(Brest State Technical University 2006 Fall Semester: Course Practice)

Contemporary Intelligent Information Technology

Akira Imada
(e-mail akira@bstu.by)

This document is still under construction and was lastly modified on

December 16, 2008

1 The Simplest Test Function — Sphere Model

The first task of this practice is to obtain the minimul value of a multi-dimensional function.

To be more specific, we now assume that we have the following function defined in a 20-dimensional space:

$$y = x_1^2 + x_2^2 + x_3^2 + \cdots + x_{20}^2. \tag{1}$$

Then obtain which point of $(x_1, x_2, x_3, \cdots, x_{20})$ gives a minimum value of y and how much is the value of minimum y. Now, try the following algorithm.

Algorithm 1 (The simplest example)

1. *Create, say, 100 chromosomes at random.*
   - *The number of gene is 20.*

     Thus our chromosomes here have the form $(x_1, x_2, x_3, \cdots, x_{20})$.
   - *Assume here each of x_i takes the continuous value from -1 to 1, that is*

     $-1 < x_i < 1$.
2. *Calculate fitness value by $y = x_1^2 + x_2^2 + x_3^2 + \cdots + x_{20}^2$. Note that the smaller the better.*
3. *Select 2 chromosomes at random from the better half of the population of 100 chromosomes.*
4. *Create a child chromosome by a crossover.*
5. *Give the child a mutation*
6. *Repeat from 2. to 5. 100 times and create the next generation.*
7. *Repeat 6. until the fitness value reaches 0.*

Then the question is as follows.

Excersize 1 (Obtaining the global minimum)
(1) Plot the average fitness value of all the 100 chromosomes versus generation. (2) Also plot the minimum fitness value of each generation.

2 A little more tricky function

Let's try a little more tricky function. for example, the one called Rastrigin's Function.

$$y = nA + \sum_{i=1}^{n} (x_i^2 - A\cos(2\pi x_i)), \quad x_i \in [-5.12 - 5.12].$$

Dimensionality n is arbitorary, but to see how its graph look like, see the Figure when $n = 1$.

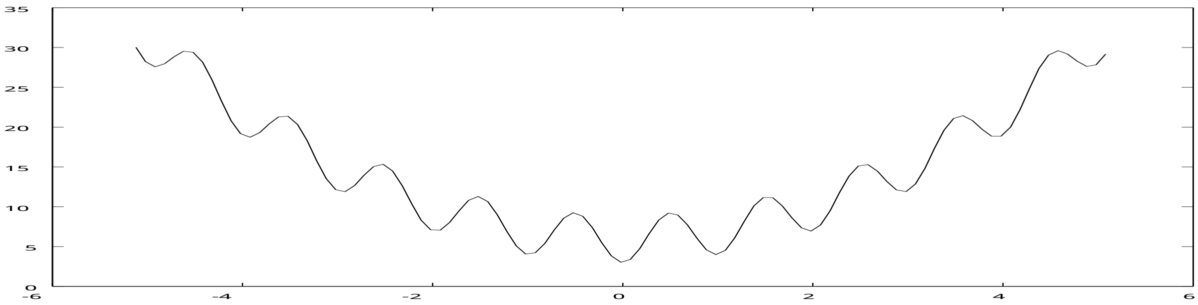


Figure 1: A 2-D version of Rastrigin function

Excersize 2 (Obtaining the global minimum)
(0) Try in the case of $n = 20$. (1) Plot the average fitness value versus generation. (2) Also plot the minimum fitness value of each generation. (3) Make an experiment with different value of mutation rate.

