Course: Natural Computing

5. Parameter Settings

J. Michael Herrmann
School of Informatics, University of Edinburgh
michael.herrmann@ed.ac.uk, +44 131 6 517177

Overview

- PSO Parameters
- DE Parameters
- Other algorithms

Convergence

- **Failure**: Swarm diverges or is stopped by search space boundaries
- **Ideally**: Global best approaches global optimum while swarm converges
- **Typically**:
  - Global best approaches a local optimum because premature collapse of the swarm
  - Global best is near global optimum and swarm remains itinerant
- Convergence is not necessary (global or local bests remember previous good solutions)
- Convergence may be useful to search the space around a good solution more carefully (see below "constriction")
- Alternatively, add a hill-climbing stage to the PSO algorithm

Analysis of PSO: Simplified algorithm

- Ignore randomness (use a homogeneous mean value)
- Ignore the global best (assume it equals personal best)
- Personal best constant (changes are rare, asymptotically)
  i.e. we had (vector equation for velocity of i-th particle)

$$\mathrm{v}_i\,(t+1) \leftarrow \omega \mathrm{v}_i + \alpha_1 r_1 \circ (\mathrm{p}_i - \mathrm{x}_i\,(t)) + \alpha_2 r_2 \circ (\mathrm{g} - \mathrm{x}_i\,(t))$$

which becomes now in component form $i = 1 \ldots n$, $d = 1 \ldots m$

$$v_{id}\,(t+1) = \omega v_{id}\,(t) + \alpha\,(p_{id} - x_{id}\,(t))$$

and for one particle in one dimension simply

$$v\,(t+1) = \omega v\,(t) + \alpha\,(p - x\,(t))$$
$$x\,(t+1) = x(t) + v(t+1)$$

PSO: Algebraic point of view

In order to study the dynamical properties of a particle swarm, consider a simplified algorithm with 1 particle, no noise, no global best:

Introduce $y(t) = p - x(t)$ in $v(t+1) = \omega v(t) + \alpha (p - x(t))$

$$x(t+1) = x(t) + v(t+1)$$
$$\Rightarrow \begin{cases} v(t+1) = \omega v(t) + \alpha y(t) \\ y(t+1) = -\omega v(t) + (1-\alpha) y(t) \end{cases}$$

Introduce state vector $z(t) = (v(t), y(t))^\top$ such that

$$z(t+1) = M\, z(t) \text{ with } M = \begin{pmatrix} \omega & \alpha \\ -\omega & 1-\alpha \end{pmatrix}$$

Starting from the initial state $z(0)$ we have $z(t) = M^t\, z(0)$

M. Clerc & J. Kennedy (2002) The particle swarm – Explosion, stability, and convergence in a multidimensional complex space. IEEE TA EC 7:1, 58-73.
(see also tutorial 2)

PSO: Algebraic point of view

Simplify further by setting $\omega = 1$ and determine eigenvalues of

$M = \begin{pmatrix} 1 & \alpha \\ -1 & 1-\alpha \end{pmatrix}$, i.e. find matrix A such that

$AMA^{-1} = L = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \Rightarrow \quad \lambda_{1/2} = 1 - \frac{\alpha}{2} \pm \frac{\sqrt{\alpha^2 - 4\alpha}}{2}$

The simplified PSO dynamics $P(t) = M^t P(0)$ is equivalent to

$$\begin{aligned} P(t+1) &= A^{-1} L A P(t) \\ A P(t+1) &= L A P(t) \\ Q(t+1) &= L Q(t) \end{aligned}$$

with $Q = AP$ and $A = \begin{pmatrix} \alpha + \sqrt{\alpha^2 - 4\alpha} & 2\alpha \\ \alpha - \sqrt{\alpha^2 - 4\phi} & 2\alpha \end{pmatrix}$

Thus $Q(t) = L^t Q(0)$ where $L^t = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}^t = \begin{pmatrix} \lambda_1^t & 0 \\ 0 & \lambda_2^t \end{pmatrix}$

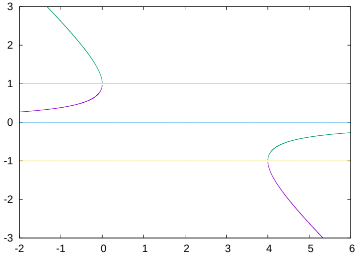
Algebraic point of view ($\omega = 1$)

