Intelligent Systems Engineering

EC6 Evolutionary Algorithms

Differential Evolution

Petr Musilek

University of Alberta

Outline

These notes are based on [Engelbrecht 2007] chapter 13; example on slides 13-31 is from slides accompanying [Weise 2011]; additional examples can be found in [Keller 2016] chapter 13.2

Introduction

- Simple Evolutionary Algorithm for numerical optimization
- Search space: vectors of real numbers, i.e., $\mathbb{G} = \mathbb{R}^n$
- Developed by Storn and Price in the mid-1990s
- Many different variants, but main idea: ternary recombination operator instead of mutation and binary crossover
- Population treatment:
  1. each offspring competes with its direct parent and
  2. replaces it if and only if it has better objective values
  3. i.e. a local form of selection!
- The rest is the same as in simple population EA

Basic Differential Evolution (DE)

- DE is a stochastic, population-based search strategy.
- Differences compared to other evolutionary algorithms:
  1. The distance and direction information from the current population is used to guide the search process.
  2. Mutation is applied first to generate a trial vector (which is then used within the crossover operator to produce one offspring).
  3. Mutation step sizes are not sampled from a prior known probability distribution function.
  4. Mutation step sizes are influenced by differences between individuals of the current population.
- Originally developed to be applied to continuous-valued landscapes, but can also be applied to discrete-valued and binary-valued ones.

Difference Vectors

- The positions of individuals provide valuable information about the fitness landscape.
- A good uniform random initialization method is used to construct the initial population; such initialization provides a good representation or the entire $\mathbb{G}$, with relatively large distances between individuals.
- Distances between individuals become smaller as the search progresses.
- The magnitude of the initial distances between individuals is influenced by the size of the population.
- Distances between individuals are a very good indication of the diversity of the current population.

Mutation:

1. Calculate one or more difference vectors.
2. Use these difference vectors to determine the magnitude and direction of step sizes.

Difference Vectors (cont.)

Advantages of using difference vectors:

- Information about the fitness landscape is used to direct the search.
- Mutation step sizes approaches a Gaussian (normal) distribution.

From using the Normal distribution:

- The population is sufficiently large to allow for a good number of difference vectors.
- The mean formed by the difference vectors are always zero: individuals used to calculate difference vectors are selected uniformly from the population; the population will not suffer from genetic drift.
- The deviation is determined by the magnitude of the difference vectors.

Total number of directions that can be explored per generation:

$$\binom{n_s}{2n_v} 2n_v! \approx \mathcal{O}(n_s^{2n_v})$$

(n_s is the population size, and n_v is the number of differentials used)

Self-Adaptation through Self-organization

In general, there are four methods of adaptation

1. Changing parameters over time independently from the search process (e.g. simulated annealing)
2. Changing parameters according to some policy which uses information about the progress of the search (e.g. evolution strategy with 1/5th rule, exogeneous method)
3. Encoding parameters as additional variables in individual records and let the evolutionary algorithm adapt them via selection and reproduction (e.g. evolution strategy with endogeneous method)
4. Self-adaptation via self-organization without parameters!

2-4 are self-adaptation methods, that allow for an algorithm behavior that is not anticipated by the designer; differential evolution uses approach 4.

DE Mutation

- Differential Evolution uses a ternary mutation operation

$$g_i = \texttt{mutationDE}(g_{i_1}, g_{i_2}, g_{i_3}) = g_{i_1} + F \cdot (g_{i_2} - g_{i_3})$$

- It produces a trial vector g_i for each individual of the current population by mutating a target vector with a weighted differential.
- At the beginning, all candidate solutions are uniformly spread in $\mathbb{G}$ and the differential information $(g_{i_2} - g_{i_3})$ is large $\Rightarrow$ large search steps
- As the population converges, the candidate solutions are located closer to each other and the distances $(g_{i_2} - g_{i_3})$ decrease $\Rightarrow$ the search steps get smaller, too
- Very simple way to perform self-adaptation without needing any additional parameter or operation!
- The step-width is self-organizing from the structure of the population and self-adapting along the optimization process

DE Mutation

DE Mutation Operator

Produces a trial vector g_i for each individual of the current population by mutating a target vector with a weighted differential.

