Opportunistic Evolutionary Method to Minimize a Sum of Squares of Nonlinear Functions

Mikhail Zhabitsky

Joint Institute for Nuclear Research, Dubna

GRID 2016, Dubna, July 4–9

Contents

Minimization of real-parameter functions

Search for a vector $\boldsymbol{x}^* = \{x_j\}\big|_{j=0,\ldots D-1}$,
which minimizes an objective function $f(\boldsymbol{x})$:

$$f\left(\boldsymbol{x}^*\right) \leqslant f(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \Omega,$$

where $\Omega \subset \mathbb{R}^D$ is a search domain.

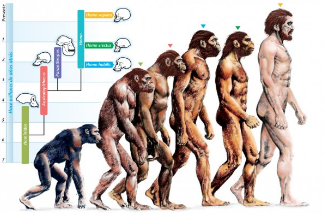
$$\boldsymbol{x}^* = \mathrm{Argmin}\ f(\boldsymbol{x}); \quad f(\boldsymbol{x}) :\ \mathbb{R}^D \mapsto \mathbb{R}.$$

Additional obstacles for minimization:

- Constraints on variables $\varphi(\boldsymbol{x}) < 0$
- Multidimensional parameter space $D = 10 \ldots 100$
- Multimodal objective functions
- Function derivatives are not available or useless
- Objective functions with “noise”
- Computationally expensive objective functions

Natural evolution

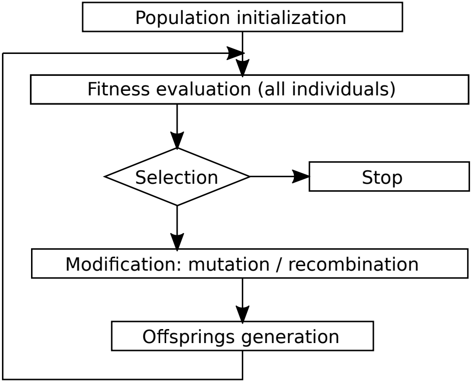
- Natural selection — organisms with favorable traits have a higher chance to survive and reproduce [Darwin]
- DNA — encoding of genetic information of the individual organism [Watson & Crick]
- Population genetics — modifications in offsprings as a result of mutations and recombinations and gene flow within the population [Mendel]



Result of the natural evolution — a population of individuals with increased fitness with respect to the external factors ("selection function")

Evolutionary algorithms

Evolutionary algorithm (EA) — an algorithm, which implements operators similar to processes in Natural evolution:



Classical Differential Evolution

- Evolutionary algorithm (EA) — an algorithm, which implements operators similar to processes in Natural evolution: selection, mutations and crossover of genes
- Classical Differential Evolution (DE) is an Evolutionary Algorithm with specific *mutation*: $\boldsymbol{v} = \boldsymbol{x}_r + F(\boldsymbol{x}_p - \boldsymbol{x}_q)$
- Invented in 1995
  [K. Price, R. Storn// J. Global of Optimization 11 (1997) 341]
- Use *population* of vectors (population size N_p)
- Every population member is a vector in continuous space $\Omega \subset \mathbb{R}^D$

[K. Price, R. Storn, J.A. Lampinen "Differential evolution — A Practical Approach to Global Optimization", Springer, 2005]
[S. Das, P.N. Suganthan// IEEE Trans. Evol. Comp. 15 (2011) 4]

Asynchronous Differential Evolution (ADE)

ADE is a steady-state variant of the classical DE

[E. Zhabitskaya, M. Zhabitsky// LNCS 7125 (2012) 328]

// population initialization $\{\boldsymbol{x}_i\}|_{i=0,\ldots N_p-1}$, $\boldsymbol{x}_i = \{x_{i,j}\}|_{j=0,\ldots D-1}$

do {

$i = choose\_target\_vector()$; // target vector choice i

// Mutation:

$\boldsymbol{v}_i = \boldsymbol{x}_r + F(\boldsymbol{x}_p - \boldsymbol{x}_q)$; // mutant vector, $r \neq p \neq q$ — random indices

