To the Graduate Council:

I am submitting herewith a thesis written by Jenny Juliany entitled "Portraits of Genetic Algorithms." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.

Dr. Michael Vose, Major Professor

We have read this thesis
and recommend its acceptance:

Accepted for the Council:

Associate Vice Chancellor
and Dean of The Graduate School

PORTRAITS OF GENETIC ALGORITHMS

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Jenny Juliany

August 1993

ACKNOWLEDGEMENTS

During the completion of this thesis, there were many people who contributed in various ways. First, I would like to thank my senior committee member, Dr. Michael Vose, without whose guidance and patience this thesis would not have been possible.

I am grateful to Dr. Michael Berry and Dr. Jean Blair for their time in reading this thesis as well as for their helpful suggestions. I extend my gratitute to Chandra Tan and Hai Li for their help, support, and encouragement.

Last but not least, my very special thanks go to my family for their infinite faith and support.

Abstract

The classical theoretical foundations of Genetic Algorithms are based on the schema theorem. This theorem is an inequality first discussed by Holland in 1975. In 1990, Vose developed the exact schema theorem, replacing the inequality by an equality. This thesis explores some graphical properties of the exact schema theorem by computing its associated fractal.

TABLE OF CONTENTS

CHAPTER		PAGE
1.	**Introduction**	**1**
1.1	General Description of Genetic Algorithms	1
1.2	A Simple Genetic Algorithm	2
1.3	Thesis Overview	5
2.	**Holland's Schema Theorem : An Overview**	**6**
2.1	The Concept of Schema	6
2.2	The Schema Theorem	7
2.3	The Minimum Deceptive Problem	10
3.	**Exact Schema Theorem: The Infinite Population Model**	**13**
3.1	Overview	13
3.2	The Infinite Population Model	13
3.2.1	Selection Operator	14
3.2.2	Recombination Operator	14
3.2.3	Operator $\mathcal{G}$	16
4.	**The Backward GA**	**18**
4.1	Overview	18
4.2	The Inverse $\mathcal{G}$	19

CHAPTER PAGE

4.3 The Big Bang 21

4.4 Floating-Point Precision 22

5. Program Implementation 24

5.1 Overview 24

5.2 Details of Implementation 24

5.2.1 Finding Fixed Points 25

5.2.2 Finding Inverse Points 25

5.2.3 Finding Slicing Plane 27

5.2.4 Generating Grid Points 27

6. The GA Fractal: A Portrait of Evolutionary System 30

6.1 Overview 30

6.2 Parameter 30

6.3 Coloring Scheme 31

6.4 Implementation Results 34

6.5 Is It Really A Fractal? 49

6.6 Conclusion 50

6.7 Future Research 51

BIBLIOGRAPHY 52

VITA 54

LIST OF FIGURES

FIGURE		PAGE
5.1	Applying backward GA from each fixed point	26
5.2	Determining the slicing plane	28
5.3	Generating grid points	29
6.1	Color scale	33
6.2	Picture for f_1 with two attracting fixed points	36
6.3	The closeup near α in Figure (6.2)	37
6.4	The closeup of small oval in Figure (6.3)	38
6.5	Picture for f_2 with two attracting fixed points	39
6.6	The closeup of small oval in Figure (6.5)	40
6.7	The closeup of triangular shape (upper corner) in Figure (6.5)	41
6.8	Picture for f_2 with different slicing plane	42
6.9	Picture for f_3 with one attracting fixed point	43
6.10	The closeup of visible egg shape towards the center of Figure (6.9)	44
6.11	The closeup of blue dot at the top left corner of Figure (6.10)	45
6.12	Picture for f_3 with different slicing plane	46
6.13	Picture for f_4 with one attracting fixed point	47
6.14	The closeup of near the center of Figure (6.13)	48

LIST OF TABLES

TABLE		PAGE
1.1	First Generation	3
1.2	Second Generation	4
1.3	Third Generation	4

CHAPTER 1

Introduction

1.1 General Description of Genetic Algorithms

Genetic Algorithms (GAs) are stochastic search algorithms whose methods model some natural phenomena: genetic inheritance and Darwinian strife for survival. The collection of organisms is traditionally represented by a collection of binary strings called the population. The basic mechanism of a GA consists of two operations: selection and recombination. Selection does not introduce new point (potential solution) for consideration from the search space. It just copies some strings from the population to form an intermediate population referred to as the gene pool. The second step of the evolution cycle is recombination. Recombination is responsible for introducing new individuals. This is done by two genetic operators: crossover and mutation, which transform the gene pool into the next generation.

In the selection process, members of a population are selected with the probability proportional to their relative fitness. In other words, the probability of each string being selected is given by its fitness with respect to the total fitness of the whole population. These probabilities are then used to choose (with replacement)

a string from the population for the gene pool. Hence, strings with better–than–average relative fitness are more likely to be selected to contribute offspring to the next generation.

Crossover consists of selecting two parent strings from the gene pool and exchanging their tails at the same random position. This is typically performed with some probability (the crossover rate), otherwise parents are copied. Mutation is implemented by flipping the bits of a string with some small probability (the mutation rate). The crossover operator is introduced to enable structured, yet randomized information exchange, while the mutation operator introduces greater variability into the population.

Thus, in each generation, evaluate the fitness of each string, select the gene pool with respect to the probability distribution based on fitness values, and recombine the strings in the gene pool by mutation and crossover to form the next generation. After some number of generations, when no further improvement is observed, the process is stopped.

1.2 A Simple Genetic Algorithm

Consider the problem of maximizing the function $f(x) = \frac{1}{x^2+1}$ where $x \in [0, 7]$. Suppose the string length ℓ is 3, and the population size is 4.

To start the process, the initial population of strings is chosen randomly:

$$010 \quad 100 \quad 001 \quad 011 \ ,$$

Their fitness, $f(x)$, and probability of being selected are computed as shown in Table (1.1).

Table 1.1: First Generation

Initial Population	x	$f(x) = \frac{1}{1+x^2}$	$\frac{f_x}{\Sigma f}$
010	2	0.20	0.220
100	4	0.11	0.121
001	1	0.50	0.549
011	3	0.10	0.110

During reproduction process (selection), the strings with higher fitness are more likely to be copied, and the gene pool could be $001, 010, 001, 011$. Crossover then proceeds to mate strings randomly by selecting a random crossover site in pairs of strings at which to exchange their tails. This process is illustrated in Table (1.2).

Assuming that the crossover rate is 1 and the mutation rate is 0.0001, the reproduction and recombination processes creates a new population as shown in the above table. Note that the average fitness is higher than before. String 000 has the highest fitness value and thus has the greatest chance to survive in the next generation. The next simulation is shown in Table (1.3).

After another generation, the members of the population will likely be 000, which optimizes $f(x) = \frac{1}{1+x^2}$.

Table 1.2: Second Generation

Gene Pool	New Population	x	$f(x) = \frac{1}{1+x^2}$	$\frac{f_x}{\Sigma f}$
0 | 01	011	3	0.10	0.059
0 | 11	001	1	0.50	0.294
01 | 0	011	3	0.10	0.059
00 | 1	000	0	1.00	0.588

Table 1.3: Third Generation

Gene Pool	New Population	x	$f(x) = \frac{1}{1+x^2}$	$\frac{f_x}{\Sigma f}$
0 | 00	000	0	1.00	0.286
0 | 00	000	0	1.00	0.286
0 | 00	001	1	0.50	0.143
0 | 01	000	0	1.00	0.286

1.3 Thesis Overview

The first attempt at developing theoretical foundations for genetic algorithms relied on a binary string representation of solutions and on the notion of schema. Schema, according to Holland [2], is a similarity template that describes a subset of strings with similarities at certain string position(s).

In the next chapter, Holland's Schema Theorem which estimates the expected number of strings matching a schema in the next generation is summarized. Chapter 3 outlines Vose's infinite population model that gives the exact number of each string expected in the next generation. In Chapter 4, the backward GA is introduced. The details of the program implementation to produce related GA fractals are discussed in Chapter 5. Finally, Chapter 6 presents the visualization of GA, conclusion, and future research topics.

CHAPTER 2

Holland's Schema Theorem : An Overview

2.1 The Concept of Schema

Holland [2] introduced schema as a subset of strings which share common attributes. In addition to the binary alphabet $\{0, 1\}$, a *don't care* symbol, the asterisk "$*$", has been adopted in building a schema.

A string x matches a schema H if every element in string x matches schema H at every position not containing "$*$". Hence, for a string length of 5, strings 11001 and 10110 belong to schema $1****$, but 01000 does not.

Two properties of schema that the schema theorem refers to are *order* and *defining length*. The order of a schema H, denoted by $o(H)$, is the number of positions that are not "$*$", also called fixed position, in the schema. Denoted by $\delta(H)$, the defining length of a schema H is the distance between the first and the last fixed string position. For example, for a string of length 5, the two schemata

$$H_1 = 1**10,$$

$$H_2 = *00**,$$

have the orders

$$o(H_1) = 3, \;\; o(H_2) = 2,$$

and the defining lengths:

$$\delta(H_1) = 5 - 1 = 4, \quad \delta(H_2) = 3 - 2 = 1 \ .$$

The order of a schema is useful in calculating its survival probability (the probability it will exist in the next generation) under mutation. The defining length of a schema is useful in calculating its survival probability under crossover. More explanation will be given in the next section.

2.2 The Schema Theorem

As discussed in the previous chapter, the formation of the next generation is composed of two cycles: selection and recombination. The effect of selection (reproduction) is easy to determine in the following manner.

Let the total fitness $F(t)$ of a finite population P^t of strings at generation t be

$$F(t) = \sum_{x \in P} f(x)\,, \tag{2.1}$$

where $f(x)$ is the fitness of string x. Then the average fitness of the population is

$$\overline{F(t)} = \frac{F(t)}{|P|} \ . \tag{2.2}$$

After selection, the expected number of strings in the gene pool P matched by the schema H is

$$\begin{aligned} \mathcal{E}(H, t+1) &= \sum_{x \in H} \text{expected number of } x \text{ in } P \\ &= \sum_{x \in H} (\text{ probability that } x \text{ in } P)\ P \end{aligned}$$

$$= \sum_{x \in H} \left(\text{number of } x \text{ in } P^t \frac{f(x)}{\overline{F(t)}|P|} \right) |P|$$

$$\mathcal{E}(H, t+1) = \frac{\sum_{x \in H} (\text{number of } x \text{ in } P^t)\, f(x)}{\overline{F(t)}} .$$

Let $f(H,t)$ and $\mathcal{N}(H,t)$ be the average fitness of the schema H and the current number of all strings that match the schema H in population P at generation t, respectively. Then,

$$f(H,t) = \frac{\sum_{x \in P^t, x \in H} f(x)}{\sum_{x \in P^t, x \in H} 1} = \frac{\sum_{x \in H} (\text{number of } x \text{ in } P^t)\, f(x)}{\mathcal{N}(H,t)} . \tag{2.3}$$

Thus, due to selection, the expected number of strings matched by the schema H in the gene pool is

$$\mathcal{E}(H, t+1) = \mathcal{N}(H,t) \frac{f(H,t)}{\overline{F(t)}} . \tag{2.4}$$

Note that the number of strings in the gene pool changes as the ratio of the fitness of the schema to the average fitness of the population changes. In other words, the "above–average" schema will dominate and the "below–average" one will die off.

However, this reproductive schema growth equation is still not complete. In nature, evolution causes new individuals to be born which differ from their parents. In GA's, new strings are introduced during the second part of the evolution cycle, recombination.

As mentioned earlier, the defining length and the order of a schema play significant roles in the probability of its survival. Consider the following schemata

and parent strings x and y

$$\begin{aligned} H_1 &= 1\ *\ *\ 1\ 0 \quad , \delta(H_1) = 4 \quad , \\ H_2 &= *\ 0\ 0\ *\ * \quad , \delta(H_2) = 1 \quad , \\ x &= 1\ 0\ 0\ 1\ 0 \quad , \\ y &= 0\ 0\ 0\ 0\ 1 \quad . \end{aligned}$$

The crossover site is chosen randomly to fall between positions 3 and 4

$$\begin{aligned} x &= 1\ 0\ 0 \mid 1\ 0 \quad , \\ y &= 0\ 0\ 0 \mid 0\ 1 \quad , \end{aligned}$$

The resulting children strings are

$$\begin{aligned} x' &= 1\ 0\ 0\ 0\ 1 \quad , \\ y' &= 0\ 0\ 0\ 1\ 0 \quad . \end{aligned}$$

Note that neither offspring belongs to H_1, but both still belong to H_2. In this example, it is easy to see that because H_1 has a longer defining length, membership in H_1 has more chances to be disrupted by crossover. The fixed positions of schema H_2, on the other hand, are more likely to be preserved in one of the offspring. In other words, if $x \in H_2$ then a child of x under crossover has a greater chance of belonging to H_2.

In general, for string length ℓ, there are $\ell - 1$ possible crossover sites. This implies that the probability of disruption of a schema H with crossover rate p_c is at most

$$p_c \frac{\delta(H)}{\ell - 1} \quad , \tag{2.5}$$

and the probability of its survival is

$$p_s(H) \geq 1 - p_c \frac{\delta(H)}{\ell - 1} \quad . \tag{2.6}$$

The inequality results from the chance for the schema will survive even if a crossover site is selected between its fixed position.

