iteration, several different types of screen plots are drawn, each describing the data extracted so far. The different plots are titled:

- LOG (ChiSq) vs. iteration number
- Entropy vs. iteration number
- Residuals
- Distribution

The first two plots will not appear until the third iteration. They are intended to keep you informed about the progress of the MaxEnt routine. The residuals (difference between MaxEnt fit and input data normalized to the input errors) are plotted as a function of data point subscript number, not Q-vector. The distribution is weighted by the bin width, dD(i) = D(i+1) - D(i), and is also plotted as a function of diametral bin subscript number. All plots will be scaled to fit within the screen boundaries.

The last information to appear in each iteration will be a report, such as:

```
 #           2 of           20,   n  =           38
test =   0.19101,  Entropy =    3.4668138
 SQRT((Chi^2)/n):  target =  12.14204774     % off =    26.2698
        f-vector:     sum =   0.00452313  % change =     0.4663
```

The report is saying that for the second iteration out of 20, where there are 38 intensity data points, the difference between the entropy and ChiSquared gradients is 0.19101, the entropy of the distribution just plotted is 3.4668138 (whose units are exact), the target for the SQRT((ChiSquared)/n), where ChiSquared is derived from intensity calculated from the distribution just plotted, of 12.14204774 was missed by 26.2698%, and the total volume fraction of

scatterers in the distribution just plotted is 0.452313 %, which did not change much from the previous iteration. The iterations will continue.

Here is the screen output from the last iteration:

```
 LOG (ChiSq) vs. iteration number
          1 point(s) per column
 0.405168291335942      units per row
 -----------
|           |
|OO         |
|  O        |
|   O       |
|           |
|    O      |
|           |
|     O     |
|      O    |
|       O   |
|           |
|         O |
|          O|
|           |
|==========O|
 -----------

 Entropy vs. iteration number
          1 point(s) per column
 5.229508533621462E-002 units per row
 -----------
|===========|
|           |
|           |
|O          |
|           |
|           |
| O         |
|           |
|           |
|           |
|  O        |
|           |
|        OOO|
|      OO   |
|   OOO     |
 -----------

 Residuals
          1 point(s) per column
 0.315439585860872      standard deviations per row
 ---------------------------------------
|                                      O |
|                                        |
|  OO                           O        |
|                                        |
|                                       O|
|====O==O======O====O=====O=============|
|                         O   O      O   |
|          OO               O           |
```

```
|     O             O  O          O      |
|        OO   O   O      O   OO          |
|OO             OO   O O                 |
|======O=========================OO====|
|                                        |
|                                        |
|            O                       O   |
 ----------------------------------------

 Distribution
          1 point(s) per column
 6.975972282206744E-005 units per row
 ----------------------------------------
|      O                                 |
|                                        |
|                                        |
|       O                                |
|                                        |
|                                        |
|                                        |
|                                        |
|        O                               |
|                  OO                    |
|     O   OOOO    O  O  O                |
|             O  O    OO OO              |
|              OO                        |
|                         OO             |
|OOOOO=====================OOOOOOOOOOOOO|
 ----------------------------------------
 #          11 of           20,  n  =           38
test =   0.01587,  Entropy =    3.1391598
 SQRT((Chi^2)/n):  target =   1.00000000     % off =     0.0504
        f-vector:     sum =   0.00809492  % change =     4.0445
```

The problem has been solved in 11 iterations of the MaxEnt routine. The two criteria for solution are that TEST <= 0.05 (5%) and that the SQRT((Chi^2)/n) target be met within 0.5%. Observe that the volume fraction has not changed very much from the previous iteration.

Here is the summary screen output of the analysis:

```
 Input file: Bimodal.Sas
 Volume weighted size dist.: V(r)N(r) versus r
  2.63513513513514      units per column
 1.395194456441349E-005 units per row
 ------------------------------------------------------------------------
|           O                                                            |
|                                                                        |
|                                                                        |
|             O                                                          |
|                                                                        |
|                                                                        |
|                                                                        |
|                                                                        |
|               O                                                        |
|                                  O O                                   |
|         O       OO O O         O    O     O                          |
|                        O     O        O O   O O                        |
|                          O O                                           |
```

```
|                                                    O O                       |
|OO O O O                                                O OO O O O O O O O O O|
 ------------------------------------------------------------------------------
 standardized residuals vs. point number
          1 point(s) per column
 0.315439585860872      standard deviations per row
 -------------------------------------
|                                    O |
|                                      |
|  OO                           O      |
|                                      |
|                                     O|
|====O==O======O====O=====O============|
|                         O   O      O |
|          OO              O           |
|     O             O  O         O     |
|        OO   O   O      O   OO        |
|OO             OO   O O               |
|======O=========================OO====|
|                                      |
|                                      |
|            O                      O  |
 -------------------------------------

Input data: Bimodal.Sas
Contrast =       1.0000000 x 10^28 m^-4.
 spheroid: D x D x D*   1.00000000000000
Data conversion factor to 1/cm =  1.00000E+00
Error scaling factor =  1.00000E+00
Histogram bins are distributed in an increasing algebraic series.
Minimum particle dimension D =        10.00 A.
Maximum particle dimension D =       400.00 A.
Number of histogram bins =   40.
Maximum number of iterations allowed =   20.
Program left MaxEnt routine after   11 iterations.
Target chi-squared (# data points) =    38.
Best value of chi-squared achieved =    38.038279.
Entropy of the final distribution =    3.1389744.
Entropy of    a flat distribution =    3.6888795.
Total particles  =     1.57329E+16 per cubic cm.
Total volume fraction of all scatterers =     0.008094741.
Part of distribution smaller than        10.00 A =     0.00000000%.
Part of distribution  larger than       400.00 A =     0.00215387%.
Volume-weighted   mode D value =     70.00000 A.
Volume-weighted   mean D value =    153.25044 A.
Volume-weighted std. deviation =     72.92106 A.
Number-weighted   mode D value =     70.00000 A.
Number-weighted   mean D value =     83.89447 A.
Number-weighted std. deviation =     33.47500 A.
Minimum Q-vector =   7.4935700E-03 1/A.
Maximum Q-vector =   9.9937470E-02 1/A.
   User-specified background =        0.000000000 input data units
        Suggested background =        0.000064250 input data units
StDev of shift in background =        0.000330552 input data units
 New background should give ChiSq =    36.6534794792953
```

By now, the tabular data of the size distribution and the intensity fit have been written to the data file BIMODAL.OUT. The summary analysis of the distribution has also been written to the output file.

The program has found about 0.8% by volume of scatterers. The ChiSquared matches the number of intensity points (a requirement for solution) and the background suggested is not far from the background used (with respect to the sigma of the last intensity of 0.000123 1/cm). Observe that the standard deviation of the suggested shift in the background is much larger than the suggested shift.

It appears that we have a reasonable solution in hand. Don't be too hasty to believe it yet. In order to test the stability of the answer, analyze the BIMODAL.SAS data again, using all the same parameters as before (just take the defaults) except use the background (0.000064251 1/cm) suggested by MaxSas.

Therefore, you should respond affirmatively to the Stability Check question. Note the default answer on the Stability Check question is "N".

```
  The change in ChiSquared should be < 5%.
  Run the Stability Check? (Y/<N>)
Y{CR}
  Setting BASE constant at    1.000000000000000E-12
  MaxEnt routine beginning ...
```

For this documentation, the results of the stability check will not be shown. There is not much change in the volume fraction after the stability check (about 1% or so) for the supplied test distribution.

After the Stability Check, you should get this question again. This time, respond like the following to quit the program.

```
  The change in ChiSquared should be < 5%.
  Run the Stability Check? (Y/<N>)
{CR}

 The program is finished.
 The output file is: BIMODAL.MAX

Size distributions from SAS data using the maximum entropy criterion
   version: 3.1 (PRJ)                ,   7 February 1990
  Input file? <Quit>
{CR}

$
```

User Interaction with the program

A FEW WORDS ABOUT NUMERICAL RESPONSES BY THE USER

If you respond to a numerical question with a "zero", the default answer will be used. That is the way this program works to give you default answers. If you want to set a parameter to be "zero", use an infinitesimal value such as 1.0E-25.

All floating point responses should include a decimal point somewhere in the mantissa of the response, otherwise the results are unpredictable and very system dependent!

EXPLANATION OF QUESTIONS ASKED BY THE PROGRAM

Q: Input file? <Quit>

The input file contains the SAS intensity data as ordered triples of Q-vector (in 1/A units), Intensity (in arbitrary units), and statistical error of intensity (same as intensity units). Note that there are no initial header lines in the input file. No more than the first 300 data points (ordered triples) will be read from the input file.

If you were to press {CR} without typing in a file name, the program would quit (as indicated by the default).

If the input file does not exist, the program will happily proceed to ask you all the remaining questions it has. Then and only then will it find out that the file you named does not exist. This will generate a program crash.

A suggestion for input file name extensions is ".SAS" but this only a suggestion. The input file name may be up to 80 characters long.

Q: Output file?

This is the only question which has no default answer. You must answer this question with something. If your answer is the same as the input file name, the program will start over asking you for the input file name. This may be used as an easy exit if you specified the wrong name.

The program does not check to see if the named output file already exists. On some systems (Macintosh and MS-DOS), the old file will be erased and a new file created. On other systems (VAX), a file with the same name but a new version number will be created. Forewarned is forearmed.

My suggestion for MaxSas solution file name extensions is ".MAX" or ".DIS". The output file name may be up to 80 characters long.

Q: Minimum (Maximum) q-vector? [1/A]

Use Q-vector (actually Q-vector magnitude) in units of 1/Angstrom. The user is allowed to exclude data points from the ends of the input data. Only those data points satisfying qMin <= Q-vector <= qMax will be analyzed. The program is designed to only handle positive Q-vectors.

Initially, qMin is set ridiculously low so that even the lowest data point will be used. Correspondingly, qMax is set high enough to include all typical SAS Q-vectors.

The user should generally cut off the data when the signal-to-noise ratio becomes poor. Truncating earlier than this will lose information about the smallest particles present in the sample. Users might note that it is not neccessary for all the intensity values to be positive, although it is probably inadvisable to include more than five negative ones.

Q: Scattering contrast? [10^28 m^-4]

The scattering contrast is the squared difference between the scattering length density of particle and matrix. If the contrast is 1.27E30 1/m**4, then enter the value 127.0. By the way, 1.E28 1/m**4 = 1.E20 1/cm**4.

The user can either enter the true contrast here or reply {CR}, in which case the final "volume fractions" obtained will have to be divided by the contrast (in units of 1.E28 1/m**4) in order to obtain genuine volume fractions. The program is coded to accept scattering contrast values no larger than one million units of 1.0E28 (1.0e34) 1/m**4.

Q: Factor to convert data to 1/cm?

If the intensity values in the input file were not in units of 1/cm, enter the constant to convert them into such units. If they were already in 1/cm units, good for you, so just press {CR} to accept the default. The program is coded to accept conversion values no larger than 1000.0.

Q: Error scaling factor?

Here is an opportunity for you to try analyzing your data with different ratios of signal to noise. If you think that the errors in the input file were underspecified, you may multiply them by this constant. More on this later as this will have a major influence upon the analysis.

Q: Background?

This program has left you the opportunity to subtract a constant intensity value. A good initial approximation will put you on the road to a good analysis of the data. Remarkably, the background may take any value, positive or negative. If you want to set the background back to zero, use infinitesimal (such as 1.E-25) rather instead. More on background later.

Q: Spheroids: D x D x vD, Aspect ratio (v)?

