TITLE A computer program for circle-fitting by a least squares method

AUTHOR D Haddon-Reece

DATE February 1980

ABSTRACT The well-documented procedure for linear regression by a least squares method can be extended to the fitting of circles. This report describes the mathematical theory and program developed to compare geophysical plotting methods used on a potential round barrow at Burntwood Farm, Kings Worthy, Hants, in 1977 (AMLab report no 2344)

The information in this report is Crown Copyright and is provided on the understanding that its author will be consulted at all stages if it is to be used for any further publication

A computer program for circle-fitting by a least squares method

I Introduction

A frequently encountered problem in the geophysical surveying of archaeological sites is that of fitting a circle to a set of co-ordinates. These points could represent spot magnetometer readings around a barrow ditch, for instance, or the positions of the stone holes in a partially or totally destroyed stone circle.

The mathematical technique described here with its associated computer program was developed by the author in July 1977 to resolve a particular problem arising the comparison of plotting methods applied to a possible round barrow. Firstly, it was required to decide whether a barrow whose remaining mound was partially overlain by a dense thicket and an iron fence had been round or long. Also, the results from two magnetic plotting systems applied to the task, both under test for archaeological work, had to be compared by an objective method. As concluded in the report on the exercise - 'Burntwood Farm', A M Lab No 2344, (Haddon-Reece, 1977) - the program successfully fitted statistically indistinguishable circles to the data in the two cases, confirming that both methods produced similar and acceptable results, and that the barrow had been round.

This report has been written to document another in the series of Ancient Monuments Laboratory working techniques. Since the report on Burntwood Farm, a similar procedure for fitting circles and ellipses to Megalithic stone rings (Angell and Barber, 1977) has been published, and reference should be made to that for the treatment of ellipses.

II Mathematical basis

II 1 Straight line fitting

Linear regression by a least squares method is a well documented procedure (see, eg, Paradine and Rivett, Chapter X). This technique will be reviewed first, as it leads directly into that for circle fitting.

To a set of n data points it is required to fit a straight line such that the sum of squares of the residual distances between it and the data is a minimum. For any data point (x_i, y_i, $i = 1,n$), the residual r_i is expressed as

$$r_i = y_i - a - bx_i$$

where a and b are the intercept and slope of the fitted line. For a best fit,

$$S = \sum_{i=1}^{n} (y_i - a - bx_i)^2 \quad \text{is a minimum, requiring that}$$

$$\partial S / \partial a = \partial S / \partial b = 0$$

Adopting the convention of representing the summation sign by square brackets, and partially differentiating w.r.t. a,

$$-2\left[(y_i - a - bx_i)\right] = 0. \quad \text{Similarly, w.r.t. b,}$$

$$-2x_i\left[(y_i - a - bx_i)\right] = 0$$

Rearranging, and putting $[1] = n$,

$$[y_i] = na + b[x_i] \tag{1}$$

$$[x_iy_i] = a[x_i] + b[x_i^2] \tag{2}$$

which are called the normal equations. Summing x_i^2, x_iy_i, x_i and y_i from the data leads to the standard equations for a and b:

$$a = \text{intercept} = \frac{[y][x^2] - [x][xy]}{n[x^2] - [x]^2} \tag{3}$$

$$b = \text{slope} = \frac{n[xy] - [x]^2}{n[x^2] - [x]^2} \tag{4}$$

II 2. Circle fitting

The general equation of the circle is:

$$x^2 + y^2 + 2gx + 2fy + c = 0$$

Rearringing,

$$(x+g)^2 + (y+f)^2 = g^2 + f^2 - c$$

from which it can be seen that the centre is the point $(-g,-f)$, and the radius is $(g^2 + f^2 - c)^{\frac{1}{2}}$. Solution of the normal equations obtained in a similar manner to the linear case will yield best fit values of the coefficients g, f, and c, and hence centre and radius.

