Differential Evolution

Prof. G. Panda, **FNAE, FNASc.**
Indian Institute of Technology, Bhubaneswar, India
Email : ganapati.panda@gmail.com / gpanda@iitbbs.ac.in
www.iitbbs.ac.in

Presentation flow

- EA and its requirement
- DE - an introduction
- Parameters of DE
- Initialization
- Mutation operation
- Crossover operation (trial vector generation)
- Types of crossover
- DE selection
- An example
- Control parameters
- Selection scheme
- Schemes of DE
- Normalization
- Previous work on parameter selection and automatic tuning
- Examples of unconstrained optimization
- Conclusion

Evolutionary Algorithm (EA)

Inspired by the natural evolution of species successfully applied to solve optimization problems in diverse fields.

EA requires

- **Appropriate encoding schemes and evolutionary operators**

- **Suitable parameter settings to ensure best possible performance – leads to more computation due to trial error**

- **To overcome this difficulty adaptation of parameters and operators has been suggested**

Differential Evolution (DE)

- Proposed by Storn and Price – 1995, is a simple, powerful population based stochastic search technique.
- It is an efficient and effective global optimizer in the continuous search domain.
- It has been successfully applied to diverse fields – mech. Engg., communication, pattern recognition

Crucial parameters in DE

Population size – NP

Scaling factor – F

Crossover rate – CR

These parameters significantly influence the optimization performance of DE.

Introduction

- **DE uses parameter vectors as individuals in a population**
- **The key element distinguishing DE from other population-based techniques is differential mutation operator and trial parameter vectors.**

DE Initialization

- **Parameter vectors in a population are randomly initialized and evaluated using the fitness function**
- **The initial NP D-dimensional parameter vectors is:**

  $x_{ji,G}$ **where *i* = 1, 2, ..., NP and j = 1, 2, ..., D**
- **NP is number of population vectors**
- **D is dimension**
- **G is generation**

DE Mutation Operator

- **DE generates new parameter vectors by adding the weighted difference between two parameter vectors to a third vector**
- **For each target vector $x_{i,G}$, i = 1,2,…,NP, a mutant vector is generated according to:**

$$\underline{\mathbf{v}_{i,G+1}} = x_{r1,G} + F\,(x_{r2,G} - x_{r3,G})$$

where $\underline{\mathbf{v}_{i,G+1}}$ is a mutant vector; $r1,r2,r3 \in \{1, 2, \ldots, NP\}$, random integer mutually different and different from index i then NP≥4; F is a real and constant factor $\in [0, 1]$ which controls the amplification of the differential variation

