Differential Evolution

A Simple Evolution Strategy for Fast Optimization

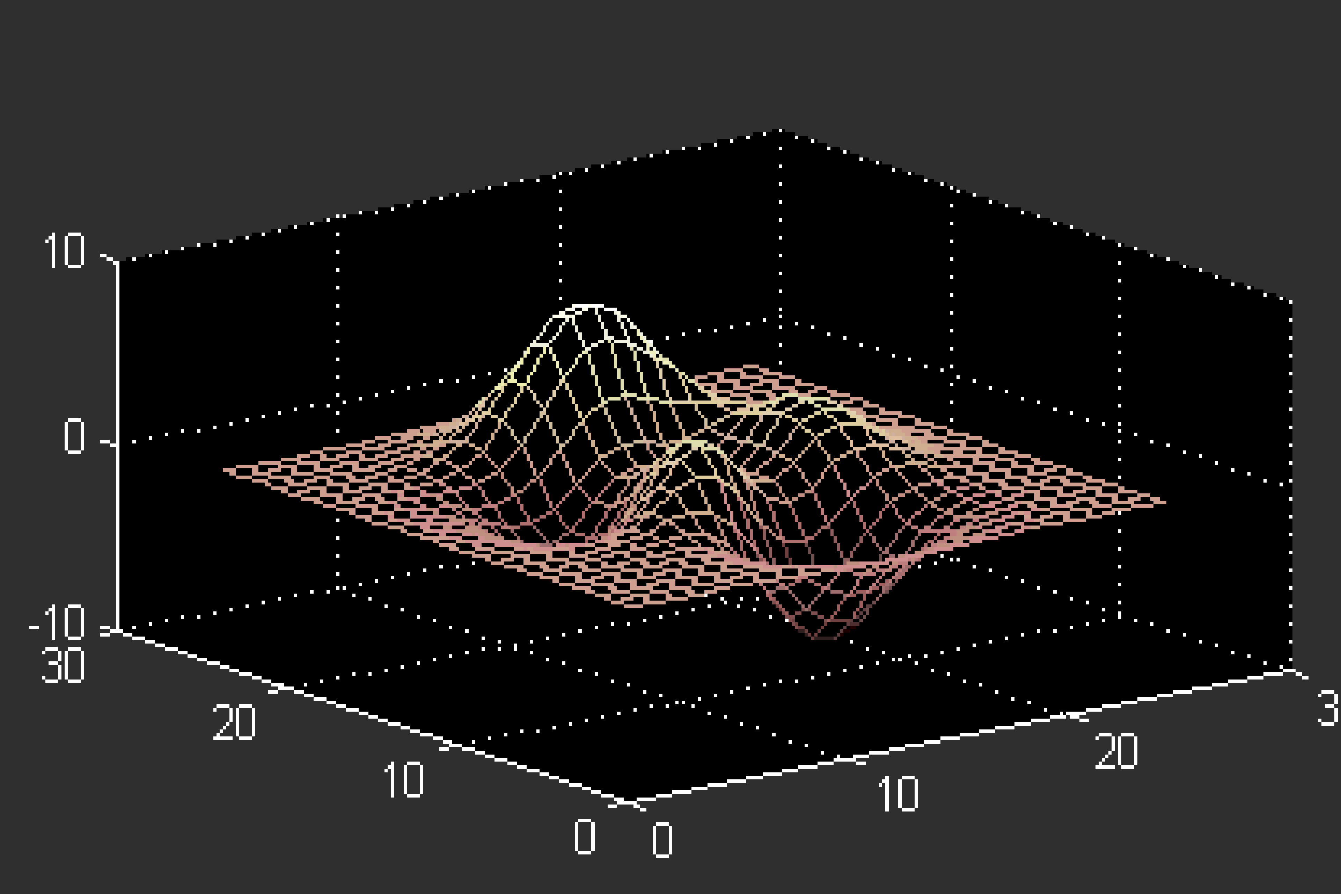
Napapan Piyasatian

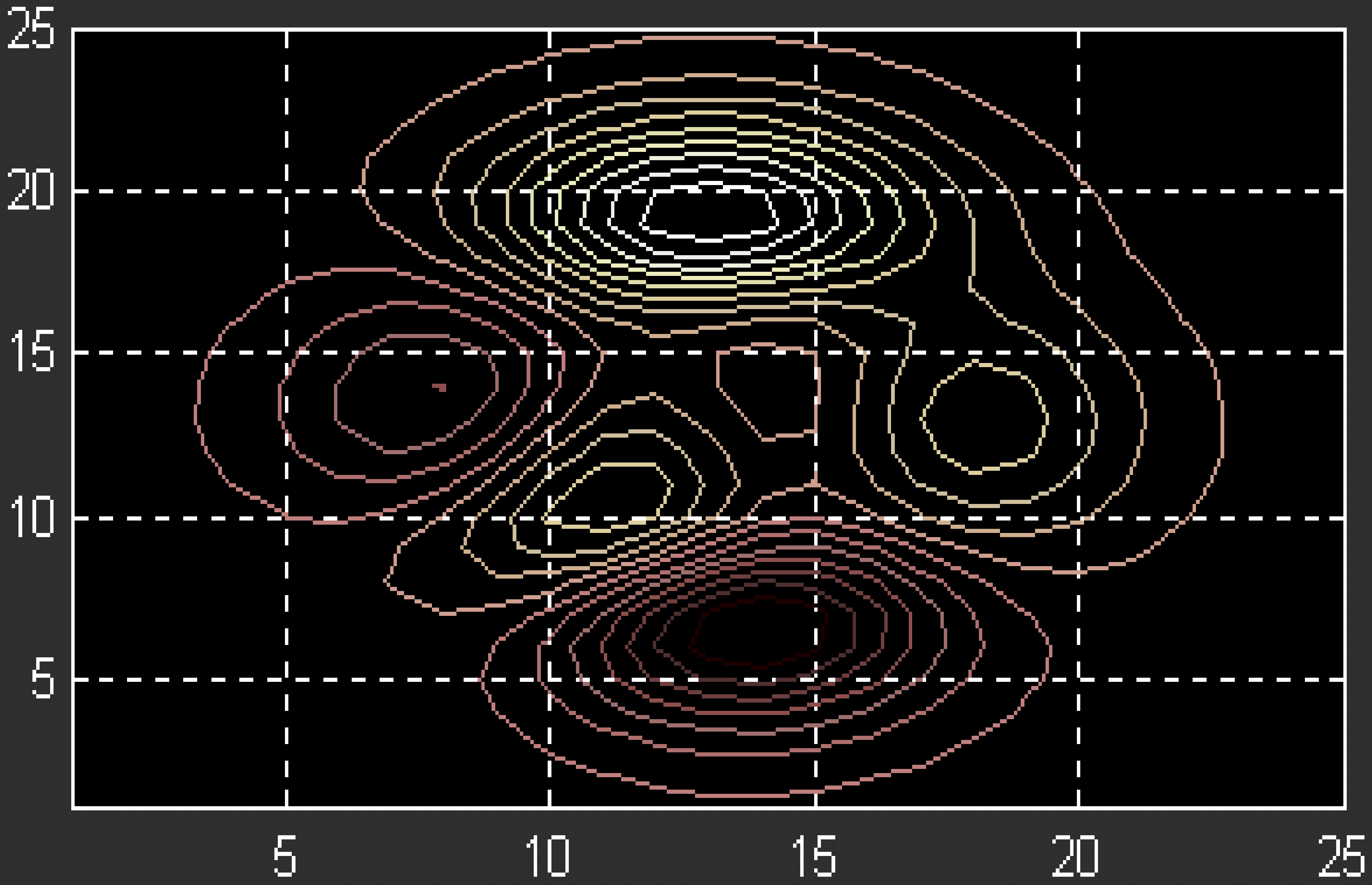
Numerical Optimization (1)

- Nonlinear objective function:
  - ☆ Many variables
  - ☆ Tortured, multidimensional topography (response surface) with many peaks and valleys
  - ☆ Example 1(a): $f(X) = X_1^2 + X_2^2 + X_3^2$
    $f(\underline{X}min) = 0$, where $\underline{X}min = \{0, 0, 0\}$

Numerical Optimization (2)

- Optimization multi-modal functions:
  - ☆ Nonrandom or deterministic search algorithms
  - ☆ Random or stochastic algorithms (more suitable)


10
0
-10
30
20
10
0
0
10
20
3


25
20
15
10
5
5
10
15
20
25

Genetic Algorithms

- Fitness or cost
- Initialization of a population of candidate solutions
- Mutation
- Recombination or crossover
- Selection

Fitness or Cost

- The value of a “Objective function” at a point
- To max. a function: the more fitness, the more optimal solutions

Initialization of a Population of Candidate Solutions

- Each solution = vector x
- Often these solutions are coded in binary
- Degree of precision determines the length of binary
- ES: floating-point number as genes
  - ☆ More suitable in continuous space

Mutation (1)