- Generation of the trial vector
  1. Select a target vector g_{i_1} from the population, such that $i \neq i_1$.
  2. Randomly select two individuals g_{i_2} and g_{i_3} from the population such that $i \neq i_1 \neq i_2 \neq i_3$ and $i_2, i_3 \backsim \boldsymbol{U}(1, n_s)$.
- The trial vector is calculated by perturbing the target vector

$$g_i = g_{i_1} + F \cdot (g_{i_2} - g_{i_3})$$

where F is the scale factor, controlling the amplification of the differential variation. Values of F are seldom grater than 1.0, i.e. $F \in (0, 1+)$.

DE Crossover

DE Crossover Operator

Implements a discrete recombination of the trial vector, g_i, and the parent vector, g_{i_1}, to produce offspring, g'_i

$$g'_{ij} = \begin{cases} g_{ij} & \text{if } j \in \mathcal{J} \\ g_{i_1 j} & \text{otherwise} \end{cases}$$

where g_{i_j} refers to the j-th element of the vector g_i, and $\mathcal{J}$ is the set of element indices to undergo perturbation.

Methods used to determine the set $\mathcal{J}$:

- Binomial
- Exponential

DE Crossover (cont.)

- DE crossover is implemented as follows
  - Genes found on the list $\mathcal{J}$ are replaced by genes from the (mutated) trial vector g_i
  - While the remaining genes are directly copied from the parent individual g_{i_3}

- In **binomial crossover**, the list of crossover points is formed by randomly drawing, with probability p_r, from the set of all possible crossover points, $\{1, 2, \cdots, n_x\}$, where n_x is the problem dimension.
- In **exponential crossover**, the list contains a sequence of adjacent crossover points, treating the list of potential crossover points as a circular array. The length of the consecutive sequence of genes is based on exponential distribution according to p_r.

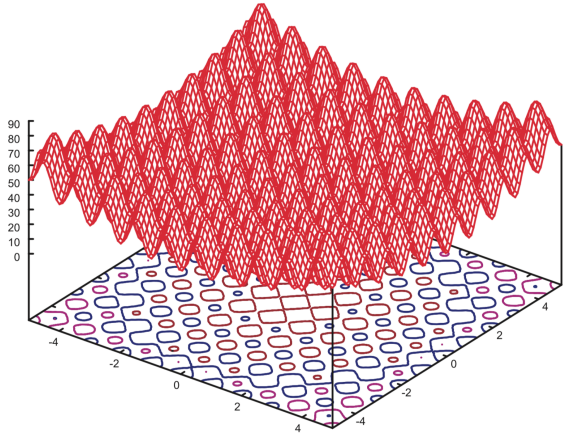
Note: Weise (2011) merges mutation and crossover into a single operator of recombination. However, here, we use separate definitions as in the original description of DE in book by Price, Storn and Lampinen (2005).

Basic DE Algorithm

1: Set the generation counter, $t = 0$ Initialize the control parameters, F and p_r
2: Create and initialize the population, $\mathcal{C}(0)$, of n_s individuals
3: **while** a termination criterion is NOT met **do**
4: **for** each individual, $g_{i_1}(t) \in \mathcal{C}(t)$ **do**
5: Evaluate the fitness, $f(g_{i_1}(t))$
6: Create the trial vector, $g_i(t)$ by applying the mutation operator
7: Create an offspring, $g'_i(t)$, by applying the crossover operator
8: **if** $fg'_i(t))$ is better than $f(g_i(t))$ **then**
9: add $g'_i(t)$ to $\mathcal{C}(t+1)$
10: **else**
11: add $g_{i_1}(t)$ to $\mathcal{C}(t+1)$
12: **end if**
13: **end for**
14: **end while**
15: Return the individual with the best fitness as the solution;

Example 1 (1/19)

