innovate achieve lead

Optimization

BITS F312 Neural Networks and Fuzzy Logic

Ankush Jahagirdar

18/11/2016

BITS Pilani
Pilani Campus

innovate achieve lead

Genetic Algorithm

BITS Pilani
Pilani Campus

- $Accuracy = \frac{x_i^{(U)} - x_i^{(L)}}{2^l}$ $\qquad x_i = x_i^{(L)} + \frac{x_i^{(U)} - x_i^{(L)}}{2^l - 1}\left[\sum_{j=0}^{l-1}\left(S_j 2^j\right)\right]$

- Fitness function $f(x_1, x_2, \ldots, x_N)$, chromosome size will be ***Nl***

- Population size : ***M*** in each generation,

- Elite Count: ***EC***

- If crossover probability is $\boldsymbol{p_c}$
  Crossover children $\boldsymbol{CC = round(p_c \times (M - EC))}$

- Mutation Child $\boldsymbol{MC = (M - EC - CC)}$

- For example, let's assume

  Dimension ***N=2***,

  No. of bits per variable ***l=5***

  Population size ***M=20***

  Elite Children ***EC=2***

  Crossover Probability p_c ***=80%*** (or crossover fraction =0.8)

  Then

  Chromosome size

  = Nl = 2 * 5 = 10

  No. of chromosomes in mating pool

  = M-EC = 20 – 2 = 18

- Will these ***18*** elements in the contain Elite Children?

  Yes.

  Such selection is termed as **Greedy**,

  because we are **biased towards stronger** chromosomes.

  No. of Crossover Children

  $$CC = round(p_c \times (M - EC)) = round(0.8 \times (20-2)) = 14$$

  No. of Mutation Children

  $$MC = (M - EC - CC) = 20-2-14 = 4$$

  Next generation will consist of ***2 EC, 14 CC, 4 MC***.

If the selection is **too greedy**, then after few generations there will be **less possible combinations of parents**.

This causes a problem of **Premature Convergence.**

There are various approaches to overcome this problem.

One of the approaches is to increase the **mutations.**

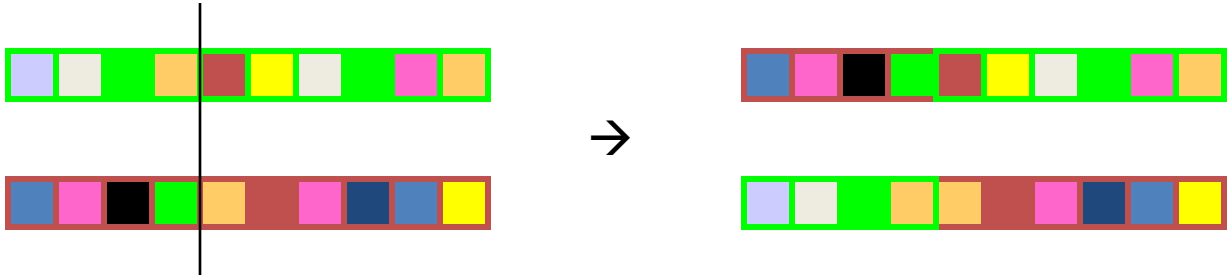
There is always a tradeoff between exploration and exploitation.

1) **Exploitation** consists of probing **a limited (but promising) region of the search space** with the hope of improving a promising solution S that we already have at hand. This operation amounts then to intensifying (refining) the search in the vicinity of S. By doing so, we would be doing, de facto, a local search.

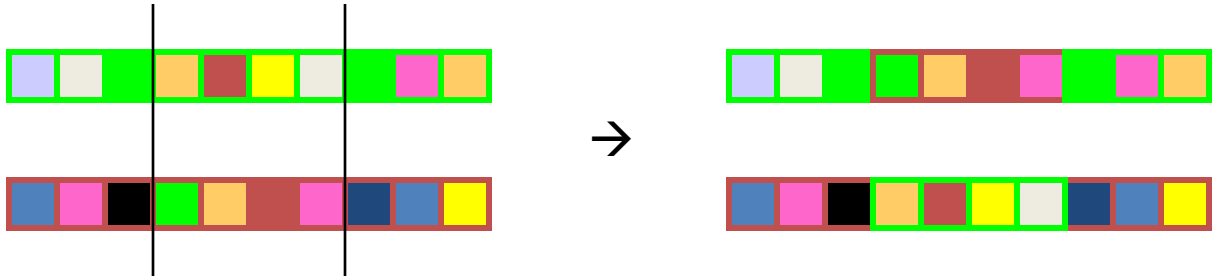
2) **Exploration**, on the other hand, consists of probing **a much larger portion of the search space** with the hope of finding other promising solutions that are yet to be refined. This operation amounts then to diversifying the search in order to avoid getting trapped in a local optimum. By doing so, we would be doing a global search.

