Differential Evolution: using differences to guide the search

Daniela Zaharie

Department of Computer Science
West University of Timisoara, Romania
e-mail:dzaharie@info.uvt.ro

ECODAM 2012 - Iasi

Outline

"Differential Evolution" as a keyword

Ten years ago ...

- Biology, medicine: evolution process leading to the differentiation of cell types, e.g. less specialized cells become more specialized
- Mathematics: differential evolution relates to a class of differential equations

Now ...

- Computer science: population based search method which uses as main source of variation differences between randomly selected elements
  - no differential calculus is involved
  - just differences between vectors

"Differential Evolution" as a keyword

Ten years ago ...

- Biology, medicine: evolution process leading to the differentiation of cell types, e.g. less specialized cells become more specialized
- Mathematics: differential evolution relates to a class of differential equations

Now ...

- Computer science: population based search method which uses as main source of variation differences between randomly selected elements
  - no differential calculus is involved
  - just differences between vectors

Roots of Differential Evolution

- developed in 1995 by Rainer Storn and Kenneth Price as a continuous optimization method
  - starting problem: Chebyshev polynomials fitting (33 variables)
  - starting variant: genetic annealing algorithm developed by Kenneth Price (1994)
- main idea: use a mutation/recombination operator based on difference(s) between pairs of elements
- similarities with older direct search methods:
  - pattern search (Hooke-Jeeves, 1961)
  - simplex methods (Nelder-Mead, 1965)
- other population based methods involving differences:
  - Particle Swarm Optimization (Kennedy & Eberhart 1995)

DE webpage http://www.icsi.berkeley.edu/ storn/code.html
Books:
K.V. Price, R.M. Storn, J.A. Lampinen; Differential Evolution. A Practical Approach to Global Optimization, 2005

U. Chakraborty, Advances in Differential Evolution, 2008

Roots of Differential Evolution

- developed in 1995 by Rainer Storn and Kenneth Price as a continuous optimization method
  - starting problem: Chebyshev polynomials fitting (33 variables)
  - starting variant: genetic annealing algorithm developed by Kenneth Price (1994)
- main idea: use a mutation/recombination operator based on difference(s) between pairs of elements
- similarities with older direct search methods:
  - pattern search (Hooke-Jeeves, 1961)
  - simplex methods (Nelder-Mead, 1965)
- other population based methods involving differences:
  - Particle Swarm Optimization (Kennedy & Eberhart 1995)

DE webpage http://www.icsi.berkeley.edu/ storn/code.html
Books:
K.V. Price, R.M. Storn, J.A. Lampinen; Differential Evolution. A Practical Approach to Global Optimization, 2005

U. Chakraborty, Advances in Differential Evolution, 2008

Roots of Differential Evolution

- developed in 1995 by Rainer Storn and Kenneth Price as a continuous optimization method
  - starting problem: Chebyshev polynomials fitting (33 variables)
  - starting variant: genetic annealing algorithm developed by Kenneth Price (1994)
- main idea: use a mutation/recombination operator based on difference(s) between pairs of elements
- similarities with older direct search methods:
  - pattern search (Hooke-Jeeves, 1961)
  - simplex methods (Nelder-Mead, 1965)
- other population based methods involving differences:
  - Particle Swarm Optimization (Kennedy & Eberhart 1995)

DE webpage http://www.icsi.berkeley.edu/ storn/code.html
Books:
K.V. Price, R.M. Storn, J.A. Lampinen; Differential Evolution. A Practical Approach to Global Optimization, 2005

U. Chakraborty, Advances in Differential Evolution, 2008

Roots of Differential Evolution

- developed in 1995 by Rainer Storn and Kenneth Price as a continuous optimization method
  - starting problem: Chebyshev polynomials fitting (33 variables)
  - starting variant: genetic annealing algorithm developed by Kenneth Price (1994)
- main idea: use a mutation/recombination operator based on difference(s) between pairs of elements
- similarities with older direct search methods:
  - pattern search (Hooke-Jeeves, 1961)
  - simplex methods (Nelder-Mead, 1965)
- other population based methods involving differences:
  - Particle Swarm Optimization (Kennedy & Eberhart 1995)

DE webpage http://www.icsi.berkeley.edu/ storn/code.html
Books:
K.V. Price, R.M. Storn, J.A. Lampinen; Differential Evolution. A Practical Approach to Global Optimization, 2005

U. Chakraborty, Advances in Differential Evolution, 2008

Roots of Differential Evolution

Why using differences?

- a difference specifies a direction of change
- scaling the difference allows to control the amount of change

Example: pattern search (Hooke-Jeeves)

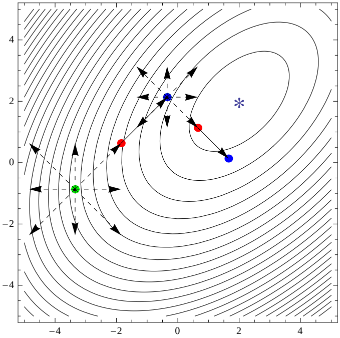
Explore:

$$x_e = x_{old} + h * d \qquad f(x_e) < f(x_{old})$$

$$d \in \{-1, 0, 1\}^n$$

Enhance:

$$x_{new} = x_e + (x_e - x_{old}) \qquad f(x_{new}) < f(x_e)$$



Usage of differences:

- Pattern search, Nelder-Mead: difference directed toward better elements
- DE: randomly constructed differences

Roots of Differential Evolution

Why using differences?

