Algorytmy ewolucyjne wspomagane metamodelami – stan wiedzy i wyzwania na przyszłość

Mateusz Zaborski
M.Zaborski@mini.pw.edu.pl

Faculty of Mathematics and Information Science
Warsaw University of Technology

16.11.2022

Table of Contest

Background

Continuous black box optimization problem

- Goal - minimize an objective function (or many objective functions):

$$f : \mathbb{R}^D \to \mathbb{R}$$

- Black Box scenario
  - Function values of evaluated search points are the only accessible information
  - Gradients are not available

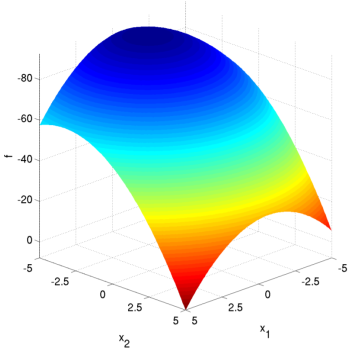


Figure: Sphere 2D function from COCO BBOB benchmark [5]

Continuous black box optimization problem

Challenges:

- Infinite number of solutions in a continuous domain
- Multidimensional problems are difficult for grid search
- The goal is to find a global (not local) optimum
- Unknown function shape
  - Non-linear, non-quadratic
  - Discontinuities, sharp ridges
  - Non-separability
  - Ill-conditioning
- Unknown optimum

Evolutionary Algorithms

Evolutionary Algorithms (+ Swarm Algorithms):

- Versatile solving methods for black-box problems
- Effective and well-researched
- Exploration and exploitation ability
- Low complexity (in non-surrogate-assisted solutions)

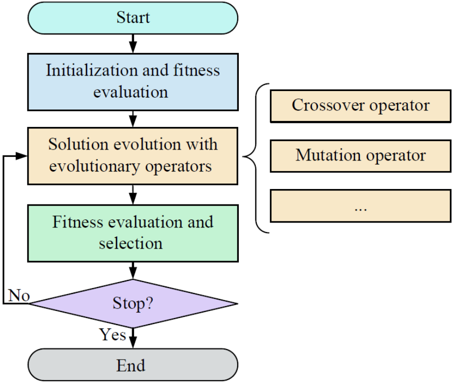


Figure: General flowchart of evolutionary computation [11]

Evolutionary Algorithms

Main groups:

- **CMA-ES-based (e.g. CMA-ES, IPOP-CMA-ES, BIPOP-CMA-ES)**
- **DE-based (e.g. jDE, JADE, SHADE, L-SHADE)**
- Swarm-based (e.g. PSO)
- Mix of EA and swarm (e.g. RFO)
- Hybrids
- Others

Evolutionary Algorithms

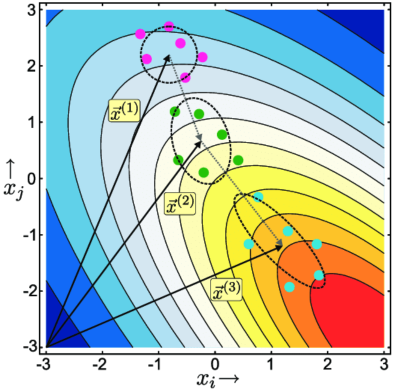


Figure: CMA-ES principle [16]

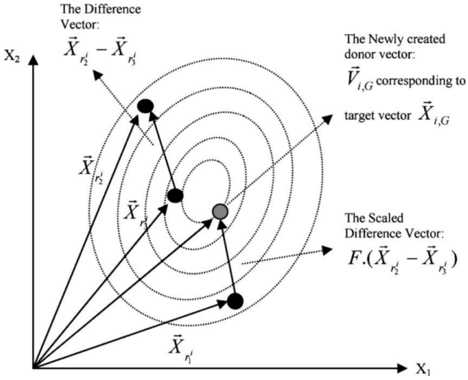


Figure: DE principle [2]

Expensive optimization

Expensive optimization

Expensive optimization assumes a significant time (cost) required for a single fitness function evaluation.

