1993-04-27

Estimating the Spatial Extent of Attractors of Iterated Function System

Canright, D.

Monterey, California. Naval Postgraduate School

NPS-MA-93-017

NAVAL POSTGRADUATE SCHOOL

Monterey, California

ESTIMATING THE SPATIAL EXTENT OF ATTRACTORS OF ITERATED FUNCTION SYSTEMS

by

D. Canright

Technical Report For Period
January 1993 - April 1993

Prepared for: Naval Postgraduate School
Monterey, CA 93943

NAVAL POSTGRADUATE SCHOOL
MONTEREY, CA 93943

Rear Admiral T.A. Mercer
Superintendent

Harrison Shull
Provost

This report was prepared in conjunction with research conducted for the Naval Postgraduate School and funded by the Naval Postgraduate School.

This report was prepared by:

D. Canright
Assistant Professor of Mathematics

Reviewed by:

RICHARD FRANKE
Chairman

Released by:

PAUL J. MARTO
Dean of Research

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a REPORT SECURITY CLASSIFICATION Unclassified	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION AVAILABILITY OF REPORT Approved for public release; distribution unlimited
2b DECLASSIFICATION / DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S) NPS-MA-93-017	5 MONITORING ORGANIZATION REPORT NUMBER(S) NPS-MA-93-017
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (If applicable) MA	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c ADDRESS (City, State, and ZIP Code) Monterey, CA 93943	7b ADDRESS (City, State, and ZIP Code) Monterey, CA 93943
8a NAME OF FUNDING / SPONSORING ORGANIZATION Naval Postgraduate School 8b OFFICE SYMBOL (If applicable) MA	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER OM,N
8c ADDRESS (City, State, and ZIP Code) Monterey, CA 93943	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)
Estimating the Spatial Extent of Attractors of Iterated Function Systems

12 PERSONAL AUTHOR(S)
D. Canright

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Technical	FROM 1-93 TO 4-93	27 April 93	21

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Iterated Function System, fractal attractor

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

From any given Iterated Function System, a small set of balls that cover the fractal attractor can be simply determined. This gives a *priori* bounds on the region of space in which the attractor may be constructed.

20 DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION Unclassified
22a NAME OF RESPONSIBLE INDIVIDUAL D. Canright	22b TELEPHONE (Include Area Code) 408-656-2782 22c OFFICE SYMBOL MA/Ca

DD Form 1473, JUN 86 Previous editions are obsolete SECURITY CLASSIFICATION OF THIS PAGE

S/N 0102-LF-014-6603

Estimating the Spatial Extent of Attractors of Iterated Function Systems

D. Canright
Mathematics Dept., Code MA/Ca
Naval Postgraduate School
Monterey, CA 93943

April 27, 1993

Abstract

From any given Iterated Function System, a small set of balls that cover the fractal attractor can be simply determined. This gives *a priori* bounds on the region of space in which the attractor may be constructed.

As Barnsley, Demko and others have shown [1, 2, 6, 4], one effective method for producing fractal shapes (in any number of dimensions) is with Iterated Function Systems (IFSs), using the "Chaos Game" algorithm (or some deterministic algorithm). This approach has been used for producing naturalistic shapes [4], finding fractal interpolants to given data [6, p. 274] and fractal approximations of given functions [8], and even for visualizing arbitrary discrete sequences [7]. Indeed, any contractive IFS will give an attractor (usually of fractal dimension); thus it is possible to generate IFSs at random to explore the graphical possibilities, as is done in some educational software [5]. Similarly, because the attractor depends continuously on the parameters in the IFS [1], a small data sets from any source could be encoded as IFSs for visualization.

In implementing the IFS method, one important question is the prediction *a priori* of the region of space containing the fractal attractor. Without such a prediction, one could only approximately estimate the spatial extent based on calculating several points of the attractor, with no guarantee that these points are near the bounds. If as a result the portion of space represented in the computation of the attractor is too small, the result will not yield the whole attractor. If the portion of space represented is overly large, then much computational space is wasted, reducing the effective resolution of the computed attractor. Another concern is when the space has natural limits, for example the space of colors in Red-Green-Blue-space representable on a video monitor is limited (more or less) to a unit cube. Then the question becomes whether the attractor (or its projection onto the limited dimensions) will fit in the space.

Here a simple algorithm is given to compute, directly from the IFS, a set of balls whose union contains the attractor as a subset (see Figure 1). The radii of the balls are minimal in a certain restricted sense. This gives reliable bounds on the region of space that must be considered in constructing the fractal. The method is general, independent of the particular space and metric. We first describe the set of balls, then show how to compute their radii and prove that the algorithm works, and lastly give a detailed example.

An IFS consists of a set of n contraction mappings $w_i : \mathbf{X} \to \mathbf{X}$ on a metric space $\mathbf{X}$ with metric $d : \mathbf{X} \times \mathbf{X} \to \mathcal{R}$. (For the "Chaos Game" algorithm, probabilities p_i are associated with each mapping; this idea has been extended to conditional probabilities [3]. Here only the former case is considered, where the attractor is independent of the non-zero p_i.) Assume

that for each contraction w_i the contractivity ratio $0 \leq s_i < 1$ and the fixed point $\mathbf{x}_i$ are known, where by definition s_i satisfies $d(w_i(\mathbf{x}), w_i(\mathbf{y})) \leq s_i d(\mathbf{x}, \mathbf{y})$ for all points $\mathbf{x}, \mathbf{y} \in \mathbf{X}$ and $\mathbf{x}_i$ satisfies $\mathbf{x}_i = w_i(\mathbf{x}_i)$. The action of the IFS W on a set S of points in $\mathbf{X}$ is defined as

$$W(S) \equiv \bigcup_{i=1}^{n} w_i(S) \tag{1}$$

where each contraction w_i is applied to the set S in a pointwise sense. The attractor A is the set of points in $\mathbf{X}$ satisfying

$$A = W(A) \quad . \tag{2}$$

That is, the attractor consists of n smaller "copies" of itself.