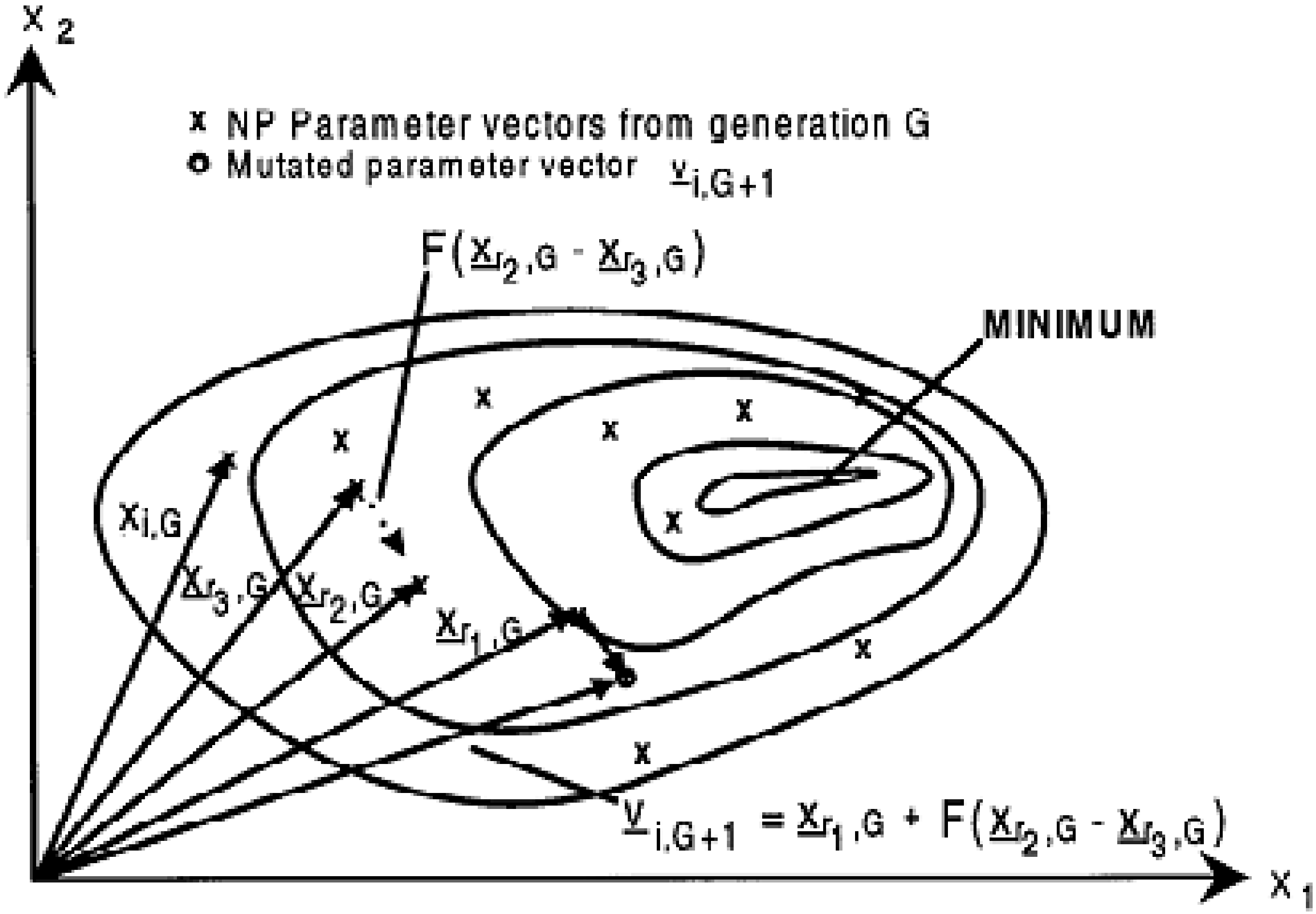


Fig. An example of a two-dimensional cost function showing its contour lines and the process for generating $\underline{v}_{i,G+1}$ by different vectors.

DE Crossover Operator

- **The parameters of mutated vector are then mixed with the parameters of another predetermined vector, <u>the target vector</u>, to yield the so-called <u>trial vector</u>**
- **Choosing a subgroup of parameters, *j* (or a set of crossover points) for mutation is similar to a process known as crossover in genetic algorithms or evolution strategies**

- **Crossover is introduced to increase the diversity of the perturbed parameter vectors**
- **The trial vector:**

$$\underline{u}_{i,G+1} = (\underline{u}_{1i,G+1}, \underline{u}_{2i,G+1}, \ldots, \underline{u}_{Di,G+1})$$

is formed by

$$\underline{u}_{ji,G+1} = \begin{cases} \underline{v}_{ji,G+1} & \text{if } (randb(j) \le CR) \text{ or } j = rnbr(i) \\ x_{ji,G} & \text{if } (randb(j) > CR) \text{ and } j \ne rnbr(i) \end{cases},$$

$$j = 1, 2, \ldots, D.$$

- **where *randb(j)* is the *j*th evaluation of a uniform random number generator with outcome [0, 1], *CR* is the crossover constant [0, 1] which has to be determined by the user, *rnbr(i)* is randomly chosen index from 1..D which ensures that $\underline{u}_{i,G+1}$ gets at least one parameter from $\underline{v}_{i,G+1}$**

