Optimization of Affine Fractal Interpolation Function for Graph Fitness using Genetic Algorithms

JIN MIN, WANG QIN, XI LIFENG
Computer Science and Information Technology College
Zhejiang Wanli University
Qianhu South Road 8, Ningbo
CHINA

Abstract: - Fractal is a tool in data fitness and interpolation. It's better to fit complex graphs. But optimization of fitness has proven to be a challenging problem in application of fractal. In this paper, a description of a genetic algorithm (GA) and the software written to optimize effect on self-affine fractal interpolation function (AFIF) implement is given. The software developed was tested on real graph. The method was capable of finding optimization fractal function, as the results presented demonstrate.

Key-Words: - Genetic Algorithm; Fractal; AFIF; Fitness; Optimization;

1 Introduction

One of the most interesting and important problems in application on self-affine fractal interpolation function (AFIF) is to fit rough curves and vibrating data, such as mountain range outlines, electrocar-diograms ...etc. AFIF is able to create complex graph by setting several parameters. As for the more information of AFIF, please refer to [1-2] To fit a presented graph, it is not rectify parameters enough. It is a challenging problem to find the optimization in areas of solution space.

Genetic algorithms (GAs) can yield accurate results providing that they are fed with suitable parameters to start with. The GA has been used to solve difficult problems with objective functions which usually are multi-modal, discontinuous, and nondifferentiable. These algorithms maintain and manipulate a family, or population, of solutions and implement a "survival of the fittest" strategy in their search for better solutions. This provides an implicit as well as explicit parallelism that allows for the exploitation of several promising areas of the solution space at the same time.

Section 1 presents the basic GA, and in Section 2 the step of optimization of affine fractal interpolation function for graph fitness using a GA is described. Section 3 briefly describes the code, presents the list of parameters of the GA implementation and experiment results. Finally, some problems that deserve to be noted in implementing this method and their mend are given in section 4.

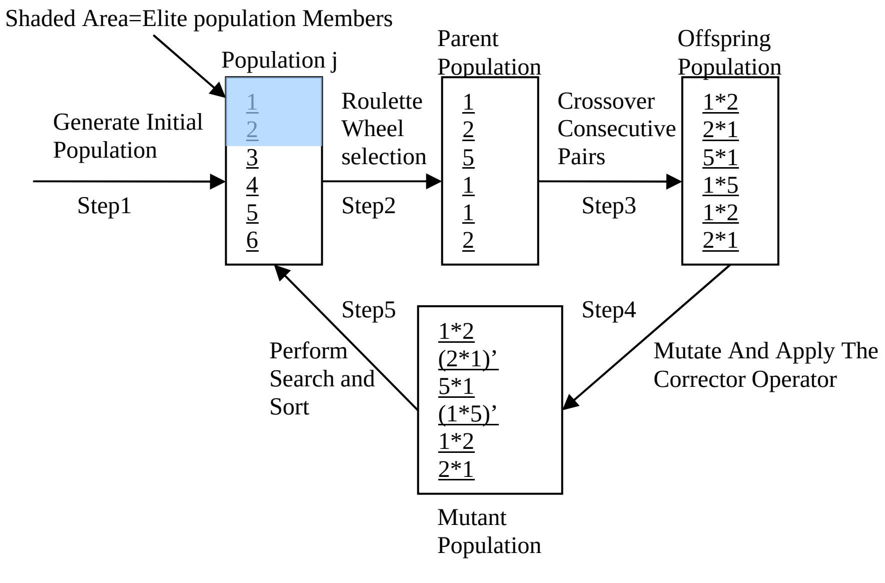

Fig.1. Schematic flow diagram for the GA program

2 The GA

The GA searches the solution space of a function through the use of simulated evolution, i.e., the survival of the fittest strategy. In general, the fittest individuals of any population tend to reproduce and survive to the next generation, thus improving successive generations. However, inferior individuals can, by chance, survive and also reproduce. The GA has been shown to solve linear and nonlinear problems by exploring all regions of the state space and exponentially exploiting promising areas through mutation, crossover, and selection operations applied to individuals in the population. As for the more complete discussion of the GA, please refer to [3-6].

The way in which our GA program operates is shown as a schematic flow diagram in Fig. 1. The characteristics and a summarized algorithm of a GA are described in the following.