We seek to cover each of the n "copies" with a closed ball B_i centered on the corresponding fixed point $\mathbf{x}_i$, so the radius r_i must be chosen large enough that $B_i \supset w_i(A)$. Call the union E of these balls the "envelope," in that $E \supset A$ by (2). Then relative to each $\mathbf{x}_i$, every point in the envelope will be within a distance $R_i = \max_j(d_{ij} + r_j)$, where $d_{ij} \equiv d(\mathbf{x}_i, \mathbf{x}_j)$, because for any point $\mathbf{x}$ in B_j, $d(\mathbf{x}_i, \mathbf{x}) \leq d_{ij} + d(\mathbf{x}_j, \mathbf{x}) \leq d_{ij} + r_j$. Applying w_i to such a point $\mathbf{x}$ will give an image point $\mathbf{y}$, where $d(\mathbf{x}_i, \mathbf{y}) \leq s_i d(\mathbf{x}_i, \mathbf{x}) \leq s_i R_i$. Hence if the radii r_i are chosen to satisfy

$$r_i = s_i \max_{j \neq i}(d_{ij} + r_j) \tag{3}$$

for $i, j = 1 \ldots n$ then B_i will contain the image $w_i(E)$ of the envelope and so E will contain its own image under the IFS:

$$E \supset W(E) \quad . \tag{4}$$

Iterating the IFS from any starting set (E in particular) yields a sequence of sets that converges to the attractor. Since (4) implies $E \supset W^k(E)$ for any positive integer k, the envelope E, subject to (3), does indeed contain the attractor. But how can the r_i be calculated from (3)?

When $n = 2$ the radii can be determined algebraically. Solving the pair of equations (3) gives:

$$\begin{aligned} r_1 &= \frac{s_1(1 + s_2)}{1 - s_1 s_2} d_{12} \\ r_2 &= \frac{s_2(1 + s_1)}{1 - s_1 s_2} d_{12} \quad . \end{aligned} \tag{5}$$

But for $n > 2$ there is apparently no closed-form general solution, and the r_i must be found algorithmically. (If one wished to minimize calculating at the expense of overestimating, one could use $r_i = d_{max} s_{max}/(1 - s_{max})$, i.e., the envelope as if all fixed points were equidistant at the maximum separation and with all s_i equal to the largest.)

A natural approach for $n > 2$ is to start with the pairwise estimates

$$r_{ij} \equiv \frac{s_i(1 + s_j)}{1 - s_i s_j} d_{ij} \tag{6}$$

$$r_i^{(1)} = \max_{j \neq i} r_{ij} \tag{7}$$

but in most cases the $r_i^{(1)}$ will not satisfy (3). The exceptional case is when $r_{ij} = r_{ik}$ for every $i, j \neq i, k \neq i$, i.e., when for each ball all the pairwise estimates for that ball give the same size. (This case is not always apparent from the attractor: Figure 1 shows such an example.) Otherwise, some of the $r_i^{(1)}$ will be too small to contain some images $w_i(B_j^{(1)})$ of the other balls. Then the obvious iterative scheme to try is

$$r_i^{(k+1)} = \max_{j \neq i} s_i(d_{ij} + r_j^{(k)}) \quad , \quad i, j = 1 \ldots n \quad . \tag{8}$$

Because this approach never overestimates the radii ($r_i^{(k)} \leq r_i$) and the iterates are nondecreasing, the algorithm must converge. What is not so obvious is that this process always succeeds in at most $n - 1$ iterations, as shown below. (This would not be true without (7).) In fact, there is a direct algorithm (not iterative) that is more efficient.

The key idea that allows the direct algorithm is that the distances d_{ij} can be rescaled to account for the contractivities s_i, and the scaled distances D_{ij} can be used to order the contractions w_i. Let

$$D_{ij} \equiv \frac{(1 + s_i)(1 + s_j)}{1 - s_i s_j} d_{ij} \quad , \quad i, j = 1 \ldots n \quad . \tag{9}$$

(While D_{ij} is clearly symmetric and non-negative, it is not a metric because it doesn't satisfy the triangle inequality.) Now *reorder* (and relabel) the w_i by decreasing maximum scaled distance, so that

$$i < j \quad \Rightarrow \quad \max_k D_{ik} \geq \max_l D_{jl} \quad , \quad i, j, k, l = 1 \ldots n \quad . \tag{10}$$

In the new order, use the pairwise formula (5) for the first two radii. Then proceed *in order* based on the previous results, letting

$$r_i = s_i \max_{j<i}(d_{ij} + r_j) \quad , \quad i = 3 \ldots n \quad . \tag{11}$$

This is the direct algorithm, which, as shown below, solves (3); an implementation in the C programming language is given in the Appendix.