// Crossover (recombination):

for ($j = 0$; $j < D$; $j = j + 1$)

$$u_{i,j} = \begin{cases} v_{i,j} & \text{rand}(0,1) < C_r \text{ or } j = j_{\text{rand}} \\ x_{i,j} & \text{otherwise} \end{cases} \quad \text{// trial vector}$$

// Selection:

if $(f(\boldsymbol{u}_i) < f(\boldsymbol{x}_i))$

$\boldsymbol{x}_i = \boldsymbol{u}_i$;

} while (termination criteria not met);

Asynchronous Differential Evolution: few parameters

ADE is a steady-state variant of the classical DE

[E. Zhabitskaya, M. Zhabitsky// LNCS 7125 (2012) 328]

// population initialization $\{\boldsymbol{x}_i\}|_{i=0,\dots N_p-1}$, $\boldsymbol{x}_i = \{x_{i,j}\}|_{j=0,\dots D-1}$
do {
  $i = choose\_target\_vector()$; // target vector choice i
  // Mutation:
  $\boldsymbol{v}_i = \boldsymbol{x}_r + F(\boldsymbol{x}_p - \boldsymbol{x}_q)$; // mutant vector, $r \neq p \neq q$ — random indices
  // Crossover (recombination):
  for ($j = 0$; $j < D$; $j = j + 1$)

$$u_{i,j} = \begin{cases} v_{i,j} & \mathrm{rand}(0,1) < C_r \text{ or } j = j_{\mathrm{rand}} \\ x_{i,j} & \text{otherwise} \end{cases} \quad \text{// trial vector}$$

  // Selection:
  if $(f(\boldsymbol{u}_i) < f(\boldsymbol{x}_i))$
    $\boldsymbol{x}_i = \boldsymbol{u}_i$;
} while (termination criteria not met);

Asynchronous DE: Classification

mutation: $\boldsymbol{v}_i = \boldsymbol{x}_r + F(\boldsymbol{x}_p - \boldsymbol{x}_q)$

$\boldsymbol{x}_i$ — target vector
$\boldsymbol{x}_r$ — base vector
$(\boldsymbol{x}_p - \boldsymbol{x}_q)$ — difference vector

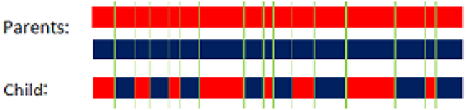
DE/w/x/y/z is classified according to Mutation and Crossover operators:
w corresponds to a target vector to be replaced;
x — a base vector;
y — number of difference vectors;
z — crossover type (binomial or exponential).

rand/rand/1/bin	$\boldsymbol{v}_{\text{rand}} = \boldsymbol{x}_{\text{rand}} + F(\boldsymbol{x}_p - \boldsymbol{x}_q)$
rand/best/1/bin	$\boldsymbol{v}_{\text{rand}} = \boldsymbol{x}_{\text{best}} + F(\boldsymbol{x}_p - \boldsymbol{x}_q)$
worst/best/bin	$\boldsymbol{v}_{\text{worst}} = \boldsymbol{x}_{\text{best}} + F(\boldsymbol{x}_p - \boldsymbol{x}_q)$

Uniform (Binomial) Crossover with fixed C_r

Combines a *mutant* vector $\boldsymbol{v}$ with a *target* vector $\boldsymbol{x}$ into a *trial* vector $\boldsymbol{u}$:

$$u_{i,j} = \begin{cases} v_{i,j} & \text{rand}(0,1) < C_r \text{ or } j = j_{\text{rand}} \\ x_{i,j} & \text{otherwise.} \end{cases}$$



Receipes for a *crossover rate* C_r:

$C_r = 0$ is suitable for separable problems

$C_r = 1$ is for non-separable problems (also rotationally-invariant)

$C_r = 0.9$ is usually used

JADE Crossover rate C_r adaptation

JADE scheme to adapt C_r

[J. Zhang and A.C. Sanderson, IEEE TEC 13 (2009) 945]

$$C_{r\,i}' = N(\mu_c, \sigma_c = 0.1) \quad \text{truncated to } [0, 1]$$

Mean μ_C is updated after *successful* iterations:

$$\mu_c' = (1 - c_c)\mu_c + c_c \langle C_{r\,i} \rangle$$
$$\mu_c^0 = 0.5$$

ADE with Adaptive Correlation Matrix

ADE-ACM: [E. Zhabitskaya, M.Zhabitsky, GECCO-2013]