The scattering form factor currently implemented is that discussed by [Roess, 1947]. A special case of this ellipsoid of revolution, whose outside dimensions are D x D x vD, is the sphere whose form factor is described in [Culverwell, 1986] and [Potton, 1988a].

It is possible to select any aspect ratio (within reason) using this model and the program only checks to see that you have entered a positive value. Special care has been taken to ensure that the volume fractions determined by this model are correct.

For a full explanation of the coding of this model (from eq. 4, 5, & 6 of [Roess, 1947]), see the source code listing. Look for the routine named "Spheroid."

Remember that the distributions that are output are in terms of the dimension "D". The volume of this type of spheroid is (4Pi/3) v r**3.

Q: Bin step scale? (1=Linear, 2=Log)

"Linear" binning means that the diametral bins will increase in size according to an algebraic series (e.g. 1, 2, 3, ...). The other method currently available is "logarithmic" binning where the increase is according to a geometric series (e.g. 1.0, 1.05, 1.1025, ...). Use whichever method gives you a sufficient number of points over all the peaks in the distribution. Be aware that the calculated volume fractions and number densities for the first few bins on the "log scale" are likely to be artificially high because of the small bin width and small particle volume corresponding to that bin (both these terms divide the quantity that MaxSas derives to give you the volume fraction").

The width of each bin indexed by "i" is dD(i) = D(i+1) - D(i) so that the number density of scatterers whose size is between D and D + dD is truly N(D) dD. The bin width appears in the output file as "dD."

Q: Number of histogram bins?

This is an integer between 2 and 100, limited by computer memory and execution CPU time. Use as few bins as you think you need to adequately describe the distribution or as many bins as you want, up to the maximum of 100.

Q: Maximum (Minimum) value of D? [A]

Use Angstrom units. Because each intensity is a statistical representation of ALL dimensions D in the sample, weighted by a particular form factor (model function), the choice of maximum and minimum D is left to the user. You may specify values that are beyond the "peripheral vision" of your data to see if there is any statistical support for such sizes in your data. Usually, one knows something about the size distribution to be solved and a maximum particle diameter can be estimated. Ideally Dmax should be an over-estimate; if too small a diameter range (Dmin to Dmax) is specified, the program will likely fail.

The largest value for Dmax is something unreasonable for most SAS data (1 million Angstroms). If you try to exceed this limit, the program will patiently ask you again for the maximum D value. The smallest Dmin value you may enter is 1.0 Angstrom. The program will always suggest Dmin = Dmax / (number of bins).

If Dmin >= Dmax, the program will start asking you questions all over. You can use this as an easy way to correct a bad input prior to this question, without having to stop and restart the program.

Q: Maximum number of iterations?

The number of iterations is best estimated by experience. Skilling and Bryan [Skilling, 1984] suggest that one should re-consider the model if more than about 20 iterations are required for convergence within the Maximum Entropy routine (MaxEnt). The largest allowed number of iterations is 200 but if you require this, your model is probably not representing the data well. The MaxEnt routine may not require as many iterations as you specify. That just means the job was easier than you "thought".

If, while the MaxEnt routine is iterating, you see that a few more iterations will be required to achieve a satisfying solution than you have specified here, all is not lost. If the limit specified is reached with no satisfying maximum entropy solution yet in hand, the program will ask you if you want to iterate more. You can then extend the process. For this reason, it is suggested that you specify a lower value (rather than higher) so that you may check the program's progress. A low limit allows the MaxEnt routine to escape should the fitting process fail to converge. In such an event, one or more of the input parameters should be adjusted to achieve a more harmonius solution.

A good general suggestion for the number of iterations is the maximum number that you are willing to see the MaxEnt routine perform and not converge. If the MaxEnt routine needs more iterations, it will ask you for permission.

Q: The change in ChiSquared should be < 5%.

```
Run the Stability Check? (Y/<N>)
```

The Stability Check will perform the same analysis on the data set with all the same parameters except that the suggested background will be used. If the answer is stable, then all the results should be the same. If the answer is unsteady, then things will look different in some way. The prompt for a stability check will not appear unless the program calculates that the shift should produce less than a 5% change in the ChiSquared.

Q: Maximum iterations have been reached.:

```
How many more iterations? <none>
```

This question occurs inside the MaxEnt routine when the maximum number of iterations that you specified have been reached. If you want the MaxEnt routine to keep trying, specify a positive integer, otherwise take the default which will generate the following output:

```
No convergence! # iter. = "IterMax"
File was: "InFile"
```

The program will then start over at the first question.

SCREEN PLOTS

LOG (ChiSq) vs. iteration number

This plot will appear after the second iteration of the MaxEnt routine. If the ChiSquared is nearly constant for 3 or more consecutive iterations, this plot will not appear. The "====" bar in the plot indicates the target value of "N", the number of intensity points.

Entropy vs. iteration number

This plot will appear for every iteration after the second. The "====" bar in the plot indicates the entropy of a flat distribution with the same number of diametral bins as have been specified.

Residuals

The standardized residuals are the difference between the intensity that is calculated from the distribution and the input intensity, all divided by the input error. For the model to fit the data well, this plot should look featureless (a.k.a. random). The "====" bars are at +1 and -1 standard deviations. 67% of the points should fit within the bars. If there is some systematic difference beween the model and the data, the residuals will reveal it by showing some shape.

Distribution

The distribution plot appears at the end of each iteration and shows the most recent distribution, whose calculated intensity is to be compared with the input intensity and errors. The values in this plot are weighted (multiplied) by the bin width. This means that when the bins are distributed in a geometric series, it will be quite difficult for the user to see a small peak at smaller diameters in this plot. Have no fear though because this method weights the volume fraction in a manner equal to that of the algebraic series.

Volume weighted size dist.: V(r)N(r) versus r

Once the MaxEnt routine has decided that it has a solution, this plot will appear. The vertical scale is the volume distribution (technically the "volume-weighted differential number distribution"). This is almost the same value as was plotted in the "Distribution" plot except the bin width has been divided out. The horizontal scale is a linear axis on which is plotted particle radii.

Note: The MaxEnt routine does its work with respect to particle radii. All answers are properly scaled to diametral units in the output files. Additionally, the MaxEnt routine works with intensities in 1/m units. It seems to do some

bad things when the intensities are in 1/cm. The FORTRAN code isolates the user from this eccentricacy. All unit conversions are corrected in the output data.

OBSERVATIONS, HINTS, SUGGESTIONS

Stability Check of the Solution

In the example case above, a second analysis of the test distribution was made to check the stability of the solution. This is a very good suggestion and is a must before you should present any data which you have analyzed. The stability check is made after a successful solution has been obtained by re-analyzing with no parameters changed except for the experimental background which the program suggests. If the answer is to be believed, the Stability Check should complete with a comparable number of iterations and determine a comparable background, volume fraction, and size distribution.

Getting the Background Close

It seems that there is a narrow thread on which the program may obtain a reasonable analysis (within 20 iterations). That thread has two adjustable parameters: error scaling and constant background. With a larger error scaling term, the exact value of the background is less important. If one is not certain of the background level (and some of the particle form models require a background different even from the experimental background), it can be very difficult to guess within the 10% or so required with an error scaling factor of unity.

An algorithm that seems to navigate that thread to an acceptable solution of a size distribution from a set of intensity data is as follows: Decide upon the aspect ratio and the largest range of dimensions that may exist in the data. Run the analysis, choosing all the data that you think will fit the model well. Specify the contrast if you know it. Increase the errors by a factor of 5.0 (or maybe 10. if conditions suggest). Take a guess at the background (the zero-order guess is zero). Let the MaxEnt routine try to solve the puzzle. If it does not converge within 20 iterations, increase the error scaling factor by double. Keep doing this until the MaxEnt routine says it has a "solution". Good! We are not interested in this solution because the residuals probably look like a smooth, curved function. What we are trying to do is get the program to tell us what it "thinks" the background should be. Now that the program has suggested a background to us, try analyzing again with this background and a slightly decreased error scaling factor. Now we are on the "thread". Keep bringing the error scaling factor down (I know this takes time) until you can be satisfied that the errors are well-specified or that there is some systematic reason why the model does not fit the data well. Whatever the background ends up as when you are satisfied with the error scaling factor, accept it and reanalyze the data again, leaving out any intensities that would be below that background.

The background suggested by the program is based on a statistically-weighted average of the difference between the intensity calculated from the distribution (^I) and the input intensity data(I). The exact equation looks like, where "s" are the input errors:

```
NewBkg = Bkg + AVERAGE( (I - ^I) / s**2 )
```

FAILURES

In ideal circumstances when the program is iterating successfully the user will observe the value of ChiSquared diminishing until it closely approaches its final target value, which is the total number of data points being used. Then in the final few iterations the entropy, which had been steadily decreasing, will be seen to increase. The residuals will become more randomly distributed with each iteration (a sign of a good fit to the data) and the size distribution will slowly converge to its final form. The program will then exit from the fitting routine. This is the behaviour observed when the program is run using the example data set.

It is quite possible for the fitting routine to fail to converge at the first attempt. If this happens the program will return to the calling routine after it has completed the maximum number of iterations specified in the input section above and display the following message:

```
No convergence! # iter. = "MaxIter"
File was: "InFile"
```

The program will then return to the input section to begin a new analysis.

There are a number of problems which can arise. Some of these are annoying bugs in the program which are gradually being sorted out. The usual symptoms of trouble are:

1. **the program suddenly suffers 'divide' or 'square-root'** execution errors
2. **it gets caught in a perpetual loop** (Hopefully, these errors have been trapped or corrected. The bulk of them are from passing a literal variable as a parameter to a subroutine or function. The size of the argument is implied by the caller but actually specified, sometimes differently, by the called subroutine or function. The error is then, "passing the wrong size argument on the stack" which has been corrected by setting a variable, of known size, to the value of the literal and then passing the variable.)

The following remedies should be considered:

1. **take out points at either end of the data to change the program's** calculation trajectory
2. change the size range or number of bins to have the same effect
3. **consider that ChiSquared is being pushed too hard so the error** scaling can be increased and the point of tragedy is never reached
4. adjust the constant background
5. **re-assess the DISTRIBUTION of assigned errors on the input** data – Do they reflect the true scatter ?
6. re-assess the particle form model with regard to the system
7. **is the scattering just too weak or noisy for a respectable program** like this one ?

Considerations (1)-(3) should resolve matters if you have encountered one of the program bugs; if not, then (4)-(7) may apply. In most situations, time spent adjusting the flat background seems to give the best return on effort expended. However it is worth considering a few points relevant to (6) above. The basic assumption is that the scattering system comprises a DILUTE assembly of identically shaped scattering particles, all of one kind, suspended in a uniform medium or matrix. Inter-particle interferences due to close packing at high concentrations are not presently considered. (J.E. Epperson is trying to develop methods for treating these.) Such inter-particle interferences are likely to result in run-failures or fictitious size distributions. A disordered interconnecting scattering interface within the sample may also lead to spurious results. Also the aspect ratio, cannot be determined from SAS data alone: if a size distribution can be obtained with one value, then size distributions should be equally obtainable over all realistic values for a given scattering particle type. Thus both the choice of shape function and the aspect ratio should be determined from independent methods such as electron microscopy, theoretical models etc.