First note that in the exceptional case mentioned above (after 7), all the D_{ij} (for $i \neq j$) are equal. In this case, the direct algorithm will obtain the correct r_i regardless of the order in which they are computed. If not all the D_{ij} are equal, then some of the r_i will need to be larger than the pairwise estimates $r_i^{(1)}$, and hence larger than the r_{ij} in (6), so in general (3) implies

$$r_i \geq \frac{s_i}{1 + s_i} D_{ij} \tag{12}$$

for $i, j = 1 \ldots n$.

For the general case, the proof is by induction, showing that each new r_i computed requires no adjustment of those previously computed. Clearly r_1, r_2 from (5) satisfy (3) for the subset $i, j = 1, 2$. Now assume the first $m - 1$ radii, in the order (10), satisfy (3), and hence (12), for $i, j = 1 \ldots m - 1$. Choose r_m by (11), and let k be the value of the index j in (11) for which the maximum is achieved. Then by (12) and the ordering (10)

$$r_k \geq \frac{s_k}{1 + s_k} D_{km} \quad . \tag{13}$$

Algebraic manipulation of (13) gives

$$r_k \geq s_k[(1 + s_m)d_{km} + s_m r_k] = s_k(d_{km} + r_m) \tag{14}$$

so the new r_m requires no alteration of r_k.

Similarly, for $i \neq k$, $i < m$

$$\begin{aligned} r_i &\geq \frac{s_i}{1 + s_i} D_{im} \\ &> \frac{s_i(1 + s_m)}{1 - s_m} d_{im} \end{aligned} \tag{15}$$

since $s_i < 1$. Combining (15) with

$$r_i \geq s_i(d_{ik} + r_k) \tag{16}$$

yields

$$\begin{aligned} r_i &> s_i[(1+s_m)d_{im} + s_m(d_{ik} + r_k)] \\ &\geq s_i[d_{im} + s_m(d_{km} + r_k)] = s_i(d_{im} + r_m) \end{aligned} \tag{17}$$

by the triangle inequality. Hence the new r_m requires no adjustment of any r_i for $i < m$, and (3) is satisfied for $i, j = 1 \ldots m$: this completes the proof. Note also that because of how the direct algorithm works, the iterative algorithm will compute at least two of the r_i (r_1, r_2 in the order (10)) in the initial step, and will find at least one of the other r_i at each successive step, and so can take at most $n-1$ iterations to arrive at the answer.

Can radii smaller than these r_i be used and still have the B_i cover the attractor? For any particular IFS, the answer is probably yes (as illustrated in Figure 2). The approach given here uses only minimal information about the IFS: the ordering of the w_i based on the maximal scaled distances s_i, and for each i one determining distance d_{ik} (where the maximum in (11) is achieved). Using more information it may be possible to reduce the size of the B_i. But if one considers the set of all the IFSs for which the direct algorithm yields the same r_i in the same way (i.e., same ordered s_i and same $n-1$ determining distances), then the r_i are *minimal for that set of attractors* (see Figure 3). In fact, one can construct one member of that set such that each image $w_i(A)$ of the attractor includes a point at a distance r_i from the fixed point x_i.

To construct this IFS, let $\mathbf{X} = \mathcal{R}$ with the Euclidean metric $d(x, y) = |x - y|$. Let $w_i(x) = -s_i x + (1 + s_i)x_i$, and let $x_1 = 0$, $x_2 = d_{12}$, say. Then the attractor includes the extremal points $x_{e1} = x_1 - r_1$, $x_{e2} = x_2 + r_2$, since $x_{e1} = w_1(x_{e2})$ and $x_{e2} = w_2(x_{e1})$. Place each succeeding x_m at the determining distance d_{mk} from the determining point x_k, in the opposite direction from x_{ek}. (Figure 4 illustrates the construction.) Then the attractor will include $x_{em} = x_m \pm r_m = w_m(x_{ek})$. Thus for this one-dimensional attractor A, each image $w_i(A)$ will include a point (x_{ei}) a distance r_i from x_i, so no smaller r_i would suffice.

As an example, consider the now familiar black spleenwort fern fractal of [4]. The IFS for the fern (in two dimensions) consists of affine contractions, each of which has the form

$$w_i \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a_i & b_i \\ c_i & d_i \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e_i \\ f_i \end{pmatrix} \quad . \tag{18}$$

or more compactly

$$w_i(\mathbf{x}) = M_i\mathbf{x} + \mathbf{b}_i \quad . \tag{19}$$

where M_i is the matrix and b_i is the offset vector. The various constants are given in [4], but in terms of scaling and rotating each axis, using p, q, θ, ϕ, where $a = p\cos\theta$, $b = -q\sin\phi$, $c = p\sin\theta$, $d = q\cos\phi$. The following table is adapted from [4, p. 1977]:

Map	Scalings		Rotations		Translations	
i	p_i	q_i	θ_i	ϕ_i	e_i	f_i
1	0	0.16	0	0	0	0
2	0.85	0.85	-2.5	-2.5	0	1.6
3	0.3	0.34	49	49	0	1.6
4	0.3	0.37	120	-50	0	0.44

where angles are given in degrees. The contractivity ratio s_i for an affine map w_i is the largest singular value of the matrix M_i. In the first three maps above, both axes rotate together, and so s_i is the larger of p_i, q_i. In w_4, the differential rotation causes a skewing effect, and the singular values of M_4 must be found. The simplest way for a real 2×2 matrix is first to factor out a pure rotation to give a symmetric matrix (S), then diagonalize it to find its eigenvectors (λ_1, λ_2) as shown below:

$$\alpha = \arctan\left(\frac{c-b}{a+d}\right) \tag{20}$$

$$S \equiv \begin{pmatrix} g & h \\ h & k \end{pmatrix} = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix} M \tag{21}$$

$$\beta = \frac{1}{2}\arctan\left(\frac{2h}{g-k}\right) \tag{22}$$

$$\begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} = \begin{pmatrix} \cos\beta & \sin\beta \\ -\sin\beta & \cos\beta \end{pmatrix} S \begin{pmatrix} \cos\beta & -\sin\beta \\ \sin\beta & \cos\beta \end{pmatrix} \quad . \tag{23}$$