2.1 Characteristics of the GA

The GA is a search technique, based on the principles of natural evolution. The important terms and principles as follow:

(1) Codes on solution have evolution.

The codes of optimal problem solution are called chromosomes. Since the solution is coded, the research on optimization of function is based on codes. The one important topic of a GA is encoding and decoding.

(2) Law of natural selection decides which chromosomes have more offspring than others.

In the GA, fitness function is created by objective function, which will be optimized. The fitness functions ensure that the more chromosomes are fitting, the more offspring is generated.

(3) New chromosomes retain characteristics of parent chromosomes.

Crossover takes two chromosomes and produces two new chromosomes. The two new chromosomes retain characteristics of parent chromosomes from parent chromosomes gene.

(4) New chromosomes is different from parent chromosomes

Randomly mutant made the difference.

2.2 A Common Algorithm

A GA is summarized as follow, and each of the major components is discussed in detail below.

(1) Supply an initial population pop(1) of N individuals and respective codes of function solutions, t: =1; (like step 1 of Fig.1)

(2) Calculate each of chromosomes $\text{pop}_i(t)$, which is in population pop(t), fitness function

$$f_i = \text{fitness}(\text{pop}_i(t));$$

(3) Calculate each of chromosomes selection probability

$$p_i = \frac{f_i}{\sum_{j=1}^{N} f_j},\ i=1,2,\cdots,\ M, \tag{1}$$

By (1), a new population is generated from pop(t) (like step 2 of Fig.1)

$\text{newpop}(t+1)=\{\ \text{pop}_j(t)|j=1,2,\cdots,\ N\}$;

Remark1: A chromosome in population pop (t) may be repeatedly selected.

(4) Generate a population crosspop (t+1) by crossing consecutive pairs chromosomes from newpop(t+1). Crossing probability is $P_{c.}$ (like step 3 of Fig.1)

(5) Mutate a gene of a chromosome by small probability p. Generate a population mutpop(t+1), t:=t+1, a new population pop(t)=mutpop(t) (like step 4 of Fig.1)

(6) Repeat step (3) until termination

(7) Print out best solution found

The use of a genetic algorithm requires the determination of six fundamental issues: the creation of the initial population, chromosomes representation, the selection method of the newpop population, the genetic operators making up the crossover function, the mutant method, and termination criteria. The rest of this section describes each of these issues.

Issue1. Initial population.

The GA must be provided an initial population as indicated in step (1). The most common method is to randomly generate solutions for the initial population. However, since GAs can iteratively improve existing solutions (i.e., solutions from other heuristics and/or current practices), the beginning population can be seeded with potentially good solutions, with the remainder of the population being randomly generated solutions.

Issue2. Chromosomes representation

For any GA, a chromosome representation is needed to describe each individual in the population of interest. The representation scheme determines how the problem is structured in the GA and also determines the genetic operators that are used. Each individual or chromosome is made up of a sequence of genes from a certain alphabet. An alphabet could consist of binary digits (0 and 1), floating point numbers, integers, symbols (i.e., A, B, C, D), matrices, etc. One useful representation of an individual or chromosome for function optimization involves genes or variables from an alphabet of floating point numbers with values within the variables upper and lower bounds.

Issue3. The selection method of the newpop population

A common selection approach assigns a probability of selection, Pi, to each individual, i based on its fitness value as indicated in step(2)(3). The chromosome, which has larger fitness function value, has more chance to be selected in new population. This selection method is call roulette wheel selection.

Issue4. The genetic operators making up the crossover function

Crossover (or mating) is the way in which "genetic" information from two parent chromosomes is combined to generate "offspring". In step(4), the parent chromosomes and crossing size are randomization.

Issue5. The mutant method

While the crossover operation leads to a mixing of genetic material in the offspring, no new genetic material is introduced, which can lead to lack of population diversity and eventually "stagnation"--where the population converges on the same, nonoptimal solution. The GA mutation operator helps to increase population diversity by introducing new genetic material. The common method is select one genetic to mutate by minimal probability.

Issue6. Termination criteria

The GA moves from generation to generation selecting and reproducing parents until a termination criterion is met. The most frequently used stopping criterion is a specified maximum number of generations. Another termination strategy involves population convergence criteria. In general, GAs will force much of the entire population to converge to a single solution. When the sum of the deviations among individuals becomes smaller than some specified threshold, the algorithm can be terminated.