Reproduction Operators comparison

- Single point crossover

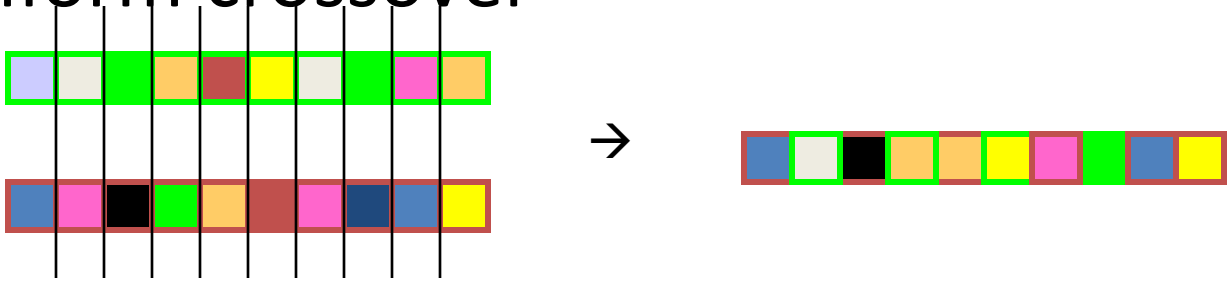


- Two point crossover



Reproduction Operators

- Uniform crossover



  - Is uniform crossover better than single crossover point?
    - Trade off between
      - Exploration: introduction of new combination of features
      - Exploitation: keep the good features in the existing solution

Differential Evolution

BITS Pilani
Pilani Campus

Differential Evolution (DE)

- Price and Storn developed DE to be a reliable and versatile function optimizer that is easy to use.

- The first written publication on DE appeared as a technical report in 1995.

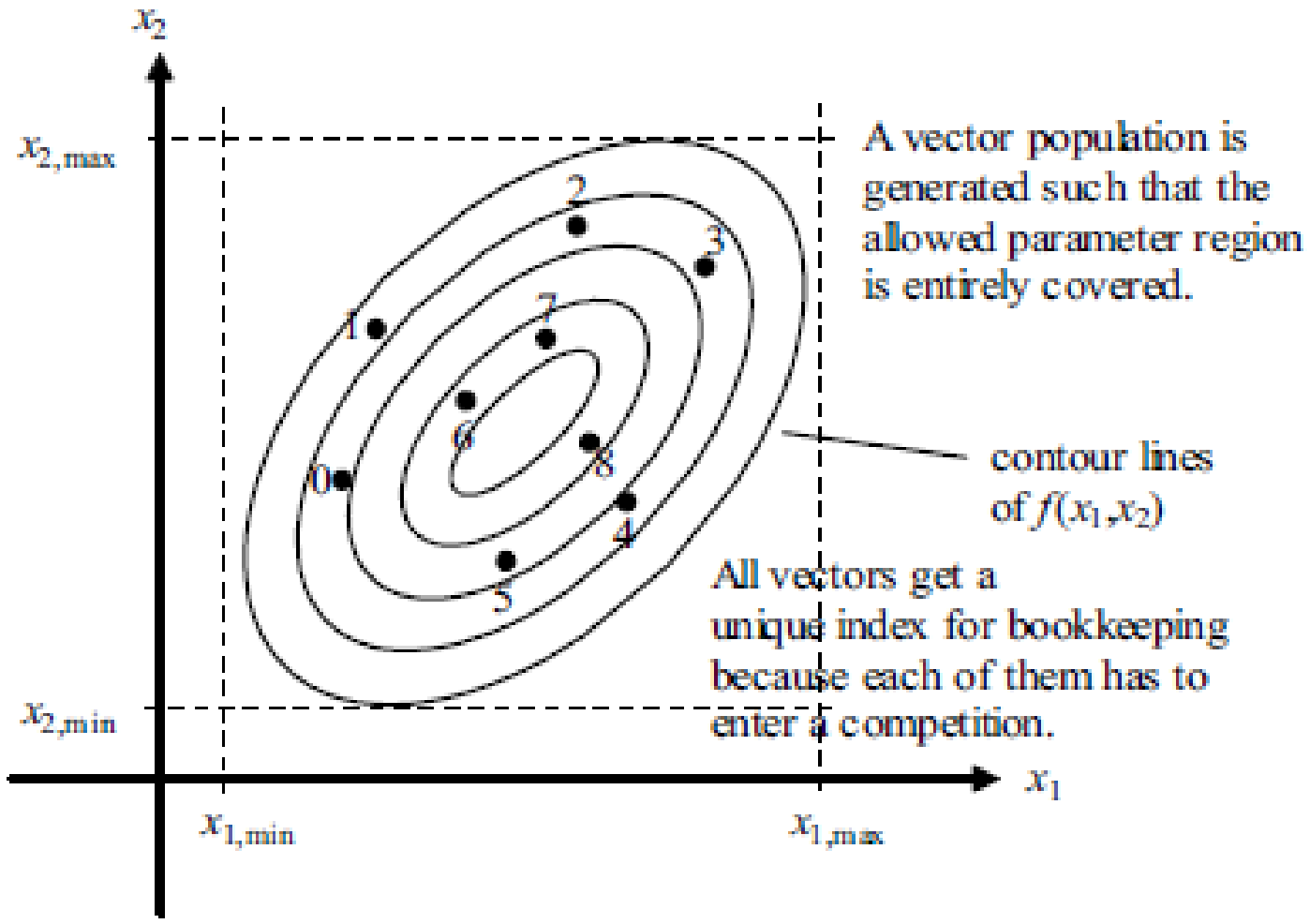
- Since then, DE has proven itself in competitions like the IEEE's International Contest on Evolutionary Optimization (ICEO) in 1996 and 1997 and in the real world on a broad variety of applications.