Then $s = \max(|\lambda_1|, |\lambda_2|)$. This approach also has a nice geometrical interpretation: the effect of multiplying a vector $\mathbf{x}$ by M is to take components of $\mathbf{x}$ in the eigenvector directions, which are at an angle β relative to the coordinate axes, scale each component by the corresponding λ, and rotate the resulting vector by α.

Proceeding as above gives $s_4 = 0.379$. The fixed point $\mathbf{x}_i$ for each map

can be found by solving

$$(I - M_i)\,\mathbf{x}_i = \mathbf{b}_i \tag{24}$$

This gives the necessary starting information, summarized as "Input" in the table below. For affine maps in higher dimensions, the contractivity ratios are found by singular value decomposition, but for nonlinear maps the ratios and fixed points may be more difficult to find.

Running the direct algorithm program ("envelope") from the Appendix on this input gives the following results:

```
% envelope 4
Map 1.  Enter scale, x, y:  .16 0 0
Map 1: s = 0.160000, x = 0.000000, y = 0.000000
Map 2.  Enter scale, x, y:  .85 2.45967 10.004734
Map 2: s = 0.850000, x = 2.459670, y = 10.004734
Map 3.  Enter scale, x, y:  .34 -0.601889 1.883961
Map 3: s = 0.340000, x = -0.601889, y = 1.883961
Map 4.  Enter scale, x, y:  0.379216 0.155336 0.630251
Map 4: s = 0.379216, x = 0.155336, y = 0.630251
radii in sorted order[orig order](sorted link):
 r1[2](->2): 16.700212
 r2[4](->1): 9.993765
 r3[3](->1): 8.628835
 r4[1](->1): 4.320459
%
```

These results are illustrated in Figure 5, and tabulated below (including the D values used in re-ordering and the determining distances):

Input				Output			
i	s_i	x_i	y_i	D-order	D_{max}	d_{det}	r_i
1	0.16	0	0	4	D_{12}: 25.59	d_{12}: 10.30	4.32
2	0.85	2.460	10.005	1	D_{24}: 36.35	d_{24}: 9.65	16.70
3	0.34	-0.602	1.884	3	D_{32}: 30.26	d_{32}: 8.68	8.63
4	0.379	0.155	0.630	2	D_{42}: 36.35	d_{42}: 9.65	9.99

(While in this example the determining distance for each map derives from the same pair that gives the maximum D, that is not always the case.) Then, If we had no idea how big the fern attractor was, we could use a computational space extending from $x_{min} = x_2 - r_2 = -14.24$ to $x_{max} = x_2 + r_2 = 19.16$

and $y_{min} = y_4 - r_4 = -9.36$ to $y_{max} = y_2 + r_2 = 26.70$ to contain the entire envelope. As it turns out, this is far more space than necessary for the fern itself, but there are many other IFSs, equivalent as far as the direct algorithm is concerned, with much larger attractors (e.g., what if $\theta_2 = \phi_2 = 177.5$ instead).

To summarize, given any IFS (along with the contractivities and fixed points of each of its constituent contraction mappings), an envelope can be constructed of one ball for each map, centered on the corresponding fixed point. (In the case of affine maps in two dimensions, an explicit procedure for finding the contractivities and fixed points was given.) We have proven that the radii of the balls can be calculated by a simple algorithm (direct or iterative) such that the envelope covers the entire attractor. The spatial extent of the envelope thus gives a reliable bound on that of the attractor. (In addition, if the balls are disjoint, the attractor is totally disconnected.) While the radii found by the direct algorithm may not be minimal for the particular IFS, they are minimal for the set of all IFSs with equivalent information (in the sense described above).

ACKNOWLEGEMENTS

Thanks to Aaron Schusteff and Ismor Fischer for stimulating, informative discussions and helpful suggestions.

References

[1] Michael F. Barnsley. *Fractals Everywhere*. Academic Press/Harcourt Brace Jovanovich, Boston, 1988.

[2] Michael F. Barnsley and Stephen Demko. Iterated function systems and the global construction of fractals. *Proc. Roy. Soc. London A*. 399:243-275, 1985.

[3] Michael F. Barnsley, John H. Elton, and Douglas P. Hardin. Recurrent iterated function systems. *Constructive Approximation*, 5:3-31, 1989.

[4] Michael F. Barnsley, V. Ervin, D. Hardin, and J. Lancaster. Solution of an inverse problem for fractals and other sets. *Proc. Nat. Acad. Sci. U.S.A.*, 83:1975-1978, 1986.

[5] David Canright. FrEd: the Fractal Editor. shareware available as FREDEX.EXE from CompuServe IBMApp forum. Graphics library, 1992.

[6] Stephen Demko, Laurie Hodges, and Bruce Naylor. Construction of fractal objects with iterated function systems. *Computer Graphics*, 19:271-278, 1985.