Examples:

- Physical/chemical experiment
- Oil exploration
- Crash test
- Aerodynamic design
- Simulation
- Hyperparameter tuning

Expensive optimization

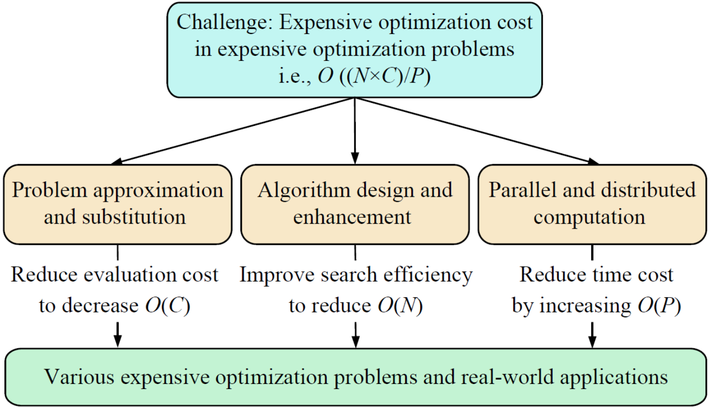


Figure: Three directions to reduce expensive optimization costs [11]

Expensive optimization

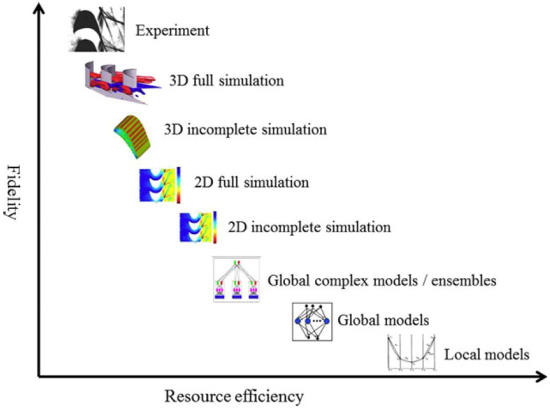


Figure: Trade-off between accuracy and computational cost [11]

Expensive optimization

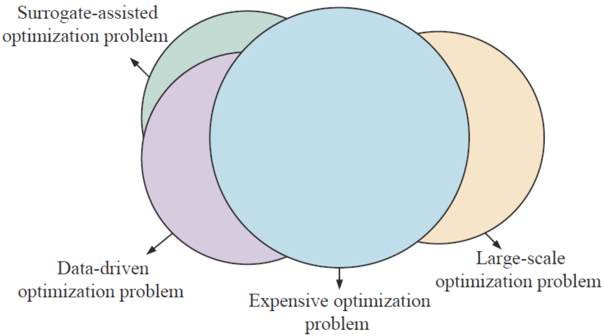


Figure: Relationship between expensive optimization problem and some relevant optimization problems [11]

Metamodels

Metamodel

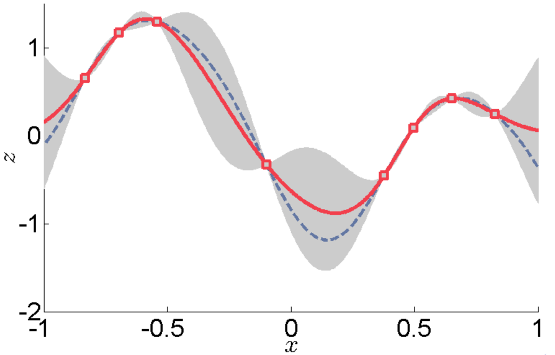
Metamodel (or surrogate model) mimics fitness function as closely as possible while being (computationally) cheaper to evaluate.