Basic Idea Behind Differential Evolution (DE)

- Like nearly all EAs, DE is a **population-based optimizer** that attacks the starting point problem by **sampling the objective function** at multiple, randomly chosen initial points.

- Preset parameter bounds **define the domain** from which the N_p vectors in this initial population are chosen (Fig. 1).

- Each vector is **indexed** with a number from **0 to ($N_p - 1$).**



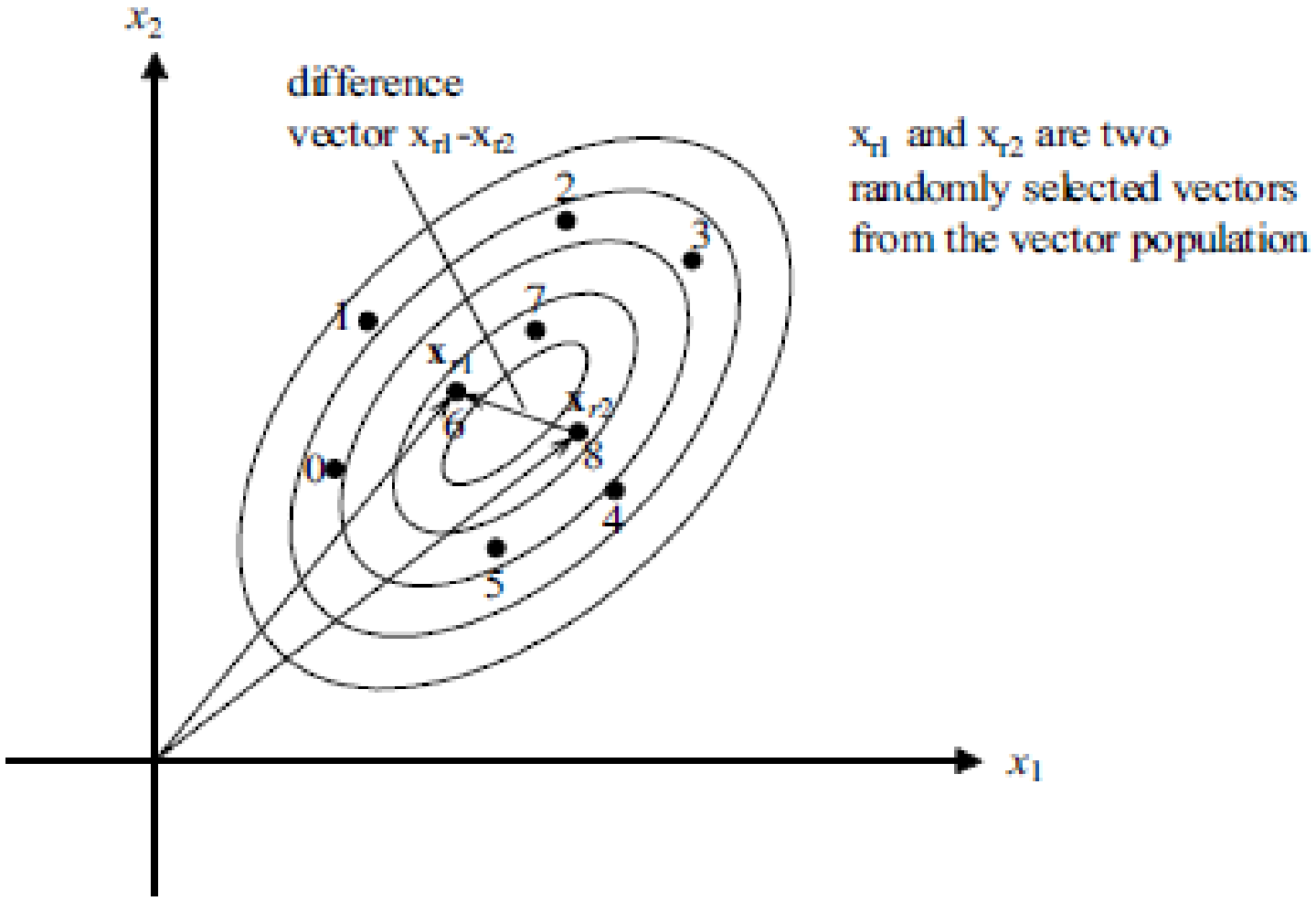
Initializing the DE population

Fig. 1

- Like other population-based methods, DE generates new points that are **perturbations of existing points**.