DE is known to adapt to the objective function landscape:
[K. Price, R. Storn, J.A. Lampinen, Springer, 2005]

Population is a sample to test pairwise correlations:

$$q_{jk} = \frac{1}{N_p-1} \sum_{i=0}^{N_p-1} (x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k),$$

$$s_{jk} = \frac{q_{jk}}{\sqrt{q_{jj} q_{kk}}},$$

Adaptive correlation matrix after *successful* iterations:

$$C' = (1-c)C + cS.$$

[A. Auger, N. Hansen, CMA-ES, IEEE CEC (2005) 1769]

ADE with Adaptive Correlation Matrix (II)

Example: composite function

$$f = f_{\text{Rosenbrock}}(x_0, x_1) + f_{\text{Rosenbrock}}(x_2, x_3)$$

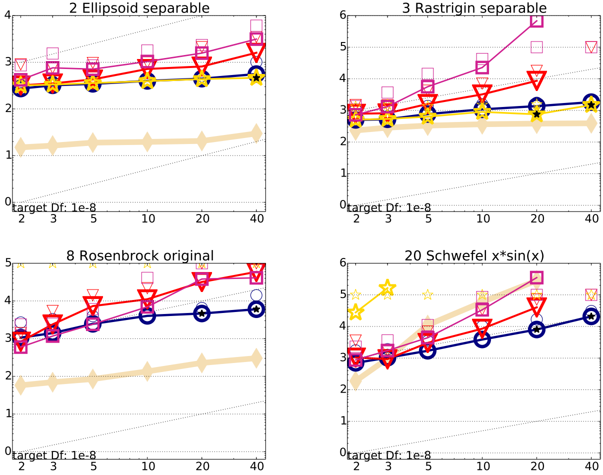
$$C = \left(\begin{array}{cc|c|c} C_{00} & 0.910 & 0.027 & 0.038 \\ 0.910 & C_{11} & 0.016 & 0.015 \\ 0.027 & 0.016 & \mathbf{C_{22}} & 0.945 \\ 0.038 & 0.015 & \mathbf{0.945} & C_{33} \end{array}\right)$$

Select some coordinate m and a threshold c_{thr}:

$$m = [D\mathrm{rand}(0,1)], \quad c_{\text{thr}} = \mathrm{rand}(0,1);$$
$$\mathbb{I}_m = \{\forall j : |c_{mj}| > c_{\text{thr}}\}.$$

$$u_{i,j} = \begin{cases} v_{i,j} & \text{if } j \in \mathbb{I}_m \\ x_{i,j} & \text{otherwise.} \end{cases} \Rightarrow \text{Crossover within the selected subspace}$$

Crossover: numerical tests



○: ADE-ACM
▽: JADE
⋆: $C_r = 0.0$
□: $C_r = 0.9$

rand/rand/1
N_p adaptation

[N. Hansen et al., BBOB-2012 test bench]

Partly-separable problems

$$D = 20: \quad f(\boldsymbol{x}) = f_{\mathrm{Ros}}(x_0, x_1) + f_{\mathrm{Ros}}(x_2, \ldots, x_4) + f_{\mathrm{Ros}}(x_5, \ldots, x_8) + f_{\mathrm{Ros}}(x_9, \ldots, x_{13}) + f_{\mathrm{Ros}}(x_{14}, \ldots, x_{19})$$

DE/rand/rand/1 strategies:

	C_r	P_{succ}	mean	median	std.dev
bin	0	0	—	—	—
bin	0.9	1	4.93e+05	4.14e+05	1.95e+05
bin	JADE	0.44	1.19e+07	—	1.11e+06
acm	—	1	6.82e+04	6.57e+04	1.74e+04

Population size N_p adaptation

	small N_p	large N_p
Probability to locate a minimum	$-$	$+$
Convergence rate	$+$	$-$

ADE with restarts [Zh&Zh// LNCS 2013]:

- Small initial population size $N_p^{\min} = 10$
- Restart with larger $N_p \leftarrow kN_p$ (after stagnation)
- Switch to $N_p^{\min}$ if $N_p^{\max} = 20D$ was exceeded

Stagnation criteria used to solve following test problems:

- $\exists j \quad \Delta x_j < \varepsilon_x \max\limits_i \{|x_{i,j}|\}$.
- $\Delta f < \varepsilon_f \max\limits_{i=0,\ldots,N_p-1} \{|f_i|\}$.