- a difference specifies a direction of change
- scaling the difference allows to control the amount of change

Example: pattern search (Hooke-Jeeves)

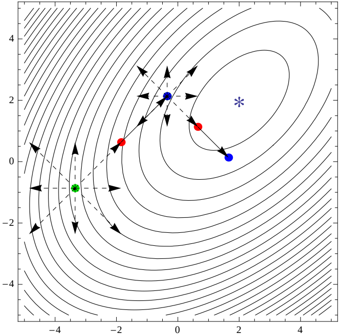
Explore:

$$x_e = x_{old} + h * d \qquad f(x_e) < f(x_{old})$$

$$d \in \{-1, 0, 1\}^n$$

Enhance:

$$x_{new} = x_e + (x_e - x_{old}) \qquad f(x_{new}) < f(x_e)$$



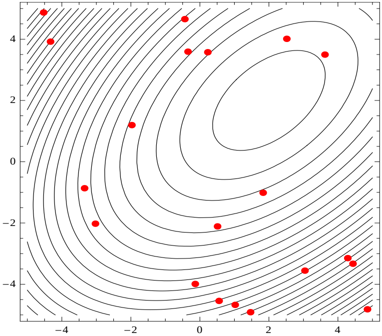
Usage of differences:

- Pattern search, Nelder-Mead: difference directed toward better elements
- DE: randomly constructed differences

Standard Differential Evolution

Problem to be solved: minimize $f : [a_1, b_1] \times \ldots \times [a_n, b_n] \to \mathbb{R}$

Initialization: $x_i = U(a_i, b_i), \qquad i = \overline{1, m}$



m - population size

Standard Differential Evolution

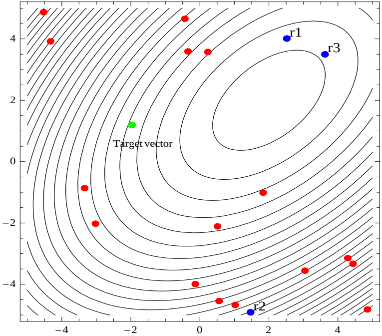
Problem to be solved: minimize $f : [a_1, b_1] \times \ldots \times [a_n, b_n] \to \mathbb{R}$

Initialization: $x_i = U(a_i, b_i), \quad i = \overline{1, m}$

`while` ⟨ NOT termination ⟩ `do`

- Mutation:

$$y_i = x_{r_1} + F \cdot (x_{r_2} - x_{r_3}), \quad i = \overline{1, m}$$



m - population size

$F \in (0, 2)$ - scale factor

Standard Differential Evolution

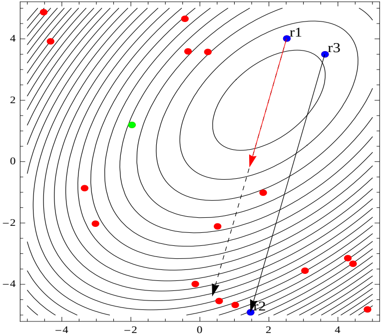
Problem to be solved: minimize $f : [a_1, b_1] \times \ldots \times [a_n, b_n] \to \mathbb{R}$

Initialization: $x_i = U(a_i, b_i), \qquad i = \overline{1, m}$

`while` ⟨ NOT termination ⟩ `do`

- Mutation:

$$y_i = x_{r_1} + F \cdot (x_{r_2} - x_{r_3}), \quad i = \overline{1, m}$$



m - population size

$F \in (0, 2)$ - scale factor

Standard Differential Evolution

Problem to be solved: minimize $f : [a_1, b_1] \times \ldots \times [a_n, b_n] \to \mathbb{R}$

Initialization: $x_i = U(a_i, b_i), \qquad i = \overline{1, m}$

`while` $\langle$ NOT termination $\rangle$ `do`

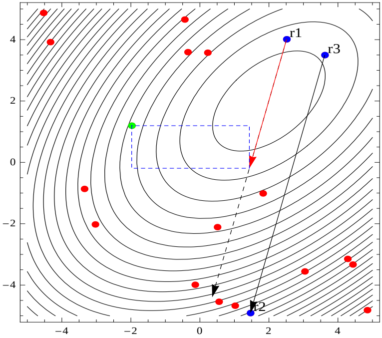
- Mutation:

$$y_i = x_{r_1} + F \cdot (x_{r_2} - x_{r_3}), \quad i = \overline{1, m}$$

- Crossover:

$$z_i^j = \begin{cases} y_i^j & \text{if } rand(0,1) < CR \text{ or } j = j_0 \\ x_i^j & \text{otherwise} \end{cases},$$

$$i = \overline{1, m}, j = \overline{1, n}$$



m - population size
$F \in (0, 2)$ - scale factor
$CR \in (0, 1)$ - crossover rate

Standard Differential Evolution

Problem to be solved: minimize $f : [a_1, b_1] \times \ldots \times [a_n, b_n] \to \mathbb{R}$

Initialization: $x_i = U(a_i, b_i), \qquad i = \overline{1, m}$

`while` ⟨ NOT termination ⟩ `do`

- Mutation:

$$y_i = x_{r_1} + F \cdot (x_{r_2} - x_{r_3}), \quad i = \overline{1, m}$$

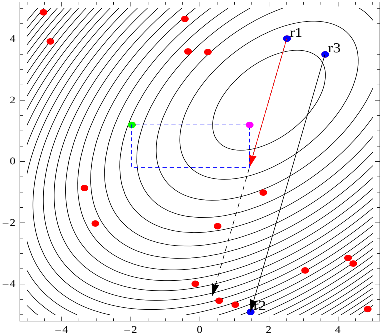
- Crossover:

$$z_i^j = \begin{cases} y_i^j & \text{if } rand(0,1) < CR \text{ or } j = j_0 \\ x_i^j & \text{otherwise} \end{cases},$$

$$i = \overline{1, m}, j = \overline{1, n}$$

- Selection:

$$x_i(g+1) = \begin{cases} z_i & \text{if } f(z_i) \leq f(x_i(g)) \\ x_i^j & \text{if } f(z_i) > f(x_i(g)) \end{cases}$$



m - population size
$F \in (0, 2)$ - scale factor
$CR \in (0, 1)$ - crossover rate