3 Optimization of AFIF Using the GA

3.1 Formulating the Optimization Problem

Given points $\{(x_i, y_i)\}_{i=0}^{N}$ in the plane, we suppose $\{\omega_1, \omega_2, \dots, \omega_N\}$ is an *iterated function system* satisfying

$$\omega_i \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a_i & 0 \\ c_i & d_i \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} g_i \\ h_i \end{pmatrix}, \qquad (2)$$

$$\omega_i \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} x_{i-1} \\ y_{i-1} \end{pmatrix}, \ \omega_i \begin{pmatrix} x_N \\ y_N \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \end{pmatrix}, \qquad (3)$$

where $|d_i| < 1$ and $a_i \in (0,1)$ for any i with $1 \le i \le N$. By (3) and (4), we notice that $\{\omega_i\}_{i=1}^{N}$ is determined by $\{(x_i, y_i)\}_{i=0}^{N}$ and. We always call $\{d_i\}_{i=1}^{N}$ vertical factors and $\{(x_i, y_i)\}_{i=0}^{N}$ interpolation points respectively.

Definition 1. Suppose $f(x)$ is a continuous function on the interval $[x_0, x_N]$. Let

$$\Gamma = \{(x, f(x)) : x \in [x_0, x_N]\}$$

be the graph of $f(x)$. We say that $f(x)$ is an **affine fractal interpolation function(AFIF)**, if

$$\Gamma = \bigcup_{i=1}^{N} \omega_i(\Gamma). \qquad (4)$$

For AFIF defined by (2)-(4), the dimension $\dim_B \Gamma$ of the graph Γ satisfies the following dimension formula ([1]):

$$\sum_{i=1}^{N} |d_i| \cdot |a_i|^{\dim_B \Gamma - 1} = 1. \qquad (5)$$

Remark 1: Formula (6) holds when the interpolation points do not lie in a line simultaneously and $\sum_{i=1}^{N} |d_i| > 1$. Any connected part of the graph Γ of AFIF has the same dimension $\dim_B \Gamma$.

From the concept of AFIF, if we have several interpolation points from original graph and suitable vertical factors, we can create graph, which approximate to the original graph.

3.2 The GA's Application on AFIF for Graph Fitness

Step1. Select a graph as a original graph. Compress axis of abscissas in the interval [0,1]. Remain aspect ratio.

Step2. Bisect axis of abscissas into *EXN* intervals. Let Terminal of every interval is interpolation points *EXP*$_i$, *i*=0,2⋯⋯*EXN*. Select suitable vertical factor Exd$_0$(exi),exi=1,2, ⋯ ⋯ *EXN*. Encode the vertical factors in *EXL* bits binary code. We use iteration method of AFIF to draw fractal graph. Select *EXM* sample point in every interval. Calculate sample variance *EXS*$_0$.

Step3. Repeat step 2 for *EXN*$_c$ times. Select randomly vertical factors Exd$_{\text{exj}}$(exi)(exj is repeat times) in every times. Calculate sample variance *EXS*$_j$.

Step4. The initial population have *EXN*$_c$ chromosomes. A chromosome is described by vertical factors codes. Length of a chromosome is *EXN*EXL*. The fitness function of the chromosome No j is *EXF*$_j$*=1/ EXS*$_j$。 From Section 2, we can evaluate the initial population until termination.

Step5. From best vertical factors found and AFIF Interpolation points, we can create best fitness graph for original graph.