For the eigenvalues $\lambda_{1/2} = 1 - \frac{\alpha}{2} \pm \frac{\sqrt{\alpha^2 - 4\alpha}}{2}$ we have three cases:

$0 < \alpha < 4$	$\alpha = 4$ or $\alpha = 0$	$\phi > 4$ or $\alpha < 0$
pair of complex EV	$\lambda_{1/2} = -1$ or $\lambda_{1/2} = 1$	$\lambda_2 < -1$ or $\lambda_1 > 1$
$\lambda_{1/2} = \cos(\theta) \pm i \sin(\theta)$ $\lambda^t_{1/2} = \cos(t\theta) \pm i \sin(t\theta)$	$z(t+1) = -z(t)$ or $z(t+1) = z(t)$	Exponentially divergent

Complex eigenvalues result in oscillatory behaviour with period k if $\theta = \frac{2k\pi}{t}$

Real eigenvalues as a function of α $\longrightarrow$

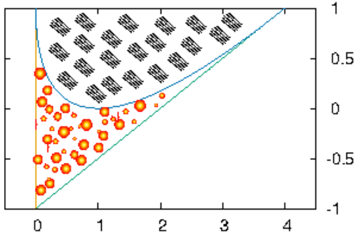


Algebraic point of view ($\omega \in [-1, 1]$)

For $\omega \in [-1, 1]$ the eigenvalues are

$$\lambda_{1/2} = \frac{1 + \omega - \alpha}{2} \pm \frac{\sqrt{(\omega - \alpha)^2 - 2(\omega + \alpha) + 1}}{2}$$

The eigenvalues are complex in the hatched area.
There and in the speckled area the dynamics is convergent.
horizontal: α, vertical: ω



Oscillation may be preferable as they tend to improve exploration near current best

Implications from the algebra

- In the standard PSO we have $\alpha = \alpha_1 + \alpha_2$, i.e. a combination of the attraction towards the personal and global best. These are typically not the same, such that the forces are not perfectly additive and a somewhat larger α might be possible.
- α slightly above 4 (e.g. $\alpha \approx 4.1$): particle stays somewhere in between or near personal and global best. If these two coincide the algorithm tends to diverge, i.e. the particles move on searching elsewhere.
- Note that this result is based on strong simplifications, i.e. it is not exactly true.

Constriction factor in canonical PSO

- If there is reason to believe that the particles are near the global optimum (e.g. after a long runtime), more exploitation might be desirable by reducing the fluctuations of the particles (see Clerc, 1999/2000)
- May help to find the global optimum up to numerical precision
- Introduce a general "constriction" factor K

$$\mathrm{v}_i \leftarrow K\left(\omega \mathrm{v}_i + \alpha_1 r_1 \circ (\mathrm{p}_i - \mathrm{x}_i) + \alpha_2 r_2 \circ (\mathrm{g} - \mathrm{x}_i)\right)$$

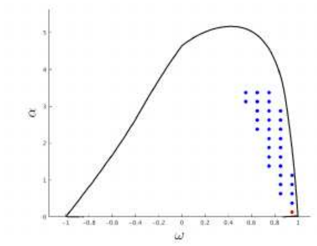
- Start with $\alpha = \alpha_1 + \alpha_2 > 4$ and $K = 1$. Later switch to

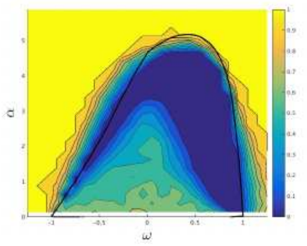
$$K = \frac{2}{\left|2 - \alpha - \sqrt{\alpha^2 - 4\phi}\right|}$$

  e.g. for $\alpha = 4.1 \Rightarrow K = 0.729$, i.e. effectively $\alpha \approx 1.5$
- For $\alpha < 4$ we set $K = 1$
- Other definitions of K are possible, e.g. including ω or based on a more complex model of the algorithm

Implications from numerical experiments

PSO performance for the minimisation of a sphere function $\sum_i x_i^2$ for the relevant pairs (α, ω) with $\alpha = \alpha_1 + \alpha_2$ and $\alpha_1 = \alpha_2$.