Minimize $f(\vec{x}) = 20 + \sum_{i=1}^{2} \left(x_i^2 - 10\cos\left(2 * \Phi * x_i\right)\right)$ for $-5.12 \geq x \geq 5.12$



Example 1 (2/19)

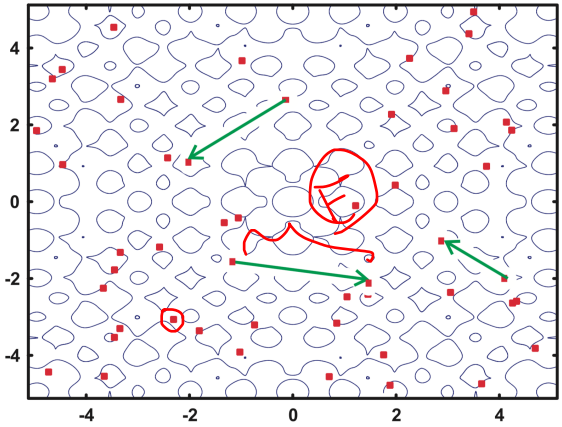
$$\text{Minimize } f(\vec{x}) = 20 + \sum_{i=1}^{2} \left(x_i^2 - 10\cos\left(2 * \Phi * x_i\right)\right) \text{ for } -5.12 \geq x \geq 5.12$$

Observations:

- Population is initially uniformly spread
- There is large, various differential information in the population
- The population then collapses to (local) optimum/optima
- As candidate solutions get closer to each other, differential information decreases
- Due to a number of different optima, differential information is still available at the advanced stages of convergence

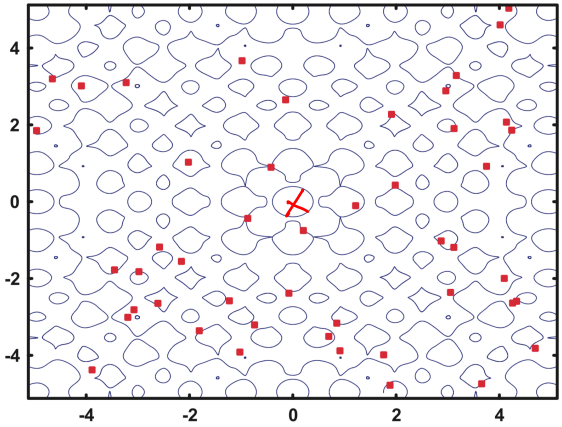
Example 1 (3/19)

Population after generation 0: uniform spread, large differences



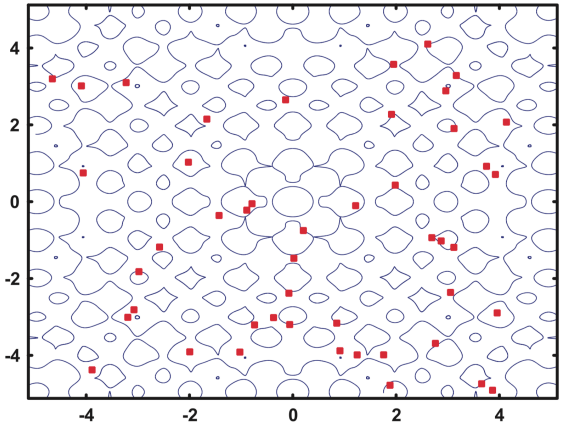
Example 1 (4/19)

Population after generation 1



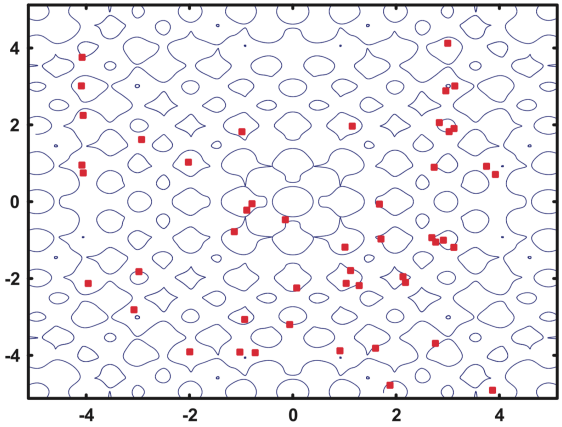
Example 1 (5/19)