ERROR SCALING

The most likely reason is that the quoted errors are too small to allow a close fit to the data by an algorithm that uses the ChiSquared test as its consistency criterion. This would probably be the case if, during the iterations, the user observed chi-squared asymptotically approaching a final value larger than the number of data points being used, the residuals becoming randomly distributed and the size distribution converging to a well behaved final form (that is to say, one that extends over more than one histogram bin, is not wildly oscillatory, and is small at either end of the diameter range). Should this occur, the easiest way to rectify it is to specify an error scaling factor that is greater than unity. An under-estimate of this factor is provided by the smallest value of SQRT(ChiSquared/N) ever achieved by the program. A reasonable error scaling factor would then be, say, 1.1 times this estimate. The user should note, however, that this device should not be abused; if the rescaled errors are much larger than their true values then statistically significant information from the scattering pattern is being thrown away. Any size distribution is consistent with data of infinite errors!

CONSTANT BACKGROUND

Another likely cause of a convergence failure is an incorrect constant background subtraction. If during the previous iterations the user observed a large spike (that was not expected or predicted) at the low diameter end of the size distribution then it is quite possible that there is a constant background remaining in the data (the program is interpreting the uniform intensity as the scattering from very small particles). Conversely, if the size distribution is unreasonably biased towards large particles then it is possible that too much background has been removed and the data is missing information about the smallest particles in the sample. In either of these cases the user should specify a different amount of background. The user is reminded that of all the input parameters, the constant background subtraction is the one that needs to be known most accurately (indeed, if this parameter is inaccurate by more than about 10% then the program will probably fail). So any length of time the user spends on a precise evaluation of the constant background is probably well spent.

OTHER PARAMETERS

From a study of the size distributions plotted during the iterations the user may be able to adjust some other input parameters in order to accelerate convergence. It might, for example, be clear that the first estimate of a maximum particle diameter was too large or too small (although if it was very far out in either direction the program would have crashed rather than simply failed to converge). And it might become clear that the size distribution can be adequately described by a histogram containing fewer bins than was originally thought. Judicious removal of some particularly doubtful data points (for example those which differ in magnitude from their neighbours by an extent far greater than their errors would suggest) is also possible, though this is unlikely to have a great effect on the convergence rate.

Precisely what to do in any particular case of convergence failure depends on experience of the program which can only be gained by experimenting with it. Prospective users, once they have analyzed the BIMODAL.SAS data set are urged to re-analyze it using different combinations of diameter range, number of histogram bins, Q-vector range, aspect ratio, error scaling, and constant background (both positive and negative) to see how these variations affect the execution of the program and the final volume fraction distributions that the program produces (if it produces any). The user should then be able to recognize when and why the program is failing in any particular fitting attempt and be able to eliminate the cause.

AN ALTERNATE TEST DISTRIBUTION: REVERSE.SAS (by P.R. Jemian)

A new, alternate test distribution, REVERSE, has been created by P.R. Jemian to test several key questions:

#1. How close can the program get to a known volume fraction?

> Note that there is no specification for the exact answer of total volume fraction for BIMODAL.SAS, only a normalized distribution [Culverwell, 1986].

#2. Does MaxSas handle data in the size range of a double-crystal intrument? #3. Do the solved distributions always look the same?

The distribution is (once again) two Gaussians in f(D) space where the Gaussian at lower diameter (1100 A, sigma = 300) is 25% the height of the other Gaussian (3400 A, sigma = 680). REVERSE.DIS is the starting distribution, from which is calculated the scattering (REVERSE.SAS) using the exact form factor for spheres. An artificial volume fraction of 1.5%, artificial scattering contrast of 10.0E28 1/m**4, an artificial background of 5.0 1/cm, and artificial random noise of 4% were added to the data. A summary of the analysis of the REVERSE.SAS dataset follows:

SUMMARY OF MAXSAS ANALYSIS OF REVERSE.SAS

term	analysis	actual
qMin	0.0005	—
qMax	0.025	—
NumPts	24	—
ChiSquared	23.972	—
Dmin	80	—
Dmax	8000	—
NumBins	100	—
flat entropy	4.605	—
entropy	3.925	—
Total vol. frac.	1.436%	1.5%
suggested background	4.78	5.0
vol-mean diameter	3231	3172
number-mean diameter	1181	905
error scaling factor	1.0	1.0

It appears that the spheres model can deliver the character of the correct distribution and volume fraction.

While the oscillations in the distribution suggest that there is statistical evidence for such irregular features, these cannot be believed as we know, a priori, the starting distribution and that distribution is smooth. We must conclude therefore that the entropy is not adequately maximized, subject to the constraint that ChiSquared equals the number of intensity points. While decreasing the maximum value allowed for TEST (currently set at 0.05) might seem to produce a better alignment between the entropy and ChiSquared gradients, a value as low as 0.0001 does not seem to alter the final entropy more than about 0.5%. A discussion with G.J. Daniell might bring us to resolve this point. Probably the oscillations have an origin in the introduction of the baseline "b" into the definition of the entropy as done by [Skilling, 1984]. This simplifies the math when calculating the entropy gradients but that probably makes the algorithm of [Skilling, 1984] very sensitive to gradients in the form factor.

One method to circumvent this unsightly "noise" in the solved distributions has been to replace the form factors that are defined with trig terms by ones defined by algebra. These approximations are only as good as the algebraic form can model the scattering and can render truly fictional volume fractions in the worst possible cases.

To answer, then, the three questions above, the volume fraction of the solution was very close to the actual volume fraction. The mean diameter was also very close, with the volume-weighted mean being the closest. The solved distribution was very close to the input distribution which differed dramatically in shape to the distribution of BIMODAL.SAS, hence the solved distributions do not always look alike. The range of diameters in the distribution for REVERSE.SAS was in the range of the double-crystal instrument and so that question can be answered affirmatively. The answers are also believable and so MaxSas is not limited by the experimental range of a particular type of scattering camera.

SOURCE CODE

This Maximum Entropy program was originally written in BASIC by G.J. Daniell (Department of Physics, Southampton University, UK) and later translated into FORTRAN and adapted for SAS analysis by J.A. Potton. Further modifications have been made by I.D. Culverwell, G.P. Clarke and A.J. Allen (UKAEA Harwell Laboratory,UK) and P.R. Jemian (Northwestern University, USA).

There is only one source code module, MaxSas.For. Compile and link it with the fastest floating point math that you can get your hands on.

Unfortunately, some data storage had to be placed in COMMON because of the limitation of the Language Systems MPW version 1.2.1 FORTRAN compiler for the Apple Macintosh. Because of this compiler's eccentricacies, there is

one compiler-dependent line of code very near the first executable statement. If you use this compiler, un-comment this line so that you get a chance to see the output. (Compiler dependence, ugh!)

As it stands on 7 February 1990, the code will now compile on:

- Digital Equipment Corporation VAX 11/785,VMS version 5.2
- Apple Macintosh, Language Systems FORTRAN v. 1.2.1
- Apple Macintosh, Microsoft (Absoft) FORTRAN v. 2.2 compiler
- MS-DOS (e.g. IBM-PC), Microsoft FORTRAN v. 5.0

Of course the program RUNS on these computers as well. Quite well!

Most of the comments in the source code have been added by P.R. Jemian. Where they exist, they are usually quite explanatory. Where they do not exist, consult the references of [Skilling, 1984] for the operation of MaxEnt.

