Wave-based Hybrid Sin Cosin and Differential Evolution Algorit for Optimization

Abhishek Chunarkar; Aditi Khandale
Sandhya Dahake; Vidhi Mehta
Department of Master In Computer Application,GHRCEM,
Nagpur, Maharashtra, India

Abstract
This paper presents a new sine-cosine algorithm (SCA) that is improved by a Differential Evolution (DE) hybridization scheme to achieve maximum optimization performance. The Hybridized SCA (H-SCA–DE) algorithm enhances the convergence rate, provides an efficient balance between exploration and exploitation, and improves the overall precision. Its efficiency was confirmed experimentally on twenty-three benchmark functions, and its performance was compared with that of the basic SCA and other optimization methods. Experiments showed that H-SCA–DE provided better solution quality in thirteen out of twenty-three test cases, showing excellent performance in all but one of the test conditions. The proposed approach also exhibited higher stability and robustness in various optimization scenarios. The results imply that H-SCA–DE is a highly efficient optimization algorithm with a quicker and more consistent solution for complex real-world problems.
Keywords— Sine Cosine Algorithm (SCA), Hybrid Metaheuristic - Algorithms, Exploration - Exploitation.

1. Introduction
Optimization methods are commonly employed to solve complex problems in engineering, artificial intelligence, and data science. Among them, the Sine Cosine Algorithm (SCA) is a popular metaheuristic that effectively balances exploitation and exploration using sine and cosine functions. However, SCA has difficulty with poor convergence and local optima trapping, which restricts its performance in computationally expensive tasks.To overcome these limitations, the Hybrid SCA-DE (H-SCA–DE) algorithm incorporates Differential Evolution (DE), which supports improved convergence and exploitation through adaptive parameter control and evolutionary mutation. Hybridization improves optimization efficiency and reduces premature convergence. H-SCA-DE was compared with twenty-three benchmark test functions and performed better than the standard SCA and other optimization algorithms in terms of solution quality, stability, and convergence, thereby proving it to be a reliable technique for solving complex optimization problems.

2. Proposed Optimization Algorithm
The key driving force of the sine-cosine algorithm (SCA) is the mathematical expression of oscillatory motion as sine and cosine functions. The algorithm was selected because it can effectively navigate the search space using trigonometric-based motion. SCA exhibited stable performance for a global search in exploring and exploiting suitably. However, SCA is plagued by issues such as

DOI: https://doi.org/10.5281/zenodo.15610091

Sr. No.	Algorithm Name	Author Name	Year
1.	Sine Cosine Algorithm	Esmaeil Rashedi et al	2009
2.	Seagull Optimization Algorithm	Seyedali Mirjalili et al	2019
3.	Brain Storm Optimization	She Cheng et al	2013
4.	Butterfly Optimization Algorithm	Sarthak S. Majumder et al	2019
5.	Differential Evolution	Rainer Storn et al	1997
6.	Genetic Algorithm	John Holland	1975
7.	Particle Swarm Optimization	James Kennedy et al	1995
8.	Grey Wolf Optimizer	Seyedali Mirjalili et al	2014

premature c
onvergence and local optima traps. This is caused by the weak capability of SCA in the following intricate landscapes,
in which the algorithm cannot break out from local traps or
explore new regions properly.This may slow down its performance in multimodal or high-dimensional optimization problems.To minimize these shortcomings, hybridization of SCA with Differential Evolution (DE) improves the optimization process through the proper blending of SCA's search capability and DE's efficient exploitation capability. DE employs adaptive mutation and crossover operations to further enhance the solutions and prevent stagnation in non-optimal regions. Hybridization offers faster convergence with precise solutions. Through the synergy of both algorithms, hybrid SCA-DE offers a better-balanced and effective search process, improving the overall optimization process for a wide variety of extremely complex problems.