The sum of squares of residuals throughout the data is

$$S = \left[(x^2 + y^2 + 2gx + 2fy + c)^2\right]$$

For a minimum, the partial derivatives of S w.r.t to g, f and c must be zero; that is

$$4\left[x.(x^2 + y^2 + 2gx + 2fy + c)\right] = 0 \tag{5}$$

$$4\left[y.(x^2 + y^2 + 2gx + 2fy + c)\right] = 0 \quad \text{and} \qquad (6)$$

$$2\left[(x^2 ++y^2 + 2gx + 2fy + c)\right] = 0\,. \qquad (7)$$

Expanding the terms under the summation sign, and consigning to to the RHS the constant terms (ie those not containing g, f or c), equations (5), (6) and (7) can be rearranged as:

$$g.2[x^2] + f.2[xy] + c.[x] = -(\,[x^3] + [xy^2]\,) \qquad (8)$$

$$g.2[xy] + f.2[y^2] + c.[y] = -(\,[y^3] + [x^2y]\,) \qquad (9)$$

$$g.2[\,x\,] + f.2[\,y\,] + c.n = -(\,[x^2] + [y^2]\,) \qquad (10)$$

This set of simultaneous equations, linear in the unknowns, is the set of normal equations which was required. It remains to solve them for g, f and c.

Denoting the summation $[x^3]$ by A
$[y^3]$ " B
$[x^2]$ " C
$[y^2]$ " D
$[\,x\,]$ " E
$[\,y\,]$ " F
$[x^2y]$ " G
$[xy^2]$ " H and
$[xy]$ " J ,

the normal equations (8), (9), and (10) may be solved by Cramer's rule to obtain the solution:

$$g = \frac{-2}{DEL}\begin{vmatrix} A+H & J & E \\ B+G & D & F \\ C+D & F & n \end{vmatrix} \;; \qquad f = \frac{-2}{DEL}\begin{vmatrix} C & A+H & E \\ J & B+G & F \\ E & C+D & n \end{vmatrix} \;; \qquad (11),(12)$$

$$c = \frac{-4}{DEL}\begin{vmatrix} C & J & A+H \\ J & D & B+G \\ E & F & C+D \end{vmatrix} \;; \quad \text{where} \quad DEL = 4\begin{vmatrix} C & J & E \\ J & D & F \\ E & F & n \end{vmatrix} \qquad (13),(14)$$

whence, as stated above,

$$\text{Radius} = (g^2 + f^2 - c)^{\frac{1}{2}} \;; \quad \text{centre is at } (-g,-f)\,.$$

III The computer program

Program CIRCLE, which is listed in full on pp 6-7, is written in FORTRAN IV to run interactively on the GEISCO Mk III time-sharing service. Up to thirty data points are acceptable, which may be entered either from the keyboard or from a previously set up ASCII data file. In both options, an integer denoting the number of points is read first followed by the coordinates pairs in either real or integer format.

After passing through all the points to sum x^3, y^3, x^2y, xy^2, xy, y^2, x and y, the program then calculates the necessary determinant values and solves for g, f and c by Cramer's rule to obtain the radius and the central coordinates. Passing through the points again, it calculates the radius to each point from the derived centre, and its deviation. By summing the squares of the deviations, dividing by the number of readings less unity and taking the square root of the quotient, the program derives the root mean square error, and, multiplying by 0.6745, the probable error. These values are then printed.

(As Paradine and Rivett point out (p 190), when deducing unknowns by least squares reasoning, equal weights may not be attributable to each observation, so that a more complicated process has to be followed to obtain the rms error. The simpler approach will be followed for the moment, and only if unacceptably large inaccuracies become visible, in the testing stage, will steps be taken to correct it.)

Next, the program scans the deviations, indicating those which exceed 5 probable errors. This follows a rule given by Wright and Hayford (Paradine and Rivett, p 161) in which surveyors are recommended to reject any observation deviating by more than this amount, and to scrutinise in the light of the circumstances in which they were made any which exceed $3\frac{1}{2}$ probable errors. In archaeological terms, the dubiety of the coordinates of any point may rest on several factors - precision of surveying, precision of estimating the centrally representative point of an amorphous magnetic anomaly, etc. Not only is the precision hard to quantify, but it is difficult to allocate weights. Re-entering the data with the exclusion of dubious values is therefore left at the discretion of the user; no option is put into this version of the program.