Complete listing of MaxSas.for

```
1       PROGRAM MaxSAS
2       IMPLICIT REAL*8 (A-H,O-Z)
3       IMPLICIT INTEGER*4 (I-N)
4       CHARACTER*25 ProgVers, EditDate
5       PARAMETER ( ProgVers = '3.6 (PRJ)' )
6       PARAMETER ( EditDate = '11 February 1992' )
7   C   Analysis of small-angle scattering data using the technique of
8   C   entropy maximization.
9
10  C   Credits:
11  C   G.J. Daniell, Dept. of Physics, Southampton University, UK
12  C   J.A. Potton, UKAEA Harwell Laboratory, UK
13  C   I.D. Culverwell, UKAEA Harwell Laboratory, UK
14  C   G.P. Clarke, UKAEA Harwell Laboratory, UK
15  C   A.J. Allen, UKAEA Harwell Laboratory, UK
16  C   P.R. Jemian, Northwestern University, USA
17
18  C   References:
19  C   1. J Skilling and RK Bryan; MON NOT R ASTR SOC
20  C       211 (1984) 111 - 124.
21  C   2. JA Potton, GJ Daniell, and BD Rainford; Proc. Workshop
22  C       Neutron Scattering Data Analysis, Rutherford
23  C       Appleton Laboratory, UK, 1986; ed. MW Johnson,
24  C       IOP Conference Series 81 (1986) 81 - 86, Institute
25  C       of Physics, Bristol, UK.
26  C   3. ID Culverwell and GP Clarke; Ibid. 87 - 96.
27  C   4. JA Potton, GK Daniell, & BD Rainford,
28  C       J APPL CRYST 21 (1988) 663 - 668.
29  C   5. JA Potton, GJ Daniell, & BD Rainford,
30  C       J APPL CRYST 21 (1988) 891 - 897.
31
32  C   This progam was written in BASIC by GJ Daniell and later
33  C     translated into FORTRAN and adapted for SANS analysis.  It
34  C     has been further modified by AJ Allen to allow use with a
35  C     choice of particle form factors for different shapes.  It
36  C     was then modified by PR Jemian to allow portability between
37  C     the Digital Equipment Corporation VAX and Apple Macintosh
38  C     computers.
39  C   The input data file format is three columns of "Q I dI" which
40  C     are separated by spaces or tabs.  There is no header line
```

```
C     in the input data file.

    PARAMETER (cm2m = 0.01) ! convert cm to m units, but why?
    PARAMETER (MaxPts = 300, MaxBin = 102)
    PARAMETER (isLin = 1, isLog = 2, ioUnit = 1)

C  point-by-point mapping between reciprocal and real space
    COMMON /space1/ grid
    DIMENSION grid(MaxBin,MaxPts)

C  terms used in entropy maximization
    COMMON /space5/ chisq, chtarg, chizer, fSum, blank
    COMMON /space2/ beta, c1, c2, s1, s2
    DIMENSION beta(3), c1(3), c2(3,3), s1(3), s2(3,3)

C  terms used only by subroutine MaxEnt, allocated here to make memory tidy
    COMMON /space3/ ox, z, cgrad, sgrad, xi, eta
    DIMENSION ox(MaxPts), z(MaxPts)
    DIMENSION cgrad(MaxBin), sgrad(MaxBin)
    DIMENSION xi(MaxBin,3), eta(MaxPts,3)

C  space for the plotting frame, allocated here to make memory tidy
C    note the limits: MaxCol <= 100, MaxRow <= 150 (really large screens!)
    PARAMETER (MaxCol = 75, MaxRow = 15)
    PARAMETER (MxC2 = MaxCol+2, MxR2 = MaxRow+2)
    COMMON /space4/ screen, nCol, nRow, nCol2, nRow2
    CHARACTER*1 screen(100, 150)

C  space for main segment arrays
    DIMENSION q(MaxPts), datum(MaxPts), sigma(MaxPts)
    DIMENSION r(MaxBin), f(MaxBin), base(MaxBin), Qty(MaxBin)
    DIMENSION fit(MaxPts), BinWid(MaxBin)
    DIMENSION SkyFit(MaxPts), SkyDis(MaxBin)
    CHARACTER*40 InFile, OutFil
    LOGICAL Yes
    CHARACTER*1 YN, aTab

    DATA one, zero /1.0, 0.0/   ! compiler-independence!
    DATA hrDamp /5.0/   ! model 7&8: sets transition range
    DATA htDamp /0.9/   ! model 7: amount of damping
C  The value "hrDamp" sets the range where the transistion occurs.
C  The value "htDamp" sets the maximum proportion of damping.

C ... Define (initially) the default responses
    DATA iOption    /4/     ! usual form factor for spheres
    DATA Aspect /1.0/       ! particle aspect ratio
    DATA LinLog /isLin/     ! linear binning scale
    DATA n      /40/        ! number of bins
    DATA Dmin, Dmax /8.00, 400.0/   ! particle diameters
    DATA IterMax    /20/        ! maximum number of iterations to try
    DATA RhoSq  /1.0/       ! scattering contrast, x10**28 1/m**4
    DATA fac, err   /1.0, 1.0/  ! scalars for intensity and errors
    DATA qMin, qMax /1.e-8, 100./   ! range to accept
    DATA Bkg    /0.0/       ! intensity to subtract
    DATA sLengt /1.0E-20/   ! rectangular slit-length, 1/A
    DATA SkyBkg /1.0E-6/    ! the so-called "sky background" of [1]

C  Next line for MPW/Language Systems version 1.2.1, Macintosh only
```

```
C  Comment this out for other compilers
C  This is the only compiler-dependent line in this source code!!!!!!
C   CALL OutWindowScroll (1000)  ! for 1-line advance screen

    pi = 4. * ATAN(1.)
    aTab = CHAR (9)

C  screen dimension variables for plots, in COMMON /space4/
    nCol = MaxCol
    nRow = MaxRow
    nCol2 = MxC2
    nRow2 = MxR2

    1   WRITE (*,*)
    WRITE (*,*) 'Size distributions from SAS data using the',
     >              ' maximum entropy criterion'
    WRITE (*,*) '       version: ', ProgVers
    WRITE (*,*) '   Last edited: ', EditDate

    CALL GetInf (InFile, OutFil, iOption, Aspect, LinLog,
     >      n, Dmin, Dmax, IterMax, RhoSq, fac, err, qMin,
     >      qMax, Bkg, sLengt, SkyBkg, hrDamp, htDamp)
    IF (InFile .EQ. ' ') STOP

C   Read in the SAS data from the file "InFile"
    WRITE (*,*)  ' Reading from file: ', InFile
        OPEN (UNIT = ioUnit, FILE = InFile, STATUS = 'old')
    DO   j = 1, MaxPts
          READ (ioUnit, *, END = 95) q(j), datum(j), sigma(j)
    END DO
   95   npt=j-1     ! ignore any lines without an explicit EOL mark
        CLOSE (UNIT = ioUnit, STATUS = 'keep')
    WRITE (*,*) npt, ' points were read from the file'

C   Subtract background, convert to 1/m units and
C   shift for the selected data range
        i = 0
        DO   j = 1, npt
      IF (q(j) .GE. Qmin .AND. q(j) .LE. Qmax) THEN
        i = i + 1
        q(i) = q(j)
        datum(i) = fac * (datum(j)-Bkg) / cm2m
        sigma(i) = fac * err * sigma(j) / cm2m
      END IF
    END DO
    npt = i
    WRITE (*,*) npt, ' points were selected from the data'

C  PRJ: 24 May 1989
C   BinWid: actual radial width of the indexed bin number
C   Step:   radial increment factor (for geometric series)
C   rWid:   radial width (for arithmetic series)
    IF (LinLog .EQ. isLog) THEN ! geometric series
      Step = (Dmax/Dmin)**(1. / FLOAT(n-1)) - 1.
      rWid = 0.
    ELSE                ! arithmetic series
      Step = 0.
      rWid = 0.5*(Dmax - Dmin) / FLOAT(n-1)
```

```
    END IF
    r(1) = 0.5 * Dmin
    BinWid(1) = r(1) * Step + rWid
    DO   i = 2, n
      r(i) = r(i-1) + BinWid(i-1)
      BinWid(i) = r(i) * Step + rWid
    END DO

    WRITE (*,*) ' Preparation of the GRID function...'
C  Calculate the form-factor pre-terms
  111   IF (iOption .EQ. 1) THEN    ! Rods, using model of AJ Allen
      alphan1 = 2. * pi * Aspect
      alphan2 = 4. * pi
      preform = alphan1
      sLengt = 0.           ! "pinhole" collimation
    ELSE IF (iOption .EQ. 2) THEN   ! Disks, using model of AJ Allen
      alphan1 = 2. * pi / (Aspect**2)
      alphan2 = 2. * pi
      preform = alphan1
      sLengt = zero
    ELSE IF (iOption .EQ. 3) THEN   ! Globules, using model of AJ Allen
      alphan1 = 4. * pi * Aspect / 3.
      IF (Aspect .LT. 0.99) THEN    ! hamburger-shaped
        sqqt = SQRT (one - Aspect**2)
        argument = (2. - Aspect**2 + 2. * sqqt) / (Aspect**2)
        surchi = (one + Aspect**2 * LOG(argument) / (2.*sqqt) )
     >      / (2. * Aspect)
      ELSE IF (Aspect .GT. 1.01) THEN   ! peanut shaped
        sqqt = SQRT(Aspect**2 - one)
        argument = sqqt / Aspect
        surchi = (one + Aspect**2 * ASIN(argument) / sqqt)
     >      / (2. * Aspect)
      ELSE              ! spheroidal
        surchi = one
      END IF
      alphan2 = 6. * pi * surchi
      preform = alphan1
      sLengt = zero
    ELSE IF (iOption .EQ. 4) THEN   ! Spheres, delta-function
      alphan1 = 4. * pi / 3.
      alphan2 = 6. * pi
      preform = 9. * alphan1
      sLengt = zero
    ELSE IF (iOption .EQ. 5) THEN   ! Spheres, box-distribution
      alphan1 = 4. * pi / 3.    ! This model by PRJ
      alphan2 = 6. * pi
      preform = 48. * pi
      sLengt = zero
    ELSE IF (iOption .EQ. 6) THEN   ! smeared, spheroidal globs
      preform = 4. * Pi / 3.    ! This model by PRJ
      alphan1 = preform
      alphan2 = 6. * Pi
      Cgs = SQRT (3. * Pi)      ! for low-Q region
      Cps = 9. * Pi / 4.        ! for med. high-Q region
      Cp = 9. / 2.          ! for high-Q region
    ELSE IF (iOption .EQ. 7) THEN   ! spheroidal globs, no smearing
      preform = 4. * Pi / 3.    ! This model by PRJ
      alphan1 = preform
```

```
      alphan2 = 6. * Pi
      sLengt = zero
    ELSE IF (iOption .EQ. 8) THEN   ! smooth spheres
      preform = 4. * Pi / 3.    ! This model by PRJ
      alphan1 = preform
      alphan2 = 6. * Pi
      sLengt = zero
    END IF

C  alphaN1 is RhoSq * the particle volume
C  alphaN2 is RhoSq * the particle surface area / the particle volume
C   ... and later divided by q**4
    alphan1 = cm2m * alphan1 * rhosq * r(1)**3
    alphan2 = cm2m * alphan2 * rhosq / r(n)
    preform = cm2m * preform * rhosq

    DO   i = 1, n
      rCubed = r(i)**3
      DO   j = 1, npt
        Qr = q(j) * r(i)
        Qr2 = Qr**2
        IF (iOption .EQ. 1) THEN
          QH = q(j) * Aspect * r(i)     ! rod 1/2 - length
          topp = one + 2.*Pi* QH**3 * Qr / (9 * (4 + Qr**2))
     >           + (QH**3 * Qr**4) / 8.
          bott = one + QH**2 * (one + QH**2 * Qr)/9
     >           + (QH**4 * Qr**7) / 16
        ELSE IF (iOption .EQ. 2) THEN
          h = r(i)          ! disk 1/2 - thickness
          Rd = h / Aspect       ! disk radius
          Qh = q(j) * h
          QRd = q(j) * Rd
          topp = one + QRd**3 / (3. + Qh**2)
     >           + (Qh**2 * QRd / 3.)**2
          bott = one + QRd**2 * (one + Qh * QRd**2) / 16
     >           + (Qh**3 * QRd**2 / 3.)**2
        ELSE IF (iOption .EQ. 3) THEN
          topp = one
          bott = one + Qr**2 * (2. + Aspect**2) / 15.
     >           + 2. * Aspect * Qr**4 / (9. * surchi)
        ELSE IF (iOption .EQ. 4) THEN
          topp = (SIN(Qr) - Qr * COS(Qr))**2
          bott = Qr**6
        ELSE IF (iOption .EQ. 5) THEN
          Qj = q(j)
          rP = r(i) + BinWid(i)
          rM = r(i)
          bP = 0.5*rP + (Qj**2)*(rP**3)/6.
     >      + (0.25*(Qj * rP**2) - 0.625/Qj) * SIN (2.*Qj*rP)
     >      + 0.75 * rP * COS (2.*Qj*rP)
          bM = 0.5*rM + (Qj**2)*(rM**3)/6.
     >      + (0.25*(Qj * rM**2) - 0.625/Qj) * SIN (2.*Qj*rM)
     >      + 0.75 * rM * COS (2.*Qj*rM)
          topp = bP - bM
          bott = Qj**6 * (rP**4 - rM**4) * rCubed
        ELSE IF (iOption .EQ. 6) THEN
          rL = r(i) * sLengt
          topp = Cgs
```

```
          bott = rL*(one + Qr2/5. + Cgs/Cps * Qr**3)
     >          + Cgs/Cp * Qr**4
        ELSE IF (iOption .EQ. 7) THEN
C  The value "hrDamp" sets the range where the transistion occurs.
C  The value "htDamp" sets the maximum proportion of damping.
C  The weight is a "step" function with a broad edge.
          weight = htDamp * EXP (-Qr2/hrDamp**2) + (one - htDamp)
          topp = 3. * (SIN(Qr) - Qr * COS(Qr)) / Qr**3
          bott = 4.5 / Qr**4    ! bott=<topp**2> for large Qr
          topp = weight * topp**2 + (one-weight) / (one + one/bott)
          bott = one
        ELSE IF (iOption .EQ. 8) THEN  ! like #7 but smoother
          Qr2 = Qr**2
          weight = EXP (-Qr2/hrDamp**2)
          IF (Qr .LE. Pi) THEN
            topp = ((-1./45360.*Qr2+1./840.)*Qr2-1./30.)*Qr2+1./3.
          ELSE
            topp = 0.0
          END IF
          topp = (3*topp)**2
          bott = 4.5 / Qr**4
          topp = weight*topp + (1-weight)/(1 + 1/bott)
          bott = one
        END IF
        grid(i,j) = preform * rCubed * topp / bott
C       factors of 4Pi/3 are already included in "preform"
      END DO
    END DO

C   Attempt to account for scattering from very large and very small
C   particles by use of the limiting forms of grid(i,j).
    DO   j = 1, npt
      grid(n+1,j) = alphan1 ! next line accounts for a slit-length
      grid(n+2,j) = alphan2 / (q(j)**3 * SQRT(q(j)**2 + sLengt**2))
    END DO

C  Try to solve the problem
C  228  basis = 1.0e-6 / RhoSq      ! Originally was just 1.0e-6
    basis = SkyBkg      ! PRJ, 18.6.90
  228   CALL MaxEnt (n+2,npt, f,datum,sigma, basis,base, max,itermax)

C   "Max" counts the number of iterations inside MAXENT.
C   If Max < IterMax, then the problem has been solved.
    IF (max .GE. itermax) THEN
      WRITE (*,*) ' No convergence! # iter. = ', max
          WRITE (*,*) ' File was: ', InFile
      GO TO 1
    END IF

C   Otherwise, SUCCESS!... so calculate the SAS from the distribution
    CALL opus (n+2, npt, f, fit)
    CALL opus (n+2, npt, base, SkyFit)  ! fit the sky background, too!

C   ... and remove the bin width effect.
C   Also, calculate the total volume fraction, the mode, mean, and
C   standard deviations of the volume and number distributions.
    SumV   = zero
    SumVR  = zero
```

```
    SumVR2 = zero
    SumN   = zero
    SumNR  = zero
    SumNR2 = zero
    modeV = 1
    modeN  = 1
    DO   i = 1, n
      size = r(i)
      frac = f(i)
          pVol = 4*Pi/3 * (size * 1.e-8)**3 ! particle volume, cm**3
          IF (iOption .EQ. 1)  pVol = pVol * Aspect ! rods
          IF (iOption .EQ. 2)  pVol = pVol / Aspect ! disks
          IF (iOption .EQ. 3)  pVol = pVol * Aspect ! globs
      amount = (frac - SkyBkg) / pVol       ! number / cm**3
      IF (amount .LT. zero) amount = zero
      f(i) = frac / BinWid(i)
      base(i) = base(i) / BinWid(i)
      Qty(i) = amount / BinWid(i)
      IF (i .GT. 3) THEN            ! ignore 1st few bins
        SumN   = SumN   + amount
        SumNR  = SumNR  + amount * size
        SumNR2 = SumNR2 + amount * size**2
      END IF
      IF (Qty(i) .GT. Qty(modeN)) modeN = i ! get the mode
      SumV   = SumV   + frac
      SumVR  = SumVR  + frac * size
      SumVR2 = SumVR2 + frac * size**2
      IF (f(i) .GT. f(modeV)) modeV = i     ! get the mode
    END DO
    DnMean = 2.0 * SumNR / SumN
    DnSDev = 2.0 * SQRT((SumNR2 / SumN) - (SumNR / SumN)**2)
    DvMean = 2.0 * SumVR / SumV
    DvSDev = 2.0 * SQRT((SumVR2 / SumV) - (SumVR / SumV)**2)

    Entropy = zero
    DO   i = 1, n
      frac = BinWid(i) * f(i) / SumV    ! Skilling & Bryan, eq. 1
      Entropy = Entropy - frac * LOG (frac)
    END DO

C  Show the final distribution, corrected for bin width.

        WRITE (*,*)
    WRITE (*,*) ' Input file: ', InFile
    WRITE (*,*) ' Volume weighted size dist.: V(r)N(r) versus r'
    CALL Plot (n, r, f)

C   Estimate a residual background that remains in the data.
    Sum1 = zero
    Sum2 = zero
    DO  j = 1, npt
      weight = one / (sigma(j)**2)
      Sum1 = Sum1 + weight * (fit(j) -  datum(j))
      Sum2 = Sum2 + weight
    END DO
    shift = Sum1 / Sum2

C  Scale the data back to 1/cm units and calculate Chi-squared
```

```
    ChiSq  = zero
    Chi2Bk  = zero
    DO  j = 1, npt
      z(j)  = (datum(j) - fit(j)) / sigma(j)
      ChiSq   = ChiSq + z(j)**2
      Chi2Bk  = Chi2Bk + (z(j) + shift/ sigma(j))**2
      datum(j) = cm2m * datum(j)
      sigma(j) = cm2m * sigma(j)
      fit(j) = cm2m * fit(j)
      SkyFit(j) = cm2m * SkyFit(j)
    END DO
    shift = cm2m * shift / fac

    WRITE (*,*) ' standardized residuals vs. point number'
    CALL ResPlt (npt, z)

C  Let the file output begin!

    OPEN (UNIT = ioUnit, FILE=OutFil, STATUS='new')
    WRITE (ioUnit,*) ' Results of maximum entropy analysis of SAS'
    WRITE (ioUnit,*) '     version: ', aTab, ProgVers
    WRITE (ioUnit,*) '      edited: ', aTab, EditDate
    WRITE (ioUnit,*)
    WRITE (ioUnit,*) '  input file: ', aTab, InFile
    WRITE (ioUnit,*) ' output file: ', aTab, OutFil
    WRITE (ioUnit,*)
    WRITE (ioUnit, 35591) 'D, A', aTab, 'f, 1/A',
     >          aTab, 'Bkg f, 1/A', aTab, 'N dD, 1/A/cm^3'
35591   FORMAT (1X, A12, 3(A1, 1X, A15))

    DO  i = 1, n
      WRITE (ioUnit,3559) 2.*r(i), aTab, 0.5*f(i), aTab,
     >                0.5*Base(i), aTab, 0.5*Qty(i)
    END DO
 3559   FORMAT (1X, F12.2, 3(A1, 1X, 1PE15.5))


    WRITE (ioUnit, 1011) 'Q 1/A', aTab, 'I 1/cm', aTab,
     >          'fit I 1/cm', aTab, 'dI 1/cm', aTab,
     >          'SkyFit 1/cm', aTab, 'z'
 1011   FORMAT (///, A12, 5(1X, A1, A12))

    DO   j = 1, npt
      WRITE (ioUnit,560) q(j), aTab, datum(j), aTab, fit(j),
     >      aTab, sigma(j), aTab, SkyFit(j), aTab, z(j)
    END DO
  560     FORMAT (1PE12.4, 4(A1, E13.5), 1X, A1, 0PF12.6)

    WRITE (ioUnit,3301) aTab, InFile
    WRITE (*,3301) aTab, InFile
 3301   FORMAT (//' Input data: ', A1, A40)

    WRITE (ioUnit,3302) RhoSq
    WRITE (*,3302) RhoSq
 3302   FORMAT (' Contrast = ', F15.7,' x 10^28 m^-4.')

    IF (iOption .EQ. 1) THEN
      WRITE (ioUnit,*) ' rods: dia=D, length=D*', Aspect
```

```
      WRITE (*,*) ' rods: dia=D, length=D*', Aspect
    ELSE IF (iOption .EQ. 2) THEN
      WRITE (ioUnit,*) ' disks: thickness=D, dia=D/', Aspect
      WRITE (*,*) ' disks: thickness=D, dia=D/', Aspect
    ELSE IF (iOption .EQ. 3) THEN
      WRITE (ioUnit,*) ' globs: D x D x D*', Aspect
      WRITE (*,*) ' globs: D x D x D*', Aspect
    ELSE IF (iOption .EQ. 4) THEN
      WRITE (ioUnit,*) ' delta-function Spheres: diameter=D'
      WRITE (*,*) ' delta-function Spheres: diameter=D'
    ELSE IF (iOption .EQ. 5) THEN
      WRITE (ioUnit,*) ' box-function Spheres: diameter=D'
      WRITE (*,*) ' box-function Spheres: diameter=D'
    ELSE IF (iOption .EQ. 6) THEN
      WRITE (ioUnit,*) ' slit-smeared spheroidal globs: diameter=D'
      WRITE (*,*) ' slit-smeared spheroidal globs: diameter=D'
      WRITE (ioUnit,*) ' slit-length (1/A) = ', sLengt
      WRITE (*,*) ' slit-length (1/A) = ', sLengt
    ELSE IF (iOption .EQ. 7) THEN
      WRITE (ioUnit,*) ' spheroidal globs: diameter=D'
      WRITE (*,*) ' spheroidal globs: diameter=D'
    ELSE IF (iOption .EQ. 8) THEN
      WRITE (ioUnit,*) ' smooth spheres: diameter=D'
      WRITE (*,*) ' smooth spheres: diameter=D'
    END IF

    WRITE (ioUnit,53303) fac
    WRITE (*,53303) fac
53303   FORMAT (' Data conversion factor to 1/cm = ', 1PE12.5)

    WRITE (ioUnit,63303) err
    WRITE (*,63303) err
63303   FORMAT (' Error scaling factor = ', 1PE12.5)

    IF (LinLog .EQ. isLog) THEN
      WRITE (ioUnit,13304) 'geometric'
      WRITE (*,13304) 'geometric'
    ELSE
      WRITE (ioUnit,13304) 'arithmetic'
      WRITE (*,13304) 'arithmetic'
    END IF
13304   FORMAT (' Histogram bins are distributed in an increasing ',
     >      A10, ' series.')

    WRITE (ioUnit,3304) 'Minimum', Dmin
    WRITE (*,3304) 'Minimum', Dmin
    WRITE (ioUnit,3304) 'Maximum', Dmax
    WRITE (*,3304) 'Maximum', Dmax
 3304   FORMAT (1X, A7, ' particle dimension D = ',F12.2,' A.')

    WRITE (ioUnit,3306) n
    WRITE (*,3306) n
 3306   FORMAT (' Number of histogram bins = ',I4,'.')

    WRITE (ioUnit,3307) itermax
    WRITE (*,3307) itermax
 3307   FORMAT (' Maximum number of iterations allowed = ',I4,'.')