Scale factor F adaptation

$$\text{mutation:} \qquad \boldsymbol{v}_i = \boldsymbol{x}_r + F(\boldsymbol{x}_p - \boldsymbol{x}_q)$$

$F > F_{\text{thr}}$ to prevent *premature convergence*

[D. Zaharie, 2002]

[E. Zhabitskaya, LNCS 7125 (2013) 322]

JADE scheme to adapt F

[J. Zhang and A.C. Sanderson, IEEE TEC 13 (2009) 945]

$$F_i' = \text{Cauchy}(\mu_F, \sigma_F = 0.1)$$

If $F_i' < F_{\min} = 0.1$, it is regenerated,
If $F_i' > F_{\max} = 1.0$, then $F_i' \hookleftarrow F_{\max}$

Location parameter μ_F is updated after *successful* iterations:

$$\mu_F' = (1 - c_F)\mu_F + c_F L_2(\{F\}) = (1 - c_F)\mu_F + c_F \frac{\sum F_i^2}{\sum F_i}$$

$$\mu_F^0 = 0.5$$

Strategies

mutation: $\boldsymbol{v}_i = \boldsymbol{x}_r + F(\boldsymbol{x}_p - \boldsymbol{x}_q)$

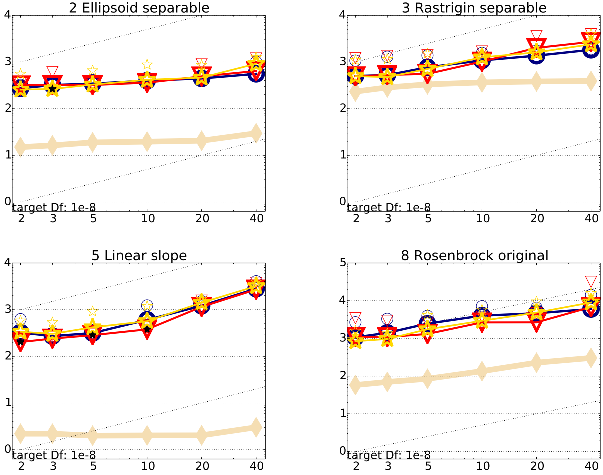
$\boldsymbol{x}_i$ — target vector
$\boldsymbol{x}_r$ — base vector
$(\boldsymbol{x}_p - \boldsymbol{x}_q)$ — difference vector

rand/rand/1	$\boldsymbol{v}_i = \boldsymbol{x}_r + F(\boldsymbol{x}_p - \boldsymbol{x}_q)$
rand/current-to-pbest/1	$\boldsymbol{v}_i = \vec{x}_i + F(\vec{x}^p_{\text{best}} - \vec{x}_i) + F(\vec{x}_r - \vec{x}_q)$ [J. Zhang and A.C. Sanderson, JADE, IEEE TEC 13 (2009) 945]
lw/current-to-pbest/1	$\boldsymbol{v}_{lw} = \vec{x}_{lw} + F(\vec{x}^p_{\text{best}} - \vec{x}_{lw}) + F(\vec{x}_r - \vec{x}_q)$

