Conference Name: International Conference on Science & Technology, 14-15 September 2026, Rome
Conference Dates: 14-Sep- 2026 to 15-Sep- 2026
Conference Venue: Courtyard by Marriott Rome Central Park, Via Moscati 7, 00168 Rome, Italy
Appears in: MATTER: International Journal of Science and Technology (ISSN 2454-5880)
Publication year: 2026

Yoel Tenne, 2026
Volume 2026, pp. 81-90
DOI- https://doi.org/10.20319/stra.2026.8190
This paper can be cited as: Tenne, Y. (2026). Accelerated Convergence Evolutionary Algorithms. International Conference on Science & Technology, 14-15 September 2026, Rome. Proceedings of Scientific and Technical Research Association (STRA), 2026, 81-90

ACCELERATED CONVERGENCE EVOLUTIONARY ALGORITHMS

Yoel Tenne
Mechanical Engineering, Ariel University, Ariel, Israel
y.tenne@ariel.ac.il

Abstract

Evolutionary Algorithms are search methods inspired by the concepts of adaptation and survival of the fittest. Their operation can require large samples to provide a good solution, which hinders their application when evaluations are limited. As such this study presents a methodology to accelerate the EA convergence by moving its population based on the positions of the population. Tests against competing algorithms show the proposed approach has performed well.

Keywords:

Evolutionary Algorithms, Optimization, Convergence, Population Movement Strategy

1. Introduction

Modern research in science and technology often requires locating an optimum of an objective/cost function. Such functions arising in real-world problems often exhibit challenging

features such as multi-modality, discontinuities, and high nonlinear variable relationships. Since classical gradient-based algorithms can often struggle in such settings this has motivated the application of Evolutionary Algorithms (EAs) in such problems since they are more robust to such difficulties and can often yield a better solution. However the inherent simplicity of EAs often results in them requiring many samples to yield a good solution, which is problematic in problems where function evaluations are restricted such as in simulation-driven design optimization (Tenne and Goh, 2010).

To address this issue this study presents a technique to improve the EA performance based on the population positions. This information is used to move the population in a beneficial direction thereby improving convergence. Tests against competing algorithms show that the technique performed well. The text which follows is structured is as follows: Section 2 contains the background information, Section 3 describes the proposed technique. Section 4 details the performance study, and lastly Section 5 has the concluding remarks.

2. Background

EAs are nature-based search methods which were first proposed in the 1960s and 1970s and have since been extensively studied (Michalewicz and Fogel, 2004; De Jong, 2006). They rely on the themes of gradual biological improvement, namely, they employ several vectors (a population) and apply operators like selection, recombination, and mutation to generate new vectors (offspring) which could be better. Algorithm 1 describes a typical EA.

Initialize a population of vectors;
Repeat
Select some vectors as *parents*;
Mix the parents' vectors to generate *offspring*;
Change several offspring vectors;
Obtain the fitness of the offspring;
Decide which vectors will comprise the next iteration;

Algorithm 1: A Typical EA.

An EA operates such that the selection operator drives the search to identify good solutions, the recombination operator yields a localized search around these solutions, while the mutation operator enables exploration of the more distant search space.

Thanks to their simplicity and demonstrated robustness EAs have been applied extensively to real-world problems. For example, Huang et al. (2023) showed the effectiveness of using a real-coded EA to optimize the layout of a wind-turbine farm under various topographic conditions and wind regimes. In another study Yin et al. (2023) used an EA for optimization of shelter location given various constraints on infrastructure. Tenne (2021) proposed an EA-based framework for solving engineering large-scale (high-dimensional) optimization problems while Tenne (2020) used an EA for airfoil shape optimization.

While EAs have been found to be effective in handling challenging optimization problems they are often inefficient, namely, their operation can produce a slow and elaborate convergence (Ivkovic et. al., 2016). As such various studies have explored means for improving the EA efficiency. One method is hybrid algorithms which combines an EA with different methods. For example Yamashita et al. (2022) proposed a hybrid algorithm of an EA and a Bayesian search for chemistry applications, while Wang et al. (2020) proposed a hybrid EA for engineering problems which integrates an island model and a steady-state replacement strategy to improve the population diversity. Another approach to address the EA inefficiency is that of surrogate modeling, namely, in which an model of the cost function is trained and then the EA uses the latter instead of the true but 'expensive' function thereby reducing convergence time. As an example of this approach Wang et al. (2022) combined an EA with surrogates to improve the efficiency of the Reinforcement Learning method for robot control, while Tenne (2022) analyzed the impact of the RBF hyperparameter on an RBF-based surrogate-assisted EA. The third approach explored in improving the EA efficiency is that of modifying some of its internal operators, (selection, recombination, and mutation). For example, Plump et al. (2021) proposed training approximate fitness functions and using prediction intervals to better select the evolutionary operators most suitable to the problem being solved, while Ye and Shen (2019) proposed an EA with dedicated operators to optimize the process of carbon fiber drawing.