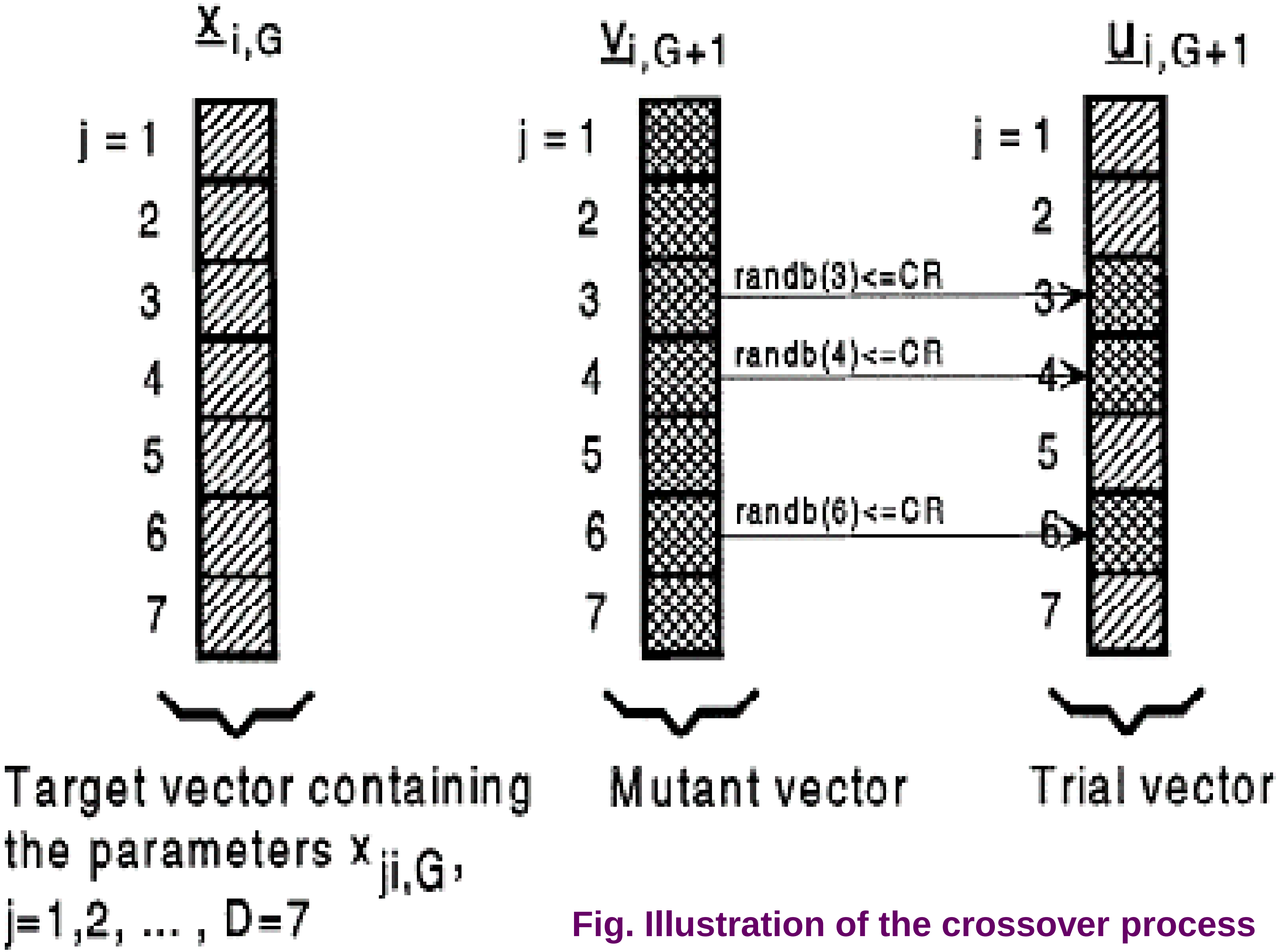


Fig. Illustration of the crossover process for D = 7 parameters.

Binomial Crossover

Copies the jth parameter of mutant vector

If $rand_j \; [0,1] <= CR$ or $j=j_{rand}$

Otherwise it is copied from the corresponding target vector, $X_{i,G}$

Exponential crossover operator

Parameters of $V_{i,G}$ are inherited from a randomly chosen parameter index till the first time $rand_j(0,1) > CR$.

The remaining parameters are copied from $X_{i,G}$.

DE Selection

- **If the trial vector yields a lower cost function value than the target vector, the trial vector replaces the target vector in the following generation**
- **In other words, the offspring replaces the parent if it is fitter. Otherwise, the parent survives and is passed on to the next iteration of the algorithm**