Finally, the program prints out the set of coordinates, together with their deviations from the calculated radius.

IV Experimental results

As a test of the program's capability, circles were fitted through three sets of data, each derived separately from an identical perfect circle of radius 6 units and cartesian centre (0,0). The polar coordinates of twelve points spaced equidistantly around the circle were calculated and subjected to the following treatment.

Firstly, using the polar to rectangular conversion facility on a TI59 programmable calculator, each point was converted to rectangular coordinates referred to an origin outside the circle at (-7,-7). These twelve coordinate pairs were put into the program, as demonstrated in the printout in the upper half of p 8, returning as the correct radius and centre. The input data is shown at the right.

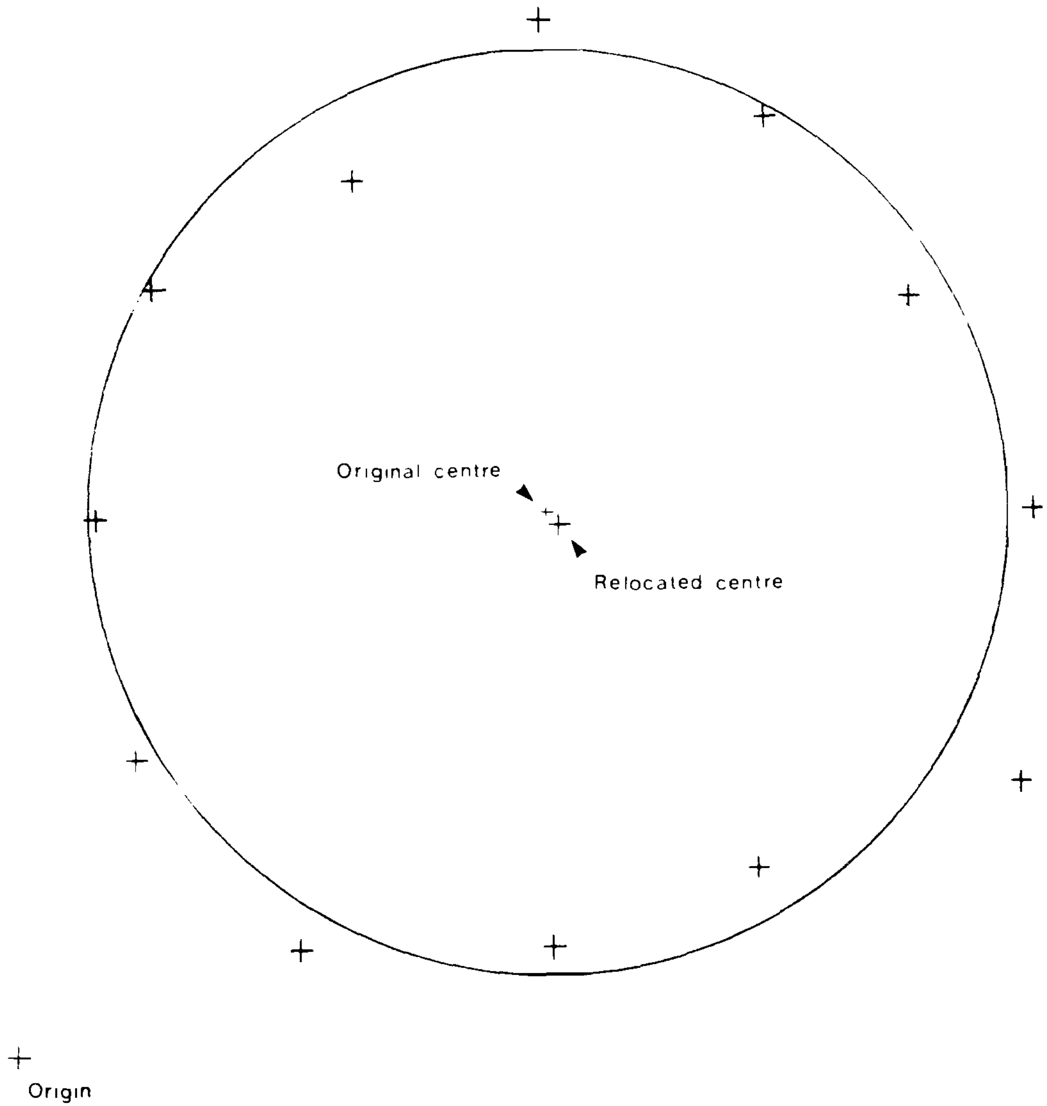

Figure 1

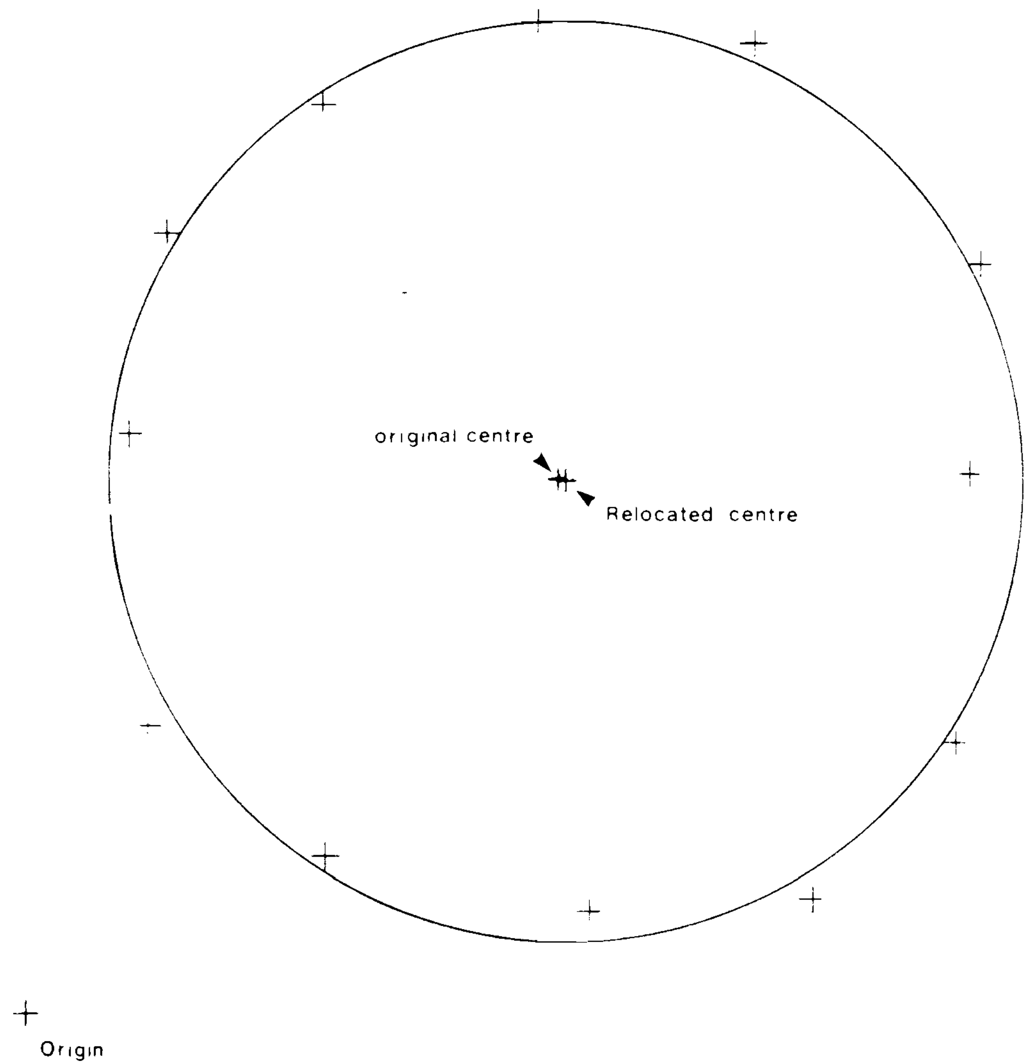

Figure 2

In the second case, each of the twelve originally equal radii were adjusted by the addition of an error term. These error terms, calculated by the TI59 pseudo-random number library program, were normally distributed about zero with a standard deviation equal to 10% of the radius, ie 0.6 units. No deviation in the angular direction was included. The printout in the lower half of p8 shows the calculated results of radius (6.02 +/- 0.535) and centre (7.14,6.84). and this circle is drawn through the data points in Figure 1.