Outline

Popularity

Common statements in DE papers:

- "Differential evolution is arguably one of the hottest topics in today's computational intelligence research." [Chakraborty - Advances in DE, 2008]
- "Since 1997 we have witnessed explosive growth in differential evolution research." [Qing, 2009]
- "DE is a simple and efficient optimizer" [Neri, Tirronen, 2010]
- "Differential evolution (DE) is arguably one of the most powerful stochastic real-parameter optimization algorithms in current use." [Das, Suganthan, 2011 - DE Survey]

Keywords:

- simple: can be implemented in a few lines of code
- powerful: flexible structure $\Rightarrow$ can be applied for a large class of problems
- efficient: estimation of optima with an acceptable cost

Flexibility

Various combinations of mutation and crossover variants

DE taxonomy: $\boxed{\text{DE/ base element/ no. of differences/ crossover type}}$

- Base element:
  - random(x_{r_1}): DE/rand/*/*
  - best (x_*): DE/best/*/*
  - combination of current and best elements ($\lambda x_* + (1-\lambda)x_i$): DE/current-to-best/*/*
  - combination of random and best elements ($\lambda x_* + (1-\lambda)x_{r_1}$): DE/rand-to-best/*/*
  - combination of current and random elements ($\lambda x_i + (1-\lambda)x_{r_1}$): DE/current-to-rand/*/*
- Number of differences: usually 1 (DE/*/1/*) or 2 (DE/*/2/*)
- Crossover type: binomial: DE/*/*/bin, exponential: DE/*/*/exp)

At least 20 DE variants ...

Flexibility

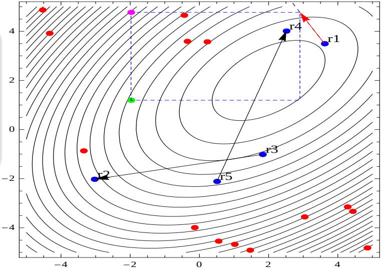
Mutation variants

DE/rand/L/*

$$y_i = x_{r_1} + \sum_{l=1}^{L} F_l \cdot (x_{r_1(l)} - x_{r_2(l)})$$

Typical variant: $L = 2$

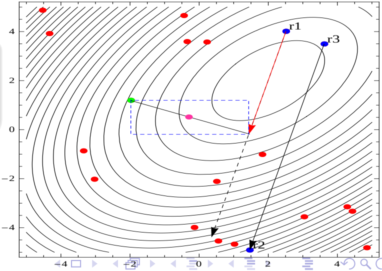
△ Allows to define new mutant directions ⇒ increased diversity



DE/current-to-best/1

$$y_i = (1-\lambda)x_i + \lambda x_* + F \cdot (x_{r_1} - x_{r_2})$$

△ Introduces a bias toward the currently best element



Flexibility

Crossover variants

Binomial (DE/*/*/bin)

$$z_i^j = \begin{cases} y_i^j & \text{if } rand(0,1) < CR \text{ or } j = j_0 \\ x_i^j & \text{otherwise} \end{cases}, \qquad i = \overline{1,m}, j = \overline{1,n}$$

Remark: similar to uniform crossover

Exponential (DE/*/*/exp)

$$z_i^j = \begin{cases} y_i^j & \text{for } j \in \{j_0, \langle j_0+1\rangle_n, \ldots, \langle j_0+K-1\rangle_n\} \\ x_i^j & \text{otherwise} \end{cases}, i = \overline{1,m}, j = \overline{1,n}$$

Remark: similar to cut points crossover

$CR \in [0,1]$ - crossover rate, $j_0 \sim U(\{1,\ldots,m\})$, $K \sim Geom(CR)$

Flexibility

Other recombination variants

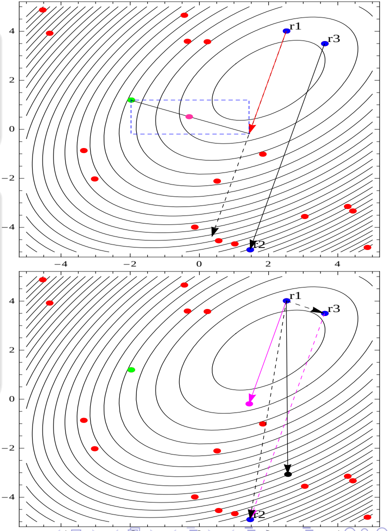
Arithmetical (DE/*/*/arithmetical)

$$z_i = (1-q)x_i + qy_i, \qquad i=\overline{1,m}, \qquad q \in [0,1]$$

Either mutation or recombination (DE/either-or)

$$z_i = \begin{cases} x_{r_1} + F \cdot (x_{r_2} - x_{r_3}) & \text{if } rand(0,1) \leq p_F \\ x_{r_1} + K \cdot (x_{r_2} - x_{r_1}) + K \cdot (x_{r_3} - x_{r_1}) & \text{if } rand(0,1) > p_F \end{cases}$$

Remark: DE/either-or was created to compensate the lack of rotational invariance of DE involving binomial crossover



Flexibility

(Self)adaptive variants

The other face of flexibility: which variant to choose ?

Recommendations

- no specific knowledge on the problem: use DE/rand/1/*
- need for an exploitative method: use DE/best/1/*
- need for a more explorative method: use DE/rand/2/*
- need for a rotationally invariant method: use DE/either-or

Remark: different variants could be appropriate in different stages of the optimization process

Flexibility

(Self)adaptive variants

The other face of flexibility: which variant to choose ?

Recommendations

- no specific knowledge on the problem: use DE/rand/1/*
- need for an exploitative method: use DE/best/1/*
- need for a more explorative method: use DE/rand/2/*
- need for a rotationally invariant method: use DE/either-or

Remark: different variants could be appropriate in different stages of the optimization process

Flexibility

(Self)adaptive variants

The other face of flexibility: which variant to choose ?