Outcome:

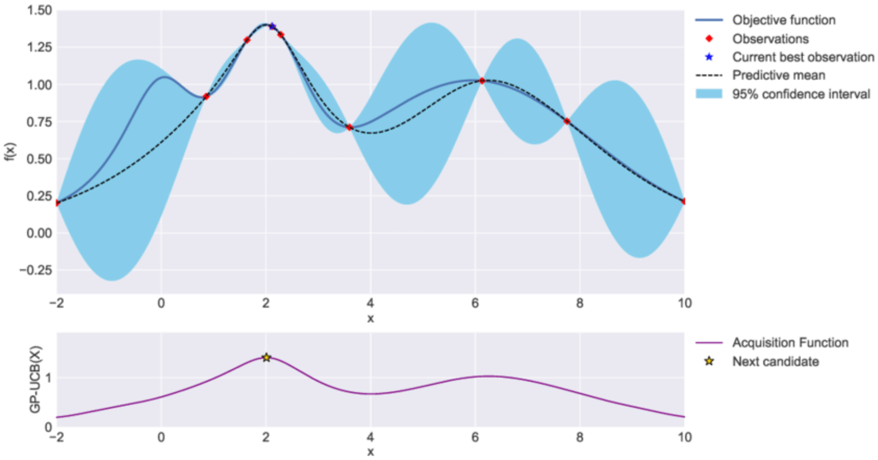
- Optimization process is cheaper
- Final result is more accurate
- Fitness function has its representation in the model

Kriging [12]

$$\hat{f}(\boldsymbol{x}) = \mu(\boldsymbol{x}) + \epsilon(\boldsymbol{x}) \tag{1}$$



Bayesian Optimization [10]



Polynomial Regression [14]

$$\hat{f} = \tilde{\boldsymbol{x}}^T \boldsymbol{\beta} \tag{2}$$

- Linear: $\tilde{\boldsymbol{x}} = [x_{i,2}, \ldots, x_{i,D}, 1]$
- Quadratic: $\tilde{\boldsymbol{x}} = [x_{i,1}^2, \ldots, x_{i,D}^2]$
- Interactions: $x_{i,d} \cdot x_{i,d'}$, where $d, d' \in \{1, \ldots, D\}$ and $d \neq d'$
- p-order PR: $\tilde{\boldsymbol{x}} = [x_{i,1}^p, \ldots, x_{i,D}^p]$

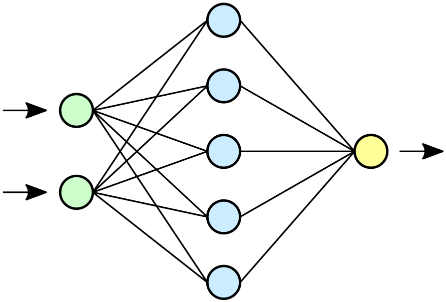
Radial Basis Function [15]

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{K} w_i h_i(\boldsymbol{x}) \tag{3}$$

- Linear: $h(r) = r$
- Cubic: $h(r) = r^3$
- Gaussian: $h(r) = e^{-\beta r^2}$, where β is a parameter
- Multiquadric: $h(r) = \sqrt{r^2 + \beta^2}$, where β is a parameter
- Thin plate spline: $h(r) = r^2 \ln r$

Artificial Neural Network [6]

$$\hat{f}(\boldsymbol{x}, \boldsymbol{w}) = f_a \left(w_0 + \sum_{j=1}^{L} w_j x_j \right) \qquad (4)$$



Other metamodels

Other metamodels:

- Support Vector Regression
- Generalized Additive Model
- Random Forest
- Kernel Partial Least Squares Regression model
- k-Nearest Neighbors Regression

Surrogate-Assisted Evolutionary Algorithms

Key observations

Key observations:

- SAEAs are popular and well-researched methods [11, 9, 8]
- First SAEAs in late 80s [3]
- Designed mainly for expensive optimization
- Many of SAEAs are Kriging-based
- Lack of proper benchmarking
- Lack of performance analysis (e.g. fixed small budget)
- Non-complex algorithms are easy to improve