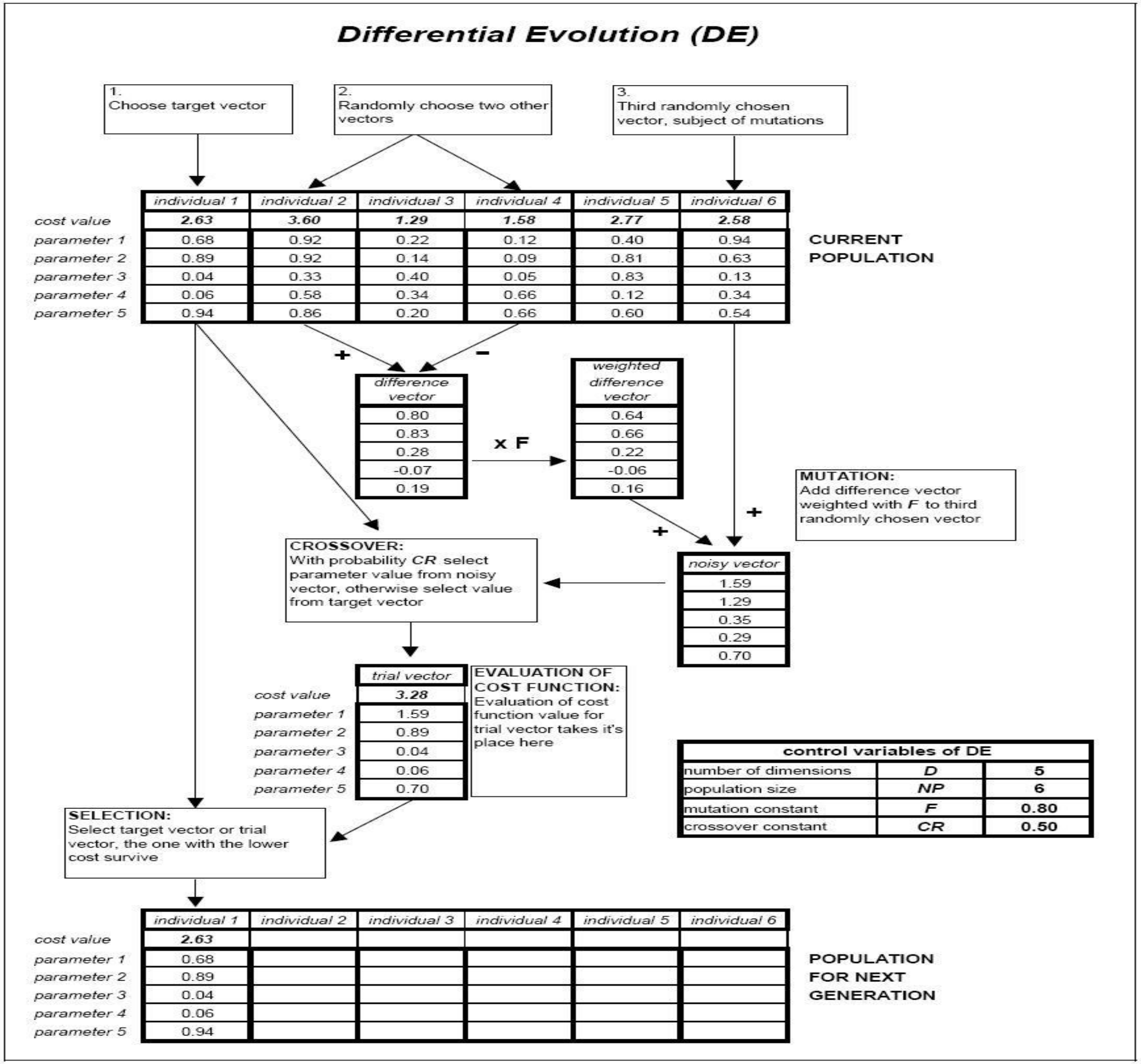


Figure 1. Differential Evolution works directly with the floating-point valued variables of the objective function, not with their (binary) encoding. The functioning of DE is here illustrated in case of a simple objective function $f(X) = x_1 + x_2 + x_3 + x_4 + x_5$ (variables bounded within the range [0,1]).

Control parameters

- ***F*** **and *CR* are DE control parameters**
- ***F*** **is a real-valued factor in the range [0.0,1.0]**
- ***CR*** **is a real-valued crossover factor in range [0.0,1.0]**
- **CR controls the probability that a trial vector parameter will come from the randomly chosen noise vector**

Control parameters

- **Optimal values are dependent both on objective function characteristics and on the population size, *NP***
- **Practical advice on how to select control parameters *NP*, *F* and *CR are reported* in the literature**