3 2-D Functioon

We now try a 2-D Function in order to observe what will be going on under an evolution. Let's try to find the minimum point of the following function as an example.

$$y = x^4 - 5x^3 - 6x^2 + 8x + 15$$

The graph looks like when $x \in [-2, 5]$

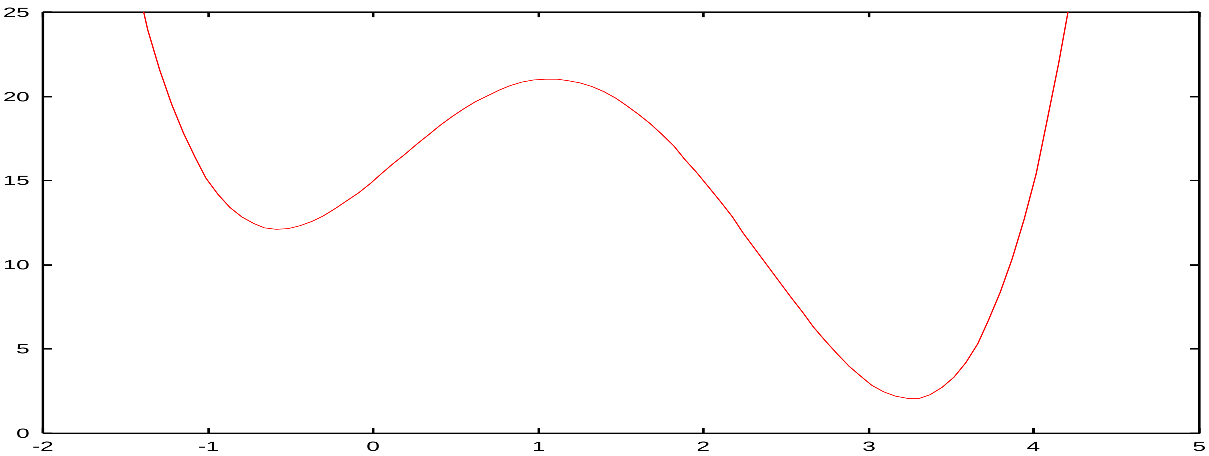


Figure 2: Yet another test function: $y = x^4 - 5x^3 - 6x^2 + 8x + 15$ with $x \in [-2, 5]$.

How you design chromosome to solve this problem?

In the previous problem, the number of genes is n if the function is difiened on n dimensional space. Then our chromosome here has only one gene? How, on earth, we crossover two chromosomes?

The answer is, we use binary chromosome. In the above example, ...

4 Neural Networks for XOR

Assuming McCulloch-Pitts neurons which take the state 1 or 0, the output Y of the neuron which receives weighted-sum of the signals X_i from other N neurons is usually specified as:

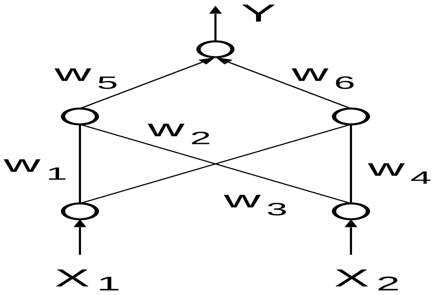
$$Y = \mathrm{sgn}(\sum_{i=1}^{N} w_i X_i - \theta),$$

where $\mathrm{sgn}(x) = 1$ if $x \geq 0$ and 0 otherwise, and w_i and θ are called *weight* and *threshold*, respectively. Here, we assume neurons take binary state but -1 or 1, instead of 0 or 1. Hence the equation is modified as

$$Y = 2 \cdot \mathrm{sgn}(\sum_{i=1}^{N} w_i X_i - \theta) - 1.$$

XOR

X_1	X_2	Y
-1	-1	-1
-1	+1	+1
+1	-1	+1
+1	+1	+1



5 Neural Network for Even-n-Parity

Even-n-Parity is a boolean function to check whether number of 1 of n-bit binary is even or not.

We now assume $n = 4$ for the sake of simplisity. Again our binary made up of -1 and 1 instead of 0 and 1 for a convenience. Hence, as in previous section, transfer function is

$$y_i = 2 \cdot \text{sgn}(\sum_{j=1}^{N} w_{ij} x_j - \theta_j) - 1,$$

where y_i is output of neuron-i, w_{ij} is weight of the synapse from neuron-j to neuron-i, x_j is state of neuron-j, θ is threashold of neuron-j, and N is the number of neurons connected to neuron-i. We assmue here $\theta_j = 0.5$ for all j.