The mutation operator randomly flips each bit with probability $p_m \ll 1$. It is clear that the probability of a single bit to survive unchanged is $1 - p_m$. Thus, the probability of membership in a schema H to survive under mutation is

$$p_s(H) = (1 - p_m)^{o(H)} \approx 1 - o(H)p_m \quad . \tag{2.7}$$

The combined effect of selection, crossover, and mutation creates a new form of the reproductive growth equation (The Schema Theorem):

$$\mathcal{E}(H, t+1) \;\geq\; \mathcal{N}(H,t) \frac{f(H,t)}{\overline{F(t)}} \left[1 - p_c \frac{\delta(H)}{\ell - 1} - o(H)p_m \right] \quad . \tag{2.8}$$

The Schema Theorem is inequality (2.8). It implies that short, low–order, and above–average schemata will receive increasing number of strings in the next generations. This means that when $f(H,t) > \overline{F(t)}$ and the probability of a schema H being disrupted by crossover and mutation is small, the representation of schema H is expected to increase in the next generation.

2.3 The Minimum Deceptive Problem

This section summarizes Goldberg's classification of two bit deceptive problems. Assuming that string 00 is optimal, there are four possible schemata con-

taining one fixed position: two contain suboptimal strings $(1*, *1)$ and two contain the optimal string $(0*, *0)$.

In some cases, a genetic algorithm may have difficulty in converging to the optimal solution. This phenomenon called *deception* refers to some short, low-order, and above-average schemata that can mislead a genetic algorithm and cause its convergence to suboptimal strings. For the problem introduced above, it means that either

$$f_{1*} > f_{0*} \quad ,$$

or

$$f_{*1} > f_{*0} \quad ,$$

Hence one of the following must hold

$$f_{1*} = \frac{f_{10} + f_{11}}{2} > \frac{f_{00} + f_{01}}{2} = f_{0*} \quad , \tag{2.9}$$

$$f_{*1} = \frac{f_{01} + f_{11}}{2} > \frac{f_{00} + f_{10}}{2} = f_{*0} \quad , \tag{2.10}$$

Without loss of generality, assume that equation (2.10) is true so that $f_{*1} > f_{*0}$ is the deceptive condition.

Normalizing the fitness with respect to the binary complement of the optimum string gives

$$r = \frac{f_{00}}{f_{11}}, \quad c = \frac{f_{01}}{f_{11}}, \quad c' = \frac{f_{10}}{f_{11}} \quad ,$$

and the globality condition can be written

$$r > c, \quad r > 1, \quad r > c' \quad .$$

It follows from equation (2.10) that

$$1 + c - r > c' \ , \quad \text{hence } c' < 1 \quad ,$$

$$c > c' + r - 1 \ , \quad \text{hence } c > c' \quad .$$

The deceptive problems are classified according to the range of values of c

$$\text{Type I} : f_{01} > f_{11} \ , \quad c > 1 \quad ,$$

$$\text{Type II} : f_{11} \geq f_{01} \ , \quad c \leq 1 \quad .$$

Goldberg [1] developed a model to analyze the behavior of a genetic algorithm on deceptive problems. However, the effect of mutation was not considered. In the next chapter, the use of $\mathcal{G}$ does not require zero mutation and is therefore characteristic of a true GA. Furthermore, note that the inequality in (2.8) implies that the schema theorem is only able to approximate the expected strings belonging to a schema H in the next generation. The next chapter will describe the exact schema theorem which is able to determine exactly the expected next population.

CHAPTER 3

Exact Schema Theorem: The Infinite Population Model

3.1 Overview

This chapter investigates the exact schema theorem, commonly known as the infinite population model. Vose [5] showed how given a representation p^t for the current population, a transition operator $\mathcal{G}$ could be defined which exactly determines the expected next population $\mathcal{G}(p^t)$.

The new approach, referred to as the infinite population model, is able to predict what the expected next generation will be, the previous schema theorem is only able to approximate it. The infinite population model, is elaborated in the following section.

3.2 The Infinite Population Model

Let Ω be the set of all length ℓ binary strings and let $N = 2^\ell$ be the number of all possible strings. Since each string is a binary representation of an integer in the interval$[0, N-1]$, may be referred to by the number it represents. For example, for $\ell = 3$ and $i = 5$, string i is 101.

Let p_i^t be defined as the proportion of string i in generation t. The vector

$p^t = \left(p_0^t, \ldots, p_{2^\ell-1}^t\right)$ is called population vector. It contains the proportion of each possible string in the population.

3.2.1 Selection Operator

Let $F = [f_{ij}]$ be an $N \times N$ diagonal matrix with f_{ii} entry containing the fitness value of string i. Given a vector v, let $|v|$ denote the sum of its elements. If v is a vector with i^{th} component equal to the proportion of string i in the population, then Fv is a vector pointing in the same direction as that vector having i^{th} component equal to the probability that i will be selected for recombination.

Hence $s^t = (Fp^t)|Fp^t|^{-1}$ is a vector with i^{th} component equal to the probability that string i will be selected as a parent for recombination.

The vector s^t is called the selection vector. The probability that string k will be in the next generation is given by

$$\sum_{i,j} s_i^t s_j^t m_{ij}(k) \quad , \tag{3.1}$$

where $m_{ij}(k)$ is the probability that string k is the child resulting from recombination (crossover and mutation) of the parent strings i and j.

3.2.2 Recombination Operator

Vose [5] has shown that $m_{ij}(k) = m_{i\oplus k, j\oplus k}(0)$, where $\oplus$ denotes *exclusive-or* on integers. The matrix M defined by $M_{ij} = m_{ij}(0)$ is called the recombination matrix.

Using the selection vector and the recombination matrix, the RHS of equation (3.1) can be written as follow:

$$\sum_{i,j} s_{i\oplus k} s_{j\oplus k} M_{ij} \quad . \tag{3.2}$$

Let σ_i be the permutation defined by

$$\sigma_i (s_0, \ldots, s_{N-1})^T = \left(s_{i\oplus 0}, \ldots, s_{i\oplus(N-1)}\right)^T \quad ,$$

where T denotes transpose. Using these permutations, equation (3.2) can be written as

$$\sum_{i,j} (\sigma_k s)_i (\sigma_k s)_j m_{ij}(0) \quad . \tag{3.3}$$

Vose [5] has shown that if recombination is one–point crossover with crossover rate χ, and mutation with mutation rate μ, then M_{ij} is

$$\frac{1-\mu}{2}\left\{\eta^{|i|}\left(1-\chi+\frac{\chi}{\ell-1}\sum_{k=1}^{\ell-1}\eta^{-\Delta_{i,j,y}}\right)+\eta^{|j|}\left(1-\chi+\frac{\chi}{\ell-1}\sum_{k=1}^{\ell-1}\eta^{\Delta_{i,j,y}}\right)\right\}, \tag{3.4}$$

where $\eta = \mu/(1-\mu)$, in which division by zero at $\mu = 0$ and $\mu = 1$ is to be removed by continuity, and $\Delta_{i,j,y} = |(2^y - 1) \otimes i| - |(2^y - 1) \otimes j|$. Here $\otimes$ is the symbol for *logical-and.*

Using vector notation to represent equation (3.3), the expected next population vector satisfies the relation

$$\mathcal{E}(p^{t+1})_k = (\sigma_k s^t)^T M (\sigma_k s^t) \quad .$$

where $\mathcal{E}$ denotes expectation. Defining the operator $\mathcal{M}$ by

$$\mathcal{M}(s^t) = \left((\sigma_0 s^t)^T M(\sigma_0 s^t), \cdots, (\sigma_{N-1} s^t)^T M(\sigma_{N-1} s^t)\right)^T \quad , \tag{3.5}$$

allows this equation to be put in the form

$$\mathcal{E}(p^{t+1}) = \mathcal{M}(s^t)$$

3.2.3 Operator $\mathcal{G}$

Let $\mathcal{G}$ be the transition operator from one population to the expected next population. In other words, $\mathcal{G}(p^t)$ is the expected next population vector p^{t+1}(given that the current population is p^t). Hence $(\mathcal{G}(p^t))_k$ is the expected proportion of string k in generation $(t+1)$. In terms of the notation in the last section

$$\mathcal{G}(p^t) = \mathcal{M}(s^t) = \mathcal{M}\left(\frac{Fp^t}{|Fp^t|}\right) \quad . \tag{3.6}$$

The matrices F and M determine the expected behavior of a simple GA. The matrix F contains fitness information appropriate for selection, the matrix M holds information appropriate for recombination.

Let Λ be the collection of vectors having 2^ℓ components such that the components are non-negative and sum to 1, i.e.

$$\Lambda = \{x \in \Re^{2^\ell} : x_j \geq 0, \sum x_j = 1\} \quad . \tag{3.7}$$

For any constant c, $\mathcal{M}(cp) = c^2\mathcal{M}(p)$ and $\mathcal{M} : \Lambda \mapsto \Lambda$. Having those properties, it is easy to show that

$$\mathcal{M}\left(\frac{Fp^t}{|Fp^t|}\right) = \frac{\mathcal{M}Fp^t}{|\mathcal{M}Fp^t|} \quad .$$

Proof:

a) $\mathcal{M}\left(\frac{Fp^t}{|Fp^t|}\right) = \frac{\mathcal{M}Fp^t}{|Fp^t|^2} \sim \mathcal{M}Fp^t \quad ,$

b) $\frac{\mathcal{M}Fp^t}{|\mathcal{M}Fp^t|} \sim \mathcal{M}Fp^t \quad ,$

where $\sim \mathcal{M}Fp^t$ represents a vector which points in the same direction as $\mathcal{M}Fp^t$. Furthermore,

$$\left|\mathcal{M}\left(\frac{Fp^t}{|Fp^t|}\right)\right| = |\lambda| = 1 \quad ,$$

where λ denotes something in Λ, and

$$\left|\frac{\mathcal{M}Fp^t}{|\mathcal{M}Fp^t|}\right| = \frac{|\mathcal{M}Fp^t|}{|\mathcal{M}Fp^t|} = 1 \quad ,$$

Hence, they both have the same magnitude. Since they are pointing in the same direction and have the same magnitude, they are equal. Thus,

$$\mathcal{G}(p^t) = \mathcal{M}\left(\frac{Fp^t}{|Fp^t|}\right) = \frac{\mathcal{M}Fp^t}{|\mathcal{M}Fp^t|} \quad . \tag{3.8}$$

CHAPTER 4

The Backward GA

4.1 Overview

As described in the previous chapter, given a representation p^t for the current population, an operator $\mathcal{G}$ exactly determines the expected next generation. The representation p of a population is a vector with i^{th} component given by the proportion of string i in the population.

Hence $p \in \Lambda$ which is the intersection of the hyperplane $\mathcal{H} : p_0 + ... + p_{2^\ell - 1} = 1$ with the non–negative orthant. The function $\mathcal{G}$ has mathematical form which allows its application to any vector in 2^ℓ–dimensional space. However, the inverse of $\mathcal{G}$ has only been determined in the case where $\ell = 2$ and where $p \in \mathcal{H}$. For this thesis, $\mathcal{G}$'s action on $\mathcal{H}$ is considered (attention is restricted to to two bit GAs) so that $\mathcal{G}^{-\infty}$ may be used.

Having the inverse of $\mathcal{G}$ enables one to find the previous population from the current one. By running the GA backwards, a "Big Bang" is computed. It can be thought as the place where things begin.

4.2 The Inverse $\mathcal{G}$

Definition 1 : A function f from A into B is *onto* if every element of B corresponds to some x in A.

Definition 2 : A function f is said to be *one-one* if it assigns different values in B to different arguments in A (or for every B there corresponds a unique A).

Definition 3 : A function f has an inverse if it is *onto* and *one-one.*

Definition 4 : A is a *proper subset* of B if A is a subset of B and at least one member of B is not a member of A.

A simple example is shown below:

$$f(x) = \frac{1}{2}x \quad ,$$

$$g(x) = 2x \quad .$$

Let $I = [0, 1]$. The function $f : I \to [0, \frac{1}{2}]$ has the inverse g. Note that $f : I \to I$ has no inverse (not even g), because it is not *onto.* In some sense, the domain I is too small or the codomain is too large. If $f : \Re \to \Re$, then f is *onto* and *one-one,* and f inverse is g.

Since $\mathcal{G}(\Lambda)$ is a *proper subset* of Λ, it is obvious that $\mathcal{G} : \Lambda \mapsto \Lambda$ can not have an inverse. However, there is a function, call it $\mathcal{G}^{-1}$, which behaves like an inverse should, just like g behaves like an inverse for f, even if an inverse for $\mathcal{G}$ on Λ

does not exist. The function $\mathcal{G}^{-1}$ can be found by manipulating the function $\mathcal{G}$. Consider $\mathcal{G}(x)$ and $\mathcal{G}^{-1}(x)$ for any $x \in \mathcal{H}$ with $|x| = 1$.

As shown in the previous chapter,

$$p^{t+1} = \mathcal{G}(p^t) = \mathcal{M}\left(\frac{Fp^t}{|Fp^t|}\right) \quad . \tag{4.1}$$

Operating on both sides with $\mathcal{M}^{-1}$ followed by F^{-1} yields

$$F^{-1}\mathcal{M}^{-1}p^{t+1} = \frac{p^t}{|Fp^t|} =\sim p^t \quad ,$$

provided that $\mathcal{M}^{-1}\mathcal{M}(x) =\sim x$.