Key observations

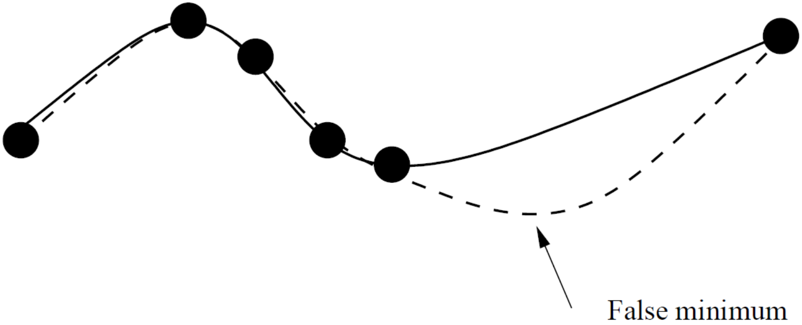


Figure: An example of a false minimum in the approximate model [7]

SAEAs classificasion

Modeling space:

- Global
- Local
- Ensemble

Scope:

- Individual
- Iteration
- Population

Integration method:

- Promising candidates selection (for evaluation or no evaluation)
- Modifying algorithm's mechanisms
- Direct optimization of the metamodel

Training set size:

- Limited
- Increasing

lq-CMA-ES [4]

Algorithm 1 Determine Population Surrogate Values

Require: A population $X = \boldsymbol{x}_1, \ldots, \boldsymbol{x}_\lambda$, a model $\mathcal{M}$ with a data queue of at most $\max(\lambda, 2\mathrm{df}_{\max})$ pairs $(\boldsymbol{y}_i, f(\boldsymbol{y}_i))$, and a fitness function f

$k \leftarrow \lfloor 1 + \max(\lambda \times 2\%, 3/0.75 - |\mathcal{M}|) \rfloor$ # incrementing evaluations
while $|X| > 0$ **do** # while not all elements are added to $\mathcal{M}$
  drop the $k - (\lambda - |X|)$ $\mathcal{M}$-best elements from X into $\mathcal{M}$
  sort the newest $\min(k, \lambda)$ elements in $\mathcal{M}$ w.r.t. f # last = best
  $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_j \leftarrow$ the last $\max(15, \min(1.2k, 0.75\lambda))$ elements in $\mathcal{M}$
  if Kendall-$\tau([\mathcal{M}(\boldsymbol{y}_i)]_i, [f(\boldsymbol{y}_i)]_i) \geq 0.85$ **then**
    break while
  $k \leftarrow \lceil 1.5k \rceil$
if $|X| > 0$ **then**
  return $\mathcal{M}(\boldsymbol{x}_1), \ldots, \mathcal{M}(\boldsymbol{x}_\lambda)$ all offset by
    $\min_{\boldsymbol{x} \in \text{last } k \text{ elements of } \mathcal{M}}(f(\boldsymbol{x})) - \min_{i=1\ldots\lambda}(\mathcal{M}(\boldsymbol{x}_i))$
else
  return $f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_\lambda)$

lq-CMA-ES [4]

$$z_{\mathrm{lin}} : \boldsymbol{x} \mapsto [1, x_1, x_2, \ldots, x_n]^\top \tag{3}$$

$$z_{\mathrm{quad}} : \boldsymbol{x} \mapsto [z_{\mathrm{lin}}(\boldsymbol{x})^\top, x_1^2, \ldots, x_n^2]^\top \tag{4}$$

$$z_{\mathrm{full}} : \boldsymbol{x} \mapsto [z_{\mathrm{quad}}(x)^\top, x_1x_2, x_1x_3, \ldots, x_1x_n, x_2x_3, \ldots x_2x_n, x_3x_4, \ldots, x_{n-1}x_n]^\top \quad . \tag{5}$$

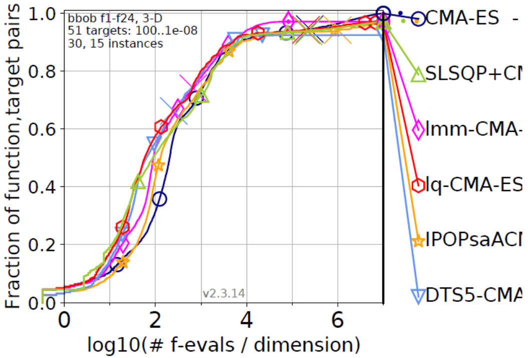
We switch to the next model when the amount of used data exceeds the degrees of freedom of the next model plus 10%.