Selection scheme

- **DE's selection scheme also differs from other evolutionary algorithms**
- **Current population is *G* --population for the next generation, *G+1*, is selected from the child population based on objective function value**

$$X_{i,G+1} = \begin{cases} U_{i,G+1} & \text{if} \quad f(U_{i,G+1}) \le f(X_{i,G}) \\ X_{i,G} & \text{otherwise} \end{cases}$$

Selection scheme

- **Each individual of the temporary population is compared with its counterpart in the current population**
- **Trial vector is only compared to one individual, not to all the individuals in the current population.**

Schemes of DE

- **Several different schemes of DE exist**
- **Differences in target vector selection and difference vector creation**

DE/rand/1: $\mathbf{V}_{i,G} = \mathbf{X}_{r_1,G} + F \cdot \left(\mathbf{X}_{r_2,G} - \mathbf{X}_{r_3,G}\right)$

DE/best/1: $\mathbf{V}_{i,G} = \mathbf{X}_{best,G} + F \cdot \left(\mathbf{X}_{r_1,G} - \mathbf{X}_{r_2,G}\right)$

DE/current-to-best/1: $\mathbf{V}_{i,G} = \mathbf{X}_{i,G} + F \cdot \left(\mathbf{X}_{best,G} - \mathbf{X}_{i,G}\right) + F \cdot \left(\mathbf{X}_{r_1,G} - \mathbf{X}_{r_2,G}\right)$

DE/rand/2: $\mathbf{V}_{i,G} = \mathbf{X}_{r_1,G} + F \cdot \left(\mathbf{X}_{r_2,G} - \mathbf{X}_{r_3,G} + \mathbf{X}_{r_4,G} - \mathbf{X}_{r_5,G}\right)$

DE/best/2: $\mathbf{V}_{i,G} = \mathbf{X}_{best,G} + F \cdot \left(\mathbf{X}_{r_1,G} - \mathbf{X}_{r_2,G} + \mathbf{X}_{r_3,G} - \mathbf{X}_{r_4,G}\right)$

Normalization

DE aims to change the candidate solutions towards global solution.

The initial values should cover the entire search space constrained by prescribed minimum and maximum parameters bounds

$$\underline{X}_{\min} = \left\{ x_{\min}^{1} \ldots\ldots\ldots\ldots\ldots\ldots x_{\min}^{D} \right\}$$

$$and$$

$$\underline{X}_{\max} = \left\{ x_{\max}^{1} \ldots\ldots\ldots\ldots\ldots\ldots x_{\max}^{D} \right\}$$

<u>Example</u>:

Initial value of jth parameter in the ith individual at generation G=0 is generated by

$$x_{i,0}^{j} = x_{\min}^{j} + rand(0,1) \quad (x_{\max}^{j} - x_{\min}^{j});$$
$$j = 1,2,\ldots\ldots\ldots, D$$

<u>Previous work</u> (Parameter selection)

Performance of DE depends on the chosen

(i) Trial vector generation strategy

(ii) Parameter values used

Otherwise

May lead to premature convergence or stagnation [1]-[5]

Solution : Empirical guidelines for choosing trial vector generation strategies and control parameter settings.

(ii) Storn and Price [6] have suggested

- **NP should be between 5D and 10D**
- **And good initial choice of F is 0.5**
- **CR is chosen between 0.1 to 0.4**
- **For unimodal problems CR=0.9 is chosen to speed up convergence**
- **If the population converges prematurely, F or NP can be increased**

(iii) In [7] it is recommended to use NP=20D and F=0.8

(iv) In [3] it is recommended to use NP=[3D, 8D], F=0.6 and CR between [0.3, 0.9]

(v) In [8] F=[0.4, 0.95], F=0.9 is a good initial choice

CR = [0, 0.2], when the function is separable

CR = [0.9 1.0], when the function parameters are dependent

(vi) Conflicting conclusions have been drawn regarding manual choice of strategy and control parameters.

Work on automatic tuning

In [5] F is chosen

(a) linearly reduced between 0.5 and 1

(b) randomly between 0.5 and 1

In [9] F is chosen from a uniform distribution between 0.5 and 1.5 with mean 1.0

Work on automatic tuning

- In [10], [4] and [11], F and CR are chosen adaptively.
- In [10] Fuzzy adaptive differential evolution – tuning of parameters by fuzzy method
- In [4] Parameters are controlled based on population density.
- In [11] self-adapted CR and F from Gaussian distribution N(0,1) are chosen.
- Parallel implementation of DE has been suggested in [12].