Hence

$$p^t = \frac{F^{-1}\mathcal{M}^{-1}p^{t+1}}{|F^{-1}\mathcal{M}^{-1}p^{t+1}|} \quad .$$

Thus, replacing p^{t+1} with x,

$$\mathcal{G}^{-1}(x) = \frac{F^{-1}\mathcal{M}^{-1}(x)}{|F^{-1}\mathcal{M}^{-1}(x)|} \quad . \tag{4.2}$$

For $\mathcal{G}^{-1}$ to behave like an inverse, it must satisfy

a) $x \overset{\mathcal{G}}{\longmapsto} \mathcal{G}(x) \overset{\mathcal{G}^{-1}}{\longmapsto} x \quad ,$

b) $x \overset{\mathcal{G}^{-1}}{\longmapsto} \mathcal{G}^{-1}(x) \overset{\mathcal{G}}{\longmapsto} x \quad .$

Let Q be the matrix

$$\begin{bmatrix}
(\chi-1)(\mu-1)^2 & (2\mu-1)(\chi-1)(\mu-1)/2 & (2\mu-1)(\chi-1)(\mu-1)/2 & \mu(\mu-1)(\chi-1)-1/2 \\
(2\mu-1)(\chi-1)(\mu-1)/2 & \mu(\mu-1)(\chi-1) & \chi(1/2-\mu+\mu^2)-\mu(\mu-1) & (2\mu-1)(\chi-1)\mu/2 \\
(2\mu-1)(\chi-1)(\mu-1)/2 & \chi(1/2-\mu+\mu^2)-\mu(\mu-1) & \mu(\mu-1)(\chi-1) & (2\mu-1)(\chi-1)\mu/2 \\
\mu(\mu-1)(\chi-1)-1/2 & (2\mu-1)(\chi-1)\mu/2 & (2\mu-1)(\chi-1)\mu/2 & \mu\mu(\chi-1)
\end{bmatrix}$$

, where χ and μ denote crossover and mutation rate, respectively. The function $\mathcal{M}^{-1}$ is defined by

$$\mathcal{M}^{-1}(x) = \left((\sigma_0 x)^T Q(\sigma_0 x), \cdots, (\sigma_{N-1} x)^T Q(\sigma_{N-1} x)\right)^T \tag{4.3}$$

The matrix Q was found using the symbolic algebra software, Maple. The functions $\mathcal{M}$ and $\mathcal{M}^{-1}$ satisfy

a') $x \overset{\mathcal{M}}{\longmapsto} \mathcal{M}(x) \overset{\mathcal{M}^{-1}}{\longmapsto} \sim x$,

b') $x \overset{\mathcal{M}^{-1}}{\longmapsto} \mathcal{M}^{-1}(x) \overset{\mathcal{M}}{\longmapsto} \sim x$.

This was checked numerically using hundreds of random vectors.

It follows that $\mathcal{G}^{-1}$ satisfies both conditions a and b :

$$x \overset{\mathcal{G}}{\longmapsto} \mathcal{M}\left(\frac{Fx}{|Fx|}\right) \overset{\mathcal{G}^{-1}}{\longmapsto} \frac{\sim F^{-1}\mathcal{M}^{-1}\left(\mathcal{M}\left(\frac{Fx}{|Fx|}\right)\right)}{|\cdots|} \overset{b'}{=} \frac{F^{-1}\sim \frac{Fx}{|Fx|}}{|\cdots|} = \frac{\sim \frac{x}{|Fx|}}{\left|\sim \frac{x}{|Fx|}\right|} = \frac{x}{|x|} = x,$$

$$x \overset{\mathcal{G}^{-1}}{\longmapsto} \frac{F^{-1}\mathcal{M}^{-1}(x)}{|F^{-1}\mathcal{M}^{-1}(x)|} \overset{\mathcal{G}}{\longmapsto} \frac{\mathcal{M}\left(F\frac{F^{-1}\mathcal{M}^{-1}(x)}{|F^{-1}\mathcal{M}^{-1}(x)|}\right)}{|\cdots|}$$

$$= \frac{\mathcal{M}\left(\frac{\mathcal{M}^{-1}(x)}{|F^{-1}\mathcal{M}^{-1}(x)|}\right)}{|\cdots|} = \frac{\frac{\sim x}{|F^{-1}\mathcal{M}^{-1}(x)|}}{|\cdots|} = \frac{\sim x}{|\sim x|} = \frac{x}{|x|} = x \quad .$$

Note that $|\cdots|$ indicates the norm of the denominator.

4.3 The Big Bang

The universe is an evolutionary system in which humans occupy only a small part in space and time. Just like the universe, the evolutionary system $\mathcal{E}$ taking place in $\mathcal{H}$ governed by the exact schema theorem $\mathcal{G}$, seems to have a Big Bang.

There are amusing parallels. The Big Bang is found by running time in reverse and to a place in space and time outside the human realm. Running $\mathcal{E}$ backwards from Λ converges to a point α in negative time ($-\infty$ it turns out) which is located in space (i.e. $\mathcal{H}$) outside of Λ. The point α is called the Big Bang.

4.4 Floating-Point Precision

Understanding what things were like near the Big Bang requires energy. Determining the structure of $\mathcal{E}$ near α requires increasing *computational* energy: α is located at $\mathcal{G}^{(-\infty)}$ and each computation is generally affected by round–off error. These errors originate from the fact that computers retain only a fixed number of significant figures during a calculation.

Typically, fractional quantities are represented in computers using a floating–point format. In this approach, the number is expressed as a fractional part, called the mantissa or significand, and an integer part called the exponent or characteristic, as in

$$m.\beta^e \quad ,$$

where $m = .d_n...d_2d_1$ is the mantissa, β is the base of the number system being used, and e is the exponent. For instance, the number 156.78 could be represented as $0.15678 \cdot 10^3$ in a floating–point base–10 system.

Furthermore, the mantissa is usually normalized if its leading digit is zero. For example, suppose the quantity of $1/34 = 0.029411765...$ is stored in a floating-

point base-10 system which allows four decimal places to be stored. Normalization causes $1/34$ to be stored as $0.2942 \cdot 10^{-1}$ instead of $0.0294 \cdot 10^{0}$.

The consequence of normalization is that the absolute value of m is limited. That is,

$$\frac{1}{\beta} \leq m < 1 \quad ,$$

where β is the base. For example, for a base-10 system, m would range between 0.1 and 1, and for base-2 system, between 0.5 and 1.

The fundamental unit whereby information is represented is called a *word.* This is an entity that consists of a string of binary digits, or *bits.* Many computers allow 24 bits to be used for the mantissa, which translates into about seven significant base–10 digits of precision.

With this acknowledged, there are still cases where round–off error become critical. For this reason, most computers allow the specification of double precision, in which an extra word is employed to store additional significant figures of the mantissa. For double precision, and extra 32 bits would be used to create 56–bit mantissa which allows about 17 significant base–10 digits.

Since the Big Bang, $\alpha = \mathcal{G}^{-\infty}(x)$, computation will still produce round–off errors that significantly affects the convergence of $\mathcal{G}$, Therefore, a higher precision floating–point numbers must be used in any program used to visualize $\mathcal{G}$. This thesis is uses a base 2^{24} floating-point representation having a 20-digit mantissa. In this case, d_i ranges from 0 to $2^{24} - 1$ and m would range between $1/2^{24}$ and 1.

CHAPTER 5

Program Implementation

5.1 Overview

Calculations show that every population vector p has a forward time evolutionary trajectory ending in a fixed point of $\mathcal{G}$ in Λ. Followed by this intuition, one might be curious of whether every p also has a backward time evolutionary trajectory ending in a point α. Should this be true, it will be interesting to view what is going on near α, the Big Bang. A program has been written to investigate this possibility.

The program is written to run on the Intel iPSC/860, a distributed memory multiprocessor machine (RX) at ORNL. Although the use of extended precision floating-point arithmetic can mitigate the effect or round–off errors, a price is paid in execution time.

5.2 Details of Implementation

The main goal of the program is to to allow the behavior of $\mathcal{G}$ to be visualized. The program is started by selecting a random initial population and iterating $\mathcal{G}$ to obtain fixed point(s). Once the fixed points are found, the next task is to find

the Big Bang by iterating $\mathcal{G}^{-1}$.

Since only two dimensions fit on computer screen and $\mathcal{H}$ is three dimensional, three points of $\mathcal{H}$ are chosen to determine a slicing plane. Most of the pictures are generated with the Big Bang α being near one of these three points. After the slicing plane has been chosen, grid points can be generated within the slicing plane and pictures are computed. More details are given below.

5.2.1 Finding Fixed Points

In order to start the GA process, some random points are generated to represent random initial populations. From an initial population, the function $\mathcal{G}$ determines the next population. $\mathcal{G}$ is repeatedly applied until the previous and current population are 'near'. The condition 'near' is satisfied when the first three mantissa positions of every element of p^t are the same as the corresponding elements in p^{t-1}. The absolute difference of the real numbers, p_i^t and p_i^{t-1} is

$$|p_i^t - p_i^{t-1}| < 2^{-(3 \cdot 24)} \approx 10^{-22}$$

Since the implementation is using string length 2, at most two fixed–points are found. For the purpose of illustration in this chapter, two fixed–points, labeled $\mathbf{fp_1}$ and $\mathbf{fp_2}$, are assumed to be found.

5.2.2 Finding Inverse Points

As mentioned in the previous chapter, given a current population, one can determine the expected previous population using the function $\mathcal{G}^{-1}$. Applying this

function $\mathcal{G}^{-1}$ infinitely many times will lead to the Big Bang.

The Big Bang is approximated as follows. From each fixed point, iterate $\mathcal{G}^{-1}$ until $\mathcal{G}^{-k}(a)$ and $\mathcal{G}^{-k}(b)$ are close. Those points are then labeled **a** and **b** respectively. Figure (5.1) illustrates this step.

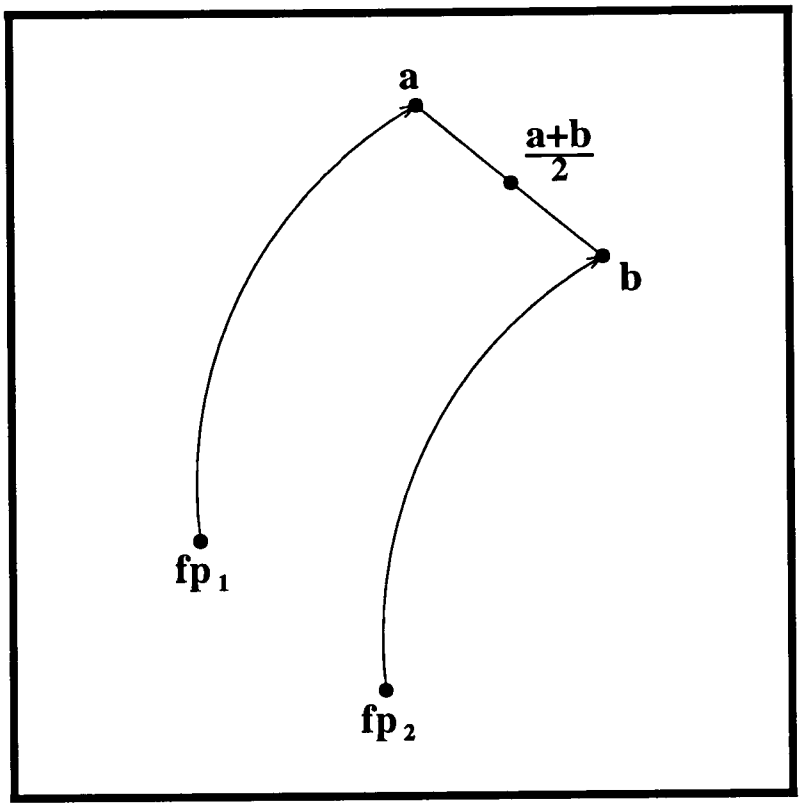


Figure 5.1: Applying backward GA from each fixed point

The midpoint $\frac{a+b}{2}$ approximates α. To get variety in the slicing plane, a randomly chosen point near $\frac{a+b}{2}$ is used.

5.2.3 Finding Slicing Plane

For the purpose of illustration and without loss of generality, three-dimensional space is used to show graphically how to obtain a slicing plane, in which grid points will be generated. In other words, the domain $\mathcal{H}$ is a flat 3-space embedded in $\Re^4$. Three points of $\mathcal{H}$ are needed to determine the slicing space.

Given a random vector $\mathbf{x}$, a random vector $\mathbf{v}$ that is perpendicular to $\mathbf{b}-\mathbf{a}$ lying in the plane containing $\mathbf{b}-\mathbf{a}$ and $\mathbf{x}$ is given by

$$\mathbf{v} = -\frac{(\mathbf{b}-\mathbf{a})(\mathbf{b}-\mathbf{a})}{\mathbf{x}(\mathbf{b}-\mathbf{a})}\mathbf{x} + (\mathbf{b}-\mathbf{a}) \quad . \tag{5.1}$$

Note that this formula can be applied to a space of any dimension.

Multiplying the vector $\mathbf{v}$ by the radius r of the sphere and adding $(\mathbf{b}+\mathbf{a})/\mathbf{2}$ will give a random approximation β to α. In short,

$$\alpha \approx \mathbf{r}\mathbf{v} + \frac{(\mathbf{b}+\mathbf{a})}{\mathbf{2}} = \beta \quad , \tag{5.2}$$

where $r = \frac{\|\mathbf{b}-\mathbf{a}\|}{\mathbf{2}}$. Figure (5.2) shows α as a point on the equator surface.