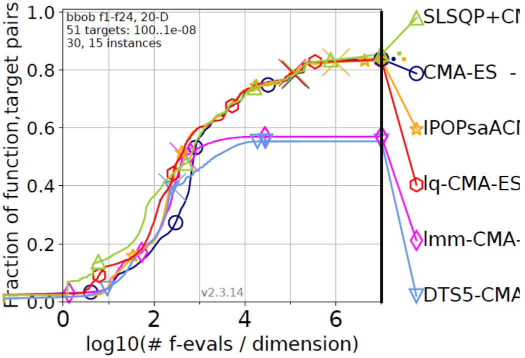
Results of lq-CMA-ES [4]

lq-CMA-ES was compared with:

- Baseline CMA-ES
- SLSQP+CMA-ES
- DTS-CMA-ES (Kriging-based but limited training size)
- lmm-CMA-ES (quadratic PR with interactions as a local metamodel)
- IPOPsaACM (ranking SVM for the entire iteration)

Results of lq-CMA-ES [4]





psLSHADE overview

psLSHADE [18]: LSHADE extended with:

- An archive of samples (N_a best-so-far solutions collected)
- A prescreening metamodel

psLSHADE prescreening

Algorithm psLSHADE main loop high–level pseudocode

1: Generate randomly $N^g \cdot N_s$ mutated vectors $\boldsymbol{v}_i^{g,j}$ ▷ Mutation phase
2: Generate randomly $N^g \cdot N_s$ trial vectors $\boldsymbol{u}_i^{g,j}$ ▷ Crossover phase
3: Estimate meta-model parameter values ▷ Meta-model estimation using Ordinary Least Squares
4: Calculate $N^g \cdot N_s$ surrogate values $f^{surr}(\boldsymbol{u}_i^{g,j})$
5: For each individual i designate the best trial vector $\boldsymbol{u}_i^{g,best}$ ▷ Meta-model prescreening
6: **for** i = 1 to N **do**
7: Do selection of $\boldsymbol{u}_i^{g,best}$ ▷ Selection phase
8: Add $\boldsymbol{u}_i^{g,best}$ and $f(\boldsymbol{u}_i^{g,best})$ to the archive ▷ Archive update
9: **end for**
10: Update memory with values of successful parameters ▷ Parameter adaptation
11: Decrease population size N^g if needed ▷ Population size reduction
12: $g = g + 1$

psLSHADE metamodel

Table: A description of transformations and the final meta-model (mm.)

Name	Form	DoF
Constant	$X_c = [1]$	$df_c = 1$
Linear	$X_l = [x_1, \ldots, x_D]$	$df_l = D$
Quadratic	$X_q = [x_1^2, \ldots, x_D^2]$	$df_q = D$
Interactions	$X_i = [x_1 x_2, \ldots, x_{D-1} x_D]$	$df_i = \frac{D(D-1)}{2}$
Inv. linear	$X_{il} = [\frac{1}{x_1}, \ldots, \frac{1}{x_D}]$	$df_{il} = D$
Inv. quad.	$X_{iq} = [\frac{1}{x_1^2}, \ldots, \frac{1}{x_D^2}]$	$df_{iq} = D$
Final mm.	$[X_c + X_l + X_q + X_i + X_{il} + X_{iq}]$	$df_{mm} = \frac{D^2+7D}{2} + 1$

psLSHADE experimental evaluation

- Experimental evaluation utilizing CEC2021 benchmark [13] (10D and 20D problems)
- Expensive scenario assumed ($10^3 \cdot D$ optimization budget)
- Comparison of:
  - psLSHADE
  - LSHADE
  - MadDE [1] (from CEC2021 Special Session and Competition)

psLSHADE parametrization

Table: LSHADE and psLSHADE parameters (following the parameterization from CEC2021 [17]).