```

```
505      WRITE (ioUnit,3314) max
506      WRITE (*,3314) max
507   3314   FORMAT (' Program left MaxEnt routine after ',
508       *    I4,' iterations.')
509
510      WRITE (ioUnit,3308) npt
511      WRITE (*,3308) npt
512   3308   FORMAT (' Target chi-squared (# data points) = ',I5,'.')
513
514      WRITE (ioUnit,3309) ChiSq
515      WRITE (*,3309) ChiSq
516   3309   FORMAT (' Best value of chi-squared achieved = ',F12.6,'.')
517
518      WRITE (ioUnit, 33091) 'the final', Entropy
519      WRITE (*, 33091) 'the final', Entropy
520      WRITE (ioUnit, 33091) 'a flat', LOG (FLOAT (n))
521      WRITE (*, 33091) 'a flat', LOG (FLOAT (n))
522  33091   FORMAT (' Entropy of ', A9, ' distribution = ', F12.7,'.')
523
524      WRITE (ioUnit,33101) SumN
525      WRITE (*,33101) SumN
526  33101   FORMAT (' Total particles  = ', 1PE15.5,' per cubic cm.')
527
528      WRITE (ioUnit,3310) SumV
529      WRITE (*,3310) SumV
530   3310   FORMAT (' Total volume fraction of all scatterers = ',
531       *    F15.9,'.')
532
533      WRITE (ioUnit,3311) 'smaller', Dmin, f(n+1)
534      WRITE (ioUnit,3311) 'larger',  Dmax, f(n+2)
535      WRITE (*,3311) 'smaller', Dmin, f(n+1)
536      WRITE (*,3311) 'larger',  Dmax, f(n+2)
537   3311   FORMAT (' Volume fraction ',A7,' than ', F12.2,
538       *    ' A = ', 1PE13.5,'.')
539
540      WRITE (ioUnit,3411) SkyBkg
541      WRITE (*,3411) SkyBkg
542   3411   FORMAT (' Sky background (minimum ',
543       *    'significant volume fraction) = ', 1PE13.5,'.')
544
545      WRITE (ioUnit,3312) 'Volume', 'mode D value', 2.0 * r(modeV)
546      WRITE (*,3312) 'Volume', 'mode D value', 2.0 * r(modeV)
547      WRITE (ioUnit,3312) 'Volume', 'mean D value', DvMean
548      WRITE (*,3312) 'Volume', 'mean D value', DvMean
549      WRITE (ioUnit,3312) 'Volume', 'std. deviation', DvSDev
550      WRITE (*,3312) 'Volume', 'std. deviation', DvSDev
551      WRITE (ioUnit,3312) 'Number', 'mode D value', 2.0 * r(modeN)
552      WRITE (*,3312) 'Number', 'mode D value', 2.0 * r(modeN)
553      WRITE (ioUnit,3312) 'Number', 'mean D value', DnMean
554      WRITE (*,3312) 'Number', 'mean D value', DnMean
555      WRITE (ioUnit,3312) 'Number', 'std. deviation', DnSDev
556      WRITE (*,3312) 'Number', 'std. deviation', DnSDev
557   3312   FORMAT (1X, A6, '-weighted ', A14, ' = ', F12.5, ' A.')
558
559      WRITE (ioUnit,3313) 'Min', q(1)
560      WRITE (*,3313) 'Min', q(1)
561      WRITE (ioUnit,3313) 'Max', q(npt)
562      WRITE (*,3313) 'Max', q(npt)
```

```
 3313   FORMAT (1X, A3,'imum Q-vector = ', 1PE15.7, ' 1/A.')

    WRITE (ioUnit,3315) 'User-specified', Bkg
    WRITE (*,3315) 'User-specified', Bkg
    WRITE (ioUnit,3315) 'Suggested', Bkg - shift
    WRITE (*,3315) 'Suggested', Bkg - shift
 3315   FORMAT (1X, A14, ' background = ', F18.9,' input data units')

    WRITE (ioUnit,*) ' New background should give ChiSq = ', Chi2Bk
    WRITE (*,*) ' New background should give ChiSq = ', Chi2Bk

    CLOSE (UNIT=ioUnit, STATUS='keep')

C  Adjust the background default setting
C  Shift the intensity data just in case the user wants a Stability Check
C  Remember: background shifts down, intensity shifts up
C  Don't forget to put the data back into units of 1/m!
        Bkg = Bkg - shift
    DO   j = 1, npt
      datum(j) = (datum(j) + shift) / cm2m
      sigma(j) = sigma(j) / cm2m
    END DO

    IF (ABS ((Chi2Bk-ChiSq)/FLOAT (npt)) .LE. 0.05) THEN
      WRITE (*,*) ' The change in ChiSquared should be < 5%.'
 4000     WRITE (*, '(X,A,$)') ' Run the Stability Check? (Y/<N>)'
      READ (*,'(A1)') YN
      IF (YN .EQ. 'y'  .OR.  YN .EQ. 'Y') GO TO 228
      IF (YN .NE. ' ' .AND. YN .NE. 'n' .AND. YN .NE. 'N') GO TO 4000
    END IF

    WRITE (*,3200) OutFil
 3200   FORMAT (/,' The program is finished.', /,
     1    ' The output file is: ', A40)
    GO TO 1

 3199   STOP
    END


    SUBROUTINE GetInf (InFile, OutFil, iOption, Aspect, LinLog,
     >      nBin, Dmin, Dmax, IterMax, RhoSq, fac, err, qMin,
     >      qMax, Bkg, sLengt, SkyBkg, hrDamp, htDamp)
    IMPLICIT REAL*8 (A-H,O-Z)
    IMPLICIT INTEGER*4 (I-N)
    CHARACTER*40 InFile, OutFil
    PARAMETER (Ro2Max = 1.e6, ItrLim = 200, AbsMax = 1.e3)
    PARAMETER (DiaMin = 1., DiaMax = 1.e6, ErrMax = 1.e6)
    PARAMETER (MaxPts = 300, MaxBin = 102)
    PARAMETER (isLin = 1, isLog = 2)

    1   WRITE (*,'(X,A,$)') ' Input file? <Quit>'
      READ (*, 2) InFile
    2     FORMAT (A40)
      IF (InFile.EQ.' ') RETURN

    3   WRITE (*,'(X,A,$)') ' Output file?'
      READ (*, 2) OutFil
```

```
      IF (OutFil .EQ. ' ') GO TO 3
      IF (OutFil .EQ. InFile) GO TO 1

     suggest = qMin
    16   WRITE (*,'(X,A,G,A,$)') ' Minimum q-vector? [1/A] <',
     >          suggest, '>'
     READ (*, '(F10.0)') qMin
     IF (qMin .LT. 0) GO TO 16
     IF (qMin .EQ. 0) qMin = suggest

     suggest = qMax
    17   WRITE (*,'(X,A,G,A,$)') ' Maximum q-vector? [1/A] <',
     >          suggest, '>'
     READ (*, '(F10.0)') qMax
     IF (qMax .EQ. 0) qMax = suggest
     IF (qMax .LE. 0) GO TO 17
     IF (qMax .LE. qMin) GO TO 1

     suggest = RhoSq
    13   WRITE (*,'(X,A,G,A,$)')
     >    ' Scattering contrast? [10^28 m^-4] <', suggest, '>'
     READ (*, '(F10.0)') RhoSq
     IF (RhoSq .EQ. 0) RhoSq = suggest
     IF (RhoSq .LT. 0 .OR. RhoSq .GT. Ro2Max) GO TO 13

     suggest = fac
    14   WRITE (*,'(X,A,G,A,$)')
     >    ' Factor to convert data to 1/cm? <', suggest, '>'
     READ (*, '(F10.0)') fac
     IF (fac .EQ. 0) fac = suggest
     IF (fac .LE. 0 .OR. fac .GT. AbsMax) GO TO 14

     suggest = err
    15   WRITE (*,'(X,A,G,A,$)')
     >    ' Error scaling factor? <', suggest, '>'
     READ (*, '(F10.0)') err
     IF (err .EQ. 0) err = suggest
     IF (err .LE. 0 .OR. err .GT. ErrMax) GO TO 15

     suggest = Bkg
    18   WRITE (*,'(X,A,G,A,$)') ' Background? <', suggest, '>'
     READ (*, '(F10.0)') Bkg
     IF (Bkg .EQ. 0) Bkg = suggest

     Last = iOption
     4     WRITE (*,*) '  Select a form model for the scatterer:'
       WRITE (*,*) '  (See the User Guide for complete explanations)'
       WRITE (*,*) ' 1: rods        2: disks       3: globules'
       WRITE (*,*) ' 4: spheres (usual form)       ',
      >          '5: spheres (integrated)'
       WRITE (*,*) ' 6: spheroids (slit-smeared)   ',
      >          '7: spheroidal globs (not smeared)'
       WRITE (*,*) ' 8: smoothed spheres (not smeared)'
       WRITE (*,'(X,A,I3,A,$)')
      >      '  Which option number?  <', Last, '>'
       READ (*, '(I4)') iOption
       IF (iOption .EQ. 0) iOption = Last
       IF (iOption .LT. 1  .OR.  iOption .GT. 8) GO TO 4
```

```
      suggest = Aspect
    6   IF (iOption .GE. 1  .AND.  iOption .LE. 3) THEN
        WRITE (*,*) ' AR = Aspect Ratio, useful ranges are indicated'
        IF (iOption .EQ. 1) THEN
          WRITE (*,*) ' diameter D, length D * AR, AR > 5'
        ELSE IF (iOption .EQ. 2) THEN
          WRITE (*,*) ' thickness D, diameter D / AR, AR < 0.2'
        ELSE IF (iOption .EQ. 3) THEN
          WRITE (*,*) ' D x D x D * AR, 0.3 < AR < 3'
        END IF
        WRITE (*,'(X,A,G,A,$)')
       >      ' Aspect ratio?  <', suggest, '>'
        READ (*,'(F10.0)') Aspect
        IF (Aspect .EQ. 0) Aspect = suggest
        IF (Aspect .LT. 0) GO TO 6
      END IF

      suggest = sLengt
   61   IF (iOption .EQ. 6) THEN
        WRITE (*,'(X,A,G,A,$)')
       >      ' Slit-smeared globs.  Slit-length [1/A]? <',
       >      suggest, '>'
        READ (*,'(F10.0)') sLengt
        IF (sLengt .EQ. 0) sLengt = suggest
        IF (sLengt .LT. 0) GO TO 61
      END IF

      suggest = htDamp
   62   IF (iOption .EQ. 7) THEN
        WRITE (*,'(X,A,G,A,$)')
       >      ' spheroidal globs.  fraction of standard function? <',
       >      suggest, '>'
        READ (*,'(F10.0)') htDamp
        IF (htDamp .EQ. 0) htDamp = suggest
        IF (htDamp .LT. 0) GO TO 62
        IF (htDamp .GT. 1) GO TO 62
      END IF

      suggest = hrDamp
   63   IF (iOption .EQ. 7  .OR.  iOption .EQ. 8) THEN
        WRITE (*,'(X,A,G,A,$)')
       >      ' smoothed spheres.  Onset Qr value? <',
       >      suggest, '>'
        READ (*,'(F10.0)') hrDamp
        IF (hrDamp .EQ. 0) hrDamp = suggest
        IF (hrDamp .LT. 0) GO TO 63
      END IF

      Last = LinLog
    7     WRITE (*,'(X,A,I2,A,$)')
       >      ' Bin step scale? (1=Linear, 2=Log) <', Last, '>'
      READ (*, '(I4)') LinLog
      IF (LinLog .EQ. 0) LinLog = Last
      IF (LinLog .NE. isLin  .AND. LinLog .NE. isLog) GO TO 7

      Last = nBin
    8     WRITE (*,'(X,A,I4,A,$)')
```

```
     >      ' Number of histogram bins? <', Last, '>'
     READ (*, '(I4)') nBin
     IF (nBin .EQ. 0) nBin = Last
     IF (nBin .LT. 2 .OR. nBin .GT. (MaxBin-2)) GO TO 8

     suggest = Dmax
     9     WRITE (*,'(X,A,G,A,$)')
      >      ' Maximum value of D? [A] <', suggest, '>'
     READ (*, '(F10.0)') Dmax
     IF (Dmax .EQ. 0) Dmax = suggest
     IF (Dmax .LT. nBin*DiaMin .OR. Dmax .GE. DiaMax) GO TO 9

     Suggest = Dmax / FLOAT (nBin)
    11     WRITE (*,'(X,A,G,A,$)')
      >      ' Minimum value of D? [A] <', suggest, '>'
     READ (*, '(F10.0)') Dmin
     IF (Dmin .EQ. 0) Dmin = suggest
     IF (Dmin .GE. DMax .OR. Dmin .LT. DiaMin) GO TO 1

     IF (IterMax .GT. ItrLim) IterMax = ItrLim
     Last = IterMax
    12     WRITE (*,'(X,A,I4,A,$)')
      >      ' Maximum number of iterations? <', Last, '>'
     READ (*, '(I4)') IterMax
     IF (IterMax .EQ. 0) IterMax = Last
     IF (IterMax .LT. 0 .OR. IterMax .GT. ItrLim) GO TO 12

     Suggest = SkyBkg
    21     WRITE (*,'(X,A,G,A,$)')
      >      ' Sky background? (positive) <', Suggest, '>'
     READ (*, '(F10.0)') SkyBkg
     IF (SkyBkg .LT. 0) GO TO 21
     IF (SkyBkg .EQ. 0) SkyBkg = Suggest ! keep default

     RETURN
     END


     SUBROUTINE opus(n,npt,x,ox) ! solution-space -> data-space
     IMPLICIT REAL*8 (A-H,O-Z)
     IMPLICIT INTEGER*4 (I-N)
     PARAMETER (MaxPts=300, MaxBin=102)
     COMMON /space1/ grid
     DIMENSION x(MaxBin), grid(MaxBin,MaxPts), ox(MaxPts)
     DO   j = 1, npt
       sum = 0.
       DO   i = 1, n
        sum = sum + x(i) * grid(i,j)
       END DO
       ox(j) = sum
     END DO
     RETURN
     END


     SUBROUTINE tropus(n,npt,ox,x)   ! data-space -> solution-space
     IMPLICIT REAL*8 (A-H,O-Z)
     IMPLICIT INTEGER*4 (I-N)
```

```
      PARAMETER (MaxPts=300, MaxBin=102)
      COMMON /space1/ grid
      DIMENSION x(MaxBin), grid(MaxBin,MaxPts), ox(MaxPts)
      DO   i = 1, n
        sum = 0.
        DO   j = 1, npt
          sum = sum + ox(j) * grid(i,j)
        END DO
        x(i) = sum
      END DO
      RETURN
      END


      SUBROUTINE MaxEnt(n,npt, f,datum,sigma, flat,base,iter,itermax)
      IMPLICIT REAL*8 (A-H,O-Z)
      IMPLICIT INTEGER*4 (I-N)
      PARAMETER (MaxPts=300, MaxBin=102)
      DIMENSION f(MaxBin), datum(MaxPts), sigma(MaxPts)
      DIMENSION base(MaxBin)

      COMMON /space1/ grid
      DIMENSION grid(MaxBin,MaxPts)

      COMMON /space5/ chisq, chtarg, chizer, fSum, blank
      COMMON /space2/ beta, c1, c2, s1, s2
      PARAMETER (m = 3)       ! number of search directions
      DIMENSION beta(m), c1(m), c2(m,m), s1(m), s2(m,m)

      COMMON /space3/ ox, z, cgrad, sgrad, xi, eta
      DIMENSION ox(MaxPts), z(MaxPts)
      DIMENSION cgrad(MaxBin), sgrad(MaxBin)
      DIMENSION xi(MaxBin,3), eta(MaxPts,3)

      PARAMETER (TstLim = 0.05)   ! for convergence
      DATA one, zero /1.0, 0.0/   ! compiler-independence!

          WRITE (*,*) ' MaxEnt routine beginning ...'

      chizer = FLOAT(npt)
      chtarg = chizer
      blank = flat
      exp1 = EXP(one)

      IF (blank .EQ. zero) THEN
        DO  i = 1, n
          blank = blank + base(i)
          f(i) = base(i)  ! given initial distribution
        END DO
        blank = blank / FLOAT(n)
        WRITE (*,*) ' Average of BASE = ', blank
      ELSE
        WRITE (*,*) ' Setting BASE constant at ', blank
        DO  i = 1, n
          base(i) = blank
          f(i) = blank    ! featureless initial distribution
        END DO
      ENDIF
```

```
853
854     iter = 0
855     6   iter = iter + 1     ! The iteration loop begins here!
856     CALL opus (n, npt, f, ox)   ! calc. the model intensity from "f"
857     chisq = zero
858     DO  j = 1, npt
859       a = (ox(j) - datum(j)) / sigma(j)
860       chisq = chisq + a**2
861       ox(j) = 2. * a / sigma(j)
862     END DO
863     CALL tropus(n,npt,ox,cgrad) ! cGradient = Grid * ox
864     test = zero ! mismatch between entropy and ChiSquared gradients
865     snorm = zero    ! entropy term
866     cnorm = zero    ! ChiSqr term
867     tnorm = zero    ! norm for the gradient term TEST
868     fSum = zero ! find the sum of the f-vector
869     DO   i = 1, n
870       fSum = fSum + f(i)
871       sgrad(i) = -LOG(f(i)/base(i)) / (base(i)*exp1)
872       snorm = snorm + f(i) * sgrad(i)**2
873       cnorm = cnorm + f(i) * cgrad(i)**2
874       tnorm = tnorm + f(i) * sgrad(i) * cgrad(i)
875     END DO
876     snorm = SQRT(snorm)
877     cnorm = SQRT(cnorm)
878     a = one
879     b = one / cnorm
880     IF (iter .GT. 1) THEN
881       test = SQRT(0.5*(one-tnorm/(snorm*cnorm)))
882       a = 0.5 / (snorm * test)
883       b = 0.5 * b / test
884     ENDIF
885     DO  i = 1, n
886       xi(i,1) = f(i) * cgrad(i) / cnorm
887       xi(i,2) = f(i) * (a * sgrad(i) - b * cgrad(i))
888     END DO
889     CALL opus (n,npt,xi(1,1),eta(1,1))
890     CALL opus (n,npt,xi(1,2),eta(1,2))
891     DO  j = 1, npt
892       ox(j) = eta(j,2) / (sigma(j)**2)
893     END DO
894     CALL tropus (n,npt,ox,xi(1,3))
895     a = zero
896     DO  i = 1, n
897       b = f(i) * xi(i,3)
898       a = a + b * xi(i,3)
899       xi(i,3) = b
900     END DO
901     a = one / SQRT(a)
902     DO  i = 1, n
903       xi(i,3) = a * xi(i,3)
904     END DO
905     CALL opus (n,npt,xi(1,3),eta(1,3))
906     DO  k = 1, m
907       s1(k) = zero
908       c1(k) = zero
909       DO  i = 1, n
910         s1(k) = s1(k) + xi(i,k) * sgrad(i)
```

```
        c1(k) = c1(k) + xi(i,k) * cgrad(i)
      END DO
      c1(k) = c1(k) / chisq
    END DO
    DO  k = 1, m
      DO  l = 1, k
        s2(k,l) = zero
        c2(k,l) = zero
        DO  i = 1, n
          s2(k,l) = s2(k,l) - xi(i,k) * xi(i,l) / f(i)
        END DO
        DO  j = 1, npt
          c2(k,l) = c2(k,l) + eta(j,k) * eta(j,l) / (sigma(j)**2)
        END DO
        s2(k,l) = s2(k,l) / blank
        c2(k,l) = 2. * c2(k,l) / chisq
      END DO
    END DO
    c2(1,2) = c2(2,1)
    c2(1,3) = c2(3,1)
    c2(2,3) = c2(3,2)
    s2(1,2) = s2(2,1)
    s2(1,3) = s2(3,1)
    s2(2,3) = s2(3,2)
    beta(1) = -0.5 * c1(1) / c2(1,1)
    beta(2) = zero
    beta(3) = zero
    IF (iter .GT. 1) CALL Move(m)

C  Modify the current distribution (f-vector)
    fSum = zero     ! find the sum of the f-vector
    fChange = zero      ! and how much did it change?
    DO  i = 1, n
      df = beta(1)*xi(i,1)+beta(2)*xi(i,2)+beta(3)*xi(i,3)
      IF (df .LT. -f(i)) df = 0.001 * base(i) - f(i)    ! a patch
      f(i) = f(i) + df      ! adjust the f-vector
      fSum = fSum + f(i)
      fChange = fChange + df
    END DO

    s = zero
    DO   i = 1, n
      temp = f(i) / fSum        ! fraction of f(i) in this bin
      s = s - temp * LOG (temp) ! from Skilling and Bryan, eq. 1
    END DO

        CALL opus (n, nPt, f, z)    ! model the data-space from f(*)
    ChiSq = zero            ! get the new ChiSquared
        DO   j = 1, nPt
          z(j) = (datum(j) - z(j)) / sigma(j)   ! the residuals
      ChiSq = ChiSq + z(j)**2   ! report this ChiSq, not the one above
    END DO

  300   IF ( MOD(iter, 5) .EQ. 0 ) THEN
      WRITE (*,*)
      WRITE (*,*) ' Residuals'
      CALL ResPlt (npt, z)