From α, vectors $\mathbf{u}$ and $\mathbf{n}$ that point to the first and second fixed–point, respectively, can be drawn. The final step is finding the orthogonal basis $\mathbf{u}$ and $\mathbf{n}'$ to represent the plane.

5.2.4 Generating Grid Points

Once the slicing plane is determined, grid points can be generated. From each grid point, apply $\mathcal{G}$ until it converges to one of the two fixed–points. The previous

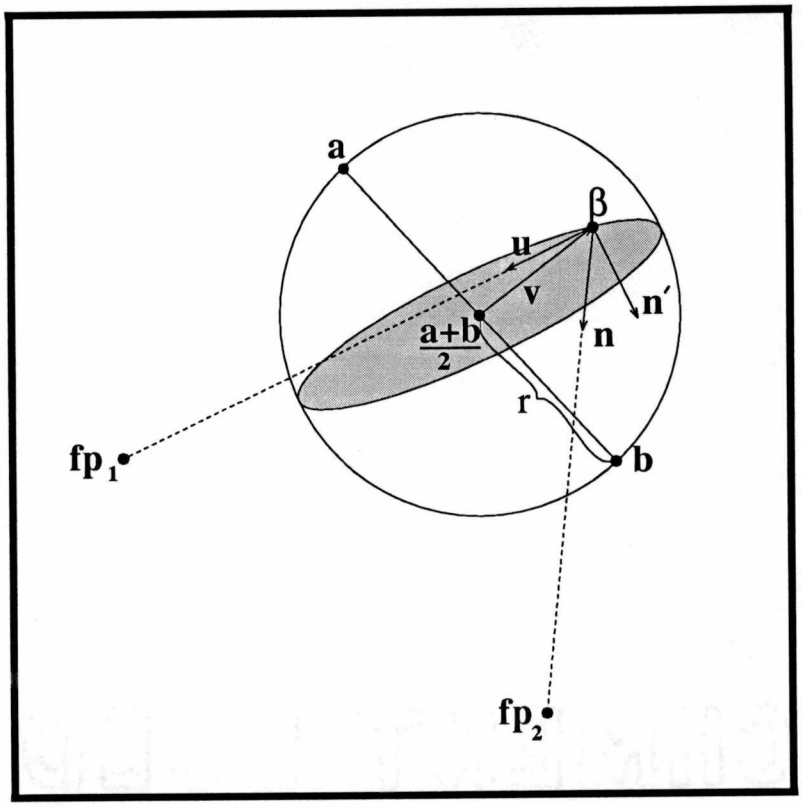


Figure 5.2: Determining the slicing plane

criteria of convergence(iteration are near a fixed point) is applied, and the number of iterations taken for each point to converge determines the picture.

In determining the position of grid points, consider a picture containing $lim \times lim$ pixels. The layout of the slicing plane in which grids are generated is shown in Figure (5.3). Note that β is arranged so that it is located in the center of the picture. Using this strategy, one would expect to see the journey that GA undergoes from near the Big Bang to the fixed-points in Λ.

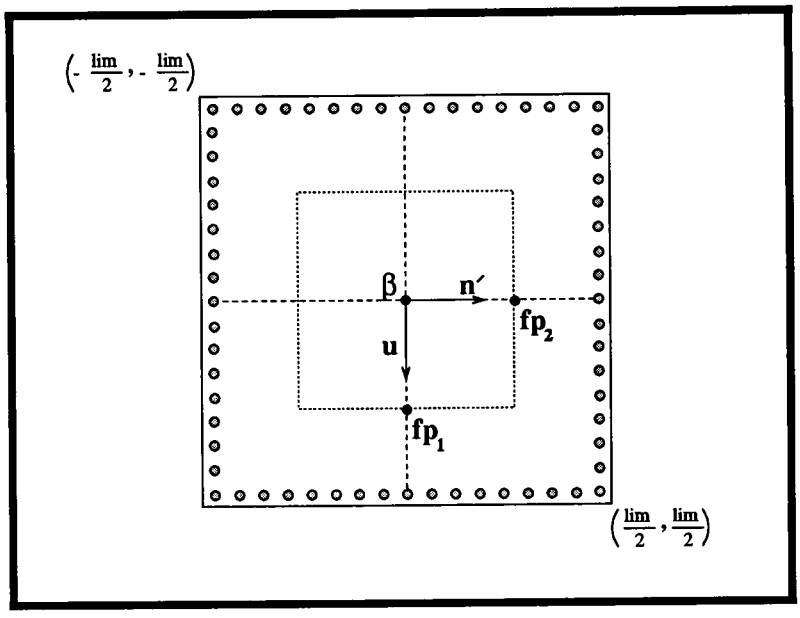


Figure 5.3: Generating grid points

The ij^{th} grid point, where $-\frac{lim}{2} \leq i, j \leq \frac{lim}{2}$, is determined by

$$ix = \frac{(i+c)*(x_2-x_1)}{lim} + x_1 \quad , \; and \tag{5.3}$$

$$jy = \frac{(j+c)*(y_2-y_1)}{lim} + y_1 \quad , \tag{5.4}$$

where $c = \frac{lim}{2}$. Note that these equations (5.3) and (5.4) move x_1 to x_2 and y_1 to y_2, respectively.

CHAPTER 6

The GA Fractal: A Portrait of Evolutionary System

6.1 Overview

With $\mathcal{G}$ in hand, one might wonder if it has an associated fractal. This chapter answers that question. Given x, consider the sequence of iterates

$$x, \quad \mathcal{G}^{(1)}(x), \quad \mathcal{G}^{(2)}(x), \quad \mathcal{G}^{(3)}(x), \ \ldots$$

The computations were restricted to iterations of $\mathcal{G}$ beginning from points x in a 256×256 planar rectangular grid within $\mathcal{H}$. Also mentioned are the parameters and the coloring scheme used. The chapter is completed by presenting some implementation results, conclusion, and future research.

6.2 Parameter

There is nothing special about the crossover and mutation rates that are used. The program read a parameter file which contains the crossover = 0.6 and mutation = 0.00505 values. As the pictures can be expensive to compute, other values have not been tried.

Since the string length is 2, it is natural to include minimal deceptive functions

of each type, as well as a completely nondeceptive function. Like the crossover and mutation rates, there is nothing special about the functions.

The dependence of the fractal on fitness functions and GA parameters is not known, but, like everything else about GAs, it should be complex. The fractal pictures were computed using the following fitness functions:

$$f_1 :: \quad 0 \mapsto 0.982335, \quad 1 \mapsto 0.143934, \quad 2 \mapsto 0.256464, \quad 3 \mapsto 0.936686 \quad ,$$

$$f_2 :: \quad 0 \mapsto 0.823359, \quad 1 \mapsto 0.439341, \quad 2 \mapsto 0.256464, \quad 3 \mapsto 0.723834 \quad ,$$

$$f_3 :: \quad 0 \mapsto 0.723359, \quad 1 \mapsto 0.639341, \quad 2 \mapsto 0.256464, \quad 3 \mapsto 0.583834 \quad ,$$

$$f_4 :: \quad 0 \mapsto 0.723359, \quad 1 \mapsto 0.639341, \quad 2 \mapsto 0.583834, \quad 3 \mapsto 0.256464 \quad .$$

6.3 Coloring Scheme

Points on the slicing plane are colored based on the number of iterations that a point requires to converge to a fixed point. Point x is colored red if it is close in time to its destiny. In other words, it takes relatively few iterations of $\mathcal{G}$ to approach a fixed point. The color blue indicates that a point x is far away in time (i.e. requires many iterations of $\mathcal{G}$). In short, the geometrical representation of x is colored red if the sequence converges quickly, blue if slowly, and through rainbow colors in between. The computer code defined "few" as the minimum number (over the collection of points to be colored) of iterations required to converge. The definition of "many" was the analogous maximum. Since there are 256 different

colors, the number of iterations of each point to converge is scaled from 0 to 255. Figure (6.3) shows the coloring scale used.

The exception to this description of coloring scheme is that black is more red than red, white is more blue than blue, and points inside Λ are in reverse color so that positive time and space can be visually identified.

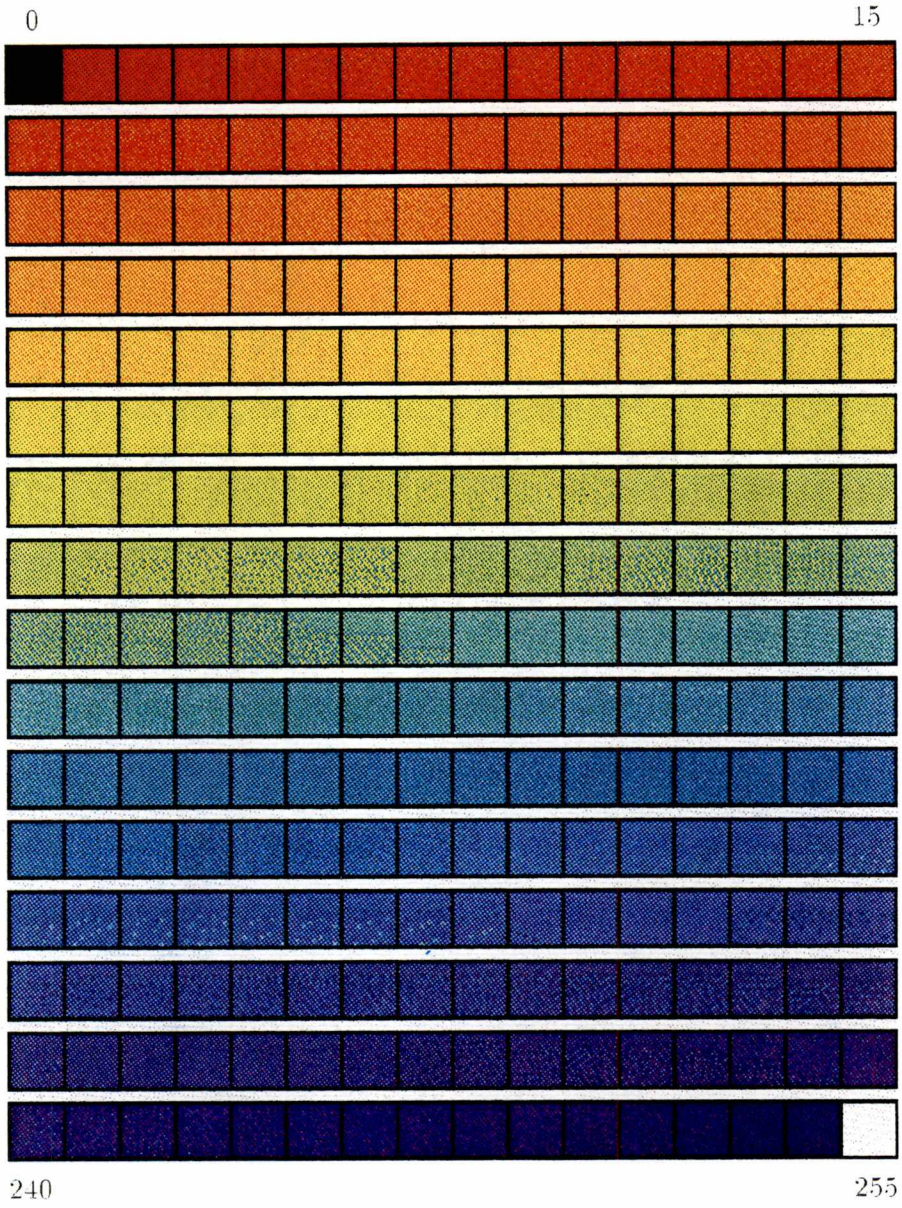


Figure 6.1: Color scale

6.4 Implementation Results

The first three pictures are for f_1. For this type-II minimal deceptive problem, $\mathcal{G}$ has two attracting fixed points in Λ (the reverse color triangle in Figure (6.2)) near its lower corners. Note how Λ is divided by a yellow line. This represents the common boundary of the basins of attraction for the two attracting fixed points. The negative time fixed point α is near the center of the picture. The closeup near α is shown in Figure (6.3). Figure (6.4) shows a closer look on the small oval on the center of Figure (6.3). Notice the similarity of shapes from the previous picture.

The next four pictures are for f_2. As before, α is near the center of the picture. This type-II minimal deceptive problem is similar to f_1, but is easier for the GA to solve in the sense that the basin of the deceptive attractor (right hand half of Λ in Figure (6.5)) is smaller. Figure (6.6) and (6.7) are closeups of the smallest visible oval and triangular shape (upper corner), respectively, in the center of Figure (6.5). Note the oval beneath it, which has an oval beneath it, which has ... the limit of this sequence is α. Note the overall regularity and how the system *inside* Λ (Figures (6.2) and (6.5)) is tame. Figure (6.8) shows the same scene from different slicing plane.

As the problem gets easier, the fractal gets more interesting. This trend is reinforced by the next three pictures corresponding to f_3. This is a type-I minimal deceptive problem for which $\mathcal{G}$ has only a single attracting fixed point in Λ, near

the lower left corner (out of view) in Figure (6.9). Figure (6.10) is a close up near the smallest visible egg shape towards the center of Figure (6.9). Zooming in the blue dot at the top left corner of Figure (6.10) gives a spiral shape in Figure (6.11). Figure (6.12) is the same scene that is viewed from different angle of slicing plane.