For the third test, the original, unaltered coordinates were subjected to separate error distributions in the x and y directions. Each was normally distributed about zero with a standard deviation of 0.425 units, which again produces a variance of $2 \times (0.425)^2 = 0.6$ in the radius. The calculated results taken from the printout on p 9 are plotted as the best fit circle through the original points in Fig 2.

V Discussion of results and conclusion

In the absence of any irregularity in the source data, the program's capacity for accurately and precisely restoring a circle is demonstrated by the results of the first test.

When fitting a circle to data which deviates from a perfect circle, the program is well able to reproduce the initial radius, to within 0.3% in the second case and closer still in the third; in both cases the relocation of the centre is more than acceptably close for archaeological purposes. In the third case, since the errors imposed were distributed about zero in both x and y directions, the relocated centre should in theory coincide with the original centre. The very slight discrepancy between original and relocated centres represents rounding errors in entering the data.

In both second and third cases the calculated rms error of the radius is actually less than the source value. This reflects the previously discussed incorrectness in choice of computing formula. However, since the errors in estimating the representative position of any archaeological anomaly in a geophysical surveying record are far greater than the magnitude of those produced in this processing, there is little point in refining the method beyond realistic bounds.

The program has been found to be a practicable way to apply this circle fitting technique to archaeological data. The only modification which suggests itself is to include the option to reject data points by user direction at the keyboard, so that the program could repeat the processing with correctly altered control parameters. The only mathematical modification which might be required for especially precise work would be to incorporate the technique for finding the rms error suggested by Paradine and Rivett.

D Haddon-Reece
Ancient Monuments Laboratory
February 1980

References

Angell, I O, and Barber, J, An algorithm for fitting circle and ellipses to Megalithic stone rings. Science and Archaeology 20,1977, 11-16.

Haddon-Reece, D, Burntwood Farm: a comparison of plotting instruments. Ancient Monuments Laboratory Report No 2344, 1977 (unpublished).