3. Literature Review

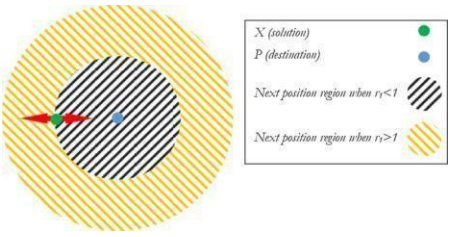


Fig 1: Basic Sine Cosine Algorithm

Optimization algorithms are responsible for solving computationally and engineering complicated

problems. The Sine Cosine Algorithm (SCA) is one of the nature-inspired optimization algorithms that utilizes sine and cosine functions to explore the search space and locate optimal solutions. While SCA is efficient and simple to use, it has a tendency to get trapped in local optima and experience premature convergence in complex or high-dimensional spaces. To avoid these limitations, scientists have coupled SCA with Differential Evolution (DE), which is a potent optimization technique that improves solutions through mutation, crossover, and selection. Testing has confirmed the efficacy of this hybrid approach. For instance, research by Gupta et al. (2021) and Zhang et al. (2022) proved that SCA-DE hybrid significantly surpasses standalone SCA and DE in solving complex optimization problems, particularly in engineering and real-world applications. The method has been successfully applied in machine learning, renewable energy optimization, medical diagnosis, and industrial process optimization. Though very efficient, the hybrid method is computationally costly and requires sensitive parameter tuning. Further research needs to focus on improving adaptability and scalability to enhance its efficiency towards application.

DOI: https://doi.org/10.5281/zenodo.15610091

3.1 Classification Of Algorithms

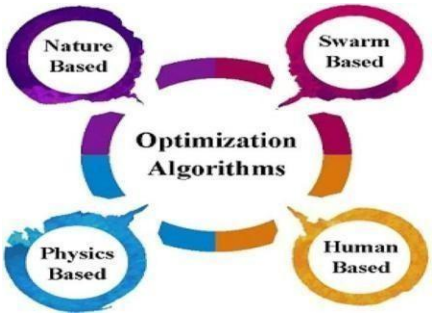


Fig 2: Classification Diagram

Table 1: Algorithm and Authors [6]

This table provides details of different metaheuristic algorithms that have been designed to solve complex optimization problems. These algorithms are based on both natural phenomena and evolutionary concepts. They are extensively used in engineering, machine learning, and problem-solving in real-world scenarios.

4. Flowchart

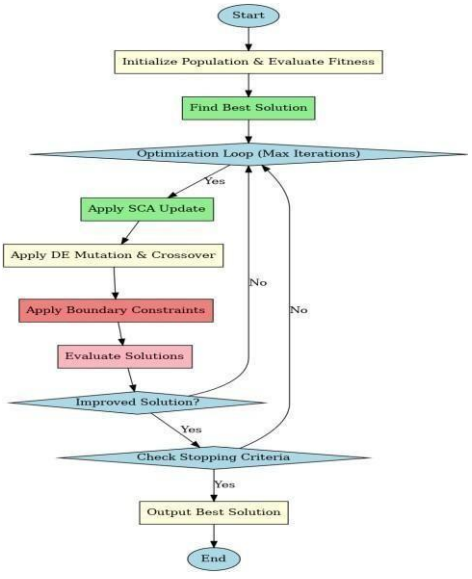


Fig 3: Hybrid SCA-DE Optimization Flowchart

5. Methodology

The Hybridization process alters the traditional SCA by incorporating the DE's evolutionary approach. The mutation mechanism of DE creates new candidate solutions through the combination of a set of existing solutions, which enhances diversity and diminishes the chances of stagnation in suboptimal areas. Through the integration of these mechanisms, Hybrid SCA-DE has an improved exploration-exploitation balance, resulting in greater convergence speed and solution accuracy.

To validate the performance of the developed algorithm, H-SCA-DE was executed and tested on twenty-three benchmark functions with varying optimization landscapes, such as unimodal, multimodal, and composite functions. The performance of H-SCA–DE compared with standard SCA and other efficient optimization algorithms was evaluated based on fundamental parameters such as convergence rate, accuracy, and reliability.

