CODING NON-HOMOGENEOUS FRACTALS USING NEURAL NETWORKS

Rashad A. Al-Jawfi

Department of Mathematics and computer Science, Faculty of Science, Ibb university, Yemen

Abstract:

In this work we formed a neural network to coding homogeneous iterated function system. Our approach to this problem consists of finding an error function which will be minimized when the network coded attractor is equal to the desired attractor. Firstly we start with a given fractal attractor; find a set of weights for the network, which will approximate the attractor. Secondly we compare the consequent image using this neural network with the original image, with the result of this comparison we can update the weight functions and the code of (IFS). A common metric or error function used to compare between the two image fractal attractors is the Hausdorff distance. The error function gets us good means to measurement the difference between the two images. The distance is calculated by finding the farthest point on each set relative to the other set and returning the maximum of these two distances.

Keywords: Image coding, fractal, neural networks, inverse problem

1. Introduction:

Fractals have been used for modeling natural images [8]. These natural images have common properties such that the magnified local subsets look identical to the whole set. This property is referred to as self-similarity. This means that they usually contain small copies of themselves buried deep within the original. On other hand, Neural networks have been hailed as the paradigm of choice for problems which require "Human Like" perception. A network could be performing its function perfectly, responding correctly to every input

that it is given, however its internal workings could still be construed as a black box, leaving its user without knowledge of what is happening internally. We are interested in different ways to tease neural networks open, to analyze what they are representing, how they are "thinking". In this work we present a novel algorithm to introduce the code of the iterated function system which generate a fractal image. Its features being that it is exact fully describing a network's function, concise, not an incremental collection of approximations and direct mapping a network's input directly to its output.

2. Preliminaries

In this section, we _rst summarize the existing knowledge which is necessary for understanding this paper.

2.1. Definition of Non-homogeneous IFSs

In this paper, we limit our consideration to an IFS consisting of affine maps with associated probabilities in the complex plane. Within this important class, we will discuss a particular IFS with non-uniform (nonhomogeneous) contraction mappings and unequal probabilities.

The IFS is called a non-homogeneous IFS with unequal probabilities, or non-homogeneous IFS for short, and defined by

$$\{C; \beta_n + \gamma_n; P_n, |\beta_n| < 1, \beta_n, \gamma_n \in C, P_n \in R, n = 0, \dots, N-1\}. \qquad (1)$$

Where N denotes the order of the IFS; C denotes the set of complex numbers; β_n and γ_n denote the deformation and displacement coefficients of the IFS's respectively. P_n denotes the associated probability, which controls the gray level of the reconstructed image. A set of coefficients $\{\beta_n, \dots, \beta_{N-1}, \gamma_1, \dots, \gamma_{N-1}, P_1, \dots, P_{N-1}\}$ is referred to as an IFS code. A fractal image generated by a homogeneous IFS of order N is called a homogeneous fractal image of order N.

2.2. Review of Basic IFS Properties

Let H(C) denote a set of images in the complex plane. Put $w_n(z) = \beta_n(z) + \gamma_n (n = 0, \dots, N-1)$. Then $w_n(z): H(C) \to H(C)$ defined by

$$\{w_n(B) = w_n(x); x \in B, \forall B \in H(C)\}$$

is a contraction mapping on the Hausdorff distance. Thus, W has a unique fixed point $F \in H(C)$, which obeys

$$F = W(F)$$

and is given by

$$F = \lim_{i \to \infty} W^i(B)$$

for any $B \in H(C)$, where W^i denotes i iteration of W. The fixed point F is called the attractor of the IFS. The IFSs can be extended to IFSs with probabilities in order to represent the gray level of an image. Then the fractal image is defined as the fixed point of a contraction mapping on the space of the probability measures $P(C)$.

2.3. The inverse problem of fractals

In several mathematical fields, many problems have inverses, for example integration, in a certain sense, is an inverse problem for differentiation, the problem of determining the forces under the action of which a particle moves along a given curve, is another example of an inverse problem in dynamics of particles, etc. In an analogous way, the problem of generating fractals by the use of IFS, calls for an inverse problem, namely: For a given set in , construct a suitable IFS whose attractor is the given set (to a certain desired degree of accuracy)[2]. The tackling of this inverse problem, as it stands, is difficult, if it is not impossible. However, if the given set is self-similar, then the required construction is almost straightforward. The IFS can be found easily by making mathematical translation of the property of self-similarity.