Performance

The convergence speed and robustness depends on

- dimensionality,
- population size,
- mutation operator and
- the setting of parameters

Performance measure

i. Average number of function calls.
ii. Success rates (SR)
iii. Average SR
iv. Number of unsolved problems
v. Number of functions for which the algorithm outperforms its competitor
vi. Acceleration rate (AR)
vii. Average of best found solutions and corresponding standard deviation after specific number of function evaluations has been reached.

Most commonly used metric in literature

NFCS : No. of function calls

A smaller NFC – higher convergence Speed

The termination criterion is to find a value smaller than the value to reach (VTR) before reaching the maximum number of function calls, Max_{NFC}

In order to minimize the effect of stochastic nature of the algorithms on the metric, the reported NFCs for each functions is the average over 50 trials.

In order to compare the convergence speeds, the acceleration rate (AR) is defined as

$$AR = NFC_{DE} / NFC_{newmethod}$$

when AR>1 means new method is faster.

The number of times, for which the algorithm successfully reaches the VTR (value to reach) for each test function is measured as the success rate (SR)

$$SR = \frac{\text{number of times reached VTR}}{\text{total number of trials}}$$

The average acceleration rate (AR_{avg}) and the Average success rate (SR_{avg}) over n test functions are calculated as follows :

$$AR_{avg} = \frac{1}{n}\sum_{i=1}^{n} AR_i$$

$$SR_{avg} = \frac{1}{n}\sum_{i=1}^{n} SR_i$$

In order to maintain a reliable and fair comparison, for all experiments

(i) The parameter settings are the same (as mentioned earlier)

(ii) For all conducted experiments the reported values the average of the results for 50 independent runs

Compare

- In terms of convergence speed and robustness
- The effect of problem dimensionality
- The effect of population size
- With mutation strategies

Handling constraints

- **Boundary Constraints---ensure that parameter values lie inside their allowed ranges after reproduction.**
- **Constraint Functions--penalty function methods have been applied with DE for handling constraint functions**

Unconstrained Optimization

$$\text{minimize } f(\mathbf{x}), \quad \mathbf{x} = (x_1, x_2, \ldots, x_{\mathrm{D}})$$

$$\text{subject to} \qquad x_j \in dom(x_j)$$

$$\text{where } \mathbf{x} \in FS = S,$$

D is dimension,

$j = 1, \ldots, \mathrm{D}$,

$dom(x_j)$ is the domain of variable $x_j \in R / Z$,

FS is feasible space, S is search space,

R is real number, and Z is integer number.