LSHADE & psLSHADE		psLSHADE only	
Initial population size N^0	$18 \cdot D$	Archive size N_a	$2df_{mm}$
Initial M_F	0.5	Surrogates per individual N_s	$2, 5, 10, 20$
Initial M_{CR}	0.5		
Best rate p	0.11		
Archive rate a	1.4		
Memory size H	5		

psLSHADE parameter tuning

Table: Scores achieved by psLSHADE with $N_s = \{2, 5, 10, 20\}$ for $10^3 \cdot D$ optimization budget.

Algo / Score	$N_s = 2$	$N_s = 5$	$N_s = 10$	$N_s = 20$
SNE	34.87	30.80	33.59	41.42
SR	101.25	93.75	133.75	171.25
Score 1	44.17	50.00	45.84	37.18
Score 2	46.30	50.00	35.05	27.37
Score	**90.46**	**100.00**	**80.89**	**64.56**

psLSHADE overall results

Table: Scores achieved by MadDE, LSHADE and psLSHADE with $N_s = 5$ and $10^2 \cdot D$, $10^3 \cdot D$, $10^4 \cdot D$ optimization budgets.

O. b. \ Algo	Score	MadDE	LSHADE	psLSHADE
$10^2 \cdot D$	Score	45.79	56.70	**100.00**
$10^3 \cdot D$	Score	54.58	73.48	**100.00**
$10^4 \cdot D$	Score	90.81	**99.18**	96.77

rmmLSHADE overview

rmmLSHADE [19]: LSHADE extended with:

- A prescreening metamodel
- Recursive Lest Squares filter for metamodel parameters estimation

rmmLSHADE prescreening

Algorithm rmmLSHADE high-level pseudocode

$P^0 = [\boldsymbol{x}_1^0, \ldots, \boldsymbol{x}_{N^0}^0]$ ▷ Population initialization using Latin Hypercube Sampling
Estimate meta-model's coefficients $\boldsymbol{w}_0$ from P^0 and fitness function values of P^0 using OLS
while evaluation budget left **do**
    Extend P^g to $P^g_{ext} = [P^g, P^g, \ldots, P^g]$, where $|P^g_{ext}| = N_m \cdot N^g$
    Generate $N_m \cdot N^g$ mutated vectors $\boldsymbol{v}_k^g$
    Generate $N_m \cdot N^g$ trial vectors $\boldsymbol{u}_k^g$
    for i = 1 to N^g **do**
        Calculate $N_m \cdot N^g$ surrogate values $f^{surr}(\boldsymbol{u}_k^g)$
        Designate the best (not already chosen) trial vector $\boldsymbol{u}_i^{g,best}$ for evaluation
        Update coefficients $\boldsymbol{w}_g$ using RLS
    end for
    Do selection of all N^g chosen trial vectors $\boldsymbol{u}_i^{g,best}$
    Do LSHADE procedures (memory and population size management)
end while

rmmLSHADE metamodel

Table: A description of transformations and the final form of the recursive meta-model (rmm.)

Name	Form	DoF
Constant	$\boldsymbol{z}_c = [1]$	$df_c = 1$
Linear	$\boldsymbol{z}_l = [x_1, \ldots, x_D]$	$df_l = D$
Quadratic	$\boldsymbol{z}_q = [x_1^2, \ldots, x_D^2]$	$df_q = D$
Interactions	$\boldsymbol{z}_i = [x_1 x_2, \ldots, x_{D-1} x_D]$	$df_i = \frac{D(D-1)}{2}$
Final rmm.	$\boldsymbol{z}_{mm} = [\boldsymbol{z}_c + \boldsymbol{z}_l + \boldsymbol{z}_q + \boldsymbol{z}_i]$	$df_{mm} = \frac{D^2+3D}{2} + 1$