Self adaptation

- use a pool of variants and assign to each element a DE variant
- record the success of the variant attached to each element
- decide which variant to select based on the success/failure information (a probability distribution is usually constructed)
- self-adaptation of mutation/crossover is usually combined with self-adaptation of parameters
- Examples: SaDE [Qin, Huang & Suganthan, 2009], EPSDE [Mallipedi et al., 2011], Competitive DE [Tvrdik, 2009] etc.

Flexibility

Extensions to other search spaces: binary, discrete, permutations

Simplest variant: use classical DE operators to evolve vectors with components belonging to a continuous domain and decode the vectors only during the evaluation step

- Binary and discrete values
  - Search domain: $[0, 1]^n$ or $[\min(D), \max(D)]^n$;
  - Decoding: $x_i \rightarrow \text{round}(x_i)$
  - Example: $(0.3, 0.7, 0.2) \rightarrow (0, 1, 0)$
  - Remark: used in various applications (e.g. engineering design, rules mining, gramatical differential evolution)
- Permutations
  - Search domain: $[a, b]^n$;
  - Decoding: $(x_1, x_2, \ldots, x_n) \rightarrow (\text{rank}(x_1), \text{rank}(x_2), \ldots, \text{rank}(x_n))$
  - Example: $(0.3, 0.7, 0.2) \rightarrow (2, 3, 1)$

Flexibility

Extensions to other search spaces: binary, discrete, permutations

Simplest variant: use classical DE operators to evolve vectors with components belonging to a continuous domain and decode the vectors only during the evaluation step

- Binary and discrete values
  - Search domain: $[0, 1]^n$ or $[\min(D), \max(D)]^n$;
  - Decoding: $x_i \to \text{round}(x_i)$
  - Example: $(0.3, 0.7, 0.2) \to (0, 1, 0)$
  - Remark: used in various applications (e.g. engineering design, rules mining, gramatical differential evolution)
- Permutations
  - Search domain: $[a, b]^n$;
  - Decoding: $(x_1, x_2, \ldots, x_n) \to (\text{rank}(x_1), \text{rank}(x_2), \ldots, \text{rank}(x_n))$
  - Example: $(0.3, 0.7, 0.2) \to (2, 3, 1)$

Flexibility

Extensions to other search spaces: binary, discrete, permutations

Another variant: exploit the idea of difference-based mutation by defining

- Binary values

$$y_i^j = \begin{cases} 1 - x_{r_1}^j & \text{if } x_{r_2}^j \neq x_{r_3}^j \text{ or } rand(0,1) \leq F \\ x_{r_1}^j & \text{otherwise} \end{cases}$$

Remark: this is the *restricted-change DE mutation* proposed in [Gong and Tuson, 2006]

- Permutation like encoding
  - Step 1: compute the *Ulam distance* d_U between x_{r_2} and x_{r3} (minimal number of "Delete-Shift-Insert" operations)
  - Step 2: apply d_U random inversions to x_{r_1}
  - Example: $x_{r_1} = (1,2,3,4,5)$, $x_{r_2} = (1,2,4,3,5)$, $x_{r3} = (3,2,1,4,5)$ $d_U(x_{r_2}, x_{r_3}) = 3$, $(1,2,3,4,5) \rightarrow (3,2,1,4,5) \leftarrow (3,2,1,5,4) \rightarrow (5,2,1,3,4)$

Remark: variant used in scheduling problems [Talukder et al., 2009])

Flexibility

Extensions to other search spaces: binary, discrete, permutations

Another variant: exploit the idea of difference-based mutation by defining

- Binary values

$$y_i^j = \begin{cases} 1 - x_{r_1}^j & \text{if } x_{r_2}^j \neq x_{r_3}^j \text{ or } rand(0,1) \leq F \\ x_{r_1}^j & \text{otherwise} \end{cases}$$

Remark: this is the *restricted-change DE mutation* proposed in [Gong and Tuson, 2006]

- Permutation like encoding
  - Step 1: compute the *Ulam distance* d_U between x_{r_2} and x_{r3} (minimal number of "Delete-Shift-Insert" operations)
  - Step 2: apply d_U random inversions to x_{r_1}
  - Example: $x_{r_1} = (1,2,3,4,5)$, $x_{r_2} = (1,2,4,3,5)$, $x_{r3} = (3,2,1,4,5)$
    $d_U(x_{r_2}, x_{r_3}) = 3$,
    $(1,2,3,4,5) \rightarrow (3,2,1,4,5) \leftarrow (3,2,1,5,4) \rightarrow (5,2,1,3,4)$

Remark: variant used in scheduling problems [Talukder et al., 2009])

Flexibility

Extensions to various classes of problems

DE has been successfully adapted for various classes of problems:

- Multi-objective optimization: PDE (Pareto DE), GDE (Generalized DE), MOEA/D (decomposition based MOEA)
- Multi-modal optimization: SDE (Sharing DE), CDE (Crowding DE)
- Dynamic optimization: DynDE, jDEdyn

Main ideas:

- keep the DE mutation as main variation operator
- adapt the selection process
- use of specific mechanisms: crowding, aging, randomness injection, archiving

Outline

Current Knowledge on DE Behavior

Mainly based on empirical parameter studies which lead to rules of thumb as:

- for the same crossover rate (CR), the number of components taken from the mutant is highly depending on the crossover type (binomial vs. exponential) ... why ?
- the control parameters (m, F, CR) influence in an interrelated manner the population diversity ... how ?
- high values of the scale factor, F, are needed to avoid premature convergence ... does there exist a lower bound ?
- a good empirical choice of parameters in DE/either-or is $K = (F + 1)/2$ ... why ?