Like the previous function, f_4 has a single associated attractor. However, f_4 has no deception at all; it is totally easy. Figure (6.14) is a closeup near the center of Figure (6.13).

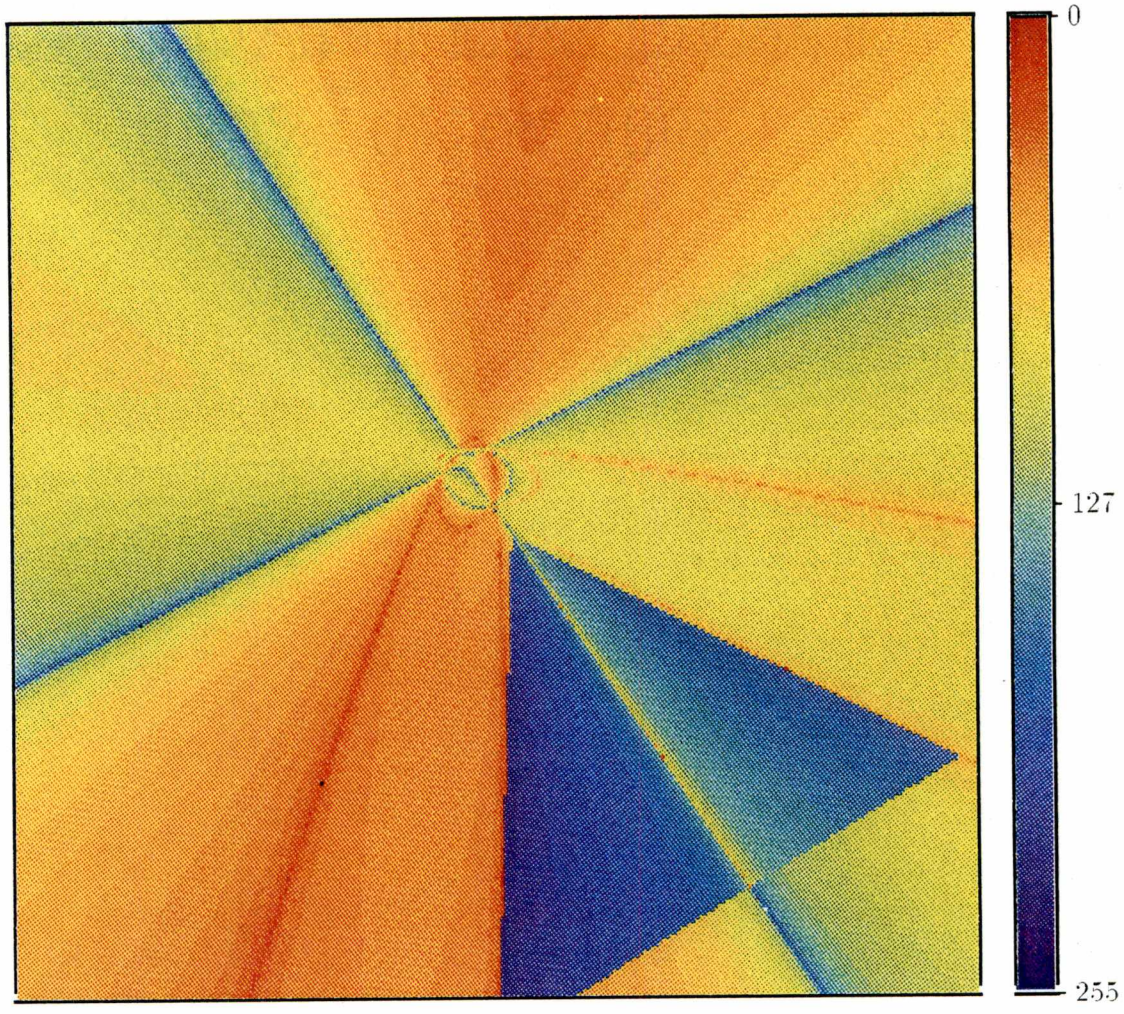


Figure 6.2: Picture for f_1 with two attracting fixed points

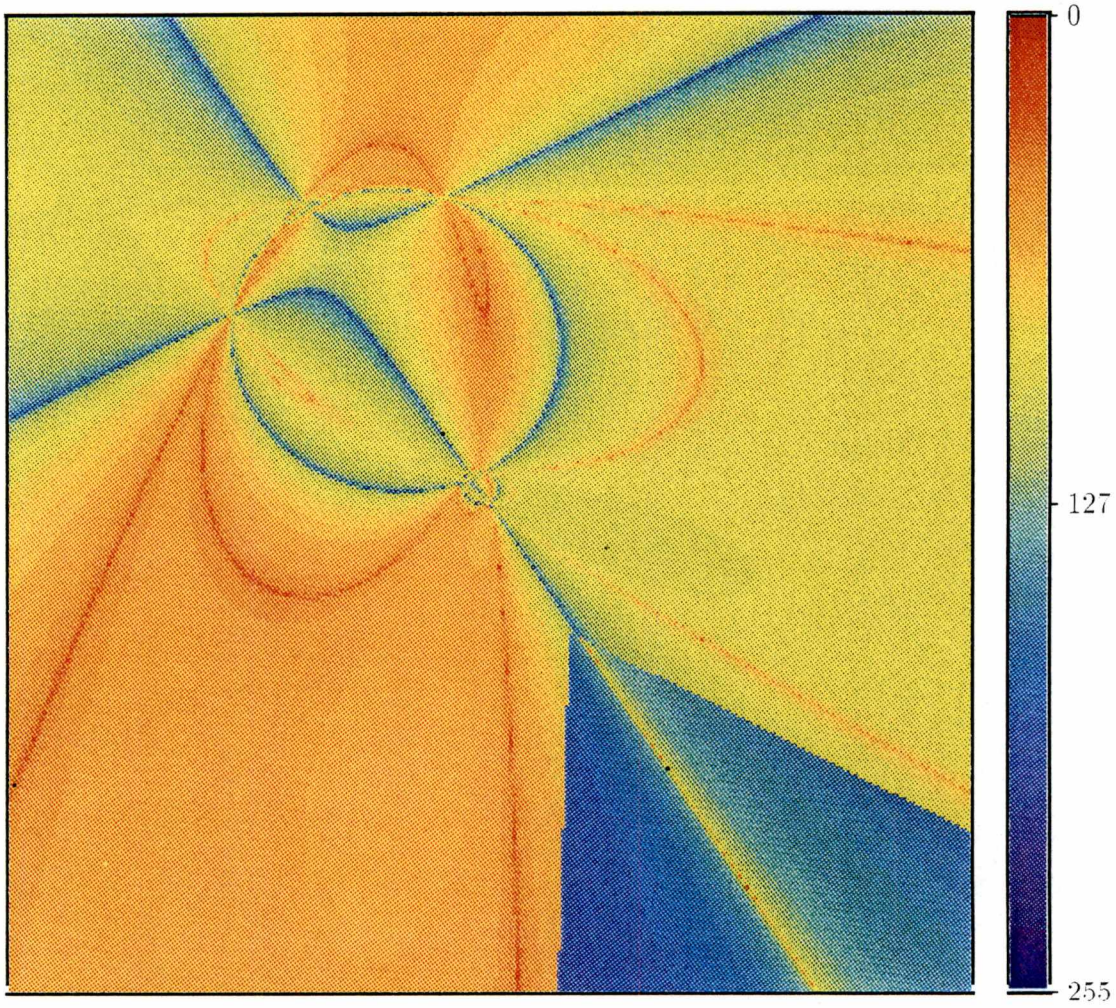


Figure 6.3: The closeup near a in Figure (6.2)

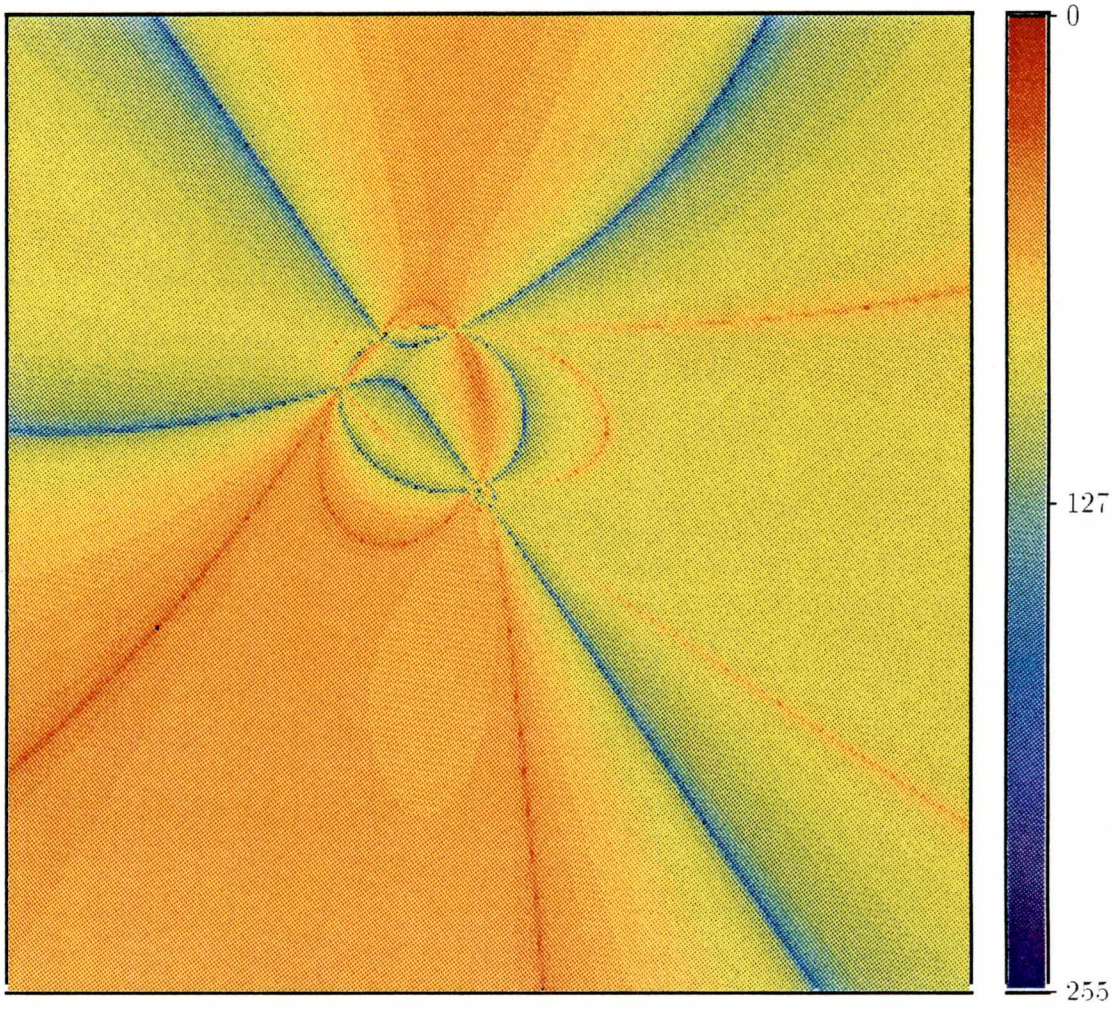


Figure 6.4: The closeup of small oval in Figure (6.3)