The experimental results show that H-SCA-DE significantly improves the optimization performance with better solution quality and robustness for an extensive range of problems. The ability of the algorithm to adapt and learn difficult optimization problems shows its prospects for real-world use in domains

Table 2: Standard UM benchmark functions

Functions	Dimensions	Range	f_{min}
$F_1(S) = \sum_{m=1}^{z} S_m^2$	(10,30,50,100)	[-100 , 100]	0
$F_2(S) = \sum_{m=1}^{z} \lvert S_m \rvert + \prod_{m=1}^{z} \lvert S_m \rvert$	(10,30,50,100)	[-10 ,10]	0
$F_3(S) = \sum_{m=1}^{z} (\sum_{n=1}^{m} S_n)^2$	(10,30,50,100)	[-100 , 100]	0
$F_4(S) = max_m\{\lvert S_m \rvert, 1 \le m \le z\}$	(10,30,50,100)	[-100 , 100]	0

DOI: https://doi.org/10.5281/zenodo.15610091

6. Benchmark Functions

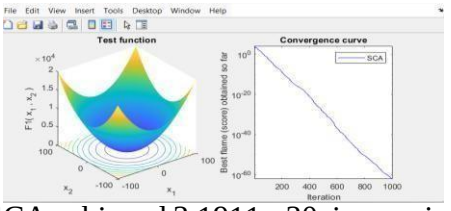
$F_5(S) = \sum_{m=1}^{z-1}[100(S_{m+1}-S_m^2)^2 + (S_m-1)^2]$	(10,30,50,100)	[-38 , 38]	0
$F_6(S) = \sum_{m=1}^{z}([S_m + 0.5])^2$	(10,30,50,100)	[-100 , 100]	0
$F_7(S) = \sum_{m=1}^{z} mS_m^4 + random[0,1]$	(10,30,50,100)	[-1.28, 1.28]	0
$F_8(S) = \sum_{m=1}^{z} -S_m sin(\sqrt{\lvert S_m \rvert})$	(10,30,50,100)	[-500,500]	-418.98295
$F_9(S) = \sum_{m=1}^{z}[S_m^2 - 10cos(2\pi S_m) + 10]$	(10,30,50,100)	[-5.12,5.12]	0
$F_{10}(S) = -20exp\left(-0.2\sqrt{\left(\frac{1}{z}\sum_{m=1}^{z} S_m^2\right)}\right) - exp\left(\frac{1}{z}\sum_{m=1}^{z} cos(2\pi S_m)\right) + 20 + d$	(10,30,50,100)	[-32,32]	0
$F_{11}(S) = 1 + \sum_{m=1}^{z} \frac{S_m^2}{4000} - \Pi_{m=1}^{z} cos\frac{S_m}{\sqrt{m}}$	(10,30,50,100)	[-600, 600]	0
$F_{12}(S) = \frac{\pi}{z}\{10sin(\pi\tau_1) + \sum_{m=1}^{z-1}(\tau_m-1)^2[1 + 10sin^2(\pi\tau_{m+1})] + (\tau_z-1)^2\} + \sum_{m=1}^{z} u(S_m,10,100,4)$ $\tau_m = 1 + \frac{S_m+1}{4}$ $u(S_m,b,x,i) = \begin{cases} x(S_m-b)^i & S_m > b \\ 0 & -b < S_m < b \\ x(-S_m-b)^i & S_m < -b \end{cases}$	(10,30,50,100)	[-50,50]	0
$F_{13}(S) = 0.1\{sin^2(3\pi S_m) + \sum_{m=1}^{z}(S_m-1)^2[1 + sin^2(3\pi S_m+1)] + (x_z-1)^2[1+sin^2 2\pi S_z)]$	(10,30,50,100)	[-50,50]	0
$F_{14}(S) = \left[\frac{1}{500} + \sum_{n=1}^{5}\frac{5}{n+\sum_{m=1}^{2}(S_m-b_{mn})^6}\right]^{-1}$	2	[-65.536, 65.536]	1
$F_{15}(S) = \sum_{m=1}^{11}\left[b_m - \frac{S_1(a_m^2+a_m S_2)}{a_m^2+a_m S_3+S_4}\right]^2$	4	[-5, 5]	0.00030
$F_{16}(S) = 4S_1^2 - 2.1S_1^4 + \frac{1}{3}S_1^6 + S_1S_2 - 4S_2^2 + 4S_2^4$	2	[-5, 5]	-1.0316
$F_{17}(S) = (S_2 - \frac{5.1}{4\pi^2}S_1^2 + \frac{5}{\pi}S_1 - 6)^2 + 10(1-\frac{1}{8\pi})cosS_1 + 10$	2	[-5, 5]	0.398
$F_{18}(S) = \left[1 + (S_1+S_2+1)^2(19-14S_1+3S_1^2-14S_2+6S_1S_2+3S_2^2)\right] \times \left[30 + (2S_1-3S_2)^2(18-32S_1+12S_1^2+48S_2-36S_1S_2+27S_2^2)\right]$	2	[-2,2]	3
$F_{19}(S) = -\sum_{m=1}^{4} d_m exp\left(-\sum_{n=1}^{3} S_{mn}(S_m - q_{mn})^2\right)$	3	[1, 3]	-3.32
$F_{20}(S) = -\sum_{m=1}^{4} d_m exp\left(-\sum_{n=1}^{6} S_{mn}(S_m - q_{mn})^2\right)$	6	[0, 1]	-3.32
$F_{21}(S) = -\sum_{m=1}^{5}[(S-b_m)(S-b_m)^T + d_m]^{-1}$	4	[0,10]	-10.1532
$F_{22}(S) = -\sum_{m=1}^{7}[(S-b_m)(S-b_m)^T + d_m]^{-1}$	4	[0, 10]	-10.4028
$F_{23}(S) = -\sum_{m=1}^{7}[(S-b_m)(S-b_m)^T + d_m]^{-1}$	4	[0, 10]	-10.5363