[7] H. Joel Jeffrey. Chaos game visualization of sequences. *Computers and Graphics*, 16(1):25-33, 1992.

[8] William Douglas Withers. Newton's method for fractal approximation. *Constructive Approximation*, 5:151-170, 1989.

APPENDIX

```
/*
   The following program in C implements the direct algorithm
for determining the "envelope" of an attractor of an Iterated
Function System on R^2, given the contractivities si and the
fixed points (xi,yi).
   written by David Canright, March 1993.
*/
#include <stdio.h>
#include <math.h>
int npts, i, j, m, n, index[64], link[64];
double d[64][64], x[64], y[64], s[64], r[64], Dmax[64],
   t, tmax, dx, dy;
int input(int argc, char *argv[]);

main(int argc, char *argv[]) {

npts = input(argc, argv);   /* get si, xi, yi */
/* compute distances dij & maximal scaled Dij */
```

```
for (i = 1; i <= npts; i++) Dmax[i] = 0.;
for (i = 1; i <= npts; i++) {
   for (j = i+1; j <= npts; j++) {
      dx = x[i]-x[j]; dy = y[i]-y[j];
      d[i][j] = d[j][i] = t = sqrt(dx*dx+dy*dy);
      t = (1.+s[i])*(1.+s[j])/(1.-s[i]*s[j]) * t;
      if (t > Dmax[i]) Dmax[i] = t;
      if (t > Dmax[j]) Dmax[j] = t;
      }
   }
/* Sort by scaled distances; index points to old order */
index[1] = 1;
for (i = 2; i <= npts; i++) {
   for (m = i; m > 1 && Dmax[i] > Dmax[index[m-1]]; m--)
      index[m] = index[m-1];
   index[m] = i;
   }

/* Direct algorithm; link points to determining distance */
i = index[1]; j = index[2]; link[1] = 2; link[2] = 1;
r[1] = ( s[i]/(1.+s[i]) ) * Dmax[i];
r[2] = ( s[j]/(1.+s[j]) ) * Dmax[j];
for (m=3; m <= npts; m++) {
   i = index[m];
   tmax = 0.;
   for (n = 1; n < m; n++) {
      j = index[n];
      if ( (t = d[i][j] + r[n]) > tmax )
         { tmax = t; link[m] = n; }
      }
   r[m] = s[i] * tmax;
   }
printf("radii in sorted order[orig order](sorted link):\n");
for (i = 1; i <= npts; i++)
   printf(" r%d[%d](->%d): %f\n",i,index[i],link[i],r[i] );
return(0);
}
```

```
/* Input function: gets si, xi, yi, or uses random numbers */
/* optional arguments: number of maps, seed for random     */
int input(int argc, char *argv[]) {
double norm;
char line[81], getmore = 1;

npts  =  3;   /* default */
if (argc > 1) sscanf(argv[1],"%d",&npts);
if (argc > 2) sscanf(argv[2],"%d",&i);
srand(i);
norm = 1./(MAXINT);    /*  machine-dep. const., to normalize */
for (i = 1; i <= npts; i++) {
   s[i] = norm*rand();   /* random by default */
   x[i] = norm*rand();
   y[i] = norm*rand();
/* get numbers from stdin until blank line, then random */
   if (getmore) {
      printf("Map %d.  Enter scale, x, y: ",i);
      gets(line);
      if (line[0]) {
         sscanf(line,"%lf%lf%lf",s+i,x+i,y+i);
         if(s[i]<0.)s[i] =  -s[i];   /* enforce 0 <= s < 1 */
         while(s[i]>=1.) s[i] *= 0.1;
         }
      else getmore = 0;
      }
   printf("Map %d: s = %f, x = %f, y = %f\n",i,s[i],x[i],y[i]);
   }
return(npts);
}
```

FIGURE CAPTIONS

Figure 1. An attractor of an IFS is shown with its envelope of three disks, as computed by the direct algorithm. (This IFS uses affine maps, with $s_1 = \frac{5}{7}$, $s_2 = \frac{1}{3}$, $s_3 = \frac{1}{2}$, $\mathbf{x}_1 = (0,0)$, $\mathbf{x}_2 = (4,0)$, and $\mathbf{x}_3 = (0,3)$.

Figure 2. For an equilateral Sierpinski's Triangle (where $w_i(\mathbf{x}) = \frac{1}{2}\mathbf{x} + \frac{1}{2}\mathbf{x}_i$), the $r_i = d_{ij}$ by this method: in this particular case the radii could be half as large.

Figure 3. Same s_i and $\mathbf{x}_i$ as the previous figure, but here $w_i(\mathbf{x}) = -\frac{1}{2}\mathbf{x} + \frac{3}{2}\mathbf{x}_i$; in this case the r_i found above are minimal.

Figure 4. A one-dimensional attractor constructed from the following ordered data: $s_1 = \frac{1}{3}$, $s_2 = \frac{1}{4}$, $s_3 = \frac{1}{7}$, $d_{12} = 1$, and $d_{13} = \frac{1}{2}$. The r_i are minimal for such attractors.

Figure 5. Barnsley's fern [4] and its envelope (see text).