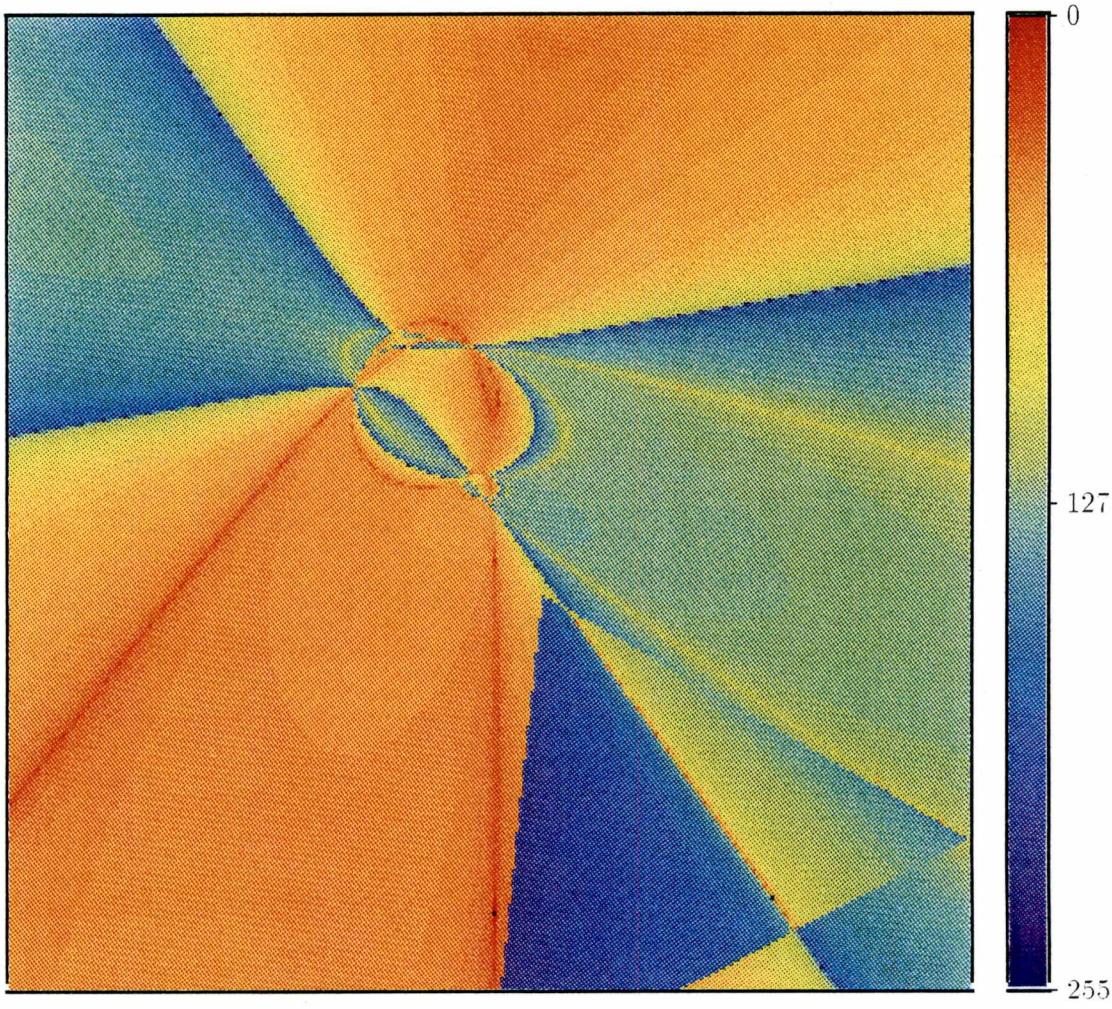


Figure 6.5: Picture for f_2 with two attracting fixed points

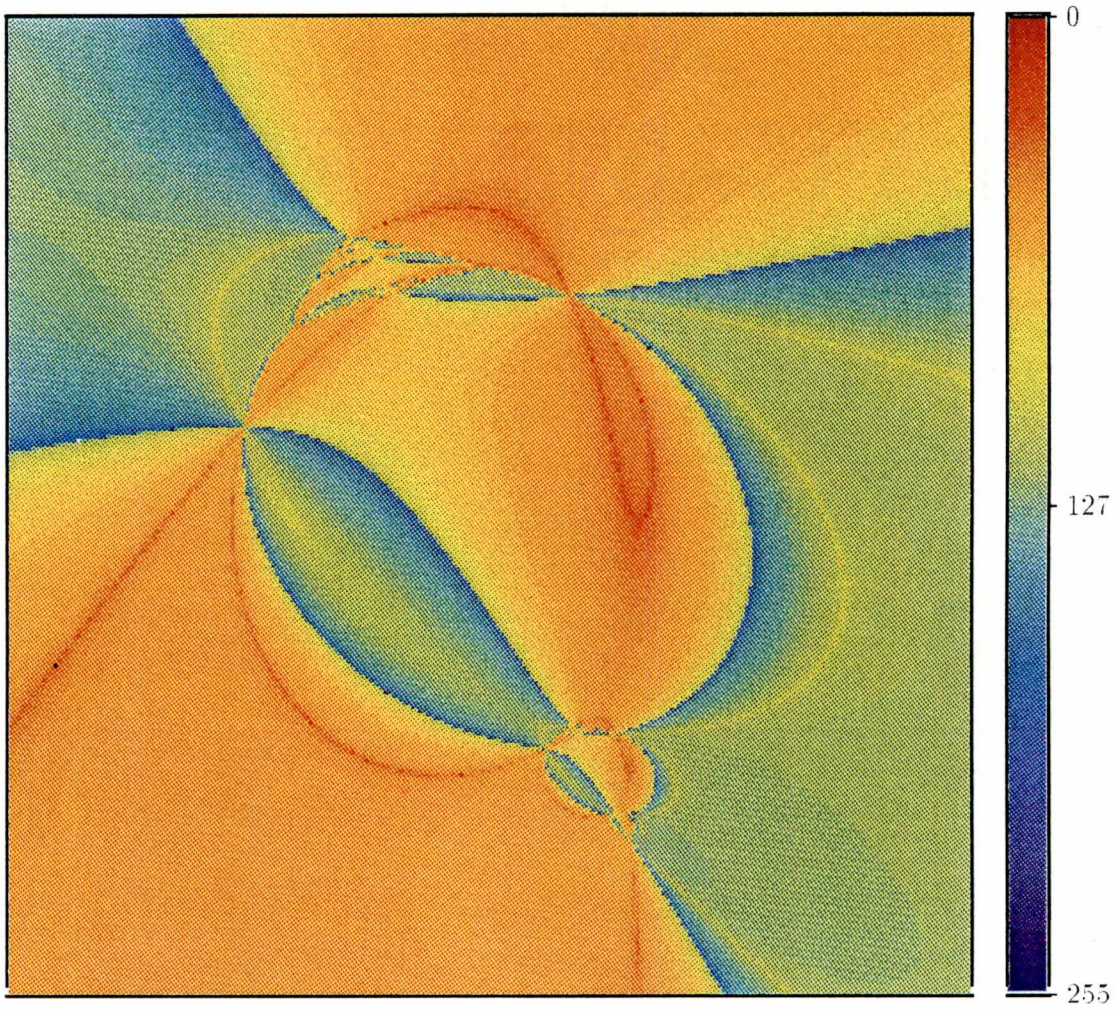


Figure 6.6: The closeup of small oval in Figure (6.5)

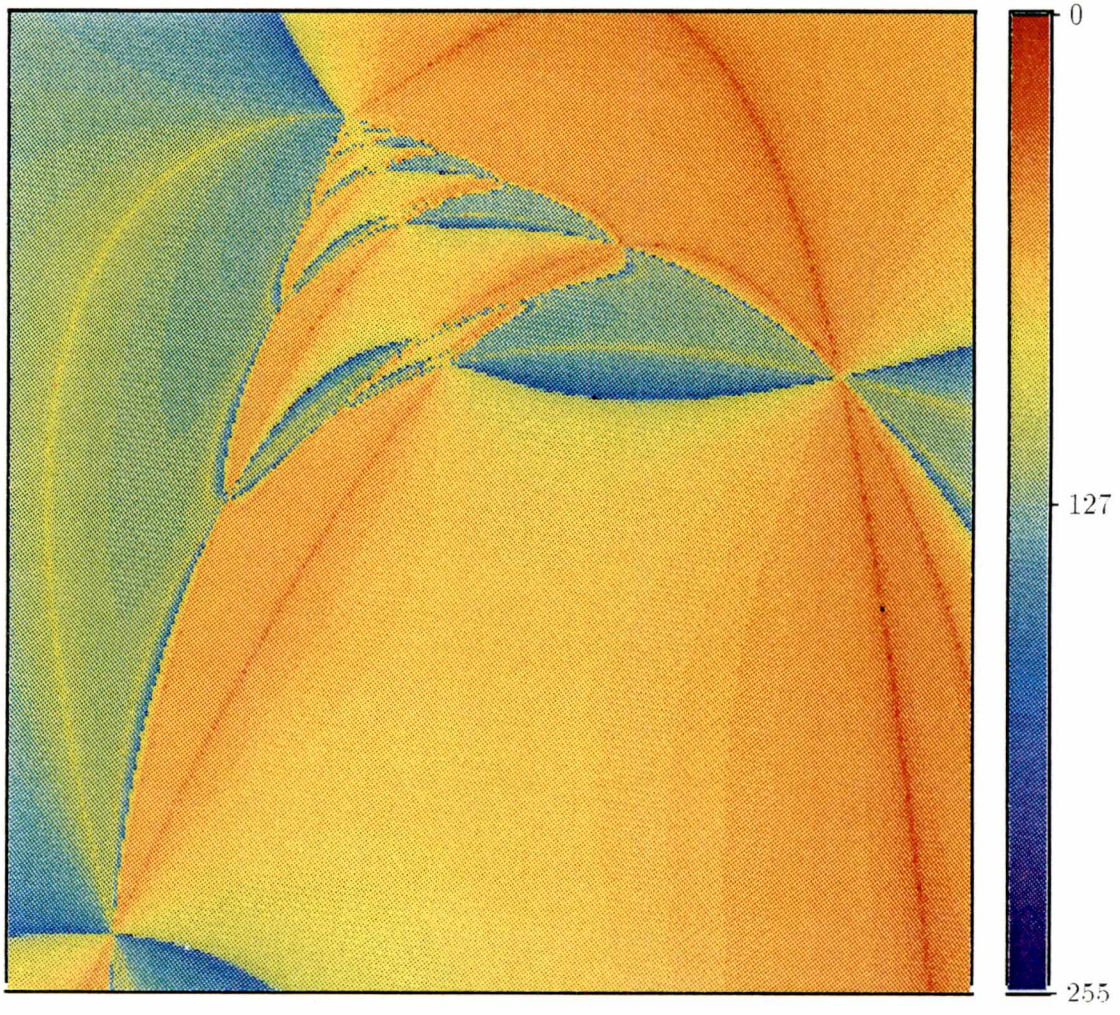


Figure 6.7: The closeup of triangular shape (upper corner) in Figure (6.5)