In contrast to the aforementioned (and similar) studies in the literature the proposed study implements a method which leverages on beneficial information recorded during EA search process to accelerate convergence, as detailed in the next section.

3. The Proposed Method

As mentioned in Section 1 the proposed method relies on tracking the trajectory of EA and then using this information to improve convergence. The main idea is that when the EA approaches an optimum, the trajectory of the population in recent iterations hints at the direction of an optimum. Therefore pushing the population along this direction could improve convergence.

Specifically, the centroid of the best *p%* members is calculated and recorded at each iteration. Next, every g generations the algorithm calculates a *move vector* which is the difference between the centroid of the current vectors $c^{(i)}$ and the previous one $c^{(i-1)}$, namely

$$r = c^{(i)} - c^{(i-1)} \quad (1)$$

The entire EA population is then moved along this direction by using the relation

$$\hat{x}_i = x_i + \omega r \quad (2)$$

where $\hat{x}_i, x_i$ are the shifted and baseline population vectors, respectively, and ω is the shift factor, which indicates the intensity of the shift. As such using $\omega > 1$ performs a larger shift which could further accelerate convergence while using $\omega < 1$ could be used to perform a more restricted shift in case progress is stalled. Figure 1 presents the mechanics of the proposed method.

The shift factor ω is adjusted at each iteration such that if the shifted population contains a better solution then $\omega_n = \omega \cdot k, k > 1$ while if the shifted population doesn't contain a better vector than $\omega_n = \omega / k\,, k > 1$. The rationale behind this update is to attempt and accelerate convergence as much as possible (increasing the shift factor) and stabilizing convergence in case progress is stalled (decreasing the shift factor).

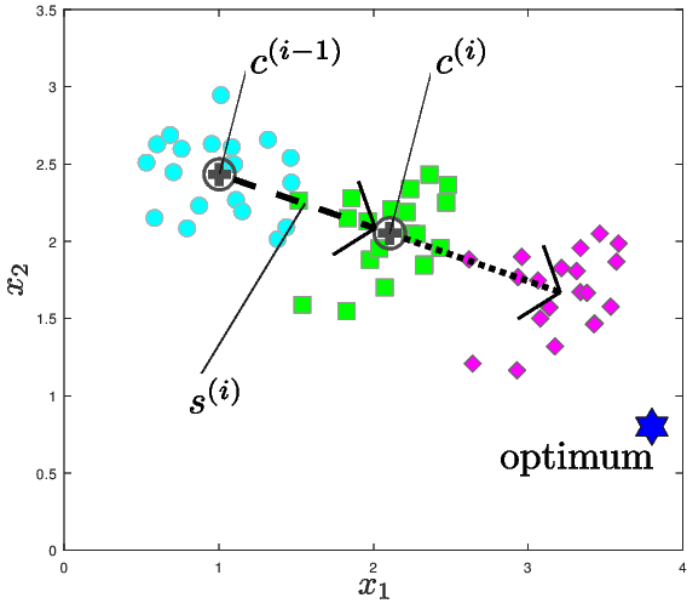


Figure 1: *A Visualization of the Proposed Method*

The proposed algorithm uses four parameters: *p*- the portion of the best vectors which determines centroid, *g* – the iterations between population shifts, ω - the initial shift factor, *k* – the amount by which the shift factor is increased/decreased. To conclude this section Algorithm 2 provides the relevant pseudocode.

4. Performance Analysis

The proposed algorithm was tested with a set of test functions from Suganthan et. al (2005), namely in dimensions 5 and 10. It was also the tested against two algorithms from the literature:

- Differential Evolution (DE) with a neighbourhood based mutation operator (Brest et al. 2025) with the standard difference factor of 0.8 and a cross-over rate of 0.9.
- Simulated Annealing (SA) with the standard temperature function $T_n = 0.8T_i$ where T_n, T_i are the new and current annealing temperatures, respectively (Vandekerckhove, 2024).