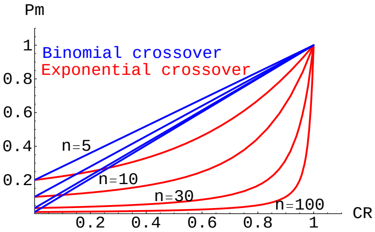
Binomial vs. Exponential Crossover

Binomial crossover:

- the probability to take a component from the mutant vector is:

$$p_m = CR\left(1 - \frac{1}{n}\right) + \frac{1}{n}$$

- the number of mutated components: binomial distribution



Exponential crossover:

- the probability to take a component from the mutant vector is:

$$p_m = \frac{1 - CR^n}{n(1 - CR)}$$

- the number of mutated components: truncated geometric distribution

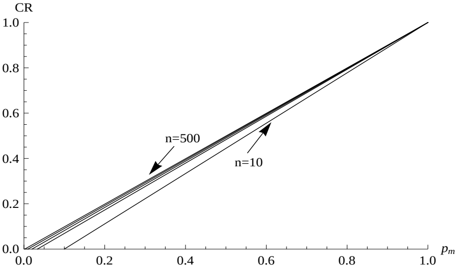
Remark: In the case of exponential crossover larger values of CR should be used in order to have the same number of mutated components as for binomial crossover [Zaharie, 2007].

Choice of crossover rate

- the DE behavior is influenced by the mutation probability, p_m, but the user provide a value for CR
- what value should have CR in order to ensure a given value for p_m ?

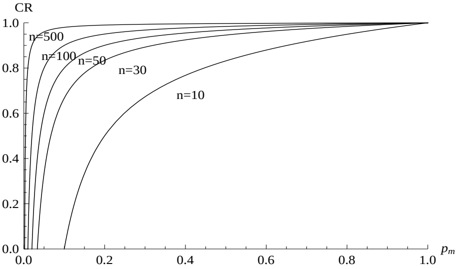
Binomial crossover

$$CR = \frac{p_m - 1/n}{1 - 1/n}$$



Exponential crossover

$$CR^n - np_m CR + np_m - 1 = 0$$

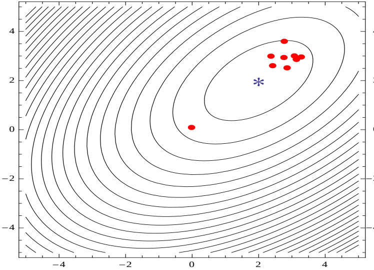


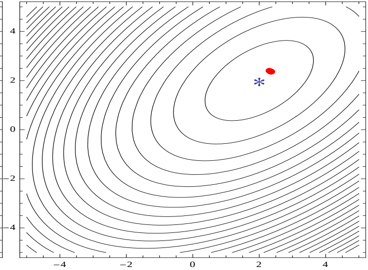
Practical remark: Exponential crossover is more sensitive to the problem size

Population diversity

Importance

- small diversity in the DE population ⇒ small values of the differences ⇒ limited progress ⇒ premature convergence
- diversity measure: population variance (component level)





Question: What is the impact of mutation and crossover on the population variance ?

Population diversity

Theoretical results

$Var(X)$=variance of current population (at component level);
$E(Var(Z))$=expected variance of the trial population

DE/rand/L/*

[Zaharie, 2002]

$$E(Var(Z)) = \left(1 + 2p_m \sum_{l=1}^{L} F_l^2 - \frac{p_m(2-p_m)}{m}\right) Var(X)$$

DE/random-to-best/1/*

[Zaharie, 2008]

$$E(Var(Z)) = \left(1 + 2p_m F^2 - \frac{p_m(2-p_m)}{m} - \lambda p_m^2 \frac{m-1}{m}\right) Var(X)$$
$$+\lambda^2 \frac{p_m(1-p_m)}{m} \sum_{i=1}^{m} (x_* - x_i)^2$$

Population diversity

Theoretical results

$Var(X)$=variance of current population (at component level);
$E(Var(Z))$=expected variance of the trial population

DE/rand/L/*

[Zaharie, 2002]

$$E(Var(Z)) = \left(1 + 2p_m \sum_{l=1}^{L} F_l^2 - \frac{p_m(2-p_m)}{m}\right) Var(X)$$

DE/random-to-best/1/*

[Zaharie, 2008]

$$E(Var(Z)) = \left(1 + 2p_m F^2 - \frac{p_m(2-p_m)}{m} - \lambda p_m^2 \frac{m-1}{m}\right) Var(X) + \lambda^2 \frac{p_m(1-p_m)}{m} \sum_{i=1}^{m} (x_* - x_i)^2$$

Population diversity

Theoretical results

$Var(X)$=variance of current population (at component level);
$E(Var(Z))$=expected variance of the trial population

DE/current-to-rand/1

(arithmetical recombination) [Zaharie, 2008]

$$E(Var(Z)) = \left(1 + 2F^2 - 2q + \frac{2m-1}{m} q^2\right) Var(X)$$

DE/either-or

[Zaharie, 2012]

$$\begin{aligned} E(Var(Z)) &= \left(p_F^2(1+2F^2-\tfrac{1}{m}) + 2p_F(1-p_F)(\tfrac{m-1}{m} + F^2 + 3K^2 - 2K)\right. \\ &\quad \left. +(1-p_F)^2\left(\tfrac{m-1}{m} + 2\tfrac{m-2}{m}(3K^2-2K)\right)\right) Var(X) \end{aligned}$$

Population diversity

Theoretical results

$Var(X)$=variance of current population (at component level);
$E(Var(Z))$=expected variance of the trial population