- Small random alterations to one or more parameters of an existing population

Mutation Point

Parent: 1 1 1 1 1 1 1 1

X

Offspring: 1 1 1 1 1 0 1 1

Mutation (2)

- Disparities between adjacent binary numbers when conducting the incremental search

16(10000) into 15(01111)

Mutation (3)

- Adding operation
  - ☆ The question is how much to add not which bits to flip
  - ☆ DE: must ensure that the mutation increment automatically scaled to the correct magnitude

Recombination or Crossover (1)

Crossover Point

Parent1:	1	1	1	1	1	1	1	1
Parent2:	0	0	0	0	0	0	0	0
Offspring1:	1	1	1	1	1	0	0	0
Offspring2:	0	0	0	0	0	1	1	1

Recombination or Crossover (2)

- Uniform crossover:
  - Inherits parameter values from parents with equal probability
- Non-uniform crossover
  - Takes parameters from one parent more often than the other

Selection

- Determine which among them will survive to the next generation
  - ☆ Random approach using "tournament selection"
    = randomly paired the winner with all possible competition.
  - ☆ DE: each child pits against one of its parents

Basic Mechanisms of DE (1)

- Initialization
  - ☆ Parameter limits should be defined
  - ☆ If not, parameter ranges should cover the suspected optimum

Basic Mechanisms of DE (2)

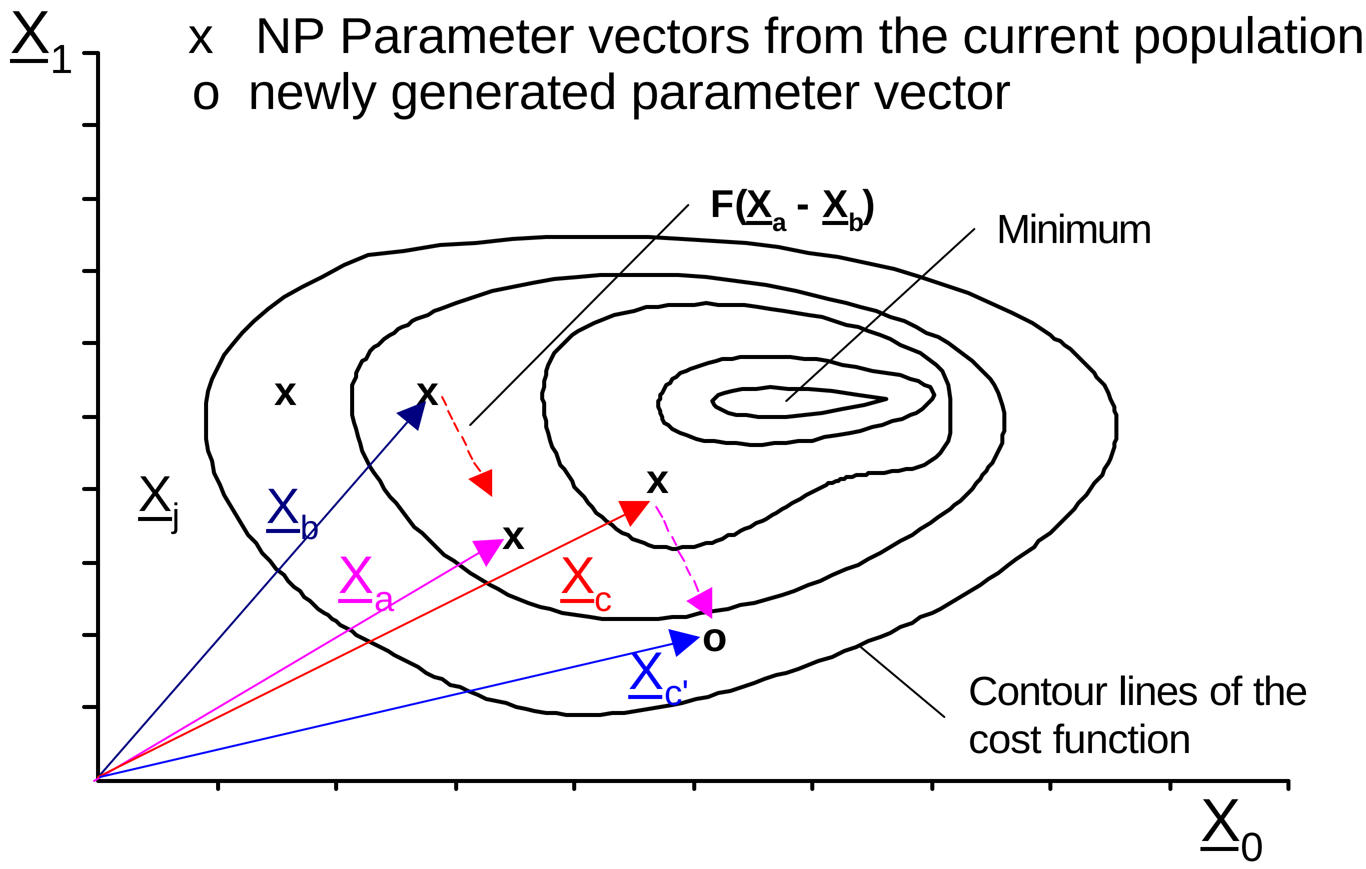
- Two arrays which represent current and the next generation
  - *NP* or the number of solutions each generation
  - Real valued vectors of parameters
  - Fitness or Cost of each vector of parameters