Population after generation 2



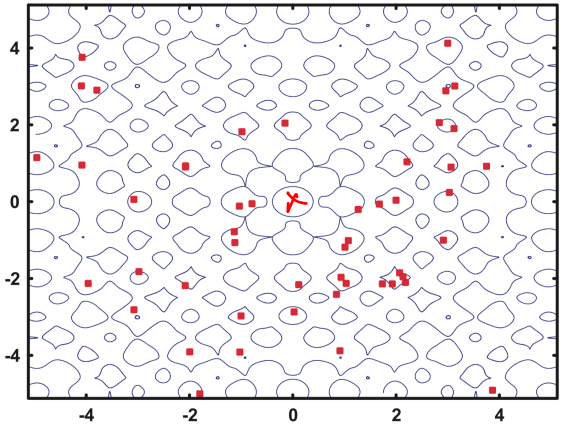
Example 1 (6/19)

Population after generation 5



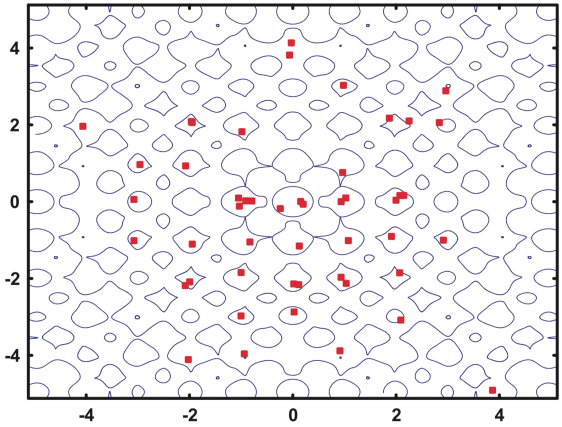
Example 1 (7/19)

Population after generation 10



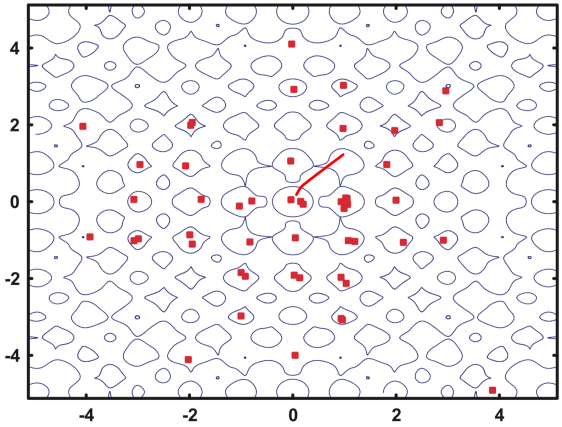
Example 1 (8/19)

Population after generation 20: concentration around center



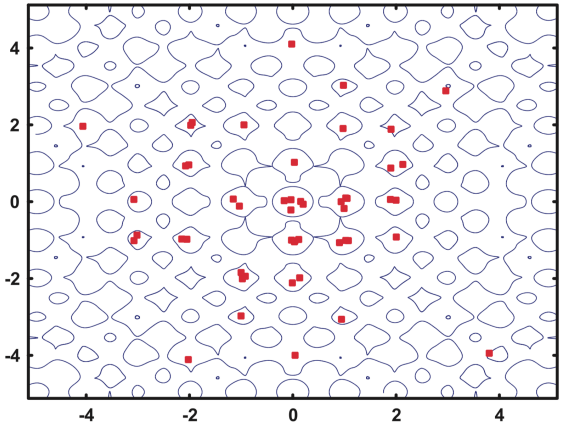
Example 1 (9/19)