Given the large number of combinations of possible parameter permutation and test functions the performance analysis sampled a representative set of test cases. In each case 30

runs were performed with combination of parameter settings for the proposed algorithm as well as for the reference algorithms (DE and SA). The test results are summarized in Tables 1-4, and relevant statistics from 30 runs in each instance.

Since the proposed algorithm relies on the user-set parameters *p*, *g*, *k* several combinations of the latter were tested where each combination is denoted by a 3 element vector *p,g,k* , *i.e.* 0.2,1,1.1 indicates using the parameters *p*=0.2, *g*=1, *k*=1 . In each test case the best mean statistic is emphasized.

Table 1: *Ackley 5*

Algorithm	Mean	SD	Best	Worst
0.2,1,1.1	4.829E+00	4.793E+00	1.005E+00	7.350E+00
0.2,1.4,2	4.744E+00	4.566E+00	2.879E+00	7.737E+00
0.8,1.4,1	**4.536E+00**	4.459E+00	8.347E-01	2.979E+00
0.8,2.1,2	6.934E+00	6.913E+00	4.051E+00	1.101E+01
DE	6.898E+00	6.965E+00	4.832E+00	9.438E+00
SA	1.909E+01	1.978E+01	6.172E+00	2.035E+01

Table 2: *Griewank 5*

Algorithm	Mean	SD	Best	Worst
0.2,1,1.1	1.232E+00	1.2222E+00	7.024E-01	1.921E+00
0.2,1.4,2	1.217E+00	1.199E+00	7.248E-01	1.702E+00
0.8,1.4,1	1.259E+00	1.287E+00	7.890E-01	1.966E+00
0.8,2.4,1	**1.133E+00**	1.171E+00	5.682E-01	1.637E+00
DE	1.926E+00	1.775E+00	1.219E+00	3.474E+00
SA	4.802E+00	2.666E+00	1.495E-01	2.284E+01

Table 3: *Rastrigin 10*

Algorithm	Mean	SD	Best	Worst
0.2,1,1.1	4.689E+01	4.919E+01	2.772E+01	5.538E+01
0.2,1.4,2	**4.381E+00**	4.285E+01	2.827E+01	5.561E+01
0.2,2.4,1	3.938E+01	4.053E+01	1.953E+01	1.282E+01
0.8,2.4,1	4.318E+01	4.329E+01	3.174E+00	5.484E+01
DE	5.698E+01	5.875E+01	3.210E+01	7.803E+01
SA	8.500E+01	8.034E+01	5.085E+01	1/265E+02

Table 4: *Rosenbrock 10*

Algorithm	Mean	SD	Best	Worst
0.2,1,1.1	**1.796E+01**	1.752E+01	1.097E+01	4.467E+01
0.2,1.4,2	1.954E+01	1.895E+01	1.168E+01	2.986E+01
0.2,2.4,1	1.833E+01	1.717E+01	1.061E+01	2.878E+01
0.8,2.4,1	1.8210E+01	1.766E+01	1.107E+01	3.113E+01
DE	5.050E+01	4.628E+01	2/523E+01	8.347E+01
SA	3.737E+01	2.731E+01	3.551E+00	1.009E+02

The results show that the proposed method performed well. For the high modality functions (Ackley, Rastrigin) it was preferable to use a smaller shift factor change (1.4) presumably since it yielded a more restricted shift which performed better in a high multi-modality landscape.

For the low modality functions (Griewank, Rosenbrock) no specific preference in the parameter combination was observed, presumably since their simpler landscape allowed for effective convergence across without high sensitivity to the parameter settings.

In terms of dimensionality, in the 5D cases it was preferable to use a larger portion of the population (p=0.8) while in the 10D cases a smaller portion (p=0.2), which is attributed to the curse of dimensionality: as the dimension increases finding an optimum becomes more

challenging hence the algorithm needs to focus on the very best population members, whereas in the simpler low dimensional problems.

initialize the vectors population;
Repeat
 calculate a centroid using the best %*p* vectors;
 Select some vectors as *parents*;
 Mix parents to generate *offspring*;
 Change some of the offspring;
 In each g*th* generation perform
  obtain the move vector based on current and previous positions
  move the offspring vectors
  if the shifted population contains a new and better solution increase ω else decrease it;
 Obtain the fitness values of the offspring;
Decide which vectors will comprise the next iteration;

Algorithm 2: The proposed EA with the population shift.