x_1	x_2	x_3	x_4	y
-1	-1	-1	-1	$+1$
-1	-1	-1	$+1$	-1
-1	-1	$+1$	-1	-1
-1	-1	$+1$	$+1$	$+1$
-1	$+1$	-1	-1	$+1$
-1	$+1$	-1	$+1$	-1
-1	$+1$	$+1$	-1	-1
-1	$+1$	$+1$	$+1$	$+1$
$+1$	-1	-1	-1	$+1$
$+1$	-1	-1	$+1$	-1
$+1$	-1	$+1$	-1	-1
$+1$	-1	$+1$	$+1$	$+1$
$+1$	$+1$	-1	-1	$+1$
$+1$	$+1$	-1	$+1$	-1
$+1$	$+1$	$+1$	-1	-1
$+1$	$+1$	$+1$	$+1$	$+1$

We now exploit a feedfoward Neural Network with 4 input neurons, 4 hidden neurons, and one output neurons. So, we have 20 synapsis and as such our chromosome has 20 genes. Create 100 chromosomes with random weight from -1 to 1. Fitness evaluation is by counting the correct answer after giving all the possible 16 cases of 4 inputs, one by one. Then evolve the population.

Excersize 3 (Neural Network for Even-4-Parity)
(1) Plot the average fitness in the population as a function of generation. (2) Plot the maximum fitness in the population as a function of generation. (3) Demonstrate the finally obtained neural network by giving 4 inputs from keybord.

6 Navigoation in gridworld

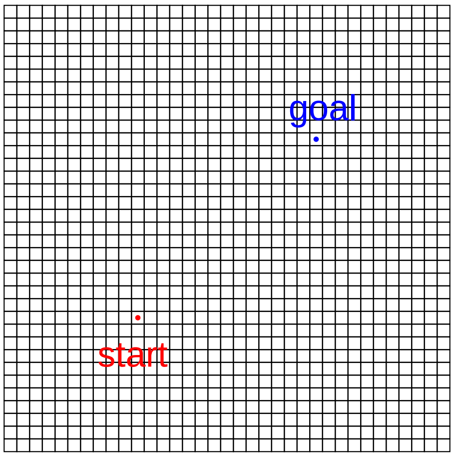


Figure 3: An example of 4 cities and a possible tour therein.

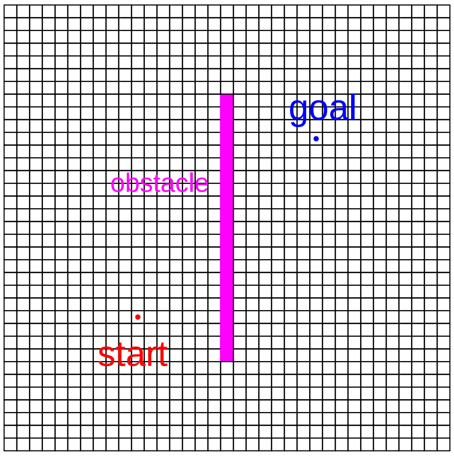


Figure 4: An example of 4 cities and a possible tour therein.

7 Traveling Salesperson Promblem (TSP)

Assuming N cities all of whose cordinate are given, Traveling Salse-person Problem (TSP) is a problem in which a sales-person should visit all of these cities once but only once with its goal being to look for the shortest tour.

We now take a look at 4 cities – A, B, C, and D – as a simplest example. We now assume the cities location are given as follows, for instance.

	(x, y)
A	$(0.83, 7.79)$
B	$(3.28, 8.32)$
C	$(1.52, 4.48)$
D	$(7.65, 3.46)$

Then the Eucledean distances between all possible pair of cities are calculated using a formula:

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{2}$$

where r_{ij} is the distance between city i and city j and (x_i, y_i) and (x_j, y_j) are coordinate of city i and city j, respectively. The distances are:

	A	B	C	D
A	0.000	2.505	3.382	8.074
B	2.505	0.000	4.232	6.539
C	3.382	4.232	0.000	6.214
D	8.074	6.539	6.214	0.000

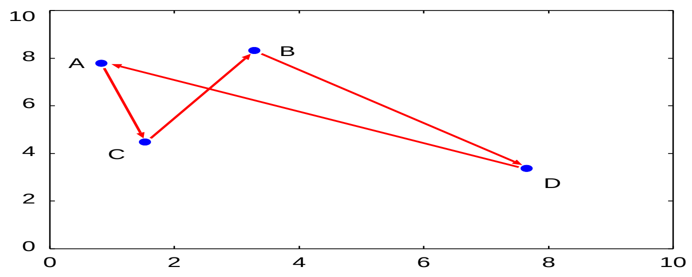


Figure 5: An example of 4 cities and a possible tour therein.

All possible routes in this example are

(A-B-C-D-A), (A-B-D-C-A), (A-C-B-D-A), (A-C-D-B-A), (A-D-B-C-A), and (A-D-C-B-A).

Notice here that lengths of a pair of tours is identical such as a pair (A-B-C-D-A) and (A-D-C-B-A). That is, we have $3!/2 = 3$ routes in total in this example.

Let's see now one root A-C-B-D-A out of them, in the map shown in Figure 5.

The length of the tour in the figure is

$$r_{A-C-B-D-A} = 3.382 + 4.232 + 6.539 + 8.074 = 22.227$$

In the same way, we can calculate the other two route. That is,

$$r_{A-B-D-C-A} = 2.505 + 6.539 + 6.214 + 3.382 = 15.640$$

$$r_{A-B-C-D-A} = 2.505 + 4.232 + 6.214 + 8.074 = 21.025$$

So, the tour of minimum length is A-B-D-C-A (or A-C-D-B-A).

But what if we have larger number of cities? Now you know even in case of 10 cities, we have $9!/2 = 181,440$ possible differnt route. Do you want calculate those distances of all the possible tour? Of course not! Further more, what about 1000 cities, for example?

Then let's apply our evolutionary algorithm. Note that, however, chromosomes like

(B D C)

for tour A-B-D-C-A and

(D C B)

for tour A-D-C-B-A, would not work, because possible child after *one-point crossover* by cutting between 1st and 2nd genes will be

(B C B) and (D D C)

would not be feasible, because both are not a leagal tour – visits one city twice neglecting one city.

Then a possible design of chromosome is as follows.

Step-1. Set $i = 1$.
Step-2. If i-th gene is n then n-th city in the list is the city to be currently visited.
Step-3. Remove the city from the list.
Step-4. Set $i = i + 1$ and repeat Step-2 to Step-4 while $i < n$.

For example, when the list of cities besides the starting city A is

{B, C, D}

chromosome: (121) is the tour:

A-B-D-C-A.

Note that genes can be any integer and *mutation* might be by simply replacing a gene with another random integer. The probability might be 1/number-of-genes (you may change the ratio as an experiment, of course.)

Excersize 4 (TSP)

(1) Create 14 cities by assign random coordinate (x_i, y_i)*.*

(2) Calculate the distance between all the possible two cities.

(3) Then evolve them until the total distance of tour converges one value.

(4) Repeat (3) until fitness value (= total distance of tour) converges a value.

Results you should show me.

- Coordinates of All the cities.
- Matric of distance between any 2 cities.
- Graphic of the location of all the cities and the shortest tour.

8 Knapsack Problem

We now assume n items whose i-th item has weight w_i and profit p_i, then we pick up x_i of the i-th item $i = 1, 2, \cdots, n$ and x_i is non-negative integer. The goal is to maximizes

$$\sum_{i=1}^{n} x_i p_i. \tag{3}$$

such that

$$\sum_{i=1}^{n} x_i w_i < C \tag{4}$$

where C is the capacity of the knapsack.

GA implementation is quite simple. Our chromosomes are in the form

$$(x_1 x_2 x_3 \cdots x_n) \tag{5}$$

with each x_i being the number of the i-th items to be in the knapsack.

Kill infeasible chromosomes

One important aspect is if a chromosome does not fulfill the condition of Eq.(4), simply kill the choromosome and repeat the procedure which resulted in the infeasible child chromosome (cross-over, mutation, or whatever.) untill creating a feasible child chromosome.