Table 2: Standard UM Benchmark Functions [6].

5. Results and Discussion

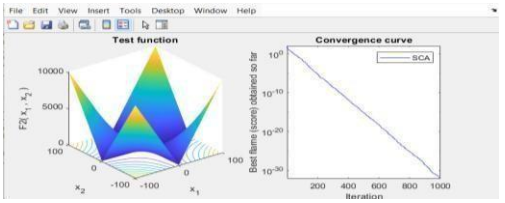
The following section presents the results and discussion
for the twenty-three benchmark functions evaluated.

Function 1:



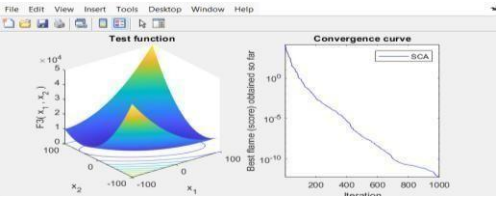
SCA achieved 3.1911e-30, improving to 1.0835-62 after hybridization.

- Function 2:



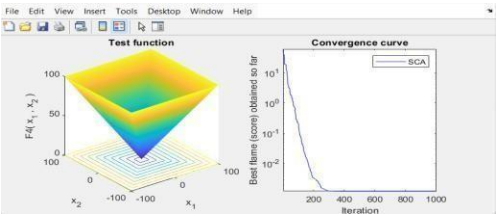
SCA achieved 9.6455e-23, improving to 1.8643e-31 after hybridization.

- Function 3:



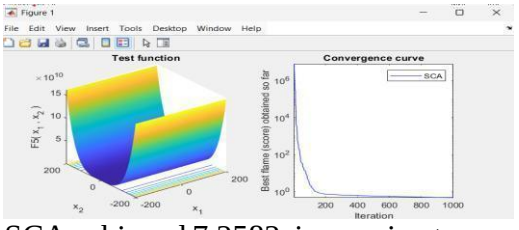
SCA achieved 6.047le-13, improving to 5.7232-13 after hybridization.

- Function 4:



SCA achieved 1.2723e-09, improving to 0.0012302 after hybridization.

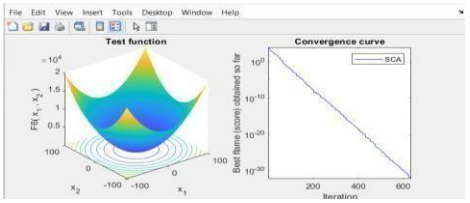
- Function 5:



SCA achieved 7.2582, improving to 2.4319 after hybridization.

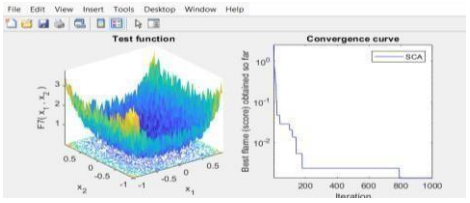
DOI: https://doi.org/10.5281/zenodo.15610091

• Function 6:



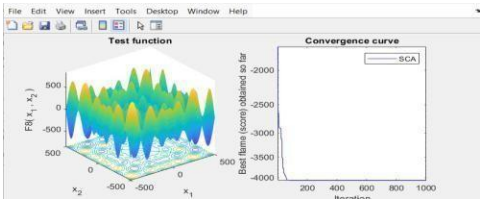
SCA achieved 0.37846, improving to 0 after hybridization.