Population after generation 30



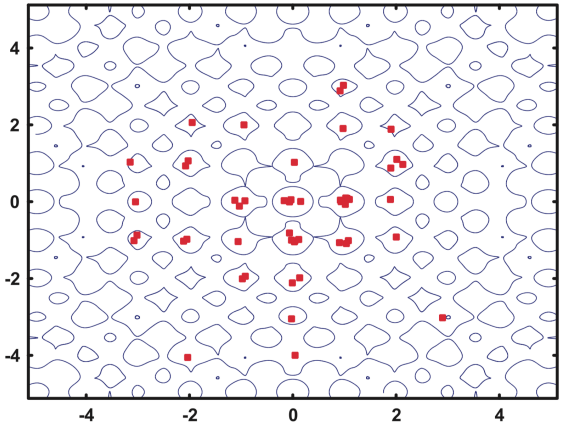
Example 1 (10/19)

Population after generation 40



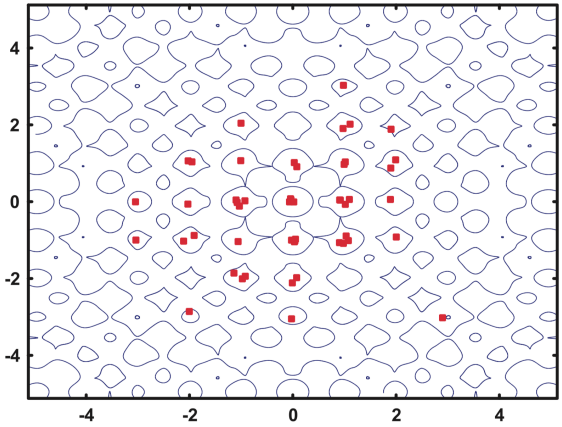
Example 1 (11/19)

Population after generation 50



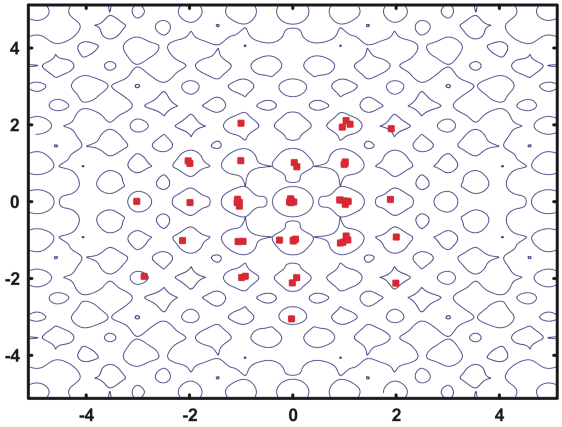
Example 1 (12/19)

Population after generation 75



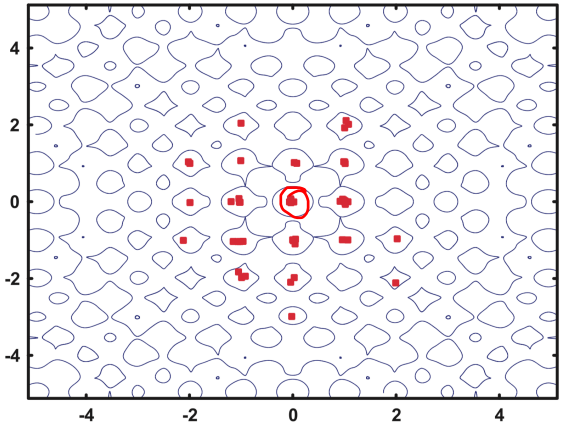
Example 1 (13/19)

Population after generation 100



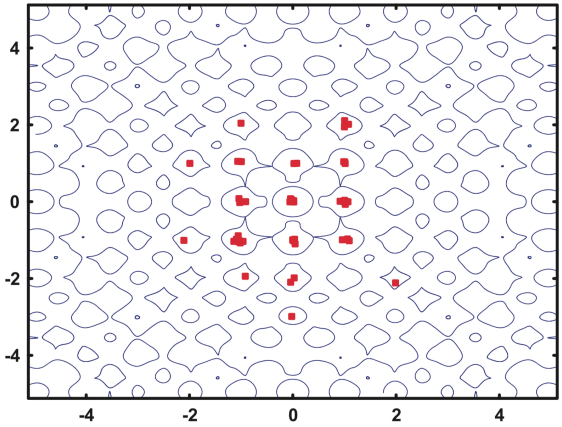
Example 1 (14/19)