- Numerical experiments show that the best results (dots in the left image) are obtained in a region similar to the oscillatory region indicated by the simplified model (see previous slides)
- In contrast to the simplified model we find
  - near $\omega = 1$, good performance is possible only for small α
  - good results are possible also for negative ω, see regions with small average deviations from global optimum (right image)
  - good results are possible also for $\alpha > 4$ for moderate ω

Implications from recent analytical studies

- The solid curve (prev. slide) is an analytical result ($\alpha_1 = \alpha_2$), see Erskine et al., *Swarm Intelligence* **11**, 295-315, 2017.
- It is based on prior research and still based on a simplification
- For $\alpha_1 \neq \alpha_2$, the curve is similar but not identical.
- All parameter pairs within the curve imply an asymptotically stable particle dynamics
- For infinitely many fitness evaluations and an infinitely large search space, parameters on the curve are optimal. For realistic experiments, parameters more inside the curve are preferrable.
- For different fitness functions, different locations near (within) the curve are performing best.
- The difference to earlier approaches lies in analysis of non-trivial noise effects, which appear to be crucial in PSO.

Parameter Settings

Course: Natural Computing (week 5)

(II) Parameters in DE

J. Michael Herrmann
School of Informatics, University of Edinburgh
michael.herrmann@ed.ac.uk, +44 131 6 517177

Differential Evolution (DE) Price & Storn 1997

- As in PSO, individuals are continuous vectors
- Apart from initialisation, there is no direct noisification of the vector.
- The diversity in the population is obtained from “mutations” based on the differences in the population, such that the algorithm is completely self-organising.
- Differences between the vectors in the population
  - point from a poor vector to a good one: a chance for improvement
  - or are traverse between vectors of similar fitness: a chance for increasing or maintaining diversity (“neutral mutation”)
- Essentially only one free parameter

Differential Evolution: Algorithm

Population of N vectors of D-dimensions: x_i, $i = 1, ..., N$

Step 1: $v_i(t+1) = x_q(t) + F \cdot (x_r(t) - x_s(t))$;

q, r, s are random indexes, all different and different from i.
Note that v_i has nothing to do with x_i (t: generation counter)
(In a sense: three parents, but this is considered as mutation in DE)

$F \in [0, 2] \subset \mathbb{R}$ (possible amplification of the differential variation)

Step 2: Choose random numbers $\rho_d \in [0, 1)$, $d \in \{1, \ldots, D\}$

(crossover)

$$u_{id}(t+1) = \begin{cases} v_{id}(t+1) & \text{if } \rho_d < p \\ x_{id}(t) & \text{if } \rho_d \geq p \end{cases}$$

or by choosing a block $d \in [n, (n+L) \bmod D]$, $L \leq D$, $1 \leq n \leq D$, where L is randomly changed.

$$u_i(t+1) = (u_{1i}(t+1), u_{2i}(t+1), \ldots, u_{Di}(t+1))$$

Step 3: $x_i(t+1) = u_i(t+1)$ if $u_i(t+1)$ is better than $x_i(t)$,
otherwise $x_i(t+1) = x_i(t)$ (selection)

Rainer Storn & Kenneth Price (1997) Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization 11: 341–359.

Differential evolution: Parameter values

- In addition to the amplification factor F, there are actually two more parameters
  - the number for vectors N
  - the crossover probability p
- Assumption: No selection (asymptotically, better values are rarely found)
- The variance of the vectors is expected to change in one generation by

$$\langle \mathrm{Var}\left(x_{t+1}\right)\rangle = \left(2F^2 p - \frac{2p}{N} + \frac{p^2}{N} + 1\right) \mathrm{Var}\left(x_t\right)$$

- The variance remains constant on average if F is *critical*:

$$F = \sqrt{\frac{1}{N}\left(1 - \frac{p}{2}\right)}$$

- In good agreement with experimental results, although (in dependence on initialisation, runtime and problem specificity) deviations from the critical value can be useful.

Daniela Zaharie (2002) Critical values for the control parameters of differential evolution algorithms.

Parameter Settings

Course: Natural Computing (week 5)

(III) ACO Convergence and Hypercube Framework

J. Michael Herrmann
School of Informatics, University of Edinburgh
michael.herrmann@ed.ac.uk, +44 131 6 517177

Does the algorithm converge?