• Function 7:



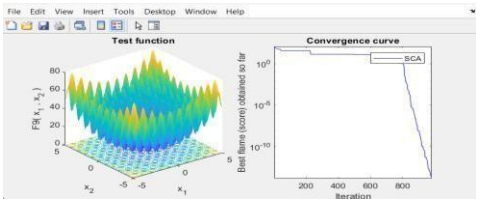
SCA achieved 0.0013593, improving to 0.0013164 after hybridization.

• Function 8:



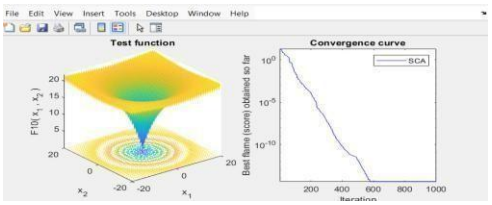
SCA achieved -2337.3626, improving to 4071.3905 after hybridization

• Function 9:



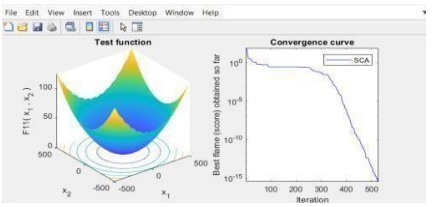
SCA achieved 0, improving to 0 after hybridization.

• Function 10:



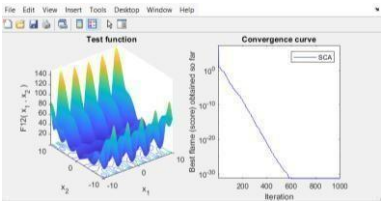
SCA achieved -3.9968e-15, improving to 3.9968e-15 after hybridization.

• Function 11:



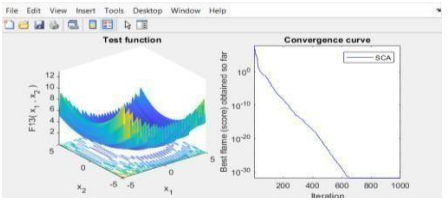
SCA achieved 0, improving to 0.01478 after hybridization.

• Function 12:



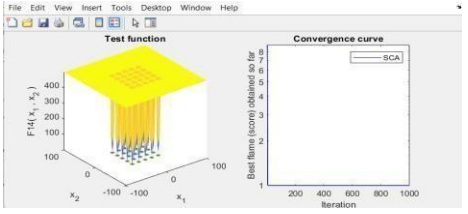
SCA achieved 0.4699, improving to 4.7116e-32 after hybridization.

• Function 13:



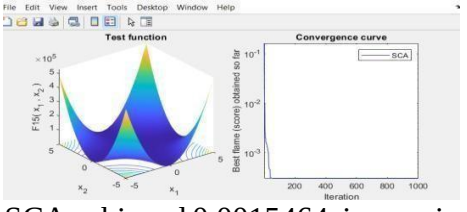
SCA achieved 0.079402, improving to 1.3498e-32 after hybridization.

• Function 14:



SCA achieved 0.99807, improving to 0.998 after hybridization.

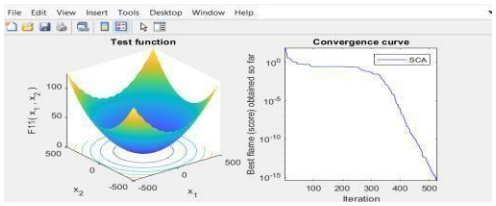
• Function 15:



SCA achieved 0.0015464, improving to 0.00064954 after

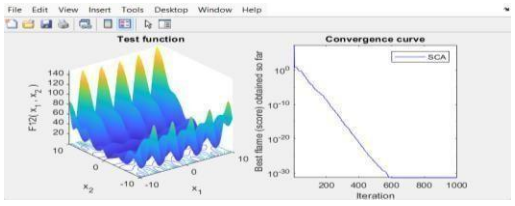
DOI: https://doi.org/10.5281/zenodo.15610091

- Function 11:



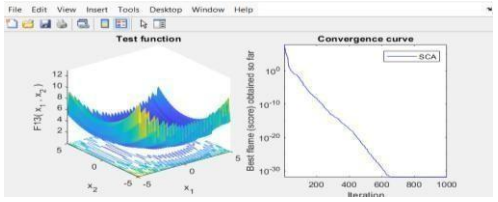
SCA achieved 0, improving to 0.01478 after hybridization.