rmmLSHADE: RLS filter

$$
\begin{aligned}
e_t &= f(\boldsymbol{u}_i^{g,best}) - \boldsymbol{z}_{mm}^\top \boldsymbol{w}_{t-1}, \qquad & \boldsymbol{g}_t &= \frac{Q_{t-1}\boldsymbol{z}_{mm}}{\lambda + \boldsymbol{z}_{mm}^\top Q_{t-1}\boldsymbol{z}_{mm}} \\
Q_t &= \frac{1}{\lambda}(Q_{t-1} - \boldsymbol{g}_t \boldsymbol{z}_{mm}^\top Q_{t-1}), \qquad & \boldsymbol{w}_g &= \boldsymbol{w}_{t-1} + \boldsymbol{g}_t e_t
\end{aligned} \tag{5}
$$

where $\lambda \in (0, \ldots 1]$ is a forgetting factor and Q_t is a matrix of size $|\boldsymbol{z}_{mm}| \times |\boldsymbol{z}_{mm}|$.

rmmLSHADE overall results

Table: Scores of LSHADE, psLSHADE and rmmLSHADE ($\lambda = 0.98,\ 0.99, 1.0$) for $10^3 \cdot D$ optimization budget. $\lambda =$ is a variant without RLS adaptation.

Score \ Algo	LSHADE	psLSHADE	$\lambda =$	$\lambda = 0.98$	$\lambda = 0.99$	$\lambda = 1.0$
SNE	30.06	17.86	36.72	15.75	14.81	21.62
SR	217.00	154.25	245.50	138.50	122.75	172.00
Score	**52.92**	**81.26**	**45.16**	**91.33**	**100.00**	**69.93**

rmmLSHADE overall results

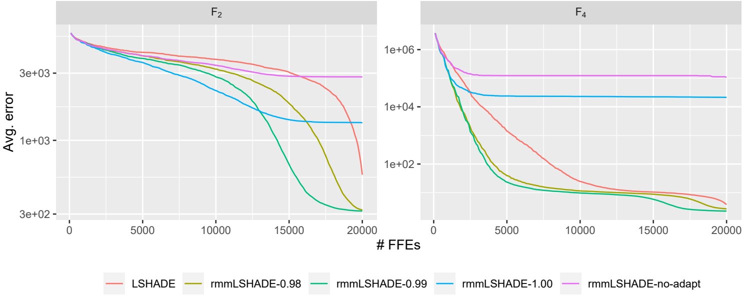


Figure: The averaged convergence of rmmSHADE with $\lambda = 0.98, 0.99, 1.00$, and without $\boldsymbol{w}$ adaptation (denoted as rmmLSHADE-no-adapt) for multi-modal F_2 and uni-modal F_4 functions for $D = 20$ with $10^3 \cdot D$ optimization budget.

Conclusions and future research

Conclusions:

- Metamodels are beneficial (in EAs)
- Various integration methods exists
- CMA-ES based SAEAs are relatively well studied
- Some observations have been confirmed by new DE-based SAEAs
- Designing universal and relatively straightforward algorithms is possible

Future research

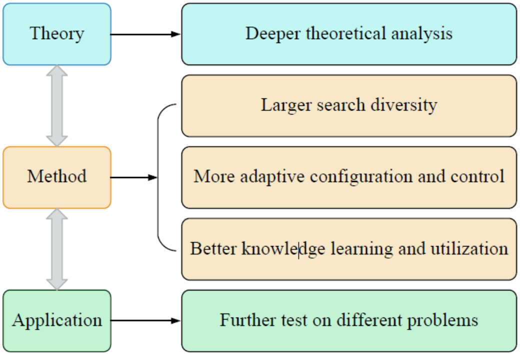


Figure: Potential future research directions and open issues [11]

Future research

Other ideas:

- Better benchmarking
- Budget-dependent adaptation
- Dynamic metamodel utilization
- Exploratory Landscape Analysis utilization
- Reducing the number of parameters
- General simplification (in the case of black-box optimization)