5. Conclusion

Evolutionary Algorithms are search methods which can perform well in different problems but their convergence can be elaborate. Accordingly in this study an EA has been described which is designed to improve convergence based on the movement of the population. This information is then used to move the population in a beneficial direction. Tests with against algorithms from the literature show that the proposed method performed well.

References

[Brest and Sepesy Mauˇcec, 2025] Brest, J. and Sepesy Mauˇcec, M. (2025). Comparative study of modern differential evolution algorithms: Perspectives on mechanisms and performance. Mathematics, 13(10).

[De Jong, 2016] De Jong, K. A. (2016). Evolutionary Computation:A Unified Approach. MIT Press, Cambridge, Massachusetts.

[Huang et al., 2023] Huang, X., Wang, Z., Li, C., and Zhang, M. (2023). A low-complexity evolutionary algorithm for wind farm layout optimization. Energy Reports, 9:5752–5761.

[Ivkovic et al., 2016] Ivkovic, N., Jakobovic, D., and Golub, M. (2016). Measuring performance of optimization algorithms in evolutionary computation. International Journal of Machine Learning and Computing, 6(3):167–171.

[Michalewicz and Fogel, 2004] Michalewicz, Z. and Fogel, D. B. (2004). How to Solve It: Modern Heuristics. Springer, Berlin; New York.

[Plump et al., 2021] Plump, C., Berger, B. J., and Drechsler, R. (2021). Improving evolutionary algorithms by enhancing an approximative fitness function through prediction intervals. In Proceedings of the 2021 IEEE Conference on Evolutionary Computation (CEC 2021), pages 127–135, Krakow, Poland. IEEE.

[Plump et al., 2021] Plump, C., Berger, B. J., and Drechsler, R. (2021). Improving evolutionary algorithms by enhancing an approximative fitness function through prediction intervals. In Proceedings of the 2021 IEEE Conference on Evolutionary Computation (CEC 2021), pages 127–135, Krakow, Poland. IEEE.

[Suganthan et al., 2005] Suganthan, P. N., Hansen, N., Liang, J. J., Deb, K., Chen, Y. P., Auger, A., and Tiwari, S. (2005). Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Technical Report KanGAL 2005005, NUS, Singapore and IIT Kanpur,India.

[Tenne, 2020] Tenne, Y. (2020). Online ensemble selection in expensive optimization problems. International Journal of Control, Automation, and Systems, 18:955–965.

[Tenne, 2021] Tenne, Y. (2021). Evaluations of an algorithm for large multivariate optimization. Journal of Physics–Conference Series, 2068. Article 0120026 (online).

[Tenne, 2022] Tenne, Y. (2022). An analysis of the RBF hyperparameter

impact on surrogate-assisted evolutionary optimization. Scientific Programming, 22. Article ID 5175941, Online.

[Tenne and Goh, 2010] Tenne, Y. and Goh, C. K., editors (2010). Computational Intelligence in Expensive Optimization Problems, volume 2 of Evolutionary Learning and Optimization. Springer, Berlin.

[Vandekerckhove, 2024] Vandekerckhove, J. (2024). A general simulated annealing algorithm. Technical report, KU Leuven.

[Wang et al., 2020] Wang, L., Janssem, P., and Ji, G. (2020). SSIEA: a hybrid evolutionary algorithm for supporting conceptual architectural design. Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 34(4):458–476.

[Wang et al., 2022] Wang, Y., Zhang, T., Chang, Y., Wang, X., Liang, B., and Bo, Y. (2022). A surrogate-assisted controller for expensive evolutionary reinforcement learning. Information Sciences, 616:539–557.

[Yamashita et al., 2022] Yamashita, T., Kino, H., Tsuda, K., Miyake, T., and Oguchi, T. (2022). Hybrid algorithm of bayesian optimization and evolutionary algorithm in crystal structure prediction. Science and Technology of Advanced Materials: Methods, 2(1):67–74.

[Ye and Shen, 2019] Ye, h. and Shen, B. (2019). An improved constrained multi-objective optimization evolutionary algorithm for carbon fibre drawing process. Systems Science and Control Engineering, 7(1):133–145.