- Function 12:



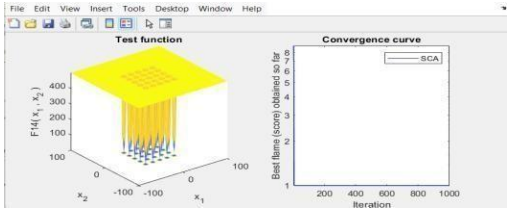
SCA achieved 0.4699, improving to 4.7116e-32 after hybridization.

- Function 13:



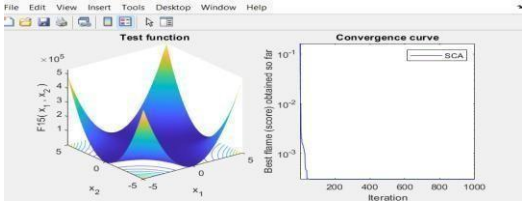
SCA achieved 0.079402, improving to 1.3498e-32 after hybridization.

- Function 14:



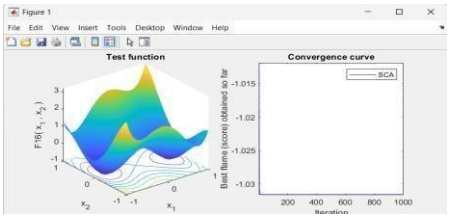
SCA achieved 0.99807, improving to 0.998 after hybridization.

- Function 15:



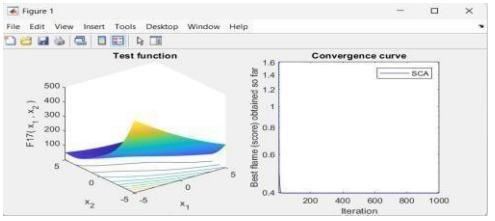
SCA achieved 0.0015464, improving to 0.00064954 after hybridization.

- Function 16:



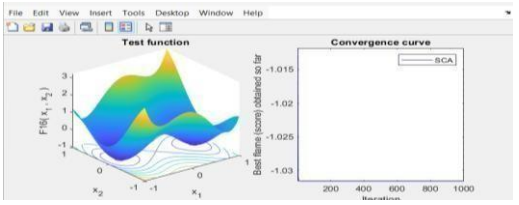
SCA achieved -1.0316, improving to -1.0316 after hybridization.

- Function 17:



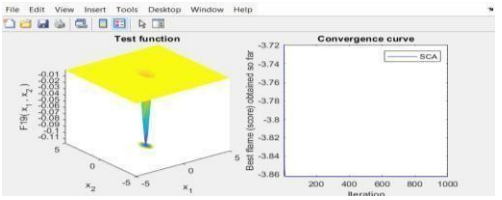
SCA achieved 0.40018, improving to 0.40018 after hybridization.

- Function 18:



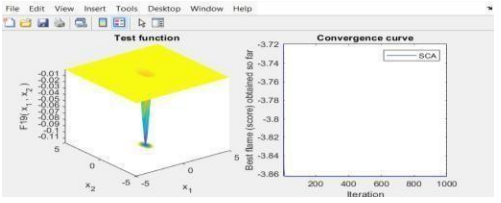
SCA achieved 3, improving to 3 after hybridization.

- Function 19:



SCA achieved -3.8544, improving - 3.8628 after
hybridization.

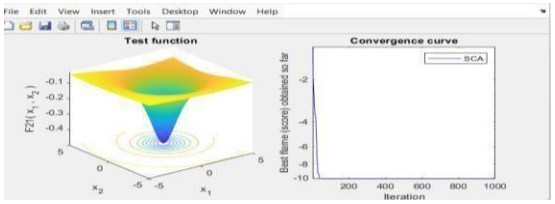
- Function 20:



SCA achieved -3.0058, improving to -3.2031 after hybridization.

