A Hybrid Jarratt-Particle Swarm Optimization Algorithm for Nonlinear Problems and Applications

Fiza Zafar [♭],[1] Francisco I. Chicharro[♮], Alicia Cordero[♮], Niaz Ali [♭], and Juan R. Torregrosa[♮]

(♭) Centre for Advanced Studies in Pure and Applied Mathematics, Bahauddin Zakariya University, Multan, Pakistan

(♮) I.U. de Matemática Multidisciplinar, Universitat Politècnica de València Camí de Vera s/n, València, Spain.

1 Introduction

In recent years, there has been a surge of interest in hybrid approaches that combine the strengths of different algorithms to tackle complex optimization problems. Among these hybrids, the fusion of the Particle Swarm Optimization (PSO) algorithm with the Newton-Raphson type methods has emerged as a promising strategy [4]. The Particle Swarm Optimization [3] is a population-based metaheuristic algorithm, inspired by the social behavior of bird flocking or fish schooling, is renowned for its simplicity, effectiveness, and ability to handle non-linear, non-convex optimization problems. The algorithm is given below.

In PSO, position of the particle i is adjusted as:

$$x_i^{t+1} = x_i^t + v_i^{t+1},$$

and velocity of the particle i is updated as follows:

$$v_i^{t+1} = \omega v_i^t + c_1 r_1 \left(p_{(i,lb)}^t - x_i^t\right) + c_2 r_2 (p_{gb}^t - x_i^t).$$

Here ωv_i^t is the momentum part or inertia component that represents the memory of previous flight direction. It prevents particles from drastically changing direction. $c_1 r_1 \left(p_{(i,lb)}^t - x_i^t\right)$ is the cognitive part that represents the memory of previous best position, and it quantifies performance relative to past performances. $c_1 r_1 \left(p_{(i,lb)}^t - x_i^t\right)$ is the social part that quantifies performance relative to neighbors.

$p_{(i,lb)}^t$ is the personal best position of ith particle in t generation. Assume minimization problem.

$$p_{(i,lb)}^{t+1} = \begin{cases} x_i^{t+1}, & \text{if } f\left(x_i^{t+1}\right) < f(p_{(i,lb)}^t) \\ p_{(i,lb)}^t, & \text{otherwise} \end{cases}$$

p_{gb}^t is the global best position in t generation which is calculated as

$$p_{gb}^t \in \left\{ p_{(1,Best)}^t, \ldots, p_{(N,Best)}^t \;/\; f\left(p_{gb}^t\right) = \min\{f(p_{(1,Best)}^t), \ldots, f(p_{(N,Best)}^t)\} \right\},$$

[1] fizazafar@bzu.edu.pk

where N is the number of particles in the swarm.

In spite of its global search properties, PSO exhibits limitations in terms of premature convergence and poor exploration-exploitation balance in high-dimensional search spaces [3,6,7]. On the other hand, the Newton-Raphson type methods excel in finding the local minima of smooth functions by iteratively adjusting the step size based on function evaluations.

Some hybrid techniques are given in [5] while recently a hybrid PSO-Newton-Raphson approach [4] is presented in solving nonlinear optimization problems. By integrating PSO's population-based search with Newton-Raphson's local refinement, the algorithm achieved improved convergence speed and solution quality compared to standalone methods.

Despite these aspects, challenges remain in optimizing the hybrid algorithm's parameters and adapting it to various optimization domains. Further research directions could include investigating hybridization strategies with higher order methods such as Jarratt method [2]. Additionally, comparative studies against other hybrid optimization techniques could provide deeper insights into the strengths and weaknesses of the PSO-Jarratt approach.

This paper presents a novel hybrid approach merging the Particle Swarm Optimization (PSO) algorithm with the Jarratt method tailored to address the complexities of solving the nonlinear systems of equations. The PSO algorithm, known for its global search capability, is integrated with the Jarratt method, renowned for its local convergence properties. This hybridization aims to capitalize on the strengths of both methods, enhancing the efficiency and effectiveness of solving nonlinear problems. The proposed hybrid algorithm's performance is evaluated on various benchmark functions such as Rosenbrock function, six-hump camelback function [1] and a system of nonlinear equation demonstrating superior convergence speed and accuracy compared to traditional methods. Experimental results underscore the efficacy of the hybrid approach in tackling complex problems, offering promising avenues for future research and practical applications.