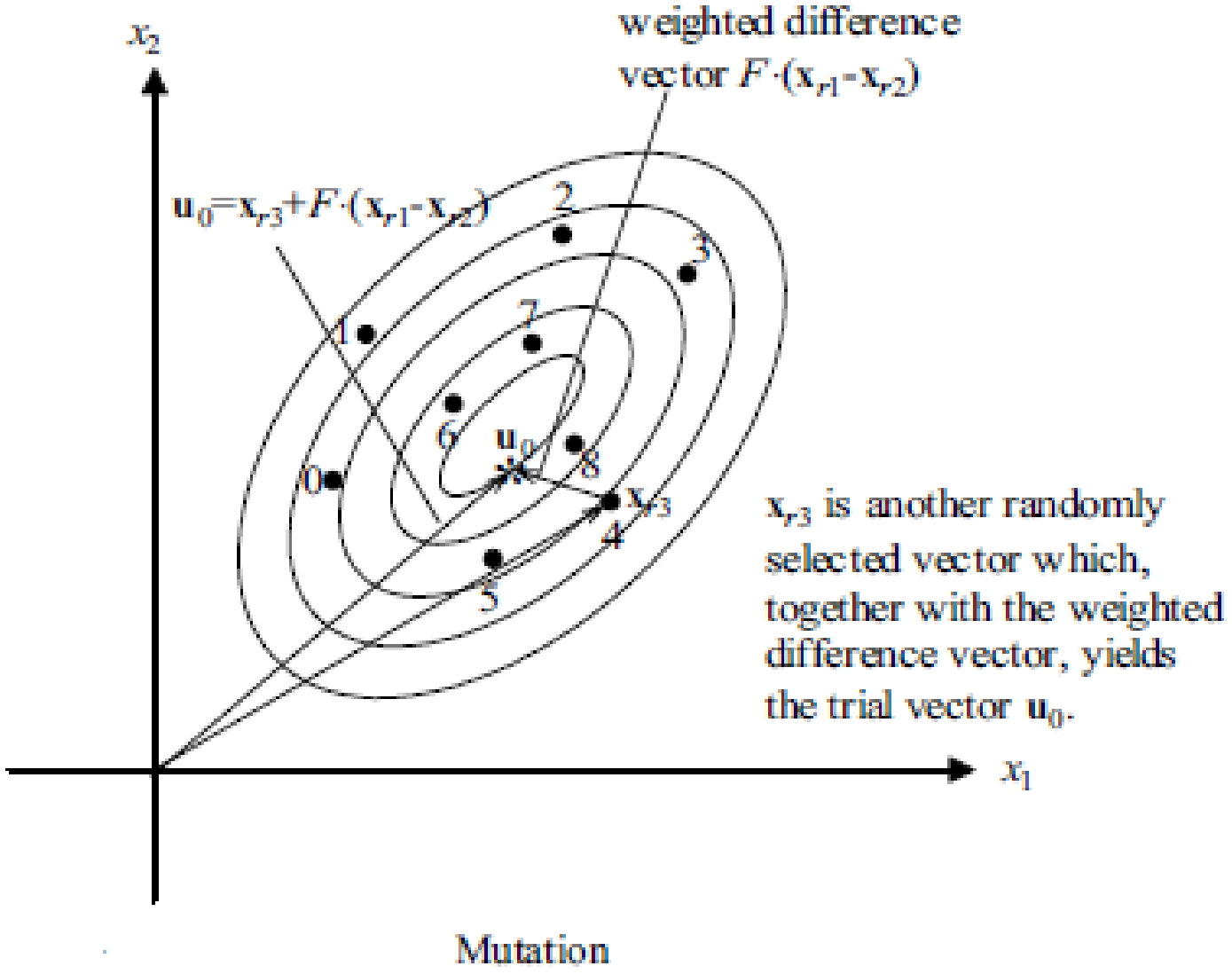
- DE perturbs vectors with the **scaled difference of two randomly selected population vectors** (Fig. 2).

- To produce the trial vector, $\mathbf{u_0}$, DE adds the scaled, random vector difference to a **third randomly selected population vector** (Fig. 3).