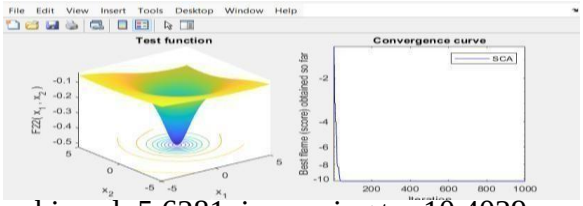
DOI: https://doi.org/10.5281/zenodo.15610091

- Function 21:



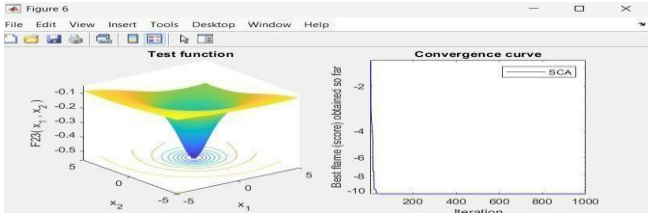
SCA achieved 4.8379, improving to -10.1532 after hybridization.

- Function 22:



SCA achieved -5.6281, improving to -10.4029 after hybridization.

- Function 23:



SCA achieved -4.7271, improving to -10.5364 after hybridization.

7.1 STEPS:

1. The original SCA algorithm was checked using twenty-three benchmark functions to get its ideal values.
2. SCA was merged with DE to improve optimization performance and convergence steadiness.
3. The hybrid algorithm was carried out for multiple iterations on each benchmark function.
4. The single DE algorithm was also evaluated using the twenty three benchmark functions for assessment.
5. The best ideal values found by SCA and DE were evaluated with the results of the Hybrid SCA-DE method.
6. The hybrid algorithm exceeded in separate methods, displaying better end results in fourteen out of twenty-three benchmark functions.

❖ From below table, conclude that hybrid DA-SCA giving more relevant and optimize value as compared to original algorithm. Some values are remained unchanged and some are showing fluctuation in values. Results of hybrid DA-SCA are impressive. Function such as F1, F2,F3, F4, F5, F6, F7, F12, F13, F14, F15,F20,F21,F22 and

Function Name	Actual Value	Hybrid Value
F1	3.1911e-30	1.0835-62
F2	9.6455e-23	1.8643e-31
F3	6.047le-13	5.7232-13
F4	1.2723 e-09	0.0012302
F5	7.2582	2.4319
F6	0.3784 6	0
F7	0.0013 593	0.0013164
F8	- 2337.3 626	4071.3905
F9	0	0
F10	- 3.9968 e-15	3.9968e-15
F11	0	0.01478
F12	0.4699	4.7116e-32
F13	0.0794 02	1.3498e-32
F14	0.9980 7	0.998
F15	0.0015 464	0.00064954
F16	- 1.0316	-1.0316
F17	0.4001 8	0.40018
F18	3	3
F19	- 3.8544	- 3.8628
F20	- 3.0058	-3.2031
F21	- 4.8379	-10.1532
F22	- 5.6281	-10.4029
F23	- 4.7271	-10.5364

DOI: https://doi.org/10.5281/zenodo.15610091

F23 shows the enhancement in value.

Table 3: Results and Discussion

8. Conclusion

The Hybrid Sine-Cosine Algorithm with Differential Evolution (H-SCA-DE) proposed in this research is aimed at addressing the shortcomings of the original SCA. From the comparative table, it can be observed that the original SCA possesses good exploration and poor exploitation and high convergence rate. Differential Evolution (DE), however, possesses low exploration but high exploitation ability and high convergence rate. By maintaining the level of exploration in SCA at a high level and **accelerating convergence rate and exploitation through DE's mutation**

Algorithm	Exploration	Exploitation	Convergence Speed
SCA	High	Low to Moderate	Moderate
DE	Moderate	High	Fast
Hybrid SCA-DE	High	High	Faster

and selection, H-SCA-DE is an improvement over the properties of the two algorithms. With improved but faster convergence, a good balance between exploration and exploitation leads to the improved algorithm.Accuracy, stability, and convergence of H-SCA-DE are significantly better than DE and SCA, as evident from the solution of 23 test problems. Test results indicate the usability, adaptability, and reliability of the hybrid strategy in solving a wide range of challenging real-life optimization problems.