Two meanings of convergence

(A) Dynamics comes to a halt

1. GA: All individual in a population are identical
2. PSO: All particles converge to a single point and velocities approach zero
3. ACO: All ants take the same path

(B) Global optimum is found (need to know maximal fitness)

1. GA: one individual has maximal fitness
2. PSO: the absolute difference of the fitness of the best-so-far solution and the maximal fitness is smaller than an appropriate threshold
3. ACO: one ant has maximal fitness

Convergence to global optimum: ACO

We ignore for the moment the dynamics question (A), and ask about the global optimum (B) for ACO (Stützle & Dorigo, 2002)

(B.1) The pheromone trails along the path representing the optimal solutions are larger than on any other solution

(B.2) Probability that an ant finds the globally optimal solution approaches 1 after sufficiently long time

In the following we will assume that only one global optimum exists, and will consider the Min-Max Ant System algorithm with best-ant pheromone update

1. Establishing a pheromone trail

Assume the best path S^* was found by an at time t^*. Let (i,j) be a component in S^*, but (worst case) $\tau_{ij}(t^*) = \tau_{\min}$ and all $(k,l) \notin S^*$ have $\tau_{kl} = \tau_{\max}$. If only the best ant lays pheromones, then the level on S^* will increase within t time steps by

$$\begin{aligned}\tau_{ij}(t^* + t) &= \rho^t \tau_{\min} + \sum_{s=1}^{t} \rho^{s-1} \Delta\tau_{\max} \\ &> t\,\rho^{t-1} \Delta\tau_{\max} = t\,\rho^{t-1}(1-\rho)\,\tau_{\max}\end{aligned}$$

because $\tau_{\max}(t) = \rho^t \tau_{\text{init}} + \sum_{s=1}^{t} \rho^{t-s} \Delta\tau_{\max}$, i.e. $\tau_{\max} = \frac{\Delta\tau_{\max}}{1-\rho}$,

whereas (assuming $\tau_{\min} < \rho^t \tau_{\max}$) $\tau_{kl}(t^* + t) = \rho^t \tau_{\max}$ such that

$$\tau_{ij}(t^* + t) > \tau_{kl}(t^* + t) \;\text{ if }\; t > \frac{\rho}{1-\rho}.$$

2. Finding a global optimum: ACO

Let $P^*(t)$ denote the probability that the best path is found at least once by time t. We need to show that (Stützle & Dorigo, 2002)

$$\forall \varepsilon > 0 \, \exists t : P^*(t) > 1 - \varepsilon.$$

We assume for simplicity that we have a single ant only, the exponent $\alpha = 1$, the local desirability is constant, and each step of the solution has at most K branches.

In the worst case the optimal path has a pheromone level $\tau_{\text{min}} > 0$, the other $K - 1$ branches have τ_{max}. The probability rule gives:

$$p_{\text{min}} = \frac{\tau_{\text{min}}}{\tau_{\text{min}} + (K-1)\,\tau_{\text{max}}}$$

Therefore, we have $P^*(1) \geq p_{\text{min}}^D > 0$, where D is the number of components in the solution. The proof is completed by noticing:

$$P^*(t) \geq 1 - \left(1 - p_{\text{min}}^D\right)^t$$

Convergence to global optimum: Remarks

- $p_{\min}^{D}$ is very small, so the convergence time bound is very large
- A tighter bound is implied for more ants, by considering that there are fewer branches for decisions down the path, and that the pheromones are usually more fortunately distributed, but
- the bound is still exponential.
- Local desirability may reduce complexity, but may also impede convergence to global optimum, if the problem is deceptive
- The update by best ant only and the fact that $\tau_{\min} > 0$ are important (convergence questionable for other ant algorithms)
- See also: W. Gutjahr (2000) A graph-based Ant System and its convergence. Future Generation Computer Systems 16, 873–888.
- A similar proof can be given for GA with mutation rate $p_m > 0$

Convergence to global optimum for PSO?

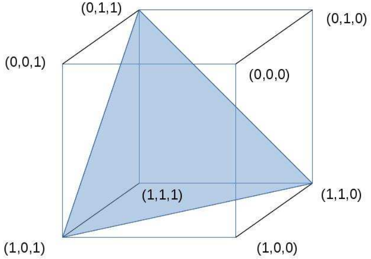
- Statements similar to (B.2) were made by F. v. d. Bergh (2001) and Liu, Abraham & Snasel (2009)
- To show that the algorithm eventually finds the global optimum, we need to assert that a particle can get close with some (possibly very small) probability to every point in state space e.g. by
  - using Gaussian noise, i.e. the forces become $\zeta_1 (\mathrm{p} - \mathrm{x})$ with $\zeta_1 \sim \mathcal{N}\left(\frac{\alpha_1}{2}, \sigma^2\right)$ and $\zeta_2 (\mathrm{g} - \mathrm{x})$ with $\zeta_2 \sim \mathcal{N}\left(\frac{\alpha_2}{2}, \sigma^2\right)$
  - including a random walk to diversify the particle positions
  - choosing parameters such that particles perform independent random movements through all dimensions of the whole search space.
- The noise-based approaches may counteract exploitation, so an appropriate choice of the parameters seems preferable to reach a good level of exploratoriness (see previous parts)