2.4. Collage theorem [2]

Let (X, d) be a complete metric space, let $L \in H(X)$ be given, and let $\varepsilon \geq 0$ be given chose an iterated function system (IFS) $\{w_1, w_2, \ldots, w_N\}$ with contractivity factor $0 \leq s \leq 1$ so that

$$h\left(L, \bigcup_{n=1}^{N} w_n(L)\right) \leq \varepsilon$$

where h(d) is the hausdorff metric. Then

$$h(L,A) \leq \frac{\varepsilon}{1-s}$$

where A is the attractor of the IFS.

3. Using Neural Networks for coding Homogeneous IFS

3.1. design of neural network

The Hopfield network uses the fixed points of the network dynamics to represent memory elements. Networks studied by [6,10] use the current activation of the network as a state in a state machine while using the dynamics of the network which is treated as an Iterated Function system that is coding for its fractal attractor[3].

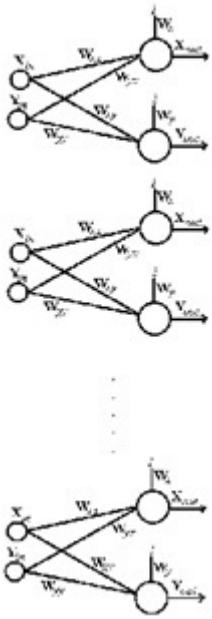


Fig. 1: Neural Network

Melnik [9], applying one of the transforms on a random point for a number of steps, until it converged. There is still no general algorithm for fractals image coding, the problem we want to solve in this paper, which is given a fractal attractor, find a set of weights for the neural network which will approximate the attractor. A neural network (figure 3.1) consists of two input units and two output units and six weights for all transform (IFS) represent scalar function. The transform is selected randomly, and all input neurons receive a coordinate of each point of fractal image, one neuron for x coordinate and the other for y coordinate for each transform. And return as x and y output, consists of TanSigmoid function [7,9] with a bias (figure 3.2). The equations of x and y output are given as

$$X_{out} = \frac{1 - e^{(-WX)}}{1 + e^{(-WX)}}$$

and

$$Y_{out} = \frac{1 - e^{(-WY)}}{1 + e^{(-WY)}}$$

where

$$WX = X_{in}W_{XX} + Y_{in}W_{YX} + W_X$$

and

$$WY = X_{in}W_{YX} + Y_{in}W_{YY} + W_Y$$

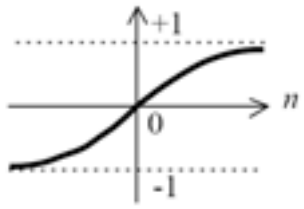


Fig. 2: Sigmoid Function

At the last of this operation for large number of points with random iterations, we get an image. Of course this image is different in general with the image we want to find the iterated function system (IFS) of. Then we must update the weight functions of the neural network to get better approximation to the target image.

This change of weights is depending with the measure of the difference between the tow images. This difference is known as error function, which must minimize with every update of weight functions. The error function used to compare fractals attractors, is the hausdorff distance[2,3,4]. The distance between two images A and B is calculated as following: We first calculate the distance between the element $a \in A$ and the set B which is the smallest distance between a and each element $b \in B$.

$$d(a,b) = min\{|a - b|; b \in B\}$$

Then the distance between A and B is the largest distance for each element $a \in A$.

$$d(A,B) = max\{d(a,B); a \in A\}$$

Also the distance between B and A is equal to

$$d(B,A) = max\{d(b,A); b \in B\}$$

And then the distance between two sets is the largest of tow distances $d(A,B)$ and $d(B,A)$

$$H(A,B) = max\{d(A,B); d(B,A)\}$$

Our error function is defined as:

$$E(T,A) = \sum_{i} d(T_i(x,y),A) + \sum_{x,y} d((x,y),T(A))$$

Where $T_i(x,y)$ is the image of the point (x,y) with respect to the transform T_i. And $T(A)$ is the all image of A. The value of the error function with respect to the iteration of neural network is shown in table (3.1).

iteration	Error
1	2.22790405672408
18	9.42187049043963E-03
39	4.47490221672919E-03
50	3.50195369895845E-03
100	1.74568350616262E-03
135	1.28726669405809E-03
150	1.15642714675241E-03
1500	1.11474721E-03
2500	1.01474721E-03
5000	1.01621654E-04
10000	1.0121654E-05

Table 1: the values of error function with some iterations

And fig (3.3) show the relation between number of iterations and the error of neural network.