Population after generation 125: similar individuals in local optima



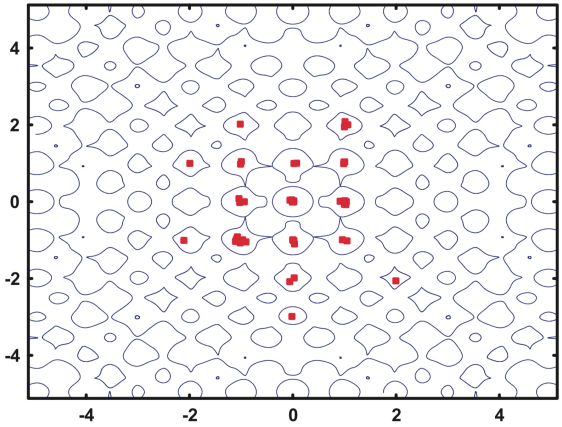
Example 1 (15/19)

Population after generation 150



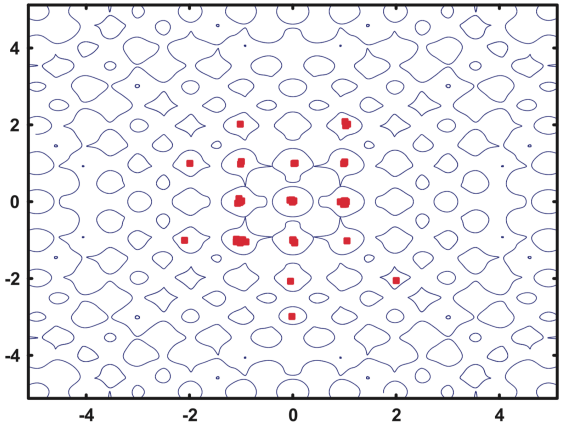
Example 1 (16/19)

Population after generation 175



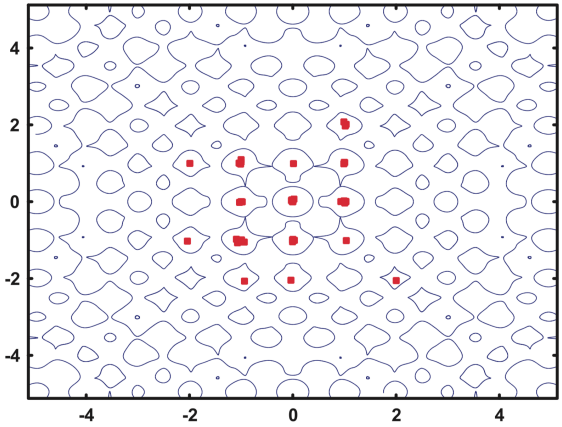
Example 1 (17/19)

Population after generation 200



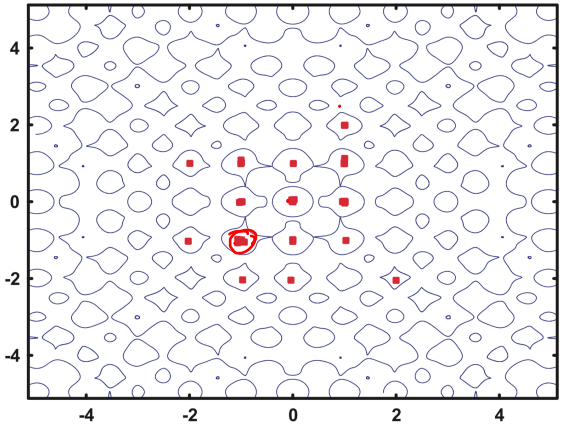
Example 1 (18/19)

Population after generation 250



Example 1 (19/19)