Conclusion on Convergence

- It is relatively easy to show that MHO algorithms can find the global optimum of an arbitrary search problem, but these proofs are not practically useful as they imply an exponentially long run time
- It is more important to find parameter settings that help to speed up the search,
  - a few rules exist how to choose parameter values in general (except for PSO and DE)
  - for a specific problem, practical experiences are needed in order to find optimal parameters (later material on applications)
  - sometimes a higher-order MHO algorithms is employed for the parameter search (later material on hyperheuristic algorithms)

The Hyper-cube framework for ACO

- Given a solution (path) $s = (s_1, \ldots, s_n)$
- The solution is a subsets of the edges E of a graph $G = (N, E)$
- Partitioning of E: if a link belongs to s: 1, otherwise 0
  solution span a hypercube of dimensions $|E|$
- For the TSP, s can be represented by a binary vector with dimension $M = n(n-1)/2 = |E|$ [generally this would be total number of available solution components]



- Pheromones are updated in the span of admissible solutions, as vectors in the volume of an M-dimensional hypercube
- HCF is not an algorithm, but a theoretical framework for ACO

Search space: "Hyper-cube framework"

Pheromone update ($c_j = s_j^i$ if it is a component of solution i)

$$\tau_j \leftarrow \rho\tau_j + \sum_{i=1}^k \Delta\tau_j^i \text{ where } \Delta\tau_j^i = \begin{cases} \frac{1}{f(s^i)} & \text{if } c_j \in s^i \\ 0 & \text{otherwise} \end{cases}$$

$\lim_{t\to\infty} \tau_i(t) \leq \frac{1}{1-\rho} \cdot \frac{k}{f\left(s^{\mathrm{opt}}\right)}$, maximal if all k ants follow forever the optimal solution: $\tau_i = \rho\tau_i + \frac{k}{f\left(s^{\mathrm{opt}}\right)}$

$\tau = (\tau_1, \ldots, \tau_M)$ is an M-dimensional vector: $\tau \in \left[\tau_{\mathrm{min}}, \tau_{\mathrm{max}}\right]^M$

M: number of solution components (e.g. all edges of a graph)

$\tau = \sum \alpha^i s^i$, $\alpha^i \in \left[\tau_{\mathrm{min}}, \tau_{\mathrm{max}}\right]$, $s^i \in \{0,1\}^M$, only used components of s^i being 1, elsewhere 0s.

We can normalise τ such that $\alpha_j \in [0,1]$

Blum, Roli, Dorigo (2001) HC-ACO. 4th Metaheuristics Int. Conf., 399-403.

Hyper-cube framework

Pheromone normalisation

$\tau_j \leftarrow \rho\tau_j + \sum_{i=1}^{k} \Delta\tau_j^i$ where

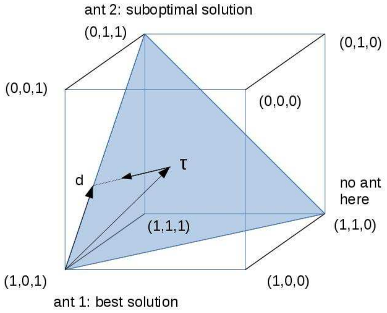
$$\Delta\tau_j^i = \begin{cases} \frac{\frac{1}{f(s^i)}}{\sum_{l=1}^{k} \frac{1}{f(s^l)}} & \text{if } c_j \in s^i \\ 0 & \text{otherwise} \end{cases}$$

Hyper-cube update rule

$$\boldsymbol{\tau} \leftarrow \boldsymbol{\tau} + (1-\rho)\,(\mathrm{d} - \boldsymbol{\tau})$$

$\mathrm{d} = (d_1, \ldots, d_M)$ where

$d_j = \sum_{i=1}^{k} \Delta\tau_j^i,\ j = 1, \ldots, M$



The pheromone vector moves by $(1-\rho)$ towards the weighted mean of the solutions produced by the current iteration.