```

```
      WRITE (*,*)
      WRITE (*,*) ' Distribution'
      CALL BasPlt (n, f, base)
    END IF

    WRITE (*,*) ' #', iter, ' of ', itermax, ',  n  = ', npt
    WRITE (*,200) test, s
    WRITE (*,201) 'target',SQRT(chtarg/npt), 'now',SQRT(chisq/npt)
    WRITE (*,202) 'sum', fSum, ' % change', 100.*fChange/fSum
  200   FORMAT (' test = ', F9.5, ',  Entropy = ', F12.7)
  201   FORMAT (' SQRT((Chi^2)/n):', A8,' = ', F12.8,A10,' = ', F12.8)
  202   FORMAT ('        f-vector:', A8,' = ', F12.8,A10,' = ', F12.8)

C  See if we have finished our task.
    IF (ABS(chisq/chizer-one) .LT. 0.01) THEN  ! hardest test first
      IF (test .LT. TstLim) THEN        ! same solution gradient?
C       We've solved it but now must check for a bizarre condition.
C       Calling routine says we failed if "iter = iterMax".
C       Let's increment iterMax so (maybe) this doesn't happen.
        IF (iter .EQ. iterMax) iterMax = iterMax + 1
        RETURN
      END IF
    END IF
    IF (iter .LT. iterMax) GO TO 6

C  Ask for more time to finish the job.
    WRITE (*,*)
    WRITE (*,*) ' Maximum iterations have been reached.'
 2001   WRITE (*,*) ' How many more iterations? <none>'
    READ (*,'(I4)') more
    IF (more .LT. 0) GO TO 2001
    IF (more .EQ. 0) RETURN
    iterMax = iterMax + more
    GO TO 6
    END


    SUBROUTINE Move(m)
    IMPLICIT REAL*8 (A-H,O-Z)
    IMPLICIT INTEGER*4 (I-N)
    PARAMETER ( MxLoop = 500 )  ! for no solution
    PARAMETER ( Passes = 1.e-3 )    ! convergence test
    COMMON /space5/ chisq, chtarg, chizer, fSum, blank
    COMMON /space2/ beta, c1, c2, s1, s2
    DIMENSION beta(3), c1(3), c2(3,3), s1(3), s2(3,3)
    DATA one, zero /1.0, 0.0/   ! compiler-independence!
    a1 = zero           ! lower bracket  "a"
    a2 = one            ! upper bracket of "a"
        cmin = ChiNow (a1, m)
    IF (cmin*chisq .GT. chizer) ctarg = 0.5*(one + cmin)
    IF (cmin*chisq .LE. chizer) ctarg = chizer/chisq
    f1 = cmin - ctarg
    f2 = ChiNow (a2,m) - ctarg
    DO   loop = 1, MxLoop
      anew = 0.5 * (a1+a2)      ! choose a new "a"
      fx = ChiNow (anew,m) - ctarg
      IF (f1*fx .GT. zero) a1 = anew
      IF (f1*fx .GT. zero) f1 = fx
```

```
      IF (f2*fx .GT. zero) a2 = anew
      IF (f2*fx .GT. zero) f2 = fx
      IF (abs(fx) .LT. Passes) GO TO 2
    END DO

C  If the preceding loop finishes, then we do not seem to be converging.
C   Stop gracefully because not every computer uses control-C (etc.)
C   as an exit procedure.
    WRITE (*,*) ' Loop counter = ', MxLoop
    PAUSE ' No convergence in alpha chop (MOVE).  Press return ...'
    STOP ' Program cannot continue.'

    2   w = Dist (m)
    IF (w .GT. 0.1*fSum/blank) THEN
      DO  k = 1, m
        beta(k) = beta(k) * SQRT(0.1 * fSum/(blank * w))
      END DO
    END IF
    chtarg = ctarg * chisq
    RETURN
    END


    REAL*8 FUNCTION Dist (m)
    IMPLICIT REAL*8 (A-H,O-Z)
    IMPLICIT INTEGER*4 (I-N)
    COMMON /space5/ chisq, chtarg, chizer, fSum, blank
    COMMON /space2/ beta, c1, c2, s1, s2
    DIMENSION beta(3), c1(3), c2(3,3), s1(3), s2(3,3)
    DATA one, zero /1.0, 0.0/   ! compiler-independence!
    w = zero
    DO   k = 1, m
      z = zero
      DO   l = 1, m
        z = z - s2(k,l) * beta(l)
      END DO
      w = w + beta(k) * z
    END DO
    Dist = w
    RETURN
    END


    REAL*8 FUNCTION ChiNow(ax,m)
    IMPLICIT REAL*8 (A-H,O-Z)
    IMPLICIT INTEGER*4 (I-N)
    COMMON /space5/ chisq, chtarg, chizer, fSum, blank
    COMMON /space2/ beta, c1, c2, s1, s2
    DIMENSION beta(3), c1(3), c2(3,3), s1(3), s2(3,3)
    DIMENSION a(3,3), b(3)
    DATA one, zero /1.0, 0.0/   ! compiler-independence!
    bx = one - ax
    DO   k = 1, m
      DO   l = 1, m
        a(k,l) = bx * c2(k,l)  -  ax * s2(k,l)
      END DO
      b(k) = -(bx * c1(k)  -  ax * s1(k))
    END DO
```

```
      CALL ChoSol(a,b,m,beta)
      w = zero
      DO   k = 1, m
        z = zero
        DO   l = 1, m
          z = z + c2(k,l) * beta(l)
        END DO
        w = w + beta(k) * (c1(k) + 0.5 * z)
      END DO
      ChiNow = one +  w
      RETURN
      END


      SUBROUTINE ChoSol(a, b, n, beta)
      IMPLICIT REAL*8 (A-H,O-Z)
      IMPLICIT INTEGER*4 (I-N)
      DIMENSION fl(3,3), a(3,3), bl(3), b(3), beta(3)
      DATA one, zero /1.0, 0.0/   ! compiler-independence!
      IF (a(1,1) .LE. zero) THEN
        WRITE (*,*) ' Fatal error in CHOSOL: a(1,1) = ', a(1,1)
        PAUSE ' Press <RETURN> to end program ...'
        STOP ' Program cannot continue.'
      END IF
      fl(1,1) = SQRT(a(1,1))
      DO   i = 2, n
        fl(i,1) = a(i,1) / fl(1,1)
        DO   j = 2, i
          z = zero
          DO   k = 1, j-1
            z = z + fl(i,k) * fl(j,k)
          END DO
          z = a(i,j) - z
          IF (j .EQ. i) fl(i,j) = SQRT(z)
          IF (j .NE. i) fl(i,j) = z / fl(j,j)
        END DO
      END DO
      bl(1) = b(1) / fl(1,1)
      DO   i=2, n
        z = zero
        DO   k = 1, i-1
          z = z + fl(i,k) * bl(k)
        END DO
        bl(i) = (b(i) - z) / fl(i,i)
      END DO
      beta(n) = bl(n) / fl(n,n)
      DO   i1 = 1, n-1
        i = n - i1
        z = zero
        DO   k = i+1, n
          z = z + fl(k,i) * beta(k)
        END DO
        beta(i) = (bl(i) - z) / fl(i,i)
      END DO
      RETURN
      END
```

```
    SUBROUTINE ResPlt (n, x)
C   Draw a plot of the standardized residuals on the screen.
C   Mark the rows of + and - one standard deviation.
    IMPLICIT REAL*8 (A-H,O-Z)
    IMPLICIT INTEGER*4 (I-N)
    DIMENSION x(1)
    CHARACTER*1 Blank, Symbol, hBordr, vBordr, resSym
    PARAMETER (Blank = ' ', Symbol = 'O', resSym = '=')
    PARAMETER (hBordr = '-', vBordr = '|')

    COMMON /space4/ screen, MaxCol, MaxRow, MxC2, MxR2
    CHARACTER*1 screen(100, 150)

        IF (n .LT. 2) RETURN    ! not enough data

C  Find out how many points to pack per column and how many columns
    nPack = 1 + INT(FLOAT (n) / MaxCol - 1./n)
    nCol = INT((n - 1./n)/nPack + 1)

C  prepare the "screen" for drawing
    DO   j = 1, nCol + 2
      DO   i = 1, MxR2
        screen(i,j) = Blank
      END DO
    END DO
    DO   i = 2, nCol + 1
      screen(MxR2,i) = hBordr
      screen(1,i) = hBordr
    END DO
    DO   i = 2, MaxRow + 1
      screen(i,nCol+2) = vBordr
      screen(i,1) = vBordr
    END DO

C  get the data limits
        xMax = 1.
    xMin = -1.
        DO  i = 1, n
      IF (x(i) .GT. xMax) xMax = x(i)
      IF (x(i) .LT. xMin) xMin = x(i)
    END DO
    RowDel = (MaxRow - 1) / (xMax - xMin)

C  show the standard deviation bars
    mPlus = 1 + INT((1 - xMin)*RowDel + 1)
    mMinus = 1 + INT((-1 - xMin)*RowDel + 1)
    DO   i = 2, nCol + 1
      screen(mMinus,i) = resSym
      screen(mPlus,i) = resSym
    END DO

C  draw the data (overdrawing the residuals bars if necessary)
    DO   i = 1, n
      mCol = 1 + INT((i - 1./n)/nPack + 1)      ! addressing function
      mRow = 1 + INT((x(i) - xMin)*RowDel + 1)  ! +1 for the plot frame
      screen(mRow, mCol) = Symbol
    END DO