Basic Mechanisms of DE (3)

- Making challenger by mutation and recombination
- Mutating with vector differentials to make noisy random vector

$$\underline{X}_{c'} = \underline{X}_c + F(\underline{X}_a - \underline{X}_b)$$

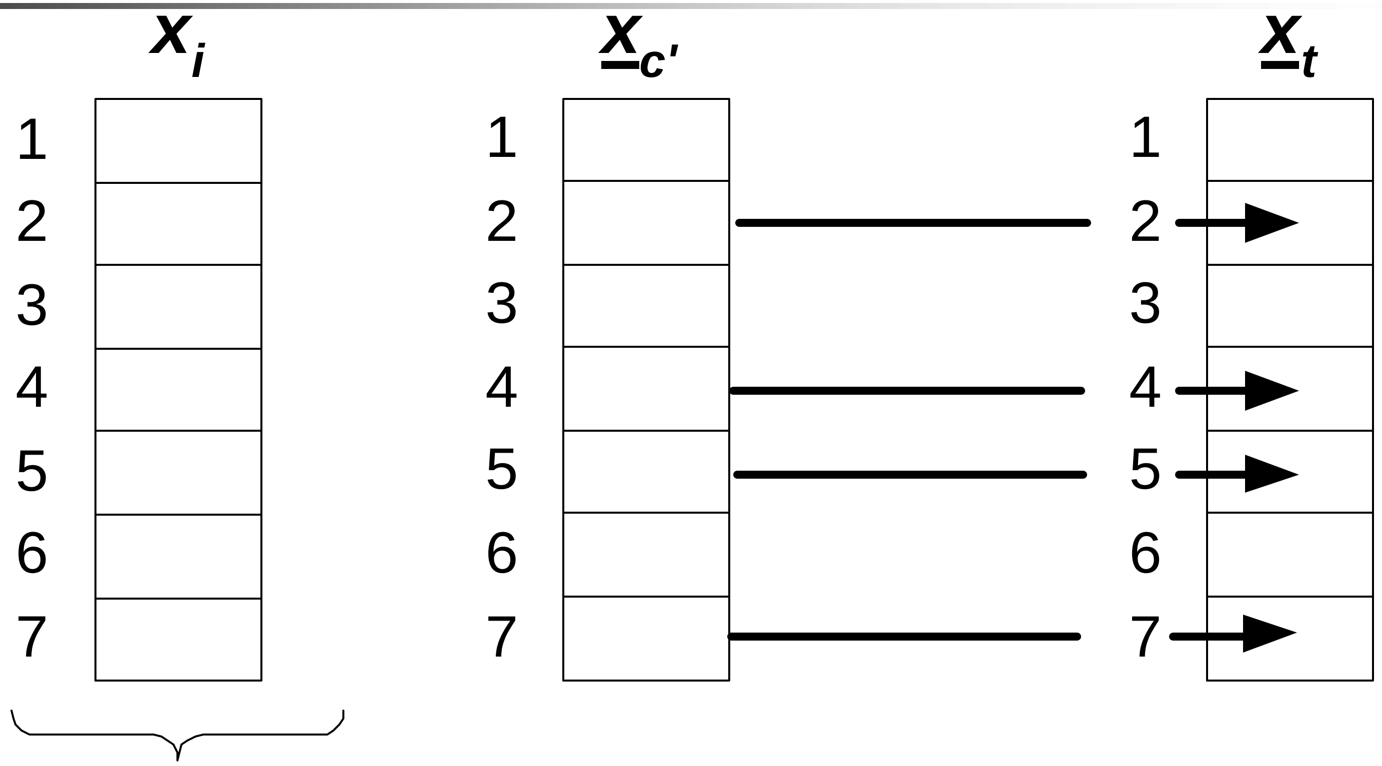
Mutation Scheme



Basic Mechanisms of DE (4)