Paradine, C G, and Rivett, B H P, Statistical methods for technologists. English Universities Press, 1960.

```
*********************************************************************
* This program fits a circle to a set of points by a least squares
* method for up to 30 points.  The coordinates of the centre, the
* radius and the probable error are printed, followed by a listing
* of the point coordinates and their deviations from the fitted
* circle.  The program then indicates whether all the points lie
* within a distance of five probable errors from the fitted line.
* Written by D Haddon-Reece, Ancient Monuments Laboratory,
* Department of the Environment, August 1977.
*********************************************************************
          OPTION NOWARN
          INTEGER IPT(30)
          REAL J,K,L,M,N
          REAL CRD(3,30),RAD(30),DEV(30)
          FILENAME FN
*
* Input section
*
          PRINT,"INPUT FROM KEYBOARD(1) OR FILE(2)"
          INPUT,REPLY
          IF(REPLY.EQ.1)GOTO 400
          PRINT,"ENTER INPUT FILENAME"
          INPUT,FN
          READ(FN,1080),NUM
          READ(FN,1080,END=390)(CRD(2,I),CRD(3,I),I=1,NUM)
      390 GO TO 440
      400 PRINT,"Enter number of points"
          INPUT,NUM
          PRINT,"Enter all pairs of coordinates"
          INPUT,(CRD(2,I),CRD(3,I),I=1,NUM)
      440 CONTINUE
*
* Calculation Section 1:  summation
*
          N = NUM
          DO 610 I=1,NUM
          X=CRD(2,I)
          Y=CRD(3,I)
          E=E+X
          F=F+Y
          C=C+X*X
          D=D+Y*Y
          A=A+X**3
          B=B+Y**3
          G=G+X*X*Y
          H=H+X*Y*Y
          J=J+X*Y
      610 IPT(I)=I
          K=A+H
          L=B+G
```

```
      M=C+D
*
* Calculation section 2: substitution into determinant
* for solution by Kramer's Rule
*
      DEL = 4*(C*D*N - C*F*F - N*J*J + 2*E*F*J - D*E*E)
      GOUT = (-2/DEL)
     &*(K*D*N - K*F*F - J*L*N + F*J*M + E*F*L - D*E*M)
      FOUT = (-2/DEL)
     &*(C*L*N - C*F*M - J*K*N + E*F*K + E*J*M - L*E*E)
      COUT = (-4/DEL)
     &*(C*D*M - C*F*L - M*J*J + E*J*L + F*J*K - E*D*K)
      XCEN = -GOUT
      YCEN = -FOUT
      RADIUS = SQRT(GOUT*GOUT + FOUT*FOUT - COUT)
      DO 830 I=1,NUM
      RAD(I) = SQRT((CRD(2,I)-XCEN)**2 + (CRD(3,I)-YCEN)**2)
      DEV(I) = RADIUS - RAD(I)
      DEVSUM = DEVSUM + DEV(I)*DEV(I)
  830 CONTINUE
      SSR = SQRT(DEVSUM/(NUM-1))
      PE = 0.6745 * SSR
*
* Output section
*
      PRINT1050,XCEN,YCEN,RADIUS,SSR
      PRINT1070,PE
      PE5 = PE*5
      DO 970 I=1,NUM
      DEVABS = ABS(DEV(I))
      IF(DEVABS.LT.PE5)GOTO 970
      PRINT,"POINT",I,"HAS DEVN. > 5 PROBABLE ERRORS"
      IFL = 1
  970 CONTINUE
      IF(IFL.EQ.1)GOTO 1010
      PRINT," "
       PRINT,"Data set is within 5 times probable error"
 1010 PRINT," "
       PRINT," Point     x        y      Radius   Dev'n"
       DO 1040 I = 1,NUM
 1040 PRINT1090,IPT(I),(CRD(NN,I),NN=2,3),RAD(I),DEV(I)
 1050 FORMAT(1H ,"Coordinates of centre are : "F6.2," , ",
     &F6.2/" Radius =    ",F6.2," +/- ",F6.3//)
 1070 FORMAT(1H ,"Probable error = ",F6.3)
 1080 FORMAT(V)
 1090 FORMAT(1H ,I7,4F8.3)
       CALL EXIT
       STOP
       END
```

```
      integer IPT(30)
      real J,K,L,M,N
      real CRD(3,30),RAD(30),dev(30)

      open (1,file = 'circinp')
c
      read(1,*)NUM
      read(1,*,end = 390) (CRD(2,I),CRD(3,i),i=1,NUM)
  390 goto 440
  400 continue
  440 continue
      write(*,441)
  441 format('input complete')
c
      N = float(NUM)
      X=0
      Y=0
      E=0
      F=0
      G=0
      H=0
      J=0
      A=0
      B=0
      C=0
      D=0
      do 610 I = 1,NUM
         X = CRD(2,I)
         Y = CRD(3,I)
         E = E + X
         F = F + Y
         C = C + X*X
         D = D + Y*Y
         A = A + X*X*X
         B = B + Y*Y*Y