```

```
C  convey the "screen" to the default output
    WRITE (*,*) nPack, ' point(s) per column'
    WRITE (*,*) 1./RowDel, ' standard deviations per row'
    DO   i = MxR2, 1, -1
      WRITE (*,*) (screen(i,j), j = 1, nCol + 2)
    END DO

    RETURN
    END


    SUBROUTINE BasPlt (n, x, basis)
C   Draw a plot of some data with reference to a basis line on the plot.
C   The basis is that line below which the data is not meaningful.
    IMPLICIT REAL*8 (A-H,O-Z)
    IMPLICIT INTEGER*4 (I-N)
    DIMENSION x(1), basis(1)
    CHARACTER*1 Blank, Symbol, hBordr, vBordr, BasSym
    PARAMETER (Blank = ' ', Symbol = 'O', BasSym = '=')
    PARAMETER (hBordr = '-', vBordr = '|')

    COMMON /space4/ screen, MaxCol, MaxRow, MxC2, MxR2
    CHARACTER*1 screen(100, 150)

        IF (n .LT. 2) RETURN    ! not enough data

C  Find out how many points to pack per column and how many columns
    nPack = 1 + INT(FLOAT (n) / MaxCol - 1./n)
    nCol = INT((n - 1./n)/nPack + 1)

C  prepare the "screen" for drawing
    DO  j = 1, nCol + 2
      DO  i = 1, MxR2
        screen(i,j) = Blank
      END DO
    END DO
    DO  i = 2, nCol + 1
      screen(MxR2,i) = hBordr
      screen(1,i) = hBordr
    END DO
    DO  i = 2, MaxRow + 1
      screen(i,nCol+2) = vBordr
      screen(i,1) = vBordr
    END DO

C  get the data limits
        xMax = x(1)
    xMin = xMax
        DO  i = 1, n
      IF (x(i) .GT. xMax) xMax = x(i)
      IF (x(i) .LT. xMin) xMin = x(i)
      IF (basis(i) .GT. xMax) xMax = basis(i)
      IF (basis(i) .LT. xMin) xMin = basis(i)
    END DO
    RowDel = (MaxRow - 1) / (xMax - xMin)


C  draw the data (overdrawing the basis bars if necessary)
```

```
    DO  i = 1, n
      mCol = 1 + INT((i - 1./n)/nPack + 1)      ! addressing function
      mRow = 1 + INT((basis(i) - xMin)*RowDel + 1)  ! basis
      screen(mRow, mCol) = basSym
      mRow = 1 + INT((x(i) - xMin)*RowDel + 1)  ! data
      screen(mRow, mCol) = Symbol
    END DO

C  convey the "screen" to the default output
    WRITE (*,*) nPack, ' point(s) per column'
    WRITE (*,*) 1./RowDel, ' units per row'
    DO  i = MxR2, 1, -1
      WRITE (*,*) (screen(i,j), j = 1, nCol + 2)
    END DO

    RETURN
    END


        SUBROUTINE Plot (n,x,y)
C   Make a scatter plot on the default display device (UNIT=*).
C   MaxRow and MaxCol correspond to the display dimensions.
    IMPLICIT REAL*8 (A-H,O-Z)
    IMPLICIT INTEGER*4 (I-N)
    DIMENSION x(1), y(1)
    CHARACTER*1 Blank, Symbol, hBordr, vBordr
    PARAMETER (Blank = ' ', Symbol = 'O')
    PARAMETER (hBordr = '-', vBordr = '|')

    COMMON /space4/ screen, MaxCol, MaxRow, MxC2, MxR2
    CHARACTER*1 screen(100, 150)

        IF (n .LT. 2) RETURN    ! not enough data

C  prepare the "screen" for drawing
    DO   j = 1, MxC2
      DO   i = 1, MxR2
        screen(i,j) = Blank
      END DO
    END DO
    DO   i = 2, MaxCol+1
      screen(MxR2,i) = hBordr
      screen(1,i) = hBordr
    END DO
    DO   i = 2, MaxRow+1
      screen(i,MxC2) = vBordr
      screen(i,1)  = vBordr
    END DO

C  get the data limits
    xMin = x(1)
        xMax = x(1)
    yMin = y(1)
        yMax = y(1)
        DO  i = 2, n
      IF (x(i).GT.xMax) xMax=x(i)
      IF (x(i).LT.xMin) xMin=x(i)
      IF (y(i).GT.yMax) yMax=y(i)
```

```
      IF (y(i).LT.yMin) yMin=y(i)
    END DO
    ColDel = (MaxCol - 1) / (xMax - xMin)
    RowDel = (MaxRow - 1) / (yMax - yMin)

C  data scaling functions are offset by +1 for plot frame
    DO   i = 1, n
      mCol = 1 + INT((x(i) - xMin)*ColDel + 1)
      mRow = 1 + INT((y(i) - yMin)*RowDel + 1)
      screen(mRow, mCol) = Symbol
    END DO

C  convey the "screen" to the default output
    WRITE (*,*) 1./ColDel, ' units per column'
    WRITE (*,*) 1./RowDel, ' units per row'
    DO   i = MaxRow + 2, 1, -1
      WRITE (*,*) (screen(i,j), j = 1, MaxCol + 2)
    END DO
        RETURN
        END
```