DE/current-to-rand/1

(arithmetical recombination) [Zaharie, 2008]

$$E(Var(Z)) = \left(1 + 2F^2 - 2q + \frac{2m-1}{m} q^2\right) Var(X)$$

DE/either-or

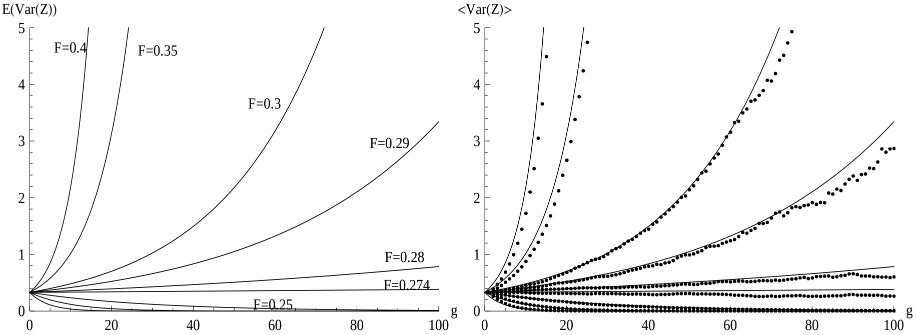
[Zaharie, 2012]

$$\begin{aligned} E(Var(Z)) \quad = \quad & \left(p_F^2 (1 + 2F^2 - \tfrac{1}{m}) + 2p_F(1-p_F)(\tfrac{m-1}{m} + F^2 + 3K^2 - 2K)\right. \\ & \left. +(1-p_F)^2 \left(\tfrac{m-1}{m} + 2\tfrac{m-2}{m}(3K^2 - 2K)\right)\right) Var(X) \end{aligned}$$

Population diversity

Theoretical vs empirical evolution

- Evolution of population variance after mutation and crossover (no selection)
- Practical remark: the population variance can decrease even in the absence of selection pressure



DE/either-or

Population diversity

From theory to practical insights

$$E(Var(Z)) = c(F, CR, p_F, q, m, n)Var(X)$$

- if $c(F, CR, p_F, q, m, n) < 1$ the algorithm will probably prematurely converge
- one can control the impact which mutation and crossover have on the population variance by changing the values of the parameters involved in the factor c
- this is a particularity of DE, as in EAs using mutation based additive perturbation involving an arbitrary distribution:

$$E(Var(Z)) = aVar(X) + b$$

with b not necessarily zero

Population diversity

From theory to practical insights

$$E(Var(Z)) = c(F, CR, p_F, q, m, n)Var(X)$$

- if $c(F, CR, p_F, q, m, n) < 1$ the algorithm will probably prematurely converge
- one can control the impact which mutation and crossover have on the population variance by changing the values of the parameters involved in the factor c
- this is a particularity of DE, as in EAs using mutation based additive perturbation involving an arbitrary distribution:

$$E(Var(Z)) = aVar(X) + b$$

with b not necessarily zero

Population diversity

From theory to practical insights

$$E(Var(Z)) = c(F, CR, p_F, q, m, n)Var(X)$$

- if $c(F, CR, p_F, q, m, n) < 1$ the algorithm will probably prematurely converge
- one can control the impact which mutation and crossover have on the population variance by changing the values of the parameters involved in the factor c
- this is a particularity of DE, as in EAs using mutation based additive perturbation involving an arbitrary distribution:

$$E(Var(Z)) = aVar(X) + b$$

with b not necessarily zero

Population diversity

From theory to practical insights

$$E(Var(Z)) = c(F, CR, p_F, q, m, n)Var(X)$$

- if $c(F, CR, p_F, q, m, n) < 1$ the algorithm will probably prematurely converge
- one can control the impact which mutation and crossover have on the population variance by changing the values of the parameters involved in the factor c
- this is a particularity of DE, as in EAs using mutation based additive perturbation involving an arbitrary distribution:

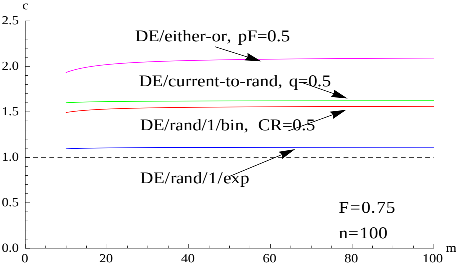
$$E(Var(Z)) = aVar(X) + b$$

with b not necessarily zero

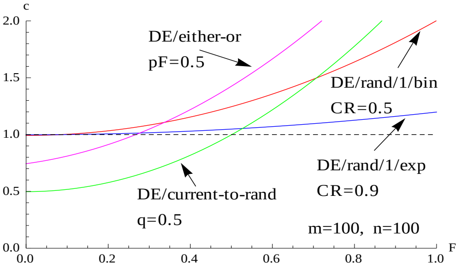
Population diversity

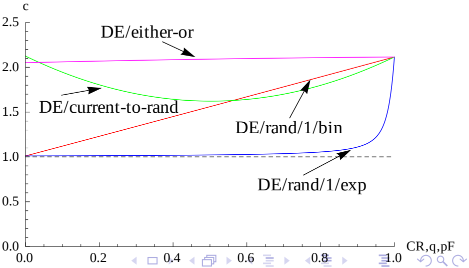
From theory to practical insights

$$E(Var(Z)) = c(F, CR, p_F, q, m, n)Var(X)$$



- the value of $c(F, CR, p_F, q, m, n)$ is highly influenced by the type of mutation and crossover

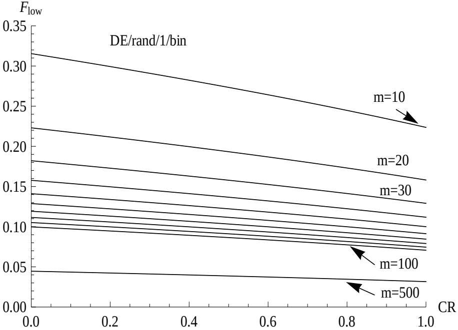




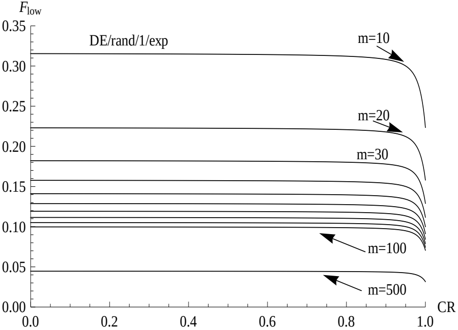
Population diversity

Avoiding premature convergence

- choose the DE control parameters (F, CR, m etc) such that the population diversity does not decrease too fast ($c(CR, F, q, m, n) > 1$)
- by solving $c(F, CR, p_F, q, m, n) = 1$ we can find a lower bound for F under which the population decreases even in the absence of selection