Note: Minimization of objective function also known as cost function

DE Algorithm for Rosenbrock function

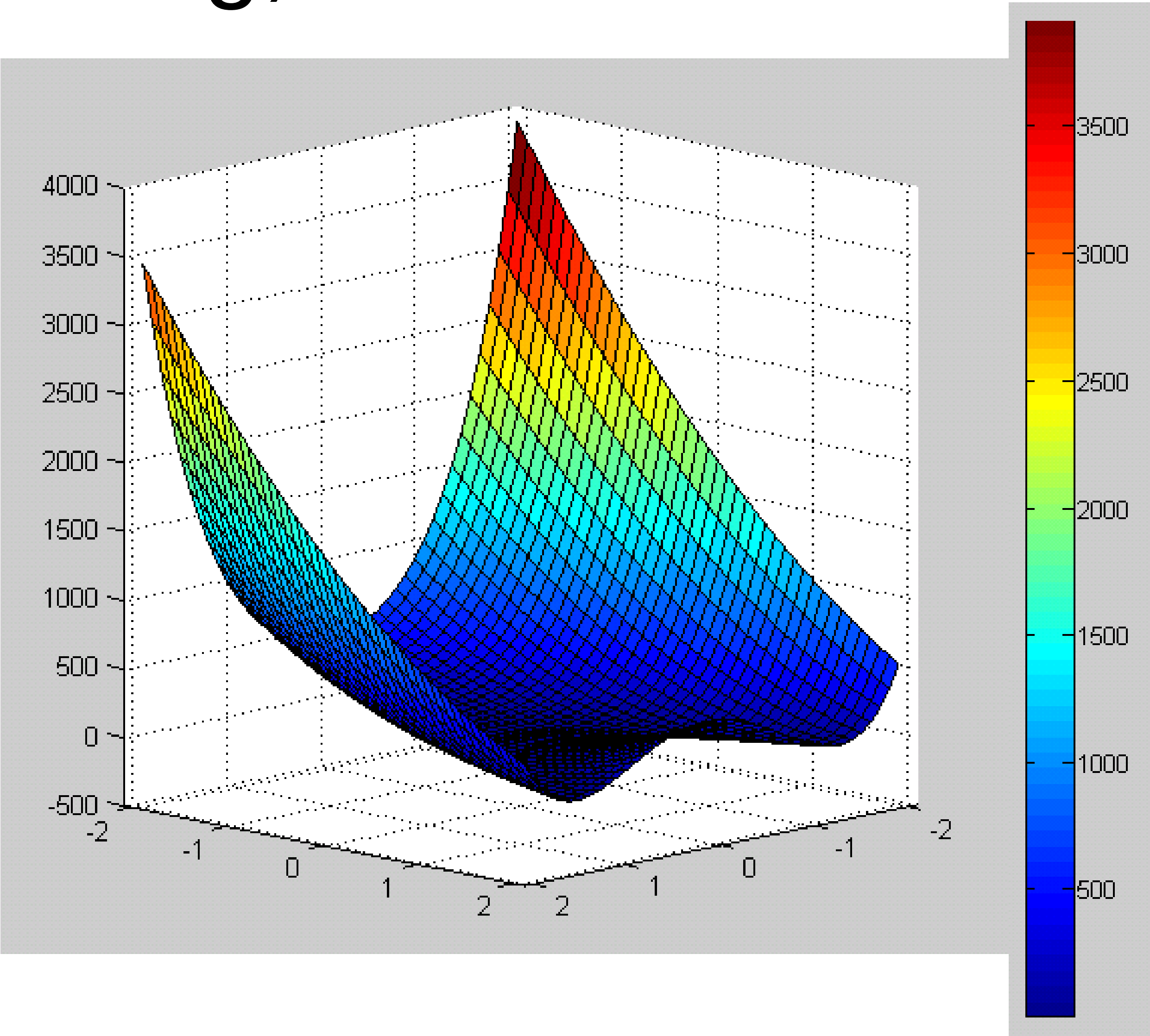
- **Minimize Rosenbrock function**:

$$f_2(\mathbf{x}) = \sum_{j=1}^{D/2} [100(x_{2j-1}^2 - x_{2j})^2 + (1 - x_{2j-1})^2]$$

$$\text{with } x_j \in [-2.048, 2.048] \text{ and } f^*(\mathbf{x}) = 0.0.$$

- if D = 2, then f^* ($x1 \approx 1$, $x2 \approx 1$) $\leq 10^{-6}$
- **Rosenbrock function D = 2 is a difficult minimization problem because the convergence to the global optimum is inside a long, narrow, parabolic shaped flat valley**

Rosenbrock Surface Plotting, D=2



Experiments and Results

- DE/rand/1/bin scheme is used
- NP=10D, F=0.85, CR=0.5
- Results:
  - Total time for 20 test runs (ms) = 156
  - Eval. (mean) = 108.750
  - Best min fitness value = 8.56340e-08
  - Best member:
    - var(0) = 0.9997081358509485
    - var(1) = 0.999414237117633

Sphere Function

- **Minimize Sphere function:**

$$f_1(x) = \sum_{j=1}^{3} x_j^2; \qquad \text{IPR: } x_j \in [-5.12, 5.12]$$

- **IPR is initial parameters range, VTR is value to reach**
- ***f*1(x) is considered to be a very simple task. The global minimum is *f*1*(0)=0 and the VTR was set to 10-6**

Sphere Function Results

- DE/rand/1/bin scheme is used
- NP=10D, F=0.85, CR=0.5
- Results:
  - Total time for 20 test runs (ms) = 109
  - Eval. (mean) = 75.2500
  - Best min fitness value = 7.06624e-08
  - Best member:
    - var(0) = -2.7408741451722535E-5
    - var(1) = 1.319472809555699E-4
    - var(2) = 2.2913106305018E-4

Step Function

- **Minimize Step function:**

$$f_3(x) = 30. + \sum_{j=1}^{5} \lfloor x_j \rfloor$$

$$\text{IPR}: x_j \in [-5.12, 5.12]$$

- **IPR is initial parameters range, VTR is value to reach**
- **The global minimum is *f*3*(-5 - ε) = 0 where ε is [0, 0.12]. VTR was chosen to be 10-6. The step function exhibits many plateaus which pose a considerable problem for many minimization algorithms**