Last Ditch Help

Any user who has, without success, tried all of the suggestions provided in the section titled *OBSERVATIONS, HINTS, SUGGESTIONS* for correcting a failure of the program should feel free to contact the authors listed here at any time for further advice and suggestions.

References

1. (a) Skilling and R.K. Bryan; MON NOT R ASTR SOC 211 (1984) 111 - 124.

10. J.A. Potton, G.J. Daniell, and B.D. Rainford; Proc. Workshop on Neutron Scattering Data Analysis, Rutherford Appleton Laboratory, UK, 1986; ed. M.W. Johnson, IOP Conference Series 81 (1986) 81 - 86, Institute of Physics, Bristol, UK.

11. I.D. Culverwell and G.P. Clarke; Proc. Workshop on Neutron Scattering Data Analysis, Rutherford Appleton Laboratory, UK, 1986; ed. M.W. Johnson, IOP Conference Series 81 (1986) 87 - 96, Institute of Physics, Bristol, UK.

12. J.A. Potton, G.J. Daniell, & B.D. Rainford; J APPL CRYST 21 (1988a) 663 - 668.

13. J.A. Potton, G.J. Daniell, & B.D. Rainford; J APPL CRYST 21 (1988b) 891 - 897.

14. L.C. Roess & C.G. Shull; J APPL PHYS 18 (1947) 308-313.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

This documentation was assembled Sep 27, 2017.

Index