Generating the perturbation: $x_{r1} - x_{r2}$

Fig. 2



Mutation

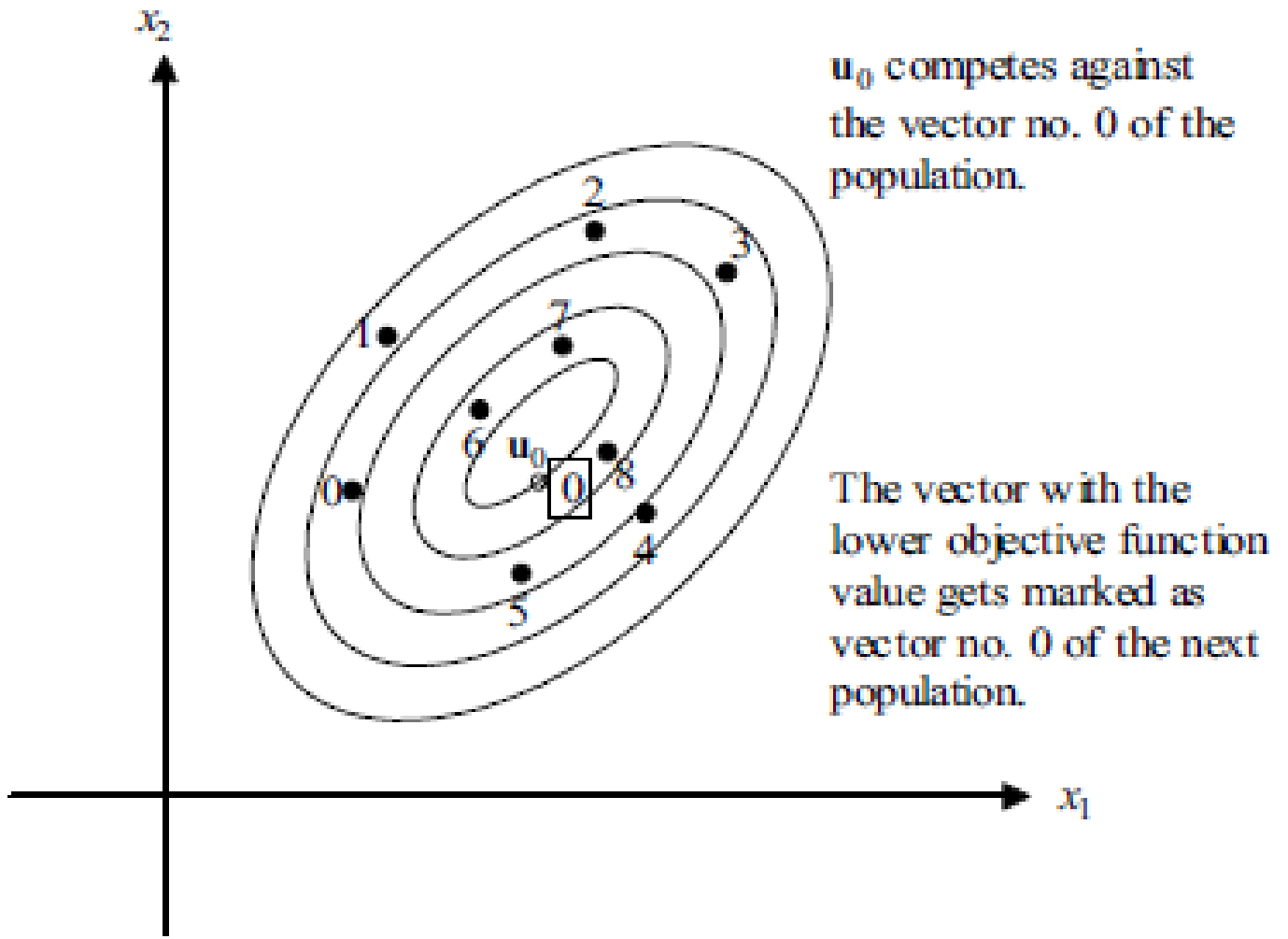
Fig. 3

- In the selection stage, the **trial vector competes against the population vector** of the same index, which in this case is number 0.

- Fig. 4 illustrates the select-and-save step in which the vector with **the lower objective function value** is marked as a member of the next generation.