Bibliography I

[1] Biswas, S., Saha, D., De, S., Cobb, A.D., Das, S., Jalaian, B.A.: Improving differential evolution through bayesian hyperparameter optimization. In: 2021 IEEE Congress on Evolutionary Computation (CEC). pp. 832–840. IEEE (2021)

[2] Das, S., Suganthan, P.N.: Differential evolution: A survey of the state-of-the-art. IEEE transactions on evolutionary computation **15**(1), 4–31 (2010)

[3] Fitzpatrick, J.M., Grefenstette, J.J.: Genetic algorithms in noisy environments. Machine learning **3**(2), 101–120 (1988)

[4] Hansen, N.: A global surrogate assisted cma-es. In: Proceedings of the Genetic and Evolutionary Computation Conference. pp. 664–672 (2019)

Bibliography II

[5] Hansen, N., Brockhoff, D., Mersmann, O., Tusar, T., Tusar, D., ElHara, O.A., Sampaio, P.R., Atamna, A., Varelas, K., Batu, U., Nguyen, D.M., Matzner, F., Auger, A.: COmparing Continuous Optimizers: numbbo/COCO on Github (2019). https://doi.org/10.5281/zenodo.2594848, `https://doi.org/10.5281/zenodo.2594848`

[6] Jain, A.K., Mao, J., Mohiuddin, K.M.: Artificial neural networks: A tutorial. Computer **29**(3), 31–44 (1996)

[7] Jin, Y.: A comprehensive survey of fitness approximation in evolutionary computation. Soft computing **9**(1), 3–12 (2005)

[8] Jin, Y.: Surrogate-assisted evolutionary computation: Recent advances and future challenges. Swarm and Evolutionary Computation **1**(2), 61–70 (2011)

[9] Jin, Y., Wang, H., Chugh, T., Guo, D., Miettinen, K.: Data-driven evolutionary optimization: An overview and case studies. IEEE Transactions on Evolutionary Computation **23**(3), 442–458 (2018)

Bibliography III

[10] Joy, T.T., Rana, S., Gupta, S., Venkatesh, S.: A flexible transfer learning framework for bayesian optimization with convergence guarantee. Expert Systems with Applications **115**, 656–672 (2019)

[11] Li, J.Y., Zhan, Z.H., Zhang, J.: Evolutionary computation for expensive optimization: A survey. Machine Intelligence Research **19**(1), 3–23 (2022)

[12] Matheron, G.: Principles of geostatistics. Economic geology **58**(8), 1246–1266 (1963)

[13] Mohamed, A.W., Hadi, A.A., Mohamed, A.K., Agrawal, P., Kumar, A., Suganthan, P.: Problem definitions and evaluation criteria for the CEC 2021 special session and competition on single objective bound constrained numerical optimization, `https://github.com/P-N-Suganthan/2021-SO-BCO/`

[14] Myers, R.H., Myers, R.H.: Classical and modern regression with applications, vol. 2. Duxbury press Belmont, CA (1990)

Bibliography IV

[15] Orr, M.J., et al.: Introduction to radial basis function networks (1996)

[16] Shir, O.M., Roslund, J., Whitley, D., Rabitz, H.: Evolutionary hessian learning: Forced optimal covariance adaptive learning (focal). arXiv preprint arXiv:1112.4454 (2011)

[17] Suganthan, P.: Code of top methods, `https://github.com/P-N-Suganthan/2021-SO-BCO/blob/main/Codes-of-top-methods%20(1).zip`

[18] Zaborski, M., Mańdziuk, J.: Improving lshade by means of a pre-screening mechanism. In: Proceedings of the Genetic and Evolutionary Computation Conference. p. 884–892. GECCO '22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3512290.3528805, `https://doi.org/10.1145/3512290.3528805`

[19] Zaborski, M., Mańdziuk, J.: Surrogate-assisted lshade algorithm utilizing recursive least squares filter. In: International Conference on Parallel Problem Solving from Nature. pp. 146–159. Springer (2022)