DE/rand/1/bin

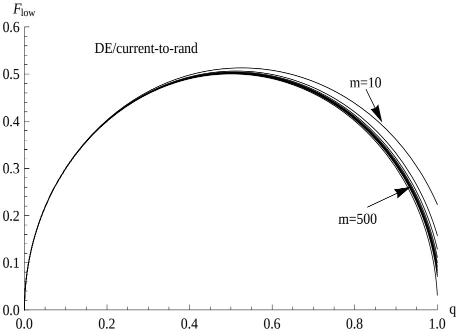


DE/rand/1/exp

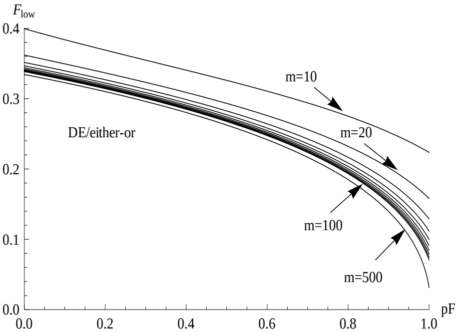
Population diversity

Avoiding premature convergence

- choose the DE control parameters (F, CR, m etc) such that the population diversity does not decrease too fast ($c(CR, F, q, m, n) > 1$)
- by solving $c(F, CR, p_F, q, m, n) = 1$ we can find a lower bound for F under which the population decreases even in the absence of selection



DE/current-to-rand



DE/either-or

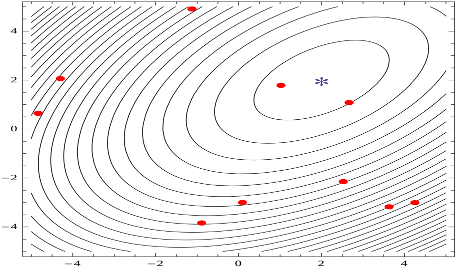
Population diversity

Avoiding premature convergence

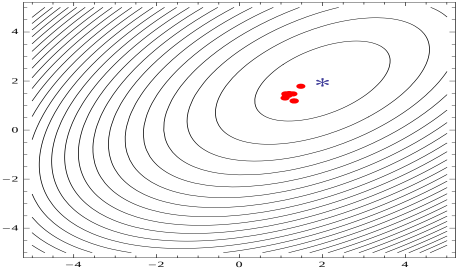
Example:

- DE/rand/1/bin for Neumaier fct, $n = 2$
- $m = 20,\ CR = 0.9,\ F = 0.2$
- lower bound $F_{low} = 0.23$

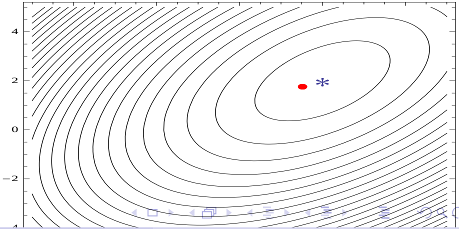
gen=1



gen=5



gen=15



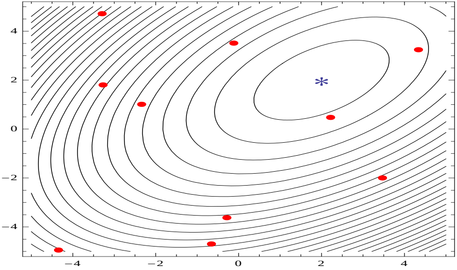
Population diversity

Avoiding premature convergence

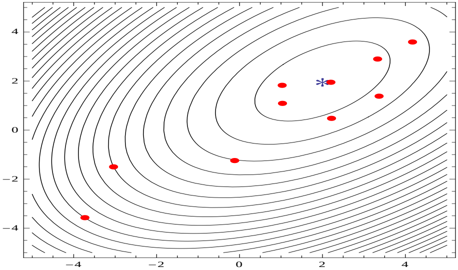
Example:

- DE/rand/1/bin for Neumaier fct, $n = 2$
- $m = 20,\ CR = 0.9,\ F = 0.5$
- lower bound $F_{low} = 0.23$

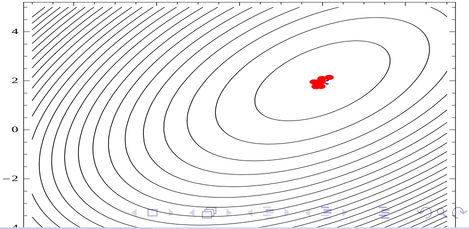
gen=1



gen=5



gen=15



Population diversity

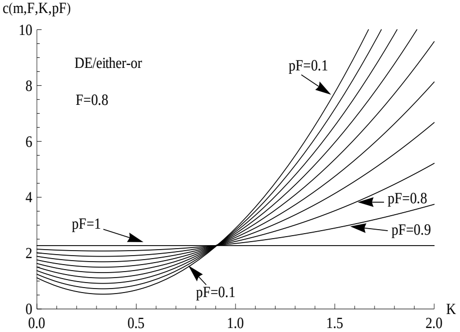
Avoiding premature convergence

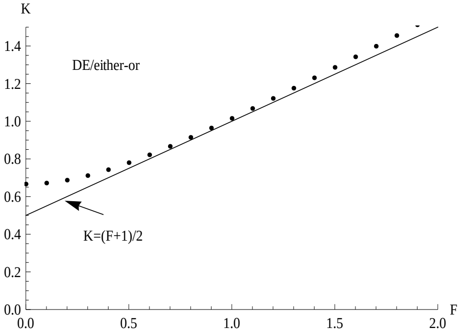
- the knowledge of lower bound is particularly important for small populations
- successfull usage of the lower bound: variant of jDE [Brest et al. 2006] adapted for Dynamic Optimization Problems (winner of CEC 2009 competition)
  - in static jDE the parameter F is sampled from [0.1,0.9]
  - in dynamic jDE the parameter F is sampled from [0.36, 0.9]