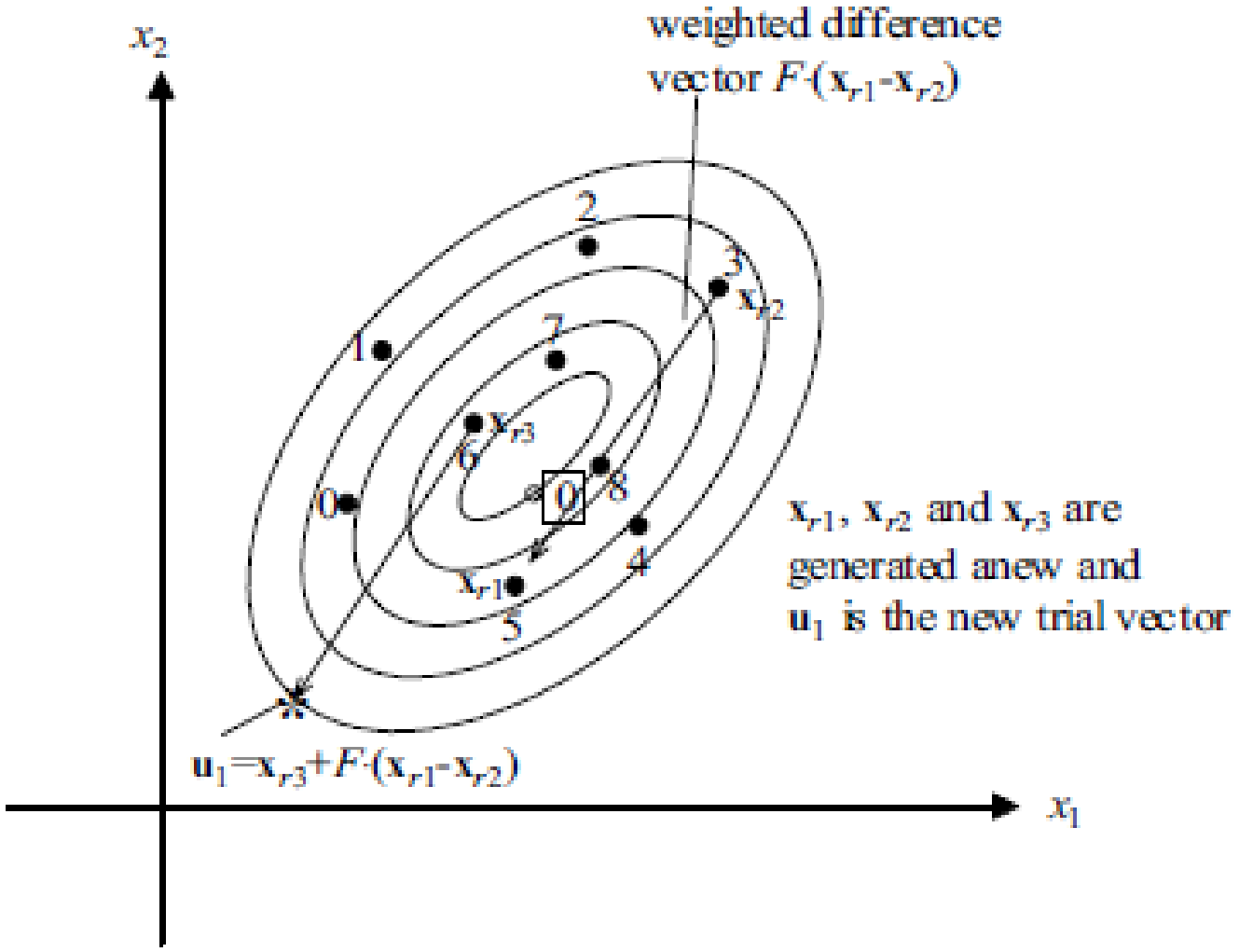
- Fig. 5–6 indicate that the procedure repeats until all N_p population vectors have competed against the randomly generated trial vector.

- The **survivors of the N_p pairwise competitions become parents** for the next generation in the evolutionary cycle.



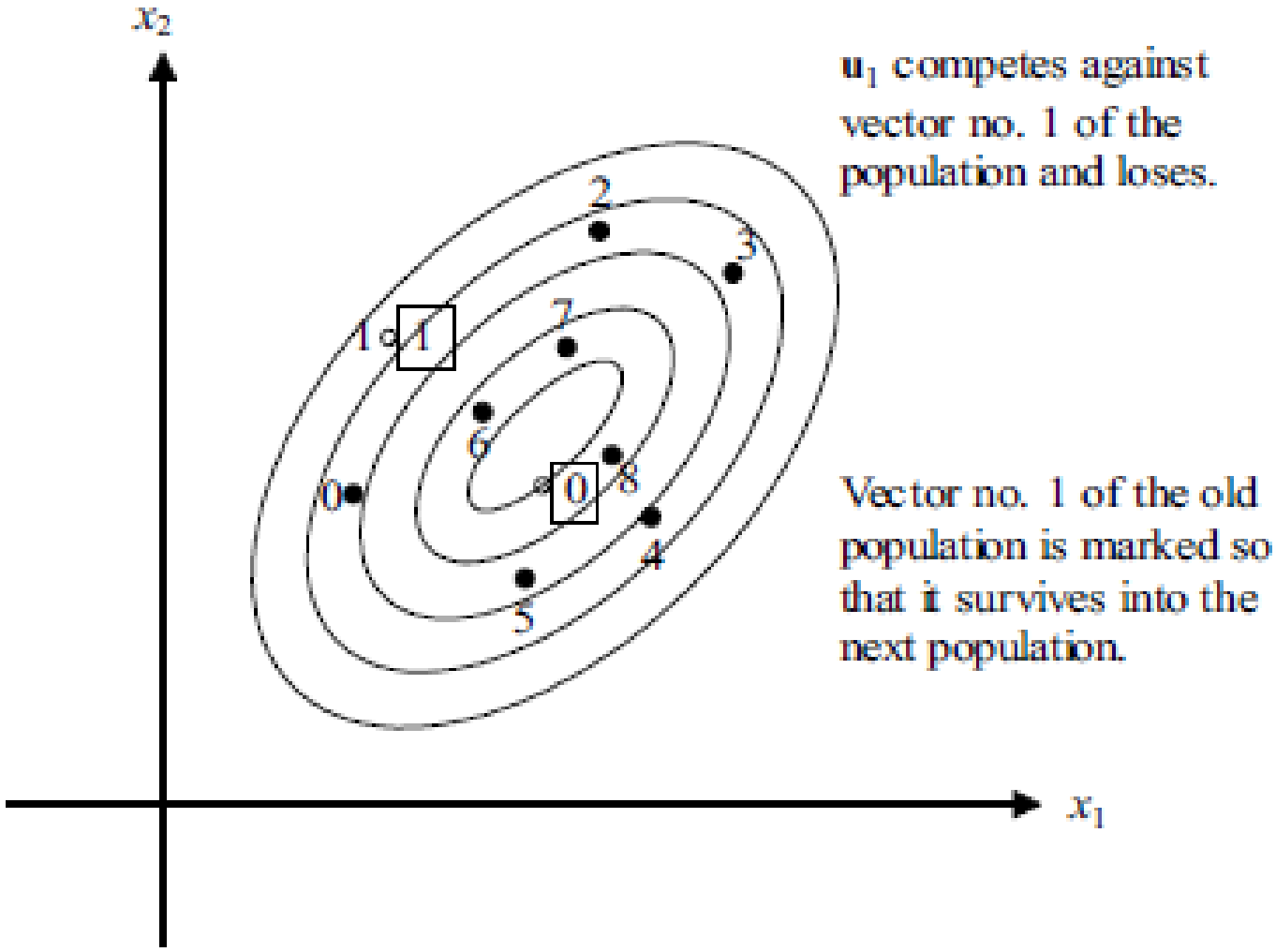
Selection. Because it has a lower function value, u_0 replaces the vector with index 0 in the next generation.