9. References

[1] W. Y. Lin, "A new 3D fruit fly optimization algorithm and its economic applications," Neural Comput. Appl., 2016, doi: 10.1007/s00521-015-1942-8....

[2] Y. Cheng, S. Zhao, B. Cheng, S. Hou, Y. Shi, and J. Chen, "Collaborative business process modeling and optimization for IoT applications," Mob. Inf. Syst., 2018, doi: 10.1155/2018/9174568. Z

[3]In 2018, X. Wang, T. M. Choi, H. Liu, and X. Yue published a paper titled "A novel hybrid ant colony optimization algorithm for emergency transportation problems during post-disaster scenarios," which can be found at doi: 10.1109/TSMC.2016.2606440.

[4] Global optimization in engineering design (nonconvex optimization and its applications), by I. E. Grossmann, vol. 9, 1996].

[5] "A new optimization algorithm for solving complex constrained design optimization problems," by R. V. Rao and G. G. Waghmare, vol. 0273, no. April, 2016, doi: 10.1080/0305215X.2016.1164855.

[6]A. Ibrahim, M. Saber, M. Eid, and E.-S. M. El-Kenawy, "MbGWO-SFS: Modified Binary Grey Wolf Optimizer Based on Stochastic Fractal Search for Feature Selection," IEEE Access, 2020, doi: 2020.300115 ttc10.1109/access.

[7] M. Nouiri, A. Bekrar, A. Jemai, S. Niar, and A. C. Ammari, "A distributed and efficient particle swarm optimization algorithm for flexible job-shop scheduling problem," J. Intell. Manuf., 2018, doi: 10.1007/s10845-015-1039-3.

[8]Y. Li, J. Wang, D. Zhao, G. Li, and C. Chen, "A two-stage approach for combined heat and power economic emission dispatch: Combining multi-objective optimization with integrated decision

10.1155/2018/9174568.
making," [8] Energy (2018), doi: 10.1016/j.energy.2018.07.200.

[9] D. Yousri, T. S. Babu, and A. Fathy, "Recent methodology based Harris hawks optimizer for designing load frequency control incorporated in multi-interconnected renewable energy plants," Sustain. Energy, Grids Networks, 2020, doi: 10.1016/j.segan.2020.100352.

[10]R. Al-Hajj and A. Assi, "Estimating solar irradiance using genetic programming technique and meteorological records," AIMS Energy,2017, doi: 10.3934/energy.2017.5.798.

[11]R. Al-Hajj, A. Assi, and F. Batch, "An evolutionary computing approach for estimating global solar radiation," in 2016 IEEE International Conference on Renewable Energy Research and Applications, ICRERA 2016, 2017. doi: 10.1109/ICRERA.2016.7884553.

[12] R. A. Meyers, "Classical and Nonclassical Optimization Methods 1 Introduction 1 1.1 Local and Global Optimality 2 1.2 Problem Types 2 1.3 Example Problem: Fitting Laser-induced Fluorescence Spectra 3 1.4 Criteria for Optimization 4 1.5 Multicriteria Optimization 4," Encycl. Anal. Chem., pp. 9678–9689, 2000, [Online]. Accessible:

https://pdfs.semanticscholar.org/5c5c/908bb00a54439dcee50ec1ada6b735694a94.pdf

[13] N. Steffan and G. T. Heydt, "Quadratic programming and related techniques for the calculation of locational marginal prices in distribution systems," in 2012 North American Power Symposium (NAPS), 2012, pp. 1–6. doi: 10.1109/NAPS.2012.6336310.

[14] In 2018, Mafarja et al. published "Evolutionary Population Dynamics and Grasshopper Optimization approaches for feature selection problems" in Knowledge-Based Syst., vol. 145, pp. 25–45, doi: 10.1016/j.knosys.2017.12.037.

[15]A. A. Heidari, R. Ali Abbaspour, and A. Rezaee Jordehi, "An efficient chaotic water cycle algorithm for optimization tasks," Neural Comput. Appl., vol. 28, no. 1, pp. 57–85, 2017, doi: 10.1007/s00521-015-2037-2.