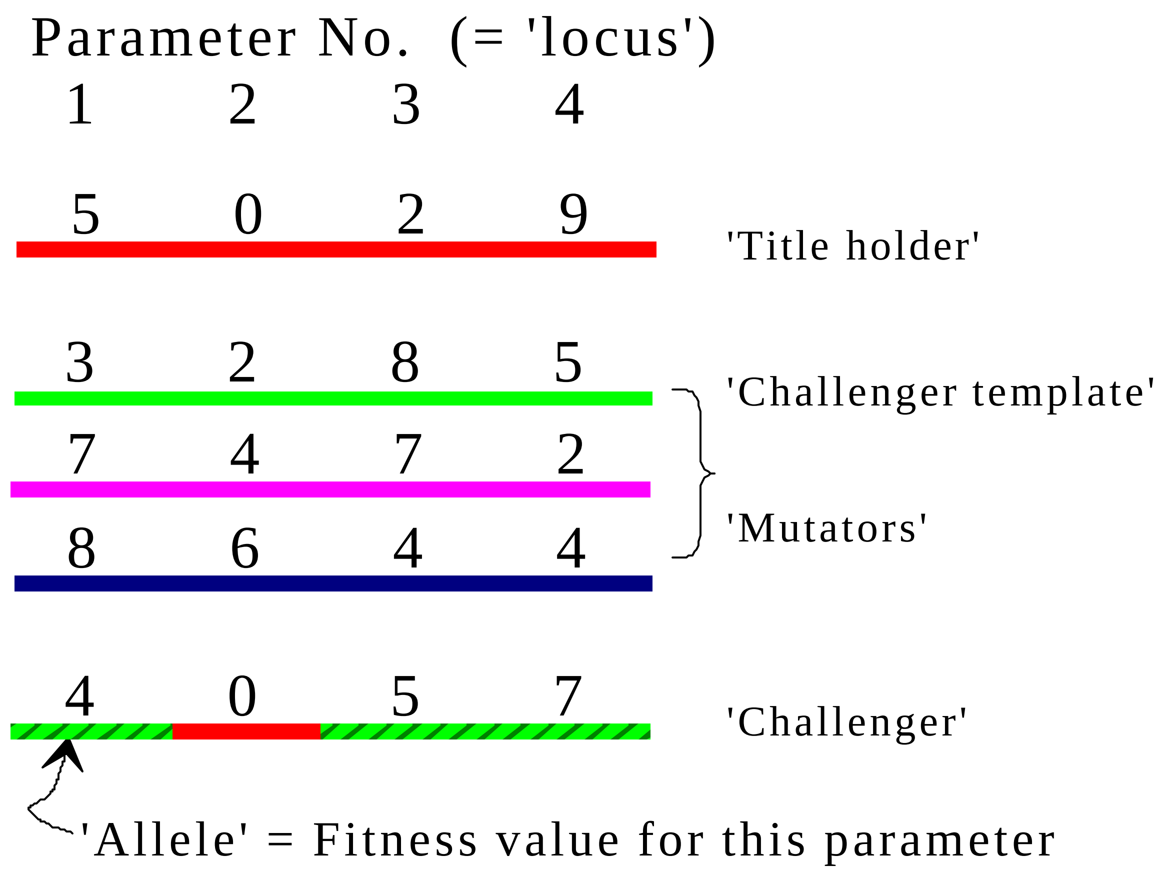
- Recombination or Crossover
  - Using "crossover constant", $CR \in [0,1)$
  - Must ensure that the "challenger" differs from the current population

Crossover Process



Parameter vector containing
D=7 parameters

Differential Evolution (DE)



A Nasty Little Function

- Non-differentiable at some points
- Many local optima that partially surrounded by very flat surface
  - ☆ Cause the solution stray outside the design limits

Practical Advice

- *NP* = 5 or 10 times of number of parameter in a vector
- If solutions get stuck:
  - ☆ F = 0.5 and then increase F or *NP*
  - ☆ $F \in [0.4, 1]$ is very effective range
- *CR* = 0.9 or 1 for a quick solution

Conclusion (1)

- Advantages
  - ☆ Immediately accessible for practical applications
  - ☆ Simple structure
  - ☆ Ease of use
  - ☆ Speed to get the solutions
  - ☆ Robustness

Conclusion (2)

- Need
  - ☆ A way to quantify the quality of the potential solutions “Fitness function” or “Objective function”