$r, p, q \in P$ — random individuals from the population P

Use an archive A to produce a directional difference vector: $q \in P \cup A$

Strategies: numerical tests

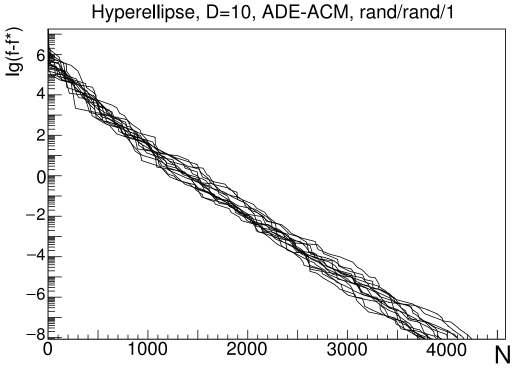


○: rand/rand/1
▽: rand/current-to-pbest/1
⋆: lw/current-to-pbest/1

no archive
N_p adaptation

[N. Hansen et al., BBOB-2012 test bench]

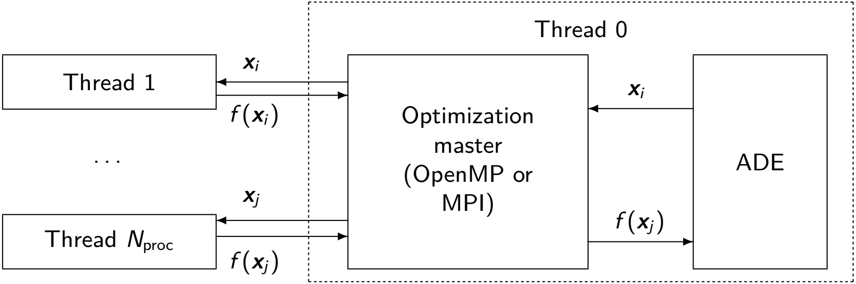
Convergence rate



$$f = \sum c_j(x_j - o_j)^2$$

- log-linear convergence
- No matrix inversions
- Algorithm internal complexity $O(D^2)$

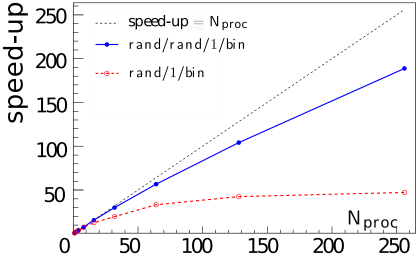
Scheme of parallelization (ADE)



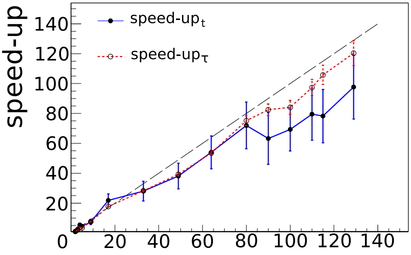
- Master/Workers model (coarse-grained parallelization)
- Complete and efficient use of all nodes
- OpenMP and MPI (MPICH2) C++ realizations

[E. Zhabitskaya, M. Zhabitsky, Math. model. 24 (2012) 33]

Speed-up on parallel systems



[E. Zhabitskaya, M. Zhabitsky // LNCS 7125 (2012) 328]



[E. Zhabitskaya, et al. // Math. model. 27 (2015) 58]

Non-linear Least Squares

Search for a vector $\boldsymbol{x}^* = \{x_j\}|_{j=0,\dots D-1}$,
which minimizes an objective function $f(\boldsymbol{x})$:

$$f(\boldsymbol{x}^*) \leqslant f(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \Omega \subset \mathbb{R}^D, \quad \text{where } f(\boldsymbol{x}) = \sum_{j=0}^{j<M} f_j^2(\boldsymbol{x}).$$

$$\boldsymbol{x}^* = \mathrm{Argmin}\ f(\boldsymbol{x}); \quad f(\boldsymbol{x}) = \sum_{j=0}^{j<M} f_j^2(\boldsymbol{x}); \quad f(\boldsymbol{x}) :\ \mathbb{R}^D \mapsto \mathbb{R}^M.$$

Additional obstacles for minimization:

- Multimodal objective functions
- Function derivatives are not available or useless

$$\text{Weighted Least Squares:} \quad \chi^2 = \sum_j \frac{(y_j - t(\boldsymbol{x}))^2}{\sigma_j^2}$$

Note: the Gauss–Newton or the Levenberg–Marquardt algorithms for a local minimum problem

How to solve NLLS?