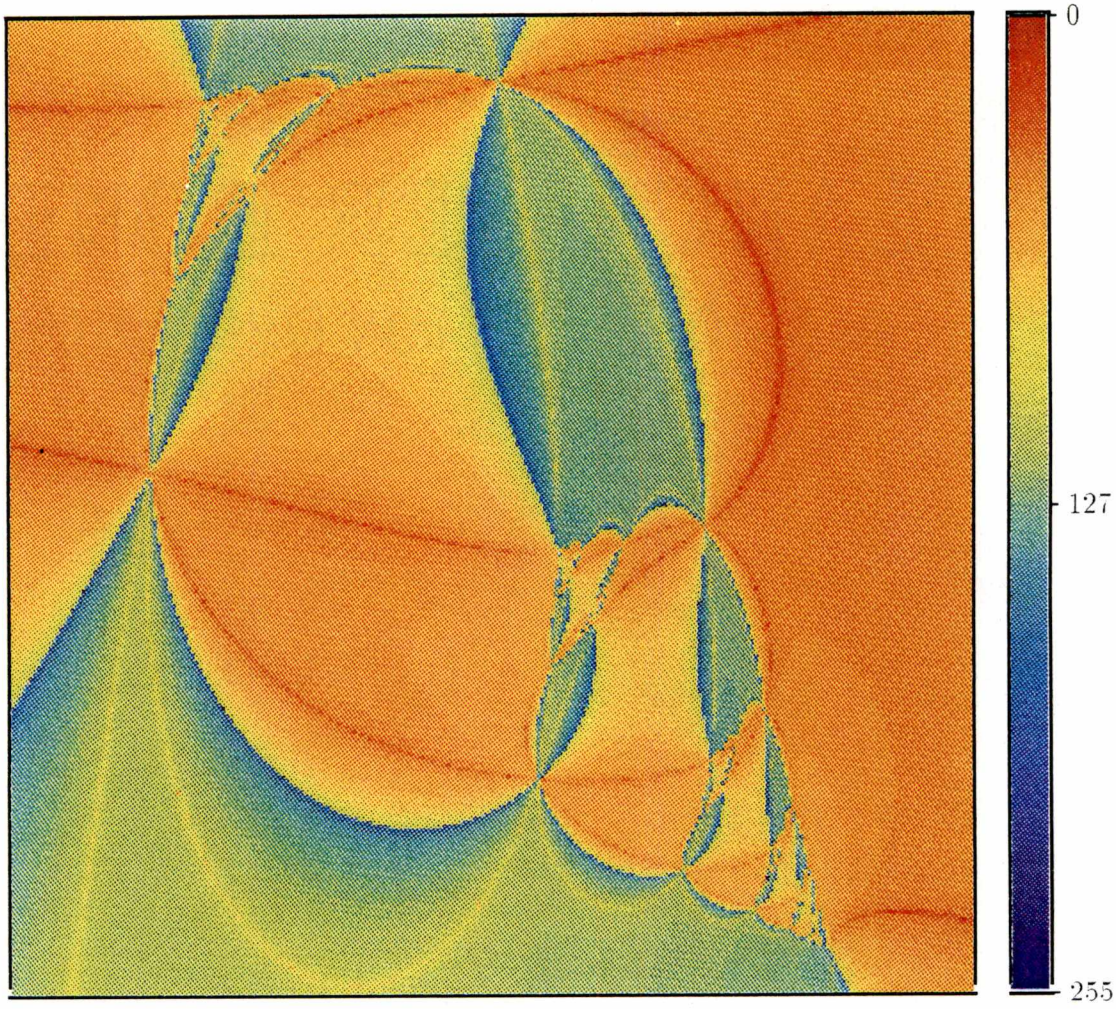


Figure 6.8: Picture for f_2 with different slicing plane

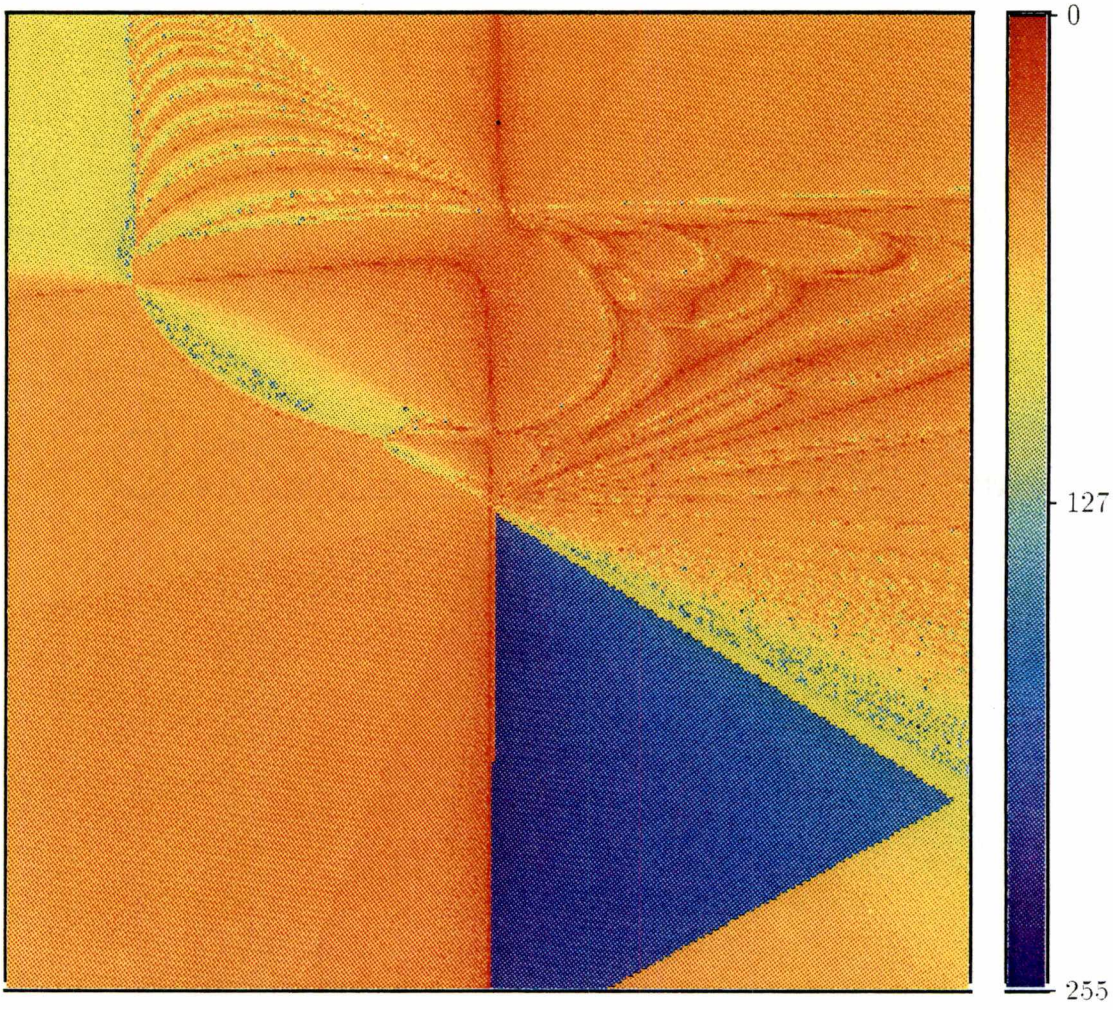


Figure 6.9: Picture for f_3 with one attracting fixed point

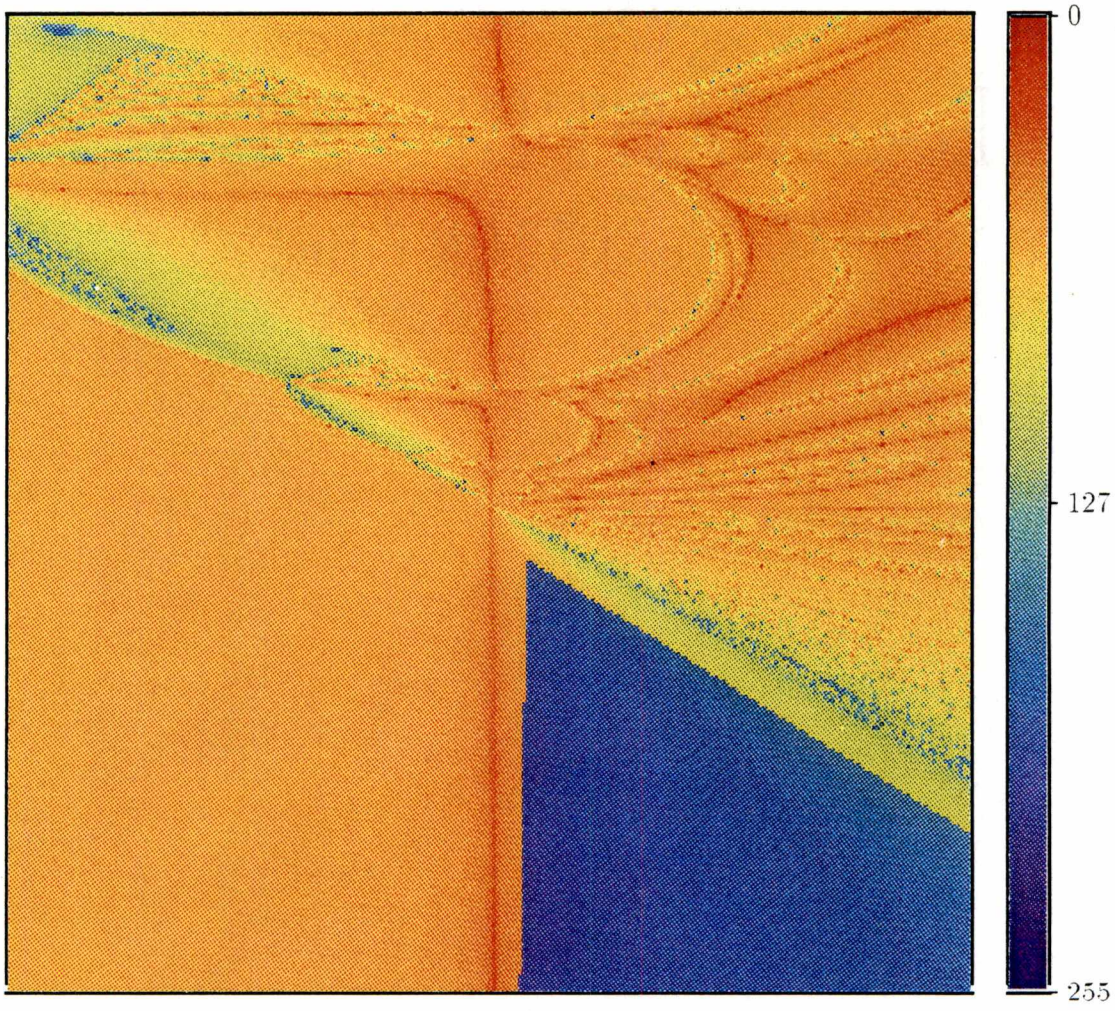


Figure 6.10: The closeup of visible egg shape towards the center of Figure (6.9)

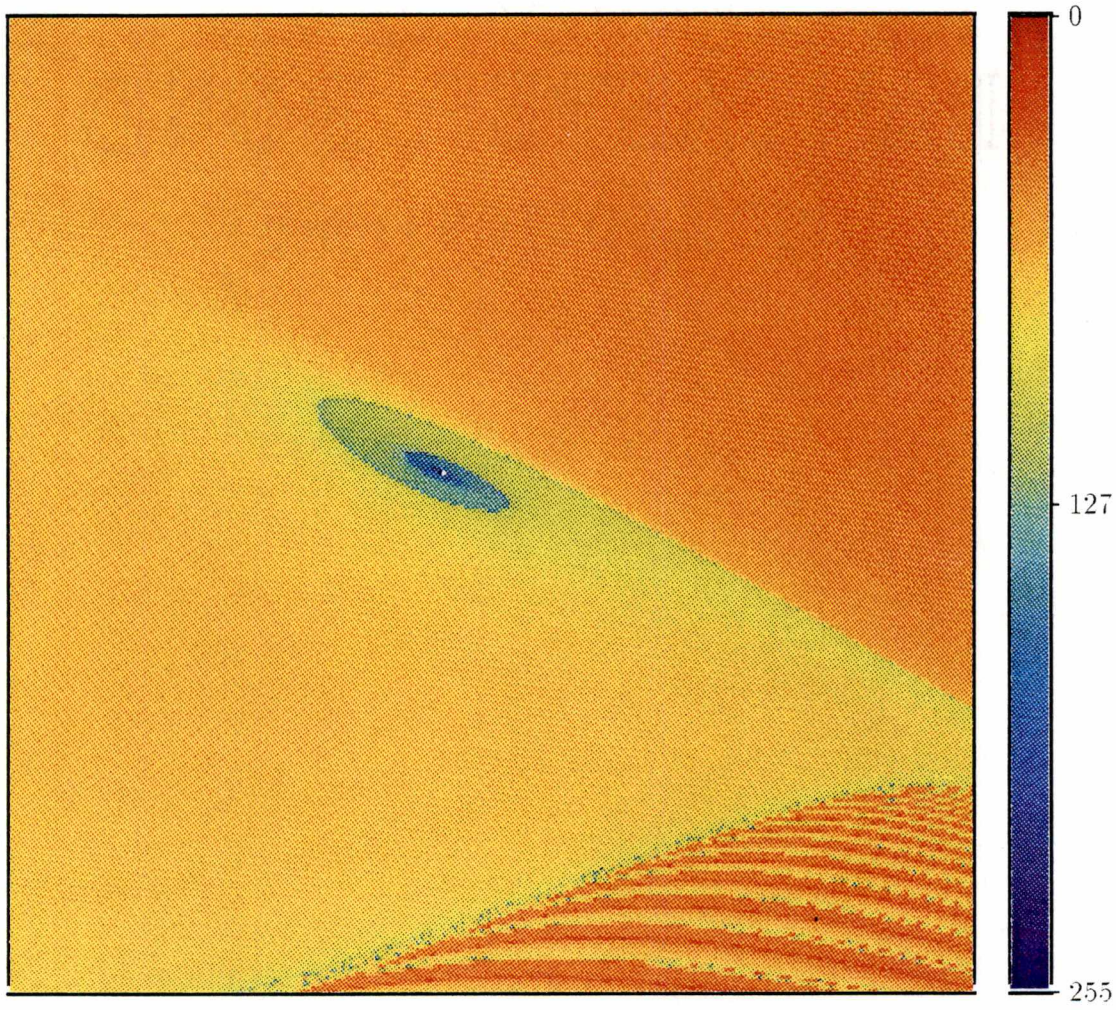


Figure 6.11: The closeup of blue dot at the top left corner of Figure (6.10)