Excersize 5 (Knapsack Problem) *Assumming the size of knapsack is, say, 60.*

(1) Create, say, 100 items, by giving each of whose price p_i and size w_i at random, both raging from 0 to 1. For example:

item	price	size
1st	0.37	0.62
2nd	0.52	0.45
3rd	0.95	0.38
$\cdots$	$\cdots$	$\cdots$
100th	0.72	0.32

(2) Creat 40 chromosomes each of which has 100 integer genes, like

$$(5,\ 7,\ 13,\ \cdots\ 2)$$

which means five 1st items, seven 2nd items 13 3rd items, $\cdots$, two 100th items.

(3) *Try to check by replace with one item with price being 0.99 and size being 0.01 Imagine this item is like diamonds small and precious. Hence all items should converge this one. And then replace all items with price being 0.01 and size being 0.01 In this case you know clearly the results.*

(4) *Try evolution and plot maximum fitness vs. generation, as well as average fitness vs. generation*

(5) *Visualize the inside of the knapsack.*

9 Multi Modal Genetic Algorithms – What if we have multipul different meaningful solution?

9.1 Target Functions

Assuming our goal is maximization, that is, we want to know when y takes the maximum value and for which x, we try two test functions.

$$y = \sin^6(5\pi x) \tag{6}$$

and

$$y = -2((x-0.2)/0.8)^2 \sin^6(5\pi x) \tag{7}$$

Now take a look what do these two function look like.

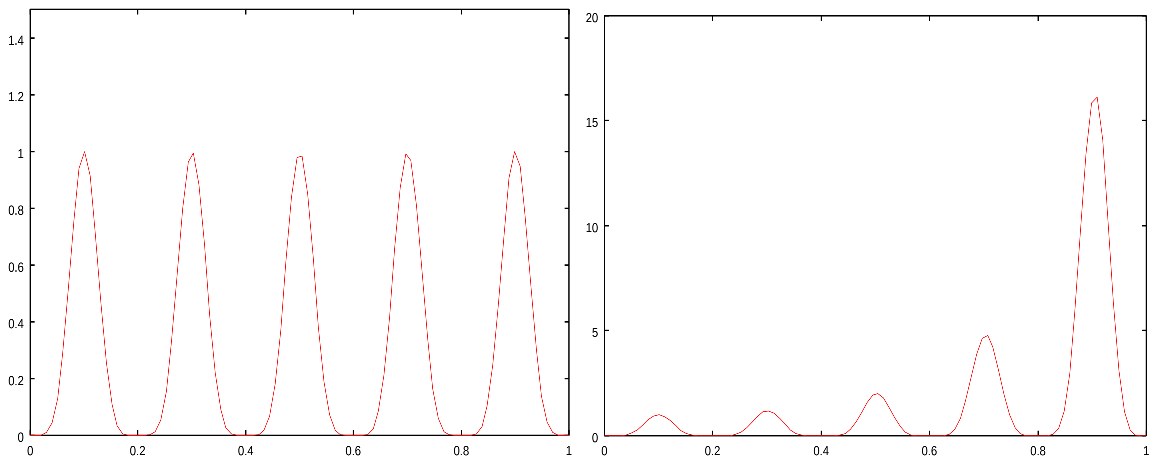


Figure 6: A multi-peak 2-D function and its variation

9.2 Two Algorithms

Here, we have two algorithms for the current purpose of finding multiple solutions at a run.

9.2.1 Fitness Sharing

Fitness of each individual is derated by an amount related to the number of similar individuals in the population. That is, shared fitness $F_s(i)$ of the individual i is

$$F_s(i) = \frac{F(i)}{\displaystyle\sum_{j=1}^{\mu} s(d_{ij})}$$

where $F(i)$ is fitness of individual i; d_{ij} is distance between individual i and j; Typically d_{ij} is *Hamming distance* if in *genotypic space Euclidean distance* if in *phenotypic space* and $s(\cdot)$ is called *sharing function* and defined as:

$$s(d_{ij}) = \begin{cases} 1 - (d_{ij}/\sigma_{\text{share}})^{\alpha} & \text{if } d_{ij} < \sigma_{\text{share}} \\ 0 & \text{otherwise} \end{cases}$$

where σ_{share} is interpreted as size of niche, and α determines the shape of the function. The denominator is called *niche count*. You see shape dependency of $s(d_{ij})$ on α in Figure 9.2.2.