$$\text{Naïve example:} \quad f(\boldsymbol{x}) = \sum_j f_j^2(x_j) = \sum_j c_j x_j^2.$$

Approach within Differential Evolution:

1. Set $C_r = 0$ (separable problem)
2. Modify a selected x_j, see improvement in $f = \sum f_j^2$

$\Rightarrow$ Coordinate descent: convergence rate $O(D^{-1})$

More efficient approach:

1. Modify all! x_j, see improvements in f_j
2. Combine $\boldsymbol{x}$ from coordinates x_j, which provides better f_j

$\Rightarrow$ Convergence rate will be (almost) independent from a problem's dimension D: $O(1)$

Heuristic approach to solve NLLS

Adaptive correlation matrices for $\mathrm{corr}(x_i, x_j)$ and $\mathrm{corr}(x_i, f_m)$:

$$C' = (1-c)C + cS.$$

1. Select a threshold $c_{\mathrm{thr}} = \mathrm{rand}(0,1)$
2. Select some coordinate m_k and identify a set of variables $\mathbb{I}_k = \{\forall j : |\mathrm{corr}(x_{m_k}, x_j)| > c_{\mathrm{thr}}\}$
3. Identify a set of functions $\mathbb{F}_k = \{\forall j, l : j \in \mathbb{I}_k,\ |\mathrm{corr}(x_j, f_l)| > c_{\mathrm{thr}}\}$
4. Identify a set of variables $\mathbb{I}_k^{ext} = \{\forall j, l : l \in \mathbb{F}_k,\ |\mathrm{corr}(x_j, f_l)| > c_{\mathrm{thr}}\}$
5. Repeat steps start from 2 within $(\mathbb{I} - \mathbb{I}_k^{ext})$ until all variables are exhausted

Trial vector: $u_{i,j} = \begin{cases} v_{i,j} & \text{if } j \in \bigcup \mathbb{I}_k \\ x_{i,j} & \text{otherwise.} \end{cases}$ $\Rightarrow$ Crossover within selected subspaces

Synthetic trial vector: $t_{i,j} = \begin{cases} u_{i,j} & \text{if } \sum_{\mathbb{F}_k} f_l(u_{i,j}) < \sum_{\mathbb{F}_k} f_l(x_{i,j}), \quad j \in \mathbb{I}_k \\ x_{i,j} & \text{otherwise.} \end{cases}$

Separable block problems: results

$$f(\boldsymbol{x}) = \sum_{j=0}^{D-1} f_j^2(x_j) = \sum_{j=0}^{D-1} c_j x_j^2.$$

Median number of function evaluations

D	ADE-ACM	Opportunistic ADE-ACM	Idealistic ADE-ACM
10	2486	2116	
100	76482	13365	
1000	$> 10^6$	22243	

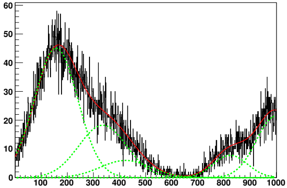
Separable block problems: results

$$f(\boldsymbol{x}) = \sum_{j=0}^{D-1} f_j^2(x_j) = \sum_{j=0}^{D-1} c_j x_j^2.$$

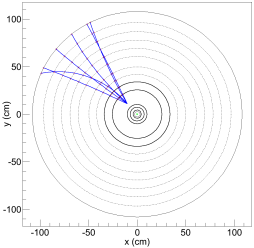
Median number of function evaluations

D	ADE-ACM	Opportunistic ADE-ACM	Idealistic ADE-ACM
10	2486	2116	596
100	76482	13365	1663
1000	$> 10^6$	22243	1893

Applications



- Curve-fitting, in particular peak-finding



- Tracking (reconstruction of particle trajectories)

Conclusions

- Opportunistic Asynchronous Differential Evolution with Adaptive Correlation Matrix is an efficient algorithm to solve partly-separable block non-linear least squares problems
- Opportunistic Asynchronous DE with ACM is (quasi) parameter-free
- Opportunistic ADE has low internal complexity $O(D^2)$
- (Almost) Linear speedup on parallel systems (up to 50 nodes)