2 Hybrid Jarratt-Particle Swarm Algorithm

Hybrid Jarratt-Particle Swarm Algorithm (JPSO)

Start

Input

N - swarm size

T - maximum number of iterations

Bounds- bounds of the search space

Output

The best position(solution) p_{gb}^t

Step 1: Solution representation

Step 2: Input: $t := 1$ (Generation counter), Maximum allowed iterations= T

Step 3: Initialize random swarm $p(t)$

Step 4: While $t \leq T$

update $p_{(i,lb)}^t$ of each particle i and find p_{gb}^t;

for $(i = 1, i \leq N, i++)$

update velocity v_i^{t+1}

update position x_i^{t+1}

update position using Jarratt method

Evaluate x_i^{t+1} and include it in $p(t+1)$;

End for

$t = t + 1$

End while

3 Numerical Experiments

We consider the following test problems to compare the performance of Jarratt, PSO and JPSO method. The values of the inertia weight, cognitive weight and social weight are taken as 0.5, 0.8 and 0.9 respectively. The number of particles is taken as 5. The tolerance is considered as 10^{-3}. We mention the comparative results at the iteration where the JPSO converges to the optimal solution.

Test Problem 1:

$$x_1^3 - 3x_1x_2^2 - 1 = 0,$$
$$3x_1^2x_2 - x_2^3 + 1 = 0,$$

subject to bounds $-5 \leq x_1 \leq 5$ and $-5 \leq x_2 \leq 5$.

Table 1. Comparison for the Test Problem 1

Methods	Best Position	Best Value
No. of Iteration: 10		
PSO	$[1.110919, -0.240698]$	0.053731
Jarratt Method	$[-0.290515, 1.084215]$	20.712385
PSO-Jarratt	$[1.084215, -0.290515]$	0.000455

Table 2. Performance of PSO for the Test Problem 1

n	Particle Current Position	Function Value	Personal Best Position	Global Best Position
1	$[1.145176, -0.565898]$	1.287907	$[1.237393, -0.391572]$	$[1.110919, -0.240698]$
2	$[1.428309, -0.191608]$	3.280796	$[1.257908, -0.284077]$	$[1.110919, -0.240698]$
3	$[1.189688, -0.334861]$	0.268395	$[1.110692, -0.240698]$	$[1.110919, -0.240698]$
4	$[0.566174, -0.556423]$	3.459169	$[0.974425, -0.258412]$	$[1.110919, -0.240698]$
5	$[1.386964, -0.256123]$	2.377395	$[1.145234, -0.371235]$	$[1.110919, -0.240698]$

Table 3. Performance of JPSO for the Test Problem 1

n	Particle Current Position	Function Value	Personal Best Position	Global Best Position
1	$[1.084215, -0.290515]$	0.000455	$[1.084215, -0.290515]$	$[1.084215, -0.290515]$
2	$[1.084215, -0.290515]$	0.000455	$[1.084215, -0.290515]$	$[1.084215, -0.290515]$
3	$[1.084215, -0.290515]$	0.000455	$[1.084215, -0.290515]$	$[1.084215, -0.290515]$
4	$[1.084215, -0.290515]$	0.000455	$[1.084215, -0.290515]$	$[1.084215, -0.290515]$
5	$[1.084215, -0.290515]$	0.000455	$[1.084215, -0.290515]$	$[1.084215, -0.290515]$

Test Problem 2: Rosenbrock Function

$$\min f(x_1, x_2) = 100\left(x_2 - x^2{}_1\right)^2 + (1 - x_1)^2,$$

subject to bounds $-2 \leq x_1 \leq 2$ and $-2 \leq x_2 \leq 2$.

The optimal solution is $x = (1, 1)^T$ and the minima is $f(x_1, x_2) = 0$.

Table 4. Comparison for Rosenbrock Function

Methods	Best Position	Best Value
No. of Iteration: 1		
PSO	$[1.207601, 1.406602]$	0.310380
Jarratt Method	$[0.441653, -0.004478]$	4.293178
PSO-Jarratt	$[1.0, 1.0]$	0.0

Table 5. Performance of PSO for Rosenbrock Function

n	Particle Current Position	Function Value	Personal Best Position	Global Best Position
1	$[1.572778, -0.440653]$	849.632099	$[1.245025, -0.891511]$	$[0.734826, 1.557518]$
2	$[1.207601, 1.406602]$	0.310379	$[1.292653, -0.482471]$	$[0.734826, 1.557518]$
3	$[1.174108, 1.675703]$	8.861529	$[0.734826, 1.557518]$	$[1.207601, 1.406602]$
4	$[-0.274909, 0.387011]$	11.324649	$[-1.836006, -0.901647]$	$[1.207601, 1.406602]$
5	$[-0.542806, -0.215216]$	28.375325	$[-1.142840, -1.024995]$	$[1.207601, 1.406602]$