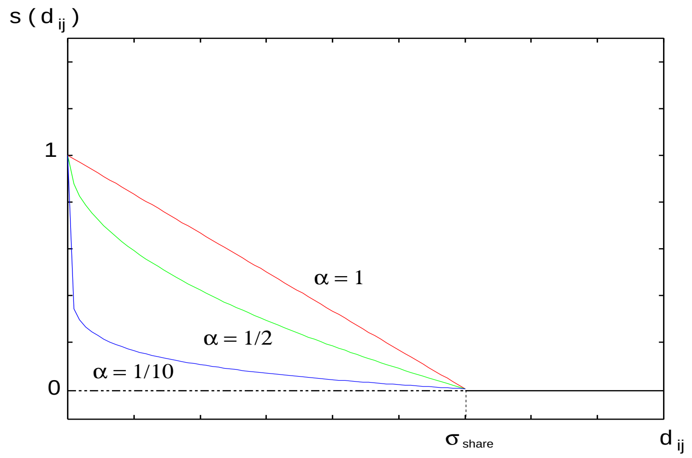


Figure 7: A shape dependency of $s(d_{ij})$ on α.

To be short (not so short though): Similar individual should share fitness. The number of individuals that can stay around any one of peaks (niche) is limited.
The number of individuals stay near any peak will theoretically be proportional to the *hight* of the peak

9.2.2 Deterministic Crowding

If the parents will be replaced or not with their childeren will be determined under a criteria of the distance between parents and children.

Algorithm Assuming crossover, mutation and fitness function are already defiened

1. *Choose two parents, p_1 and p_2, at random, with no parent being chosen more than once.*
2. *Produce two children, c'_1 and c'_2.*
3. *Mutate the children yielding c_1 and c_2, with a crossover.*
4. *Replace parent with child as follows:*
   - *IF* $d(p_1, c_1) + d(p_2, c_2) > d(p_1, c_2) + d(p_2, c_1)$
     * *IF* $f(c_1) > f(p_1)$ *THEN replace* p_1 *with* c_1
     * *IF* $f(c_2) > f(p_2)$ *THEN replace* p_2 *with* c_2
   - *ELSE*
     * *IF* $f(c_2) > f(p_1)$ *THEN replace* p_1 *with* c_2
     * *IF* $f(c_1) > f(p_2)$ *THEN replace* p_2 *with* c_1

where $d(\zeta_1, \zeta_2)$ *is the Hamming distance between two points* (ζ_1, ζ_2) *in pattern configuration space. The process of producing child is repeated until all the population have taken part in the process. Then the cycle of reconstructing a new population and restarting the search is repeated until all the global optima are found or a set maximum number of generation has been reached.*

Hopefully the following two figures would help you understand why.

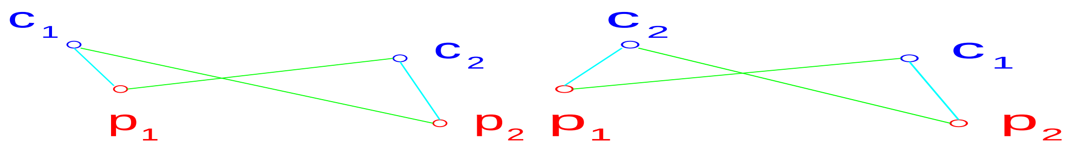


Figure 8: Two cases of parents-children's distance relation.

9.3 Results you should show.

Hopefully you apply two algorithms to each of two test functions. Besides fitness-generation graph, as usual, you try visualize how your individual change their location as generation goes.

That is to say, show all points of individuals in, say, every 20 generations in order to see how theyconverge to the peaks.

10 Multi Modal Genetic Algorithms

11 All-One Problem