Population after generation 300: concentration in optima



Example 2

- Numerical example for

$$\texttt{mutateDE}(g_1, g_2, g_3) = g_1 + F(g_2 - g_3)$$

$$g_1 = (0.25, 0.4, 0.6)^T$$

$$g_2 = (0.3, 0.45, 0.7)^T$$

$$g_3 = (0.2, 0.5, 0.7)^T$$

$F = 0.3 \leftarrow$ strengh parameter

$$g' = \begin{pmatrix} 0.25 & + & 0.3 \cdot (0.3 - 0.2) \\ 0.4 & + & 0.3 \cdot (0.45 - 0.5) \\ 0.6 & + & 0.3 \cdot (0.7 - 0.7) \end{pmatrix} = \begin{pmatrix} 0.28 \\ 0.385 \\ 0.6 \end{pmatrix}$$

Control Parameters

Population size has a direct influence on the exploration ability of DE algorithms. The larger the population, the more differentials are available, and more directions explored.

Scaling factor $F \in (0, 1+)$ controls the amplification of the differential variations $(g_{i_2} - g_{i_3})$.

- The smaller the value of F, the smaller the mutation step sizes, and the longer it will take for the algorithm to converge.
- Larger values for F facilitate exploration, but may cause the algorithm to overshoot good optima and possibly result in premature convergence.

$F = 0.5$ generally provides good performance.

Recombination probability p_r has a direct influence on the diversity of DE; it controls the number of elements of the parent, $g_i(t)$, that will change. Increasing p_r often results in faster convergence, while decreasing it increases search robustness.

Variations DE/x/y/z

- DE strategies differ in
  - x - the way that the target vector is selected
  - y - the number of difference vectors used
  - z - the way that crossover points are determined
- DE/rand/1/bin refers to binomial crossover and DE/rand/1/exp refers to exponential crossover.
- DE/best/1/z - the target vector is selected as the best individual, $\hat{g}(t)$, from the current population; any of the crossover methods can be used; the trial vector is calculated as follows

$$g_i(t) = \hat{g}(t) + F \cdot \left(g_{i_2}(t) - g_{i_3}(t)\right)$$

- DE/x/n_v/z - more than one difference vector is used; the trial vector is calculated as follows

$$g_i(t) = g_{i1}(t) + F \cdot \textstyle\sum_{k=1}^{n_V} \left(g_{i_2,k}(t) - g_{i_3,k}(t)\right)$$

Variations DE/x/y/z (cont.)

- DE/rand-to-best/n_v/z combines the *rand* and *best* strategies to calculate the trial vector as follows

$$g_i(t) = \gamma\hat{g}(t) + (1-\gamma)g_{i_1}(t) + F \cdot \sum_{k=1}^{n_V}\left(g_{i_2,k}(t) - g_{i_3,k}(t)\right)$$

where $g_{i_1}(t)$ is randomly selected and $\gamma \in [0,1]$ controls the greediness of the mutation operator.

- DE/current-to-best/$1 + n_v$/z - the parent is mutated using at least two difference vectors; one difference vector is calculated from the best vector and the parent vector, while the rest of the difference vectors are calculated using randomly selected vectors

$$g_i(t) = g_{i_1}(t) + F \cdot (\hat{g}(t) - g_{i_1}(t)) + F \cdot \sum_{k=1}^{n_V}(g_{i_2,k}(t) - g_{i_3,k}(t))$$

- DE/rand/1/bin maintains good diversity, DE/current-to-best/2/bin shows good convergence characteristics.
- Dynamically switching between these two strategies

Applications

- DE has mostly been applied to optimize functions defined over continuous-valued landscapes.
- For the initial population, each individual is initialized using:

$$x_{ij} \backsim \boldsymbol{U}(x_{\min,j}, x_{\max,j})$$

- The fitness function is simply the function to be optimized.
- DE has also been applied to train neural networks (NN). In this case, an individual represents a complete NN.

Summary

- DE is an EA for numerical optimization
- Simple self-adaption without any parameter by ternary recombination
  - Ternary mutation forms *trial* vector from a parent and two differential vectors
  - The trial vector is the recombined with the original parent based on a set of randomly generated indices
- Parents compete with their children in the population
- A number of variants under the DE nomenclature