4 Experiment and Result

4.1 Create Initial Population by Step1-Step3

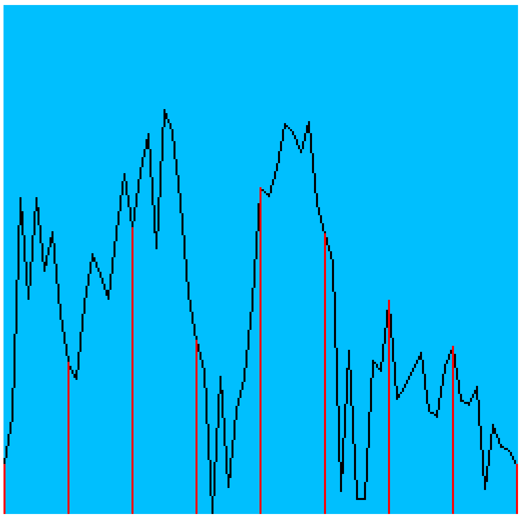

Fig.2 Original graph A

Let intervals number *EXN*=8, length of vertical factors code *EXL*=4, sample point in every interval *EXM*=16, chromosomes of population number *EXNc*=20.Here we give the original graph A, which has been compressed. to obtain *EXN* interpolation points, EXP_i, i=0,2···*EXN*. By step2, their axes are

EXP_0=(0,0.1),EXP_1=(0.125,0.3),
EXP_2=(0.25,0.56),EXP_3=(0.375,0.34),
EXP_4=(0.625,0.64),EXP_5=(0.5,0.55),
EXP_6=(0.75,0.42),EXP_7=(0.875,0.33),
EXP_8=(1,0.1)

From randomization algorithm, we get vertical factors Exd_0(exi)(exi=1,2,……*EXN*) for each interval. For example,

$Exd_0(1)$=0.559,$Exd_0(2)$=0.677,
$Exd_0(3)$=-0.573,$Exd_0(4)$=-0.574,$Exd_0(5)$=0.18,
$Exd_0(6)$=0.803,$Exd_0(7)$=-0.448,$Exd_0(8)$=0.18

Vertical factors must satisfy in formula (2). By function x'=(x+1)/2 ($x \in [-1,1]$), we can translate vertical factors in the interval [0,1]. Translate decimal to binary. We make up vertical factors code by four binary digital after decimal point. For example, $Exd_0(1)$=0.559, the code is 1100. A chromosome of population is made up by orderly jointing $Exd_{exj}(1)$ to $Exd_{exj}(8)$. For example, orderly jointing $Exd_0(1)$ to $Exd_0(8)$ is 11001101001100111001111001001001, which is a chromosome code.

From AFIF and factors, which are described hereinbefore, we can generate fractal graph. Calculate sample variance.

The initial population we selected as table 1.

Table 1. The initial population

No	Chromosome	*EXS*
1	01001001110100100011010100101110	0.05689
2	01000001100001111101000101110101	0.05449
3	10101100011101001010110100000000	0.06826
4	11010010110001111010111010000110	0.06264
5	01001101101101000111110000100010	0.07068
6	01000101011010010010100001100111	0.07846
7	10001010101111000011100100001101	0.08256
8	11010110101010010000011010001001	0.05262
9	10011010110001110001000100001011	0.04716
10	11010001001010100010001010000011	0.07849
11	00111100000010111111000000011101	0.09056
12	10101101100111010110100010101101	0.07839
13	01111001000100001010110011101001	0.09258
14	11000101000011000111100101110101	0.08794
15	11011101101010100000010110100001	0.06721
16	00010011000011010001011000110111	0.14221
17	00101010101000101011001101111100	0.04761
18	01110110100111101000000000011000	0.06246
19	10101011111000101001000110011110	0.02658
20	00010111110111100110110100101010	0.12389

4.2 The Evolution of Population

Let crossing probability P_c=1, mutant probability *p*=0.05. Population evaluates 40 generations. We translate each four binary digitals of binary code of new generation chromosomes to decimal, which can be translated into the interval [-1,1] from function x' =2x -1($x \in [0,1]$). That is vertical factor for each interval. By step4, we can generate the final population as table 2. Using Vertical factors and interpolation points, AFIF create the optimization graph as fig.3.