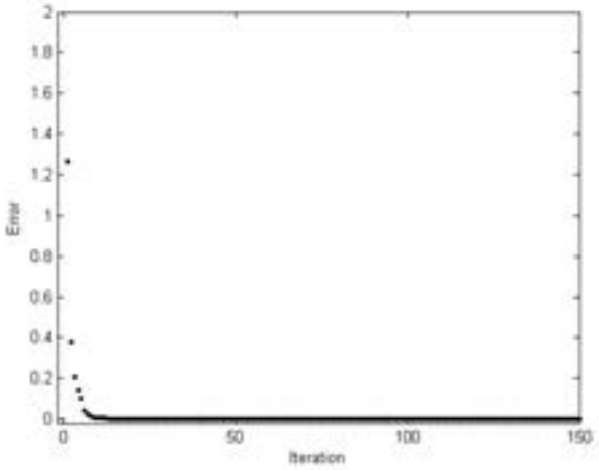


Fig. 3: error function

3.2. The procedure of fractals image coding

1. Input: fractal image, random weights, n the number of (IFS), ε, and μ.

2. compute the error function E(A,T).

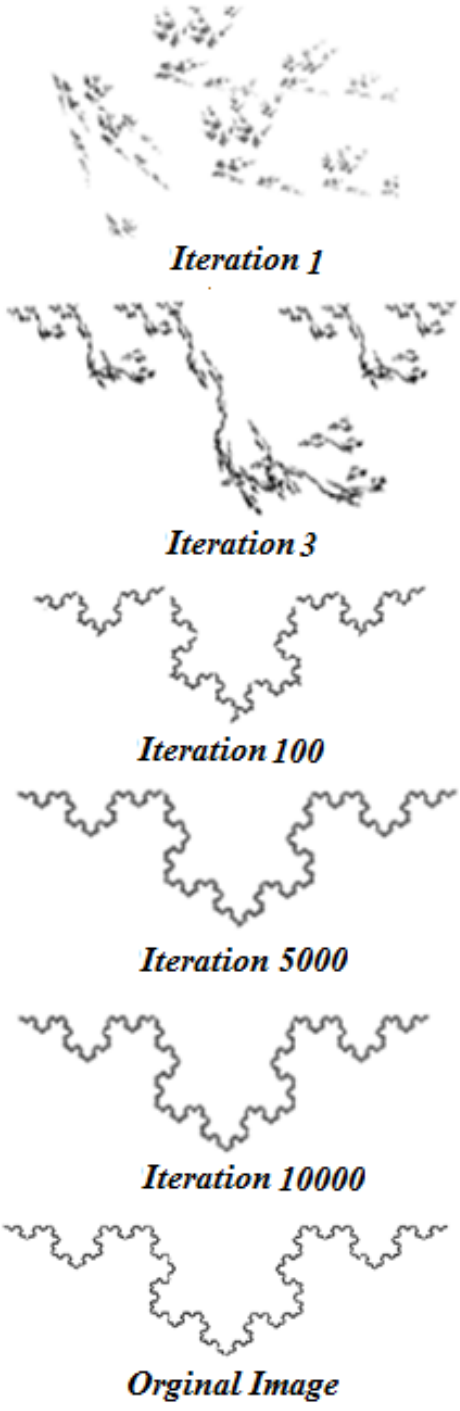
3. If $\varepsilon > \mu$ then.

3.1. Compute $\frac{\partial E}{\partial X_{ij}}$

3.2. Update the weights.

3.3. Go to step 2

4. Else stop



4. 4 Future Work

This paper focused on the inverse problem of fractals with related to iterated function systems for 2- dimension linear fractals. Solving the inverse problem of 3-dimension fractals remains an open problem, as does the same problem of non linear iterated function systems.

References:

M. F. Barnsley, Fractals Everywhere, Academic Press, New York, 1988.

M. F. Barnsley, V. Ervin, D. Hardin, and J.J Lancaster, Solution of an inverse problem for fractals and other sets, Proceedings of the National Academy of Science 83, 1986, pp. 117-134.

F. Bruno, Solving the inverse problem for function / image approximation systems II. Algorithm and computations,fractals,vol.2 No.3, 1994, pp.335-346.

L. Fausett, Fundamentals of Neural Networks, Pretice Hall.Inc., 1994.

C. L. Giles, C. B. Miller, D. Chen, H. H. Chen, G. Z. Sun and Y. C. Lee, Learning and extracting _nite state automata with second-order recurrent neural networks, Neural Computation, vol. 4, no. 3, 1992, pp. 393-405.

J. E. Hutchinson, Fractals And Self-similarity, Indiana University Mathematics Journal, vol. 30, No. 4, July-August, 1981, pp.713-747.

O. Melnik, Representation of information in neural networks, Ph.D. thesis, Department of Computer Science, brandies university, 2000.

J. B. Pollack, The induction of dynamical recognizers, Machine Learning, vol. 7, 1991, pp.227-252.