Step function results

- DE/rand/1/bin scheme is used
- NP=10D, F=0.85, CR=0.5
- Results:
  - Total time for 20 test runs (ms) = 344
  - Eval. (mean) = 116.150
  - Best min fitness value = 0.00000
  - Best member:
    - var(0) = -5.09539553399197
    - var(1) = -5.108987212819033
    - var(2) = -5.067089590952487
    - var(3) = -5.007127039774492
    - var(4) = -5.06186779248186

Quartic Function

- **Minimize Quartic function:**

$$f_4(x) = \sum_{j=1}^{30} (j \cdot x_j^4 + \eta); \qquad \text{IPR}: x_j \in [-1.28, 1.28]$$

- **IPR is initial parameters range, VTR is value to reach**
- **This *f*4 function is designed to test the behavior of a minimization algorithm in the presence of noise. In the original De Jong function, *η* is a random variable produced by Gaussian noise having the distribution N(0,1). In contrast to the original version of De Jong's quartic function, we have also included *η* inside the summation instead of just adding *η* to the summation result. This change makes *f*4(x) more difficult to minimize. The *f*4*(0)≤30E[*η*]=15=VTR, where E[*η*] is the expectation of *η***

Shekel's Foxholes Function

- Minimize Foxholes function:

$$f_5(x) = \frac{1}{0.002 + \sum_{i=0}^{24} \frac{1}{i + \sum_{j=1}^{2} (x_j - a_{ij})^6}};$$

$$\text{IPR: } x_j \in [-65.536, 65.536]$$

with $a_{i1} = \{-32, -16, 0, 16, 32\}$ for $i = 0, 1, 2, 3, 4$ and $a_{i1} = a_{i \bmod 5, 1}$ as well as $a_{i2} = \{-32, -16, 0, 16, 32\}$ for $i = 0, 5, 10, 15, 20$ and $a_{i2} = a_{i+k,2}$, $k = 1, 2, 3, 4$

The global minimum for this function is $f_6(-32, -32) \cong 0.998004$, the VTR was defined to be 0.998005.

- IPR is initial parameters range, VTR is value to reach

Shekel's Foxholes Function Results

- DE/rand/1/bin scheme is used
- NP=10D, **F=0.99, CR=0.0**
- Results:
  - Total time for 20 test runs (ms) = 859
  - Eval. (mean) = 41.1500
  - Best min fitness value = 0.998004
  - Best member:
    - var(0) = -31.980486176217152
    - var(1) = -31.97553739359431

Conclusion

- Basic principle of DE is outlined.
- Mutation, Crossover and selection operations explained.
- Types of DE discussed.
- Types of parameter selection outlined.
- Results of some examples presented.
- Comparison criteria of EC algorithms outlined.

References

1. Lampinen, J. and Zelinka, I. (2000). *On Stagnation of the Differential Evolution Algorithm*.In: Ošmera, P. (ed.) (2000). Proceedings of MENDEL 2000, 6th Int. Conf. On Soft Computing, June 7.–9. 2000, Brno University of Technology, Brno, Czech Republic,pp. 76–83. ISBN 80-214-1609-2. Available via Internet:`http://www.lut.fi/~jlampine/MEND2000.ps` .
2. Price, K.V. (1999). *An Introduction to Differential Evolution*. In: Corne, D., Dorigo,M. and Glover, F. (eds.) (1999). New Ideas in Optimization, pp. 79–108. McGraw-Hill, London. ISBN 007-709506-5.
3. Storn, R. and Price, K.V. (1995). *Differential evolution - a Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous paces*. Technical Report TR-95-012, ICSI, March 1995. Available via Internet: `ftp://ftp.icsi.berkeley.edu/pub/techreports/1995/tr-95-012.ps.Z` .
4. Storn, R. and Price, K.V. (1997). *Differential Evolution – a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces*. Journal of Global Optimization,11(4):341–359, December 1997. Kluwer Academic Publishers.
5. (C) Slick n Slide Version 1.29, Timo Kauppinen 5/1995
6. http://www.it.lut.fi/kurssit/05-06/Ti5216300/sources.html [cited on 25th January, 2008][Availaible Online]
7. Lampinen, J. ,Multi-Constrained Nonlinear Optimization by the Differential Evolution Algorithm,

THANK YOU