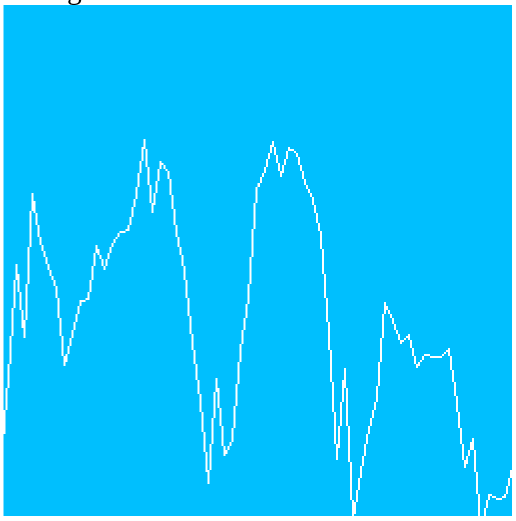

Fig.3. Optimization graph of AFIF a

Table 2. The final population

No	Chromosome	*EXS*
1	111010011100001010000000101110101	0.00621
2	111010011100001010100000101110101	0.00498
3	111010011100001010100000101110101	0.00498
4	111010011100001010100000101110101	0.00498
5	111010011100001010100000101110101	0.00498
6	111010011100001010100000101110101	0.00498
7	111010011100001010100000101110101	0.00498
8	111010011100001010100000101110101	0.00498
9	111010011100001010100000101110101	0.00498
10	111010011100001010100000101110101	0.00498
11	111010011100001010100000101110101	0.00498
12	111010011100001010100000101110101	0.00498
13	111010011100001010100000101110101	0.00498
14	111010011100001010100000101110101	0.00498
15	111010011100001010100000101110101	0.00498
16	111010011100001010100000101110101	0.00498
17	111010011100001010100000101110101	0.00498
18	111010011100001010100000101110101	0.00498
19	111010011100001010100000101110101	0.00498
20	111010011100001010100000101110101	0.00498

4.3 The Result of the Experiment

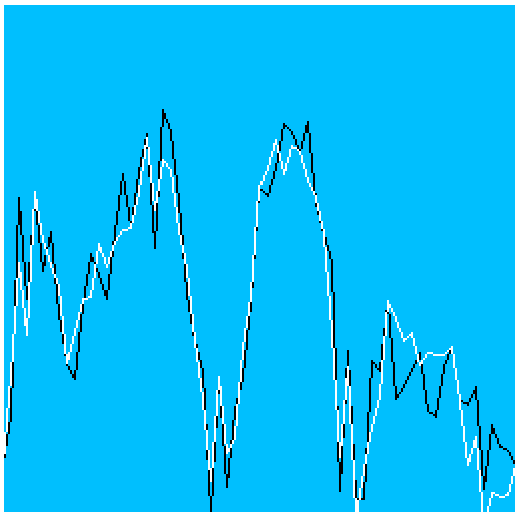

Fig.4 Original graph A and optimization graph of AFIF a

Coupling GA with AFIF yields optimization solution for graph fitness.

5 Conclusion

The experiment we have performed showed that GAs are powerful variable selection tools for graph fitness. Coupling GA with AFIF yields more precise predictions than other methods. Application of an AFIF, as a flexible nonlinear calibration, to the data selected by GA also improves the performance of the model considerably.

Some problems deserve to be noted in our experiment. The space of the initial population must be satisfied. We should use schema theorem to generate the initial population. If selection method isn’t suitable, convergence will be quick. The result of GA may be not the best. Using simulated annealing algorithm may be useful. Termination criteria should be studied. For example, we should accord sample variance to terminate implementation. However, the specific advantages of the proposed methods are low cost, simplicity, and rapidness.

References:

[1] Barnsley, M. F., Fractal functions and Interpolation. Constr. *Approx*., No. 2, 1986, pp. 303–329.

[2] Dalla L. and Drakopoulos V., On the parameter identification problem in the plane and the polar fractal interpolation functions. J. *Approx. Theory*, Vol.101, 1999, pp.289–202.

[3] Rivera SL, Karim MN, Use of micro-genetic algorithms in bioprocess optimization. In: *Proc 12th IFAC World Congress*, Vol.8, 1993, pp.217–220.

[4] Angelov P, Guthke R, A genetic-algorithm-based approach to optimization of bioprocesses described by fuzzy rules, *Bioprocess*, Vol.16, 1997, pp.299–303.

[5] Roubos JA, van Straten G, van Boxtel AJB, An evolutionary strategy for fed-batch bioreactor optimization; concepts and performance. J *Biotechnol* , Vol.7, No.2, 1999, pp.173–187.

[6] Na JG, Chang YK, Chung BH, Lim HC, Adaptive optimization of fed-batch culture of yeast by using genetic algorithms. *Bioproc Biosyst*, Vol.24, 2002, pp.299–308.