Figure 1

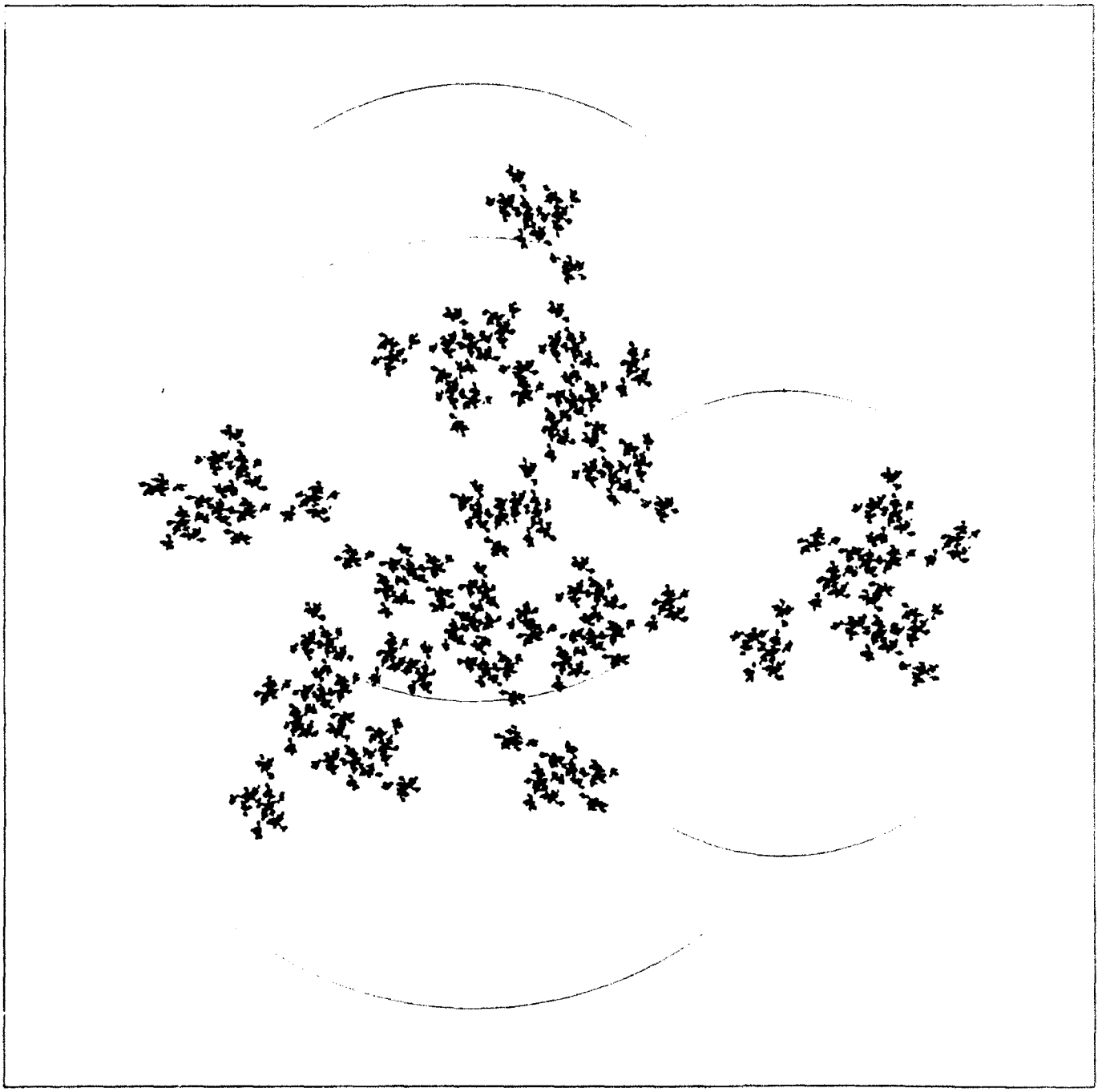

Figure 2

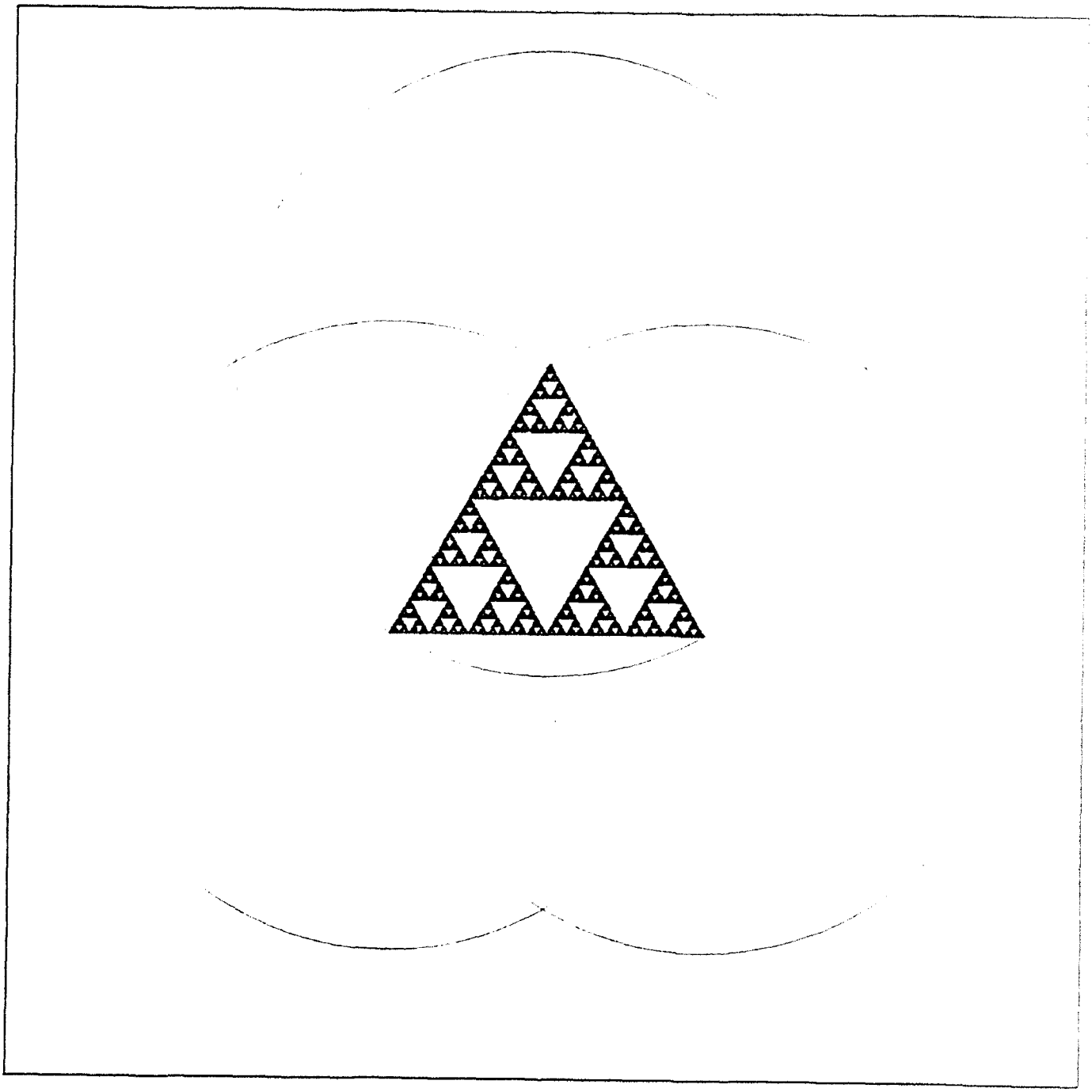

Figure 3

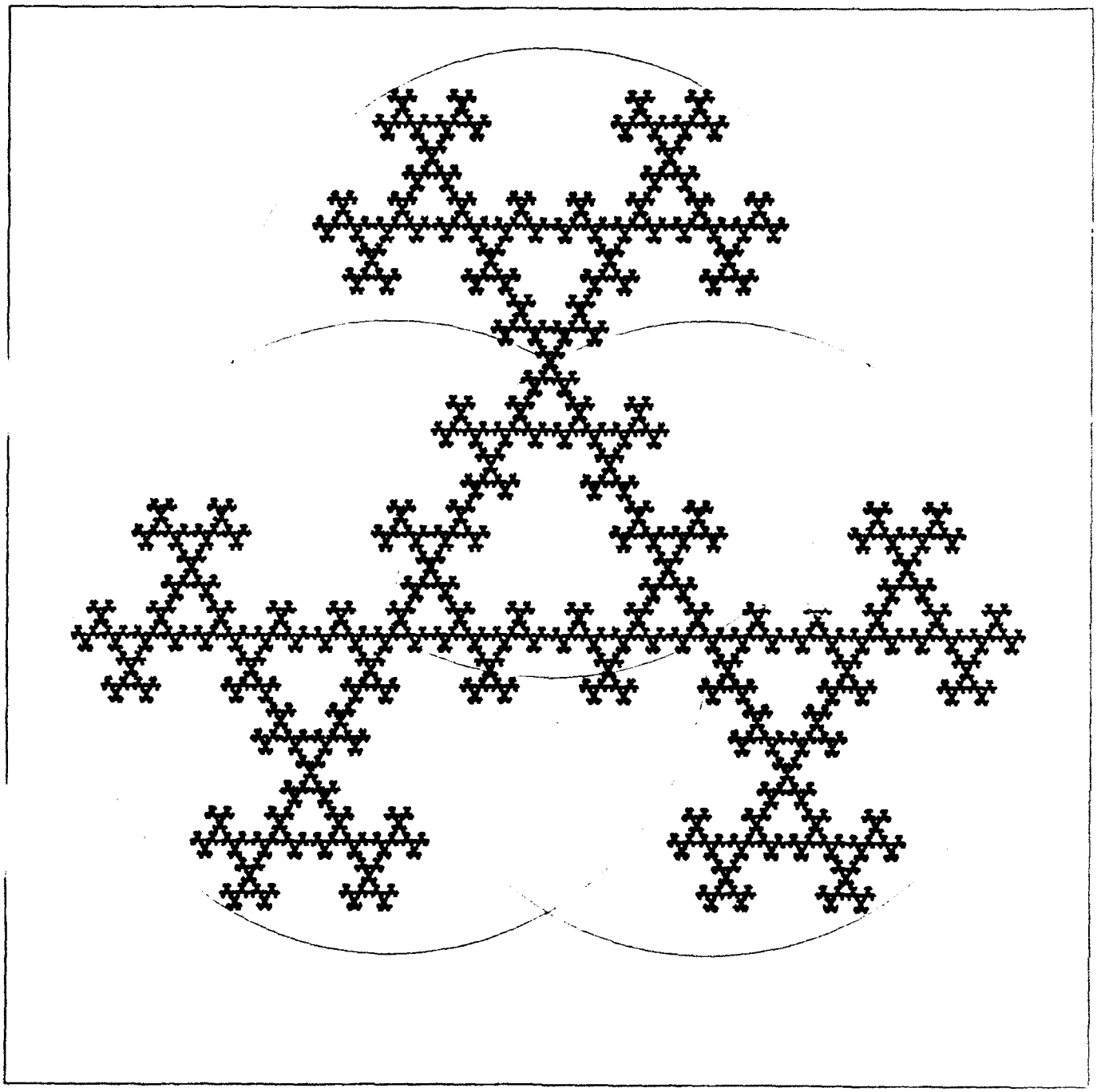

Figure 4

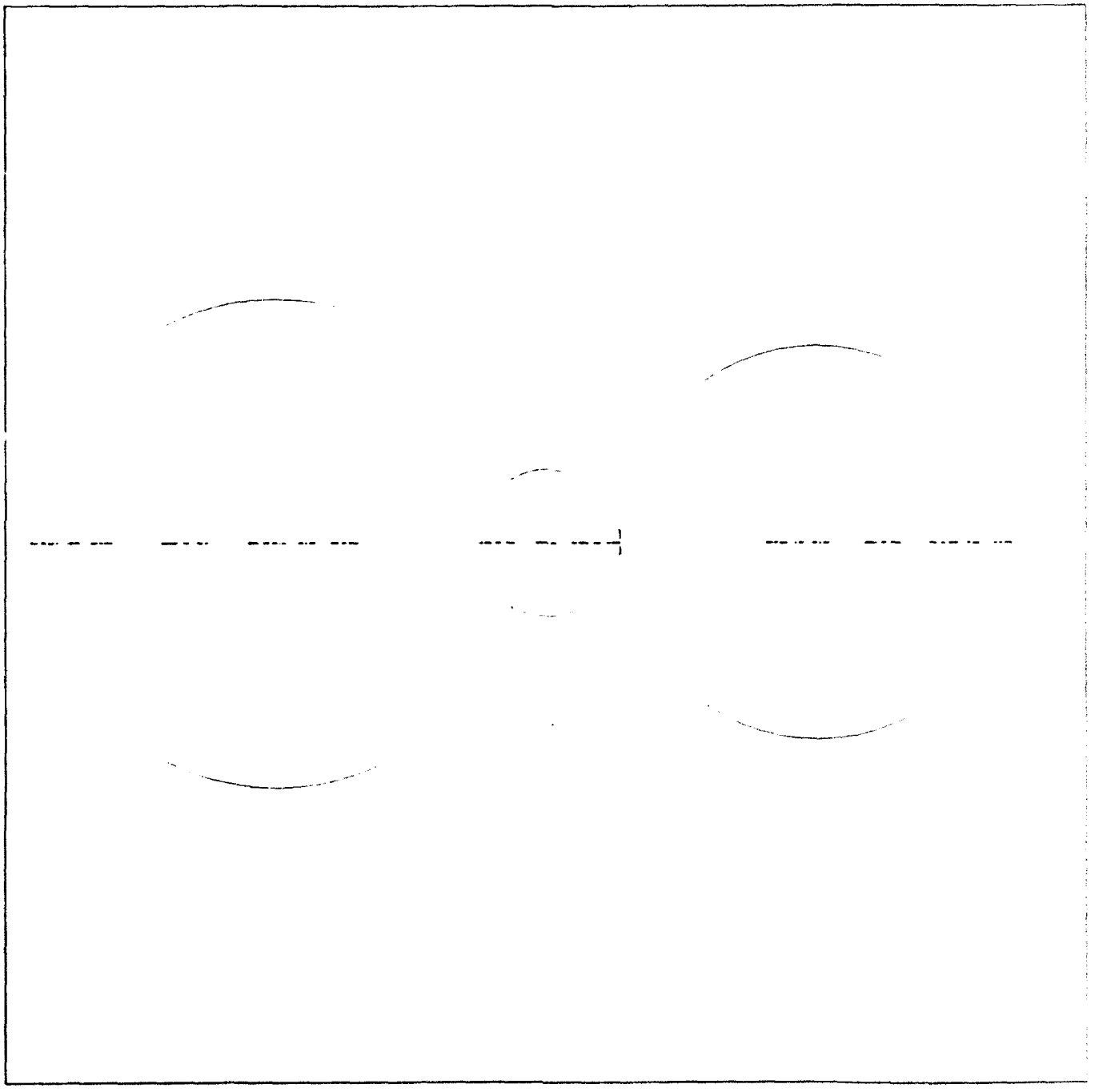

Figure 5

DISTRIBUTION LIST

Director (2)
Defense Tech Information Center
Cameron Station
Alexandria, VA 22314

Research Office (Code 81) (1)
Naval Postgraduate School
Monterey, CA 93943

Library (Code 52) (2)
Naval Postgraduate School
Monterey, CA 93943

Professor Richard Franke (1)
Department of Mathematics
Naval Postgraduate School
Monterey, CA 93943

Dr. Neil L. Gerr (1)
Mathematical Sciences Division
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5000

Dr. Richard Lau (1)
Mathematical Sciences Division
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5000

Aaron Schusteff (1)
Department of Mathematics
University of California
Los Angeles, CA 90024

Ismor Fischer (1)
Department of Mathematics
Naval Postgraduate School
Monterey, CA 93943

Professor David Canright (15)
Department of Mathematics
Naval Postgraduate School
Monterey, CA 93943