Fig. 4



A new population vector is mutated with a randomly generated perturbation.

Fig. 5



Selection. This time, the trial vector loses.

Steps for Differential Evolution (DE)

- Consider a maximization or minimization problem:
  - Maximize or Minimize Fitness Function $F(\boldsymbol{x}) = f(\boldsymbol{x})$
  - Initial population range is given as $\boldsymbol{x}_L \leq \boldsymbol{x} \leq \boldsymbol{x}_U$ where $\boldsymbol{x} = [\boldsymbol{x}^{(0)}\ \boldsymbol{x}^{(1)}\ \ldots\ \boldsymbol{x}^{(Np)}]^T$ and $\boldsymbol{x}^{(i)} = [x_1^{(i)}, x_2^{(i)}, \ldots., x_N^{(i)}]$ where i = 0, 1, 2, ..., $(N_p - 1)$
  - For unconstrained optimization, boundary values are not applicable to evolved variables whereas for constrained optimization, evolved variables should follow it strictly.
- Important steps for DE realization
  - Uniform initial population creation
  - Fitness function Evaluation
  - Application of DE operators: Mutation, Crossover, Selection
  - Check boundary conditions and termination criteria

Initial Population Creation

- DE uses following code for uniform creation of initial population $\boldsymbol{x} = [\boldsymbol{x}^{(0)}\ \boldsymbol{x}^{(1)}\ \boldsymbol{x}^{(2)}\ \ldots\ \boldsymbol{x}^{(Np)}]^{T}$ of size $N_p \times N$, where N = no. of variables and N_p = no. population members:

$$
\begin{aligned}
&for\ i = 0 : \left(N_p - 1\right) \\
&\quad for\ j = 1 : N \\
&\quad\quad x_j^{(i)} = x_{jL}^{(i)} + rand \times \left(x_{jU}^{(i)} - x_{jL}^{(i)}\right) \\
&\quad end \\
&end
\end{aligned}
$$

Note: Row no. varies from 0 to (N_p – 1) and column no. varies from 1 to N.

Evaluation of DE Fitness Function

- For **minimization**, it is called as **Cost Function.**
- For **maximization**, it is called as **Profit Function.**

Differential Mutation

- For the $1 \times N$ parameter vector $\boldsymbol{x}^{(i)}$, we have differential mutation defined as

  $\boldsymbol{v}^{(i)} = F \times (\boldsymbol{x}^{(r_1)} - \boldsymbol{x}^{(r_2)}) + \boldsymbol{x}^{(r_3)}$

  where

  $i = 0, 1, 2, \ldots, (N_p - 1)$

  F is Amplification factor

  random integer indexes $r_1 \neq r_2 \neq r_3 \neq i \in \{0, 1, 2, \ldots, (N_p - 1)\}$

Crossover and Trial vector

- Generate a $1\times N$ random number vector and compare it with constant Crossover Rate (CR), which results in a logical constant given as follows:

  $K_1^{(i)} = \text{rand}(1\times N) \leq CR$

  For example: $K_1^{(i)} = [0.9 \;\; 0.3 \;\; 0.8 \;\; 0.5 \;\; 0.2] \leq 0.8 \;\; = [0 \;\; 1 \;\; 1 \;\; 1 \;\; 1]$

- Generate another logical constant $K_2^{(i)}$ which is complement of $K_1^{(i)}$ as follows:

  $K_2^{(i)} = K_1^{(i)} < 0.5$

  For example: $K_2^{(i)} = [0 \;\; 1 \;\; 1 \;\; 1 \;\; 1] < 0.5 \;\; = [1 \;\; 0 \;\; 0 \;\; 0 \;\; 0]$

- Generate trial vector as a result of crossover operation between mutation and population vectors as given below:

$$\boldsymbol{u}^{(i)} = K_1^{(i)}\,\boldsymbol{v}^{(i)} + K_2^{(i)}\,\boldsymbol{x}^{(i)}$$

- Above equation means that if random values in crossover operation are
  - less than or equal to *CR*, take element resulting from mutation operation as trial population element,
  - greater than *CR*, take current population element as trial population element.

Selection and Next Generation Population

- Evaluate fitness function $F(\boldsymbol{u})$ for the trial vector $\boldsymbol{u}$.

- Compare $F(\boldsymbol{u}^{(i)})$ with $F(\boldsymbol{x}^{(i)})$ such that if
  - $F(\boldsymbol{u}^{(i)}) \leq F(\boldsymbol{x}^{(i)})$, then $\boldsymbol{x}^{(i)} := \boldsymbol{u}^{(i)}$ i.e. trial vector replaces current population vector for next generation DE optimization.
  - $F(\boldsymbol{u}^{(i)}) > F(\boldsymbol{x}^{(i)})$, then $\boldsymbol{x}^{(i)} := \boldsymbol{x}^{(i)}$ i.e. keep current population vector for next generation DE optimization.

Termination criteria

- Following termination criteria can be used to Stop DE procedure.
  - Maximum number of Generations, say G = 100, is reached.
  - After certain number of generations, say G = 50, change in fitness function values for two consecutive generations is than threshold value, say

    $\{(f_{G=52})-(f_{G=51}) < 10^{-6}$.

Choice of Key Parameters

- Population size should be 5-10 times more than the dimension of the problem.

- Choose F=0.5 initially. If this leads to premature convergence, then increase F.

  Optimal range of F is **0.4 < F < 1**

- Choose CR=0.9 first, then CR=0.1.

  Based on the speed, choose a value between 0 and 1.

Numerical Example for DE

Use unconstrained DE to minimize the objective function:

$f(x_1, x_2) = x_1 - x_2 + 2x_1^2 + 2x_1x_2 + x_2^2$

Following additional information are provided:

(a) Boundary values of variables: $0 \leq x_1, x_2 \leq 6$.

(b) Initial population of variables (x_1, x_2), given in Table 1.

(c) Take crossover rate (CR) = 0.8, amplification factor (F) = 1.2

(d) Random indexes r_1, r_2 and r_3 for mutation operation in first generation are given in Table 1 with difference vector represented as $(x^{(r_1)} - x^{(r_2)})$.

(e) Random number vector K_1 for crossover operation in first generation is given in Table 1.

Show calculations for first generation to obtain (i) fitness function (ii) mutation vector (iii) trial vector (iv) next generation population.

Table 1

Pop no	Initial x_1	Initial x_2	r_1	r_2	r_3	K_1
0	3.677337	0.222648	8	14	7	[0.460200 0.699338]
1	2.338032	3.289922	14	15	3	[0.894391 0.591400]
2	0.528440	5.870270	4	6	16	[0.317430 0.260761]
3	1.187137	4.523740	2	7	6	[0.381619 0.315992]
4	3.133475	4.677932	15	11	13	[0.058299 0.882461]
5	3.216689	2.511420	19	15	0	[0.659134 0.978825]
6	0.447552	1.098901	15	10	14	[0.115894 0.613161]
7	4.642198	1.944511	1	19	3	[0.086053 0.824881]
8	2.412028	0.662900	3	18	13	[0.095357 0.457195]
9	5.517039	5.075050	19	16	17	[0.006763 0.796946]
10	4.422031	4.354949	16	11	7	[0.214108 0.073992]
11	3.360381	3.322859	15	2	9	[0.845929 0.354696]
12	4.689647	2.206954	2	6	10	[0.781496 0.255899]
13	4.020225	3.727019	9	19	17	[0.062814 0.146946]
14	4.645876	2.039149	8	3	5	[0.564136 0.544190]
15	4.023861	3.511431	17	11	18	[0.244590 0.716581]
16	5.000684	5.525994	11	14	4	[0.005858 0.551914]
17	3.093430	4.681739	8	14	7	[0.604323 0.201341]
18	4.498183	2.810695	9	7	13	[0.889923 0.197326]
19	5.460903	3.884746	0	9	6	[0.127865 0.064069]