Benefits of the Hyper-cube framework

- Probabilistic interpretation
- Proof that expected quality of solutions strictly increases (without the assumption of an infinitesimal step size as in standard gradient methods!)
- A diversification scheme
    - global desirability: $v_j^{\text{des}} \leftarrow \max\left\{\frac{1}{f(s)} : s \in S_{\text{ants}}, s_j = 1\right\}$
    - global frequency: $v_j^{\text{fr}} \leftarrow \sum_{s \in S_{\text{ants}}} s_j$

      S_{ants} : all solutions generated since the start
    - At stagnation the algorithm may be restarted with a pheromone matrix constructed from v^{des} or the regularised inverse of v^{fr} in order to keep good solutions, but also to favour regions where few ants have been before.
- The HC update rule can be used to adapt ρ such that the expected variance of τ remains constant (depends on fitness!)

Dorigo & Blum (2005) Ant colony optimization theory: A survey. Theoretical Computer Science 344, 243-278.

Conclusion on Parameter setting

- It is relatively easy to show that MHO algorithms can find the global optimum of an arbitrary search problem, but these proofs are not practically useful as the imply an exponentially long run time
- It is more important to find parameter settings that help to speed up the search,
  - For some algorithms, rules exist how to choose parameter values in general
  - for a specific problem, practical experiences are needed in order to find optimal parameters
  - sometimes a higher-order MHO algorithms is employed for the parameter search

Differential Evolution

Course: Natural Computing

J. Michael Herrmann
School of Informatics, University of Edinburgh
michael.herrmann@ed.ac.uk, +44 131 6 517177

Differential Evolution (DE) Price & Storn 1997

- As in PSO, individuals are continuous vectors
- Apart from initialisation, there is no direct noisification of the vectors.
- The diversity in the population is obtained from “mutations” based on the differences in the population, such that the algorithm is fully self-organising.
- Differences between the vectors in the population
  - point from a poor vector to a good one: a chance for improvement
  - or are traverse between vectors of similar fitness: a chance for increasing or maintaining diversity (“neutral mutation”)
- Essentially only one free parameter (F, in addition to p_d, N)
- DE is an interesting algorithm that is often used as a component in hybrid metaheuristic algorithms.

Differential Evolution: Algorithm

Population of N vectors of D-dimensions: x_i, $i = 1, ..., N$

Step 1: $v_i(t+1) = x_q(t) + F \cdot (x_r(t) - x_s(t))$;

q, r, s are random indexes, all different and different from i.
Note that v_i has nothing to do with x_i (t: generation counter)
(In a sense: three parents, but this is considered as mutation in DE)

$F \in [0, 2] \subset \mathbb{R}$ (possible amplification of the differential variation)

Step 2: Choose random numbers $\rho_d \in [0, 1)$, $d \in \{1, \ldots, D\}$
(crossover)

$$u_{id}(t+1) = \begin{cases} v_{id}(t+1) & \text{if } \rho_d < p \\ x_{id}(t) & \text{if } \rho_d \geq p \end{cases}$$

or by choosing a block $d \in [n, (n+L) \bmod D]$, $L \leq D$, $1 \leq n \leq D$, where L is randomly changed.

$$u_i(t+1) = (u_{1i}(t+1), u_{2i}(t+1), \ldots, u_{Di}(t+1))$$

Step 3: $x_i(t+1) = u_i(t+1)$ if $u_i(t+1)$ is better than $x_i(t)$,
otherwise $x_i(t+1) = x_i(t)$ (selection)

Rainer Storn & Kenneth Price (1997) Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization 11: 341–359.

Differential evolution: Parameter values

- In addition to the amplification factor F, there are actually two more parameters
  - the number for vectors N
  - the crossover probability p_d
- Assumption: No selection (asymptotically, better values are rarely found)
- The variance of the vectors is expected to change in one generation by

$$\langle \mathrm{Var}\left(x_{t+1}\right)\rangle = \left(2F^2 p_d - \frac{2p_d}{N} + \frac{p_d^2}{N} + 1\right) \mathrm{Var}\left(x_t\right)$$

- The variance remains constant on average if F is *critical*:

$$F_{\text{crit}} = \sqrt{\frac{1}{N}\left(1 - \frac{p_d}{2}\right)}$$

- In good agreement with experimental results, although (in dependence on initialisation, runtime and problem specificity) deviations from the critical value can be useful ($F \leq F_{\text{crit}}$).

Daniela Zaharie (2002) Critical values for the control parameters of differential evolution algorithms.