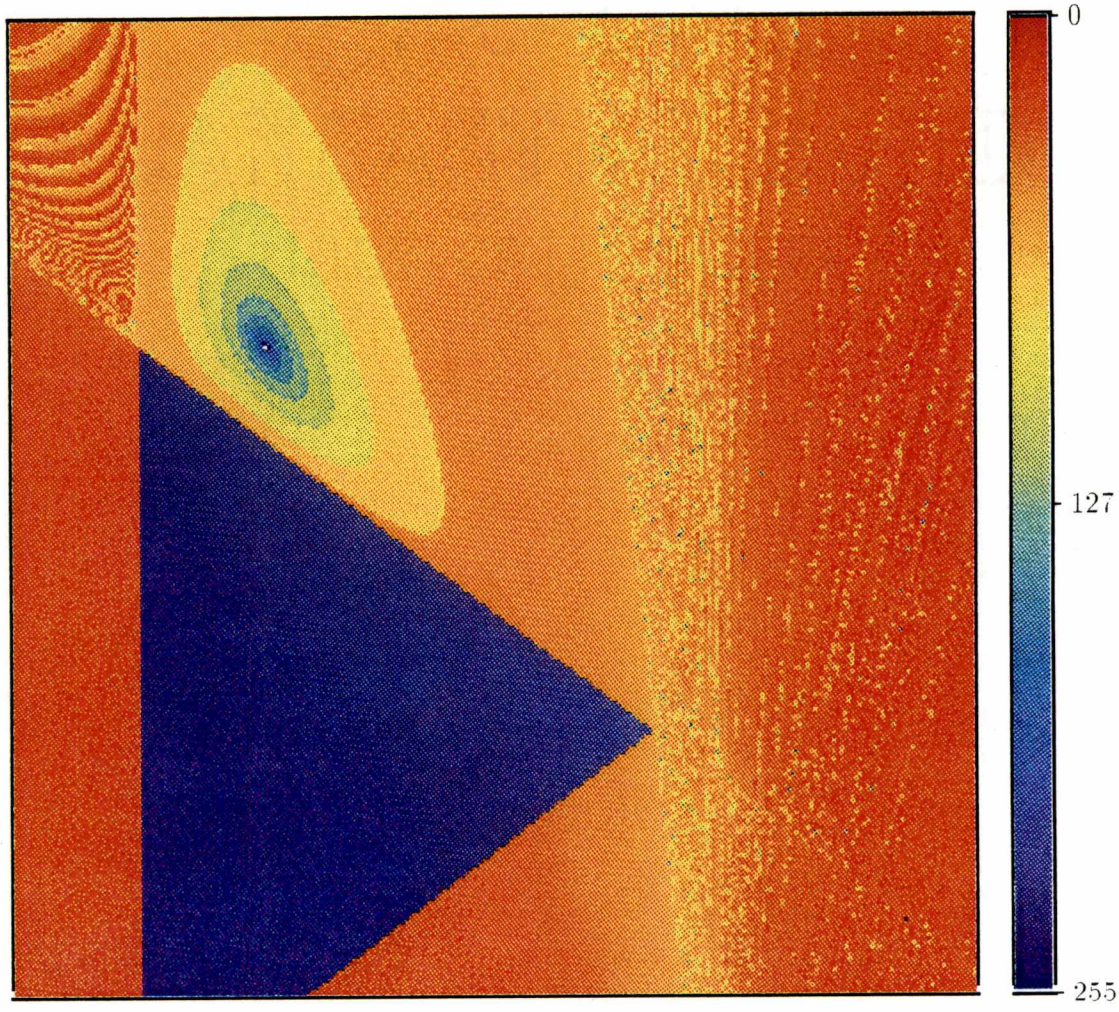


Figure 6.12: Picture for f_3 with different slicing plane

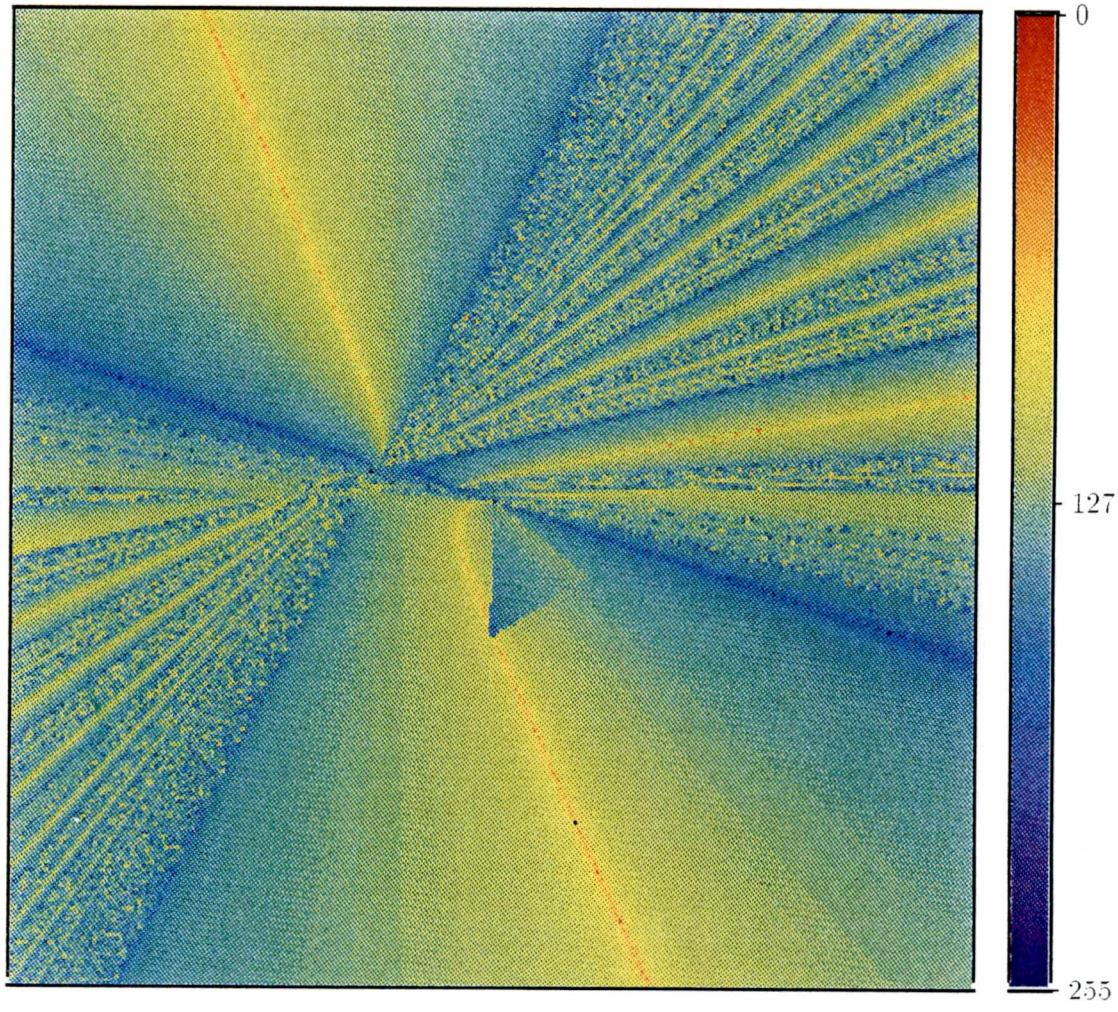


Figure 6.13: Picture for f_4 with one attracting fixed point

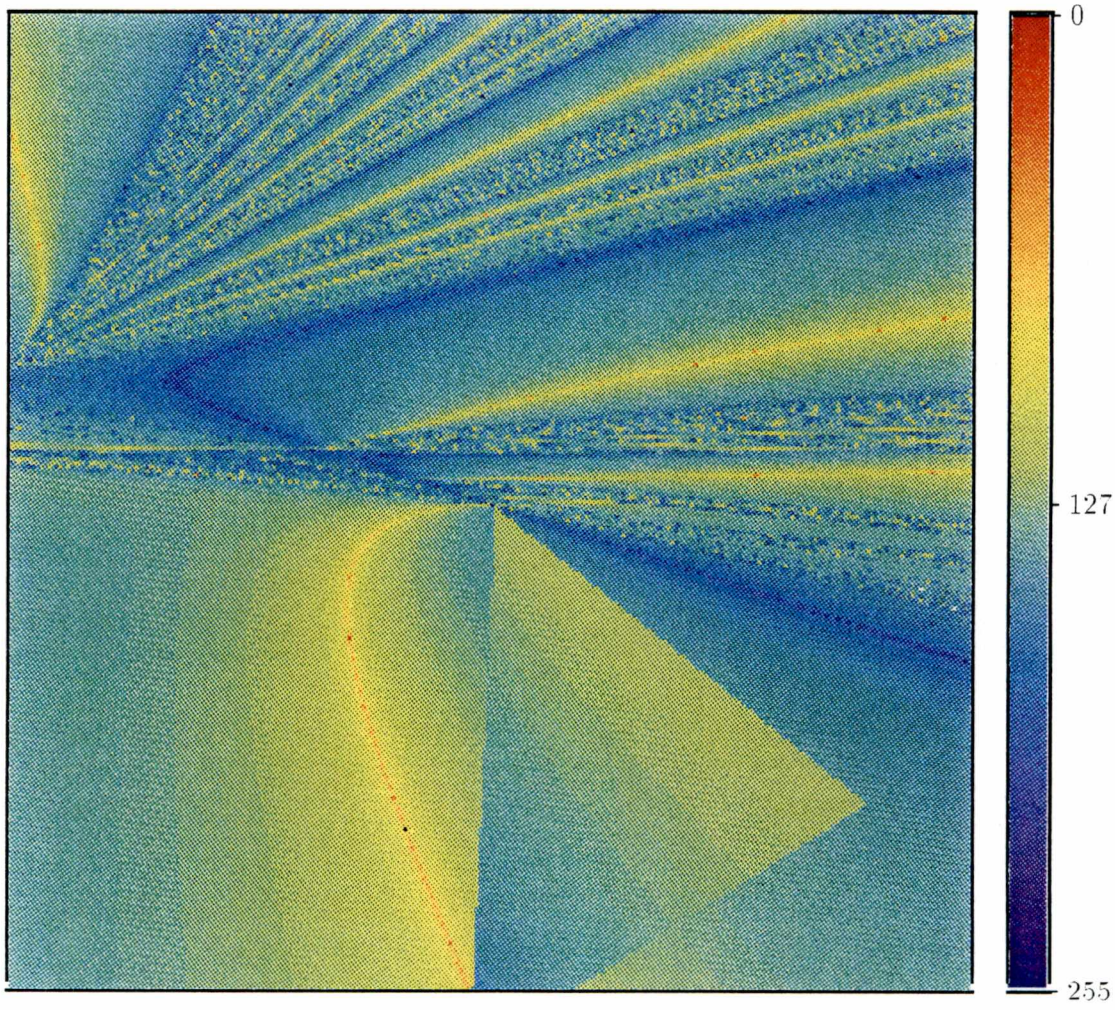


Figure 6.14: The closeup of near the center of Figure (6.13)

6.5 Is It Really A Fractal?

The proof that the pictures are actually fractals is straightforward, given a convenient definition of fractal. In this thesis, "fractal" means self similar. That is, given any region near α, there is a smaller region closer to α and similar to the original.

By the definition of the differential,

$$\mathcal{G}^{-1}(x+\delta) = \mathcal{G}^{-1}(x) + \mathrm{d}\mathcal{G}_x^{-1}\delta + o(\delta) \quad . \tag{6.1}$$

If x is a fixed–point, then $x = \mathcal{G}^{-1}(x)$ and

$$\mathcal{G}^{-1}(x+\delta) = x + \mathrm{d}\mathcal{G}_x^{-1}\delta + o(\delta) \quad . \tag{6.2}$$

Considering the fixed–point as the origin of the coordinates,

$$\mathcal{G}^{-1}(\delta) = \mathrm{d}\mathcal{G}_x^{-1}\delta + o(\delta) \quad . \tag{6.3}$$

So, near the fixed point, $\mathcal{G}^{-1}$ acts like a linear transformation.

Note that because α is a limit obtained by iterating $\mathcal{G}^{-1}$, it is an attracting fixed point. Hence $\mathcal{G}^{-1}$ maps the given region closer to α. Moreover, the coloring scheme will preserve relative colors because if iterations of $\mathcal{G}$ from x take longer to converge than from y, then iterations from $\mathcal{G}^{-1}(x)$ take longer than from $\mathcal{G}^{-1}(y)$.

Therefore, as one moves closer and closer to α, one sees the same kinds of patterns repeated on a smaller and smaller scale but with slight variation. The linear variation is due to $\mathrm{d}\mathcal{G}_x^{-1}$, and small nonlinear variations are due to $o(\delta)$ in

the equation (6.3). This causes $\mathcal{G}^{-1}$ to stretch and distort the region that it is mapping. However, as one moves very close to α, even the distortions settle into a predictable pattern since the action of $\mathcal{G}^{-1}$ near α is given by its differential, which is a linear map.

It should be noted that the fractal is three dimensional. The pictures included are from a two dimensional slice through the fractal. Some of the "repeated patterns" referred to above may be moving outside the viewing plane while others are entering into the viewing plane as α is being approached.

6.6 Conclusion

The schema theorem gives approximate information concerning the propagation of schema from one generation to the next. However, if the composition of the population is unknown, then determining the representation of schema in the population is impossible. Conversely, if the schema is not known, then neither is the population. In fact, knowing the schema is equivalent to knowing the population. This points to the fundamental importance of $\mathcal{G}$, since $\mathcal{G}$ determines the expected next population exactly.

The function $\mathcal{G}$ is of manifest importance. Surprisingly, there has been only scant research directed towards understanding $\mathcal{G}$. Visualization of the time to convergence behavior of the sequence of iterates

$$x, \quad \mathcal{G}^{(1)}(x), \quad \mathcal{G}^{(2)}(x), \quad \mathcal{G}^{(3)}(x), \ \ldots$$

lead to the discovery of the GA fractal. The resulting graphic illustrates a fascinatingly complex evolutionary system governed by selection, crossover, and mutation. However, a deeper knowledge of $\mathcal{G}$ is needed to understand the fractal completely.

6.7 Future Research

This thesis makes it possible to present a portrait of a GA behavior that before has not been possible. However, a major bottleneck lies in the expensive computation which generates the fractals. Should one be able to minimize the computational time, more pictures can be produced which would facilitate an investigation of the dependence of the fractal on fitness and other GA parameters. One could also implement the program under different environment such as MasPar (SIMD machine), Delta (256 processors), PVM (heterogeneous machines), or others. Also, since the fractal is three dimensional, it will be interesting to see three dimensional pictures.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Goldberg, D. E. (1989), *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison Wesley, Reading, MA.

[2] Holland, J. H. (1975), *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor.

[3] Michalewicz, Z. (1992), *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag Berlin Heidelberg.

[4] Nix, A. E. and Vose M. D. (1992), *Modeling Genetic Algorithms with Markov Chains*, Annals of Mathematics and Artificial Intelligence 5, page 79-88.

[5] Vose M. D. (1990), *Formalizing Genetic Algorithms*, Proc. IEEE wksp. on G.A.s, N.N.s, & S.A. applied to problems in Signal & Image Processing, May 1990, Glasgow, U.K.

VITA

Jenny Juliany is an international student from the Republic of Indonesia. She was born in Surabaya, East-Java on July 28, 1969, and first came to the United States of America on August of 1987 as a high school exchange student. She graduated in 1988 and enrolled Wofford College, South Carolina in Fall of 1988. In Spring of 1991, she received a Bachelor of Arts degree with major in Mathematics. Moving to Knoxville, she joined the graduate program in the University of Tennessee in the subsequent semester as a graduate teaching and research assistant. As a graduate research assistant, she worked with Dr. Michael Vose at the University of Tennessee. She graduated with a Master of Science degree in Summer 1993, in computer science.