Table 6. Performance of JPSO for Rosenbrock Function

n	Particle Current Position	Function Value	Personal Best Position	Global Best Position
1	$[1.0, 1.0]$	$1.24496e-30$	$[0.913033, -0.729693]$	$[-0.591929, -0.083947]$
2	$[1.0, 1.0]$	0.0	$[0.397187, 0.794623]$	$[1.0, 1.0]$
3	$[1.0, 1.0]$	0.0	$[-1.584231, -1.704015]$	$[1.0, 1.0]$
4	$[1.0, 1.0]$	$2.0215e-30$	$[-0.383914, -1.403466]$	$[1.0, 1.0]$
5	$[1.0, 1.0]$	0.0	$[-0.591929, -0.083947]$	$[1.0, 1.0]$

Test Problem 3: Six-Hump Camelback Function

The Six-Hump Camelback has six local optima, two of which are global. The global optima are located at either $\boldsymbol{x} = (-0.08984,\ 0.71266)$ *or* $\boldsymbol{x} = (0.08984, -0.71266)$, each with a corresponding function value equal to $f(\boldsymbol{x}) = -1.0316285$.

Table 7. Comparison for Six-Hump Camelback Function

Methods	Best Position	Best Value
No. of Iteration: 3		
PSO	$[-0.058390, -0.252294]$	-0.210058
Jarratt Method	$[-0.090221, 0.715655]$	-1.031555
PSO−Jarratt	$[0.089842, -0.712656]$	-1.031629

Table 8. Performance of PSO for Six-Hump Camelback Function

n	Particle Current Position	Function Value	Personal Best Position	Global Best Position
1	$[-0.365793, -0.532881]$	-0.119966	$[-0.365793, -0.532881]$	$[-0.058390, -0.252294]$
2	$[0.990940, -1.993692]$	47.540045	$[-1.179122, 0.337791]$	$[-0.058390, -0.252294]$
3	$[0.048341, 0.213653]$	-0.154591	$[0.048341, 0.213650]$	$[-0.058390, -0.252294]$
4	$[-0.058390, -0.252294]$	-0.210058	$[-0.058390, -0.252290]$	$[-0.058390, -0.252294]$
5	$[-1.4641240, 1.009918]$	0.810808	$[-1.464124, 1.009920]$	$[-0.058390, -0.252294]$

Table 9. Performance of JPSO for Six-Hump Camelback Function

n	Particle Current Position	Function Value	Personal Best Position	Global Best Position
1	$[-1.109205, 0.768268]$	0.543719	$[-1.703607, 0.796084]$	$[0.089842, -0.712656]$
2	$[1.109205, -0.768268]$	0.543719	$[0.089842, -0.712657]$	$[0.089842, -0.712656]$
3	$[0.089842, -0.712656]$	-1.031629	$[0.089842, -0.712657]$	$[0.089842, -0.712656]$
4	$[1.109205, 0.768268]$	0.543719	$[-1.703607, 0.796084]$	$[0.089842, -0.712656]$
5	$[-1.638068, -0.228674]$	2.229358	$[-1.703607, 0.796084]$	$[0.089842, -0.712656]$

4 Conclusion

In conclusion, the fusion of the Particle Swarm Optimization algorithm with Jarratt's method represents a compelling synergy of evolutionary and iterative methods paradigms, offering enhanced performance and versatility across various domains. Continued research efforts are warranted to fully exploit the potential of this hybrid approach and address the evolving challenges in optimization.

Acknowledgements

This study has been funded by "Ayuda a Primeros Proyectos de Investigación (PAID-06-23), Vicerrectorado de Investigación de la Universitat Politècnica de València (UPV)", in the framework of project MERLIN.

References

[1] Jaberipour, M., Khorram, E.,Karimi, B. Particle swarm algorithm for solving systems of nonlinear equations, *Comput. Math. Appl.*, 62: 566-576, 2011.

[2] Jarratt, P., Some fourth order multipoint iterative methods for solving equations, *Mathematics of Computation*, 20: 434–437, 1966.

[3] Kennedy, J., Eberhart, R., Swarm Intelligence, Morgan Kaufmann Publishers, 2001.

[4] Oh, R. , Shi, Y., Choi, J.W., A hybrid Newton–Raphson and particle swarm optimization method for target motion analysis by batch processing, *Sensors*, 21(6): Art. 2033, 2021.

[5] Sengupta, S., Basak, S., Peters, R.A., Particle swarm optimization: A survey of historical and recent developments with hybridization perspectives, *Mach. Learn. Knowl. Extr.* 1: 157–191, 2019.

[6] Shami, T.M., El-Saleh, A.A., Alswaitti, M., Al-Tashi, Q., Summakieh, M.A., Mirjalili, S., Particle Swarm Optimization: A comprehensive survey, *IEEE Access* 10: 10031–10061, 2022.

[7] Wang, D., Tan, D., Liu, L., Particle swarm optimization algorithm: An overview, *Soft Comput.* 22: 387–408, 2018.