Population diversity

Explaining empirical rules

- Rule of thumb for DE/either-or: $K = (F+1)/2$
- When $K = (F+1)/2$ the variance evolution is not sensitive to p_F





Differential Evolution without differences ?

Question: can the DE behaviour be reproduced by mechanisms which do not involve differences?

Mimicking the distribution of DE trial population

- Use a Gamma-like probability distribution to generate trial vectors [Ali& Fatti, 2006]
- Advantage: all trial vectors are in the search domain (no repairing rule is needed)
- Disadvantage: more complicated than DE

Mimicking the DE trial population variance

- Variance-based mutation [Zaharie, 2008]
  $y_i = r_1 + \xi_i, \quad \xi_i^j \sim N(0,\sigma) \quad \sigma^2 = F^2 Var(X^j)\frac{m}{m-1}$
- $E(Var(Z)) = (1 + p_m^2 F^2 - \frac{p_m(2-p_m)}{m})Var(X)$ - as in the case of DE/rand/1/*
- however, the performance is not identical

Variance Based Mutation - Numerical Results

CR	F	DE/rand/1/bin		var/bin	
		$\langle f^* \rangle$ $stdev(f^*)$	Success $\langle nfe \rangle$	$\langle f^* \rangle$ $stdev(f^*)$	Success $\langle nfe \rangle$
0.1	0.5	$9 \cdot 10^{-9}$ $\pm 10^{-10}$	30/30 (380416)	$9 \cdot 10^{-9}$ $\pm 10^{-10}$	30/30 (190290)
0.5	0.5	10^{-4} $\pm 10^{-5}$	0/30 (500000)	$9 \cdot 10^{-9}$ $\pm 10^{-10}$	30/30 (204703)
0.9	0.5	0.0078 ± 0.0125	18/30 (306933)	$1.27 \cdot 10^{-8}$ $\pm 10^{-8}$	27/30 (470792)
0.1	0.2	$9 \cdot 10^{-9}$ $\pm 2 \cdot 10^{-10}$	30/30 (137090)	0.0158 0.0318	24/30 (131887)
0.5	0.2	0.0959 ± 0.1657	18/30 (87666)	1.3469 1.5373	0/30 (500000)

Test function: $f(x_1, \ldots, x_n) = \frac{1}{4000} \sum_{j=1}^{n} x_j^2 - \prod_{j=1}^{n} \cos(x_j/\sqrt{n}) + 1$ (Griewank, n=100)

Outline

Non-separable problems

- Separable functions:

  $argmin_{(x_1,x_2,\ldots,x_n)} f(x_1,x_2,\ldots,x_n) = (argmin_{x_1} f(x_1,*,\ldots,*), argmin_{x_2} f(*,x_2,\ldots$

  - Example (additively separable): $f(x_1,x_2,\ldots,x_n)=\sum_{i=1}^{n} f_i(x_i)$
  - DE with small values of CR (e.g. $CR \leq 2$) explores the separability
- Nonseparable functions: the variables are correlated
  - Example: by a rotation of the axes a separable problem can become nonseparable
  - DE is rotationally invariant only when $CR = 1$ (only mutation)

Non-separable problems

- using only mutation $\Rightarrow$ reduces the number of trial vectors $\Rightarrow$ stagnation
  - DE/rand/1/bin: when $CR = 1$ there are $(m-1)(m-2)$ possible trial vectors instead of $(m-1)(m-2)(2^n-1)$
- Idea: use of "recombination differentials" (differences involving the current element x_i)
  - DE/either-or [Price, 2005], drift free DE [Price, 2008]

$$z_i = \begin{cases} x_i + F\cdot(x_{r1}-x_{r2}) \text{ if } rnd < p_F \\ x_i + K\cdot(x_{r3}-x_{r4}-2x_i) \text{ otherwise} \end{cases}$$

  - Combinatorial Differential Evolution [Iorio, Li, 2008] - alternatively applies:

$$z_i = x_i + F\cdot(x_i - x_r) \qquad z_i^j = \begin{cases} x_i^j + F\cdot(x_i^j - x_r^j) \text{ if } rnd < 0.5 \\ x_i^j + F\cdot(x_r^j - x_i^j) \text{ otherwise} \end{cases}$$

when $f(x_i) < f(x_r)$

Remark. Not strict rotationally invariant but generates new trial vectors around the current one

High-dimensional problems

- the problem size influences directly the relationship between p_m and CR (especially for exponential crossover)
  - CR values tuned for small size problems are not necessarily good for large size problems
- Most non hybrid DE variants are based on cooperative coevolution which split the problem into smaller sub-problems:
  - a potential solution consists of several components
  - evolve independently the population corresponding to each component (coevolution)
  - each component is evaluated in the context of other components (cooperation)

Noisy problems

- standard DE behaves rather poor for noisy optimization problems
- Cause: the difference based mutation does not ensure enough level of randomness
- Solution: increase the level of randomness
  - random control parameters (F and CR)
  - extend the pool of perturbations (e.g. opposition based DE)
  - hybridization

Conclusions

- DE should be in the "bag of tools" of practitioners, but ...
- attention should be paid on the choice of variant and parameters
- use the existing theoretical results to collect useful practical insights