         G = G + X*X*Y
         H = H + X*Y*Y
         J = J + X*Y
  610 IPT(I) = I
      K = A + H
      L = B + G
      M = C + D
c
      write(*,611)
  611 format('at 611')

      DEL = 4*(C*D*N - C*F*F - N*J*J + 2*E*F*J - D*E*E)
      FC = 2./DEL
      GOUT = FC * (K*D*N - K*F*F - J*L*N + F*J*M + E*F*L - D*E*M)
      FOUT = FC * (C*L*N - C*F*M - J*K*N + E*F*K + E*J*M - L*E*E)
      COUT = 2.*FC * (C*D*M - C*F*L - M*J*J + E*J*L + F*J*K - E*D
      XCEN =  GOUT
      YCEN =  FOUT
      RADIUS = sqrt (GOUT*GOUT + FOUT*FOUT + COUT)
      do 830 I = 1,NUM
         RAD(I) = sqrt ((CRD(2,I) - XCEN)**2 + (CRD(3,I) - YCEN)*
         DEV(I) = RADIUS - RAD(I)
         DEVSUM = DEVSUM + DEV(I)*DEV(I)
  830 continue
c
      SSR = sqrt (DEVSUM/(N - 1))
      PE = 0.6745*SSR
c
c
```

```
CIRCLE       18:05       02/29/80

INPUT FROM KEYBOARD(1) OR FILE(2)? 2

ENTER INPUT FILENAME? RING.IN

Coordinates of centre are :    7.00 ,    7.00
     Radius =       6.00 +/-  0.000

Probable error =  0.000

Data set is within 5 times probable error

  Point     x        y      Radius    Dev'n
      1  13.000    7.000    6.000   -0.000
      2  12.196   10.000    6.000    0.000
      3  10.000   12.196    6.000    0.000
      4   7.000   13.000    6.000   -0.000
      5   4.000   12.196    6.000    0.000
      6   1.804   10.000    6.000   -0.000
      7   1.000    7.000    6.000    0.000
      8   1.804    4.000    6.000   -0.000
      9   4.000    1.804    6.000   -0.000
     10   7.000    1.000    6.000    0.000
     11  10.000    1.804    6.000   -0.000
     12  12.196    4.000    6.000    0.000
```

```
RING.IN

12
13,7
12.196,10
10,12.196
7,
13
4,12.196
1.8038,10
1,7
1.8038,4
4,1.8038
7,1
10,1.8038
12.196,4
```

```
CIRCLE       18:06       02/29/80

INPUT FROM KEYBOARD(1) OR FILE(2)? 2

ENTER INPUT FILENAME? RINGINP

Coordinates of centre are :    7.14 ,    6.84
     Radius =       6.02 +/-  0.535

Probable error =  0.361

Data set is within 5 times probable error

  Point     x        y      Radius    Dev'n
      1  13.389    7.000    6.254   -0.233
      2  11.823    9.785    5.534    0.487
      3   9.954   12.116    5.980    0.041
      4   7.000   13.424    6.584   -0.563
      5   4.474   11.375    5.258    0.763
      6   1.832    9.984    6.166   -0.145
      7   1.082    7.000    6.057   -0.036
      8   1.606    3.885    6.271   -0.250
      9   3.733    1.341    6.468   -0.447
     10   7.000    1.391    5.452    0.569
     11   9.684    2.351    5.162    0.859
     12  13.064    3.499    6.804   -0.783
```

```
RINGINP

12
13.389,7
11.823,9.785
9.9535,12.116
7,13.424
4.474,11.375
1.832,9.984
1.082,7
1.606,3.885
3.733,1.341
7,1.391
9.684,2.351
13.064,3.499
```

```
CIRCLE      13:26       03/03/80

INPUT FROM KEYBOARD(1) OR FILE(2)?  2

ENTER INPUT FILENAME?  RINGINP2

Coordinates of centre are :   7.10 ,   6.94
Radius =      6.00 +/-  0.317

Probable error =  0.214

Data set is within 5 times probable error

  Point     x        y      Radius    Dev'n
     1   12.439    7.086    5.344    0.655
     2   12.532    9.811    6.146   -0.147
     3    9.601   12.648    6.231   -0.233
     4    6.737   12.913    5.982    0.017
     5    3.945   11.866    5.847    0.152
     6    1.810   10.135    6.177   -0.178
     7    1.399    7.547    5.730    0.268
     8    1.595    3.744    6.364   -0.366
     9    3.944    2.042    5.827    0.171
    10    7.407    1.306    5.644    0.354
    11   10.375    1.527    6.330   -0.332
    12   12.237    3.355    6.268   -0.269

USED     6.32 UNITS


RINGINP2    13:27       03/03/80

12
12.439   7.086
12.532   9.811
9.601    12.648
6.737    12.913
3.945   11.866
1.8098   10.135
1.399    7.547
1.5948   3.744
3.944    2.0418
7.407    1.306
10.375   1.5268
12.237   3.355
```