Using Variable Interaction Graphs to Improve Particle Swarm Optimization

Caz L. Czworkowski
Johns Hopkins University
Baltimore, Maryland, USA
cazczworkowski@gmail.com

John W. Sheppard
Montana State University
Bozeman, Montana, USA
john.sheppard@montana.edu

Abstract

This paper presents Variable Interaction Graph Particle Swarm Optimization (VIGPSO), an adaptation to Particle Swarm Optimization (PSO) that dynamically learns and exploits variable interactions during the optimization process. PSO is widely used for real-valued optimization problems but faces challenges in high-dimensional search spaces. While Variable Interaction Graphs (VIGs) have proven effective for optimization algorithms operating with known problem structure, their application to black-box optimization remains limited. VIGPSO learns how variables influence each other by analyzing how particles move through the search space, and uses these learned relationships to guide future particle movements. VIGPSO was evaluated against standard PSO on eight benchmark functions (three separable, two partially separable, and three non-separable) across 10, 30, 50 and 1000 dimensions. VIGPSO achieved statistically significant improvements ($p < 0.05$) over the standard PSO algorithm in 28 out of 32 test configurations, with particularly strong performance extending to the 1000-dimensional case. The algorithm showed increasing effectiveness with dimensionality, though at the cost of higher variance in some test cases. These results suggest that dynamic VIG learning can bridge the gap between black-box and gray-box optimization effectively in PSO, particularly for high-dimensional problems.

CCS Concepts

• **Theory of computation → Bio-inspired optimization**; • **Mathematics of computing → Evolutionary algorithms**; **Optimization with randomized search heuristics**.

Keywords

Particle swarm optimization, variable interaction graphs, large-scale Optimization

ACM Reference Format:
Caz L. Czworkowski and John W. Sheppard. 2025. Using Variable Interaction Graphs to Improve Particle Swarm Optimization. In *Proceedings of The Genetic and Evolutionary Computation Conference 2025 (GECCO '25).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3712255.3726589

GECCO '25, Málaga, Spain

ACM ISBN 979-8-4007-1464-1/2025/07
https://doi.org/10.1145/3712255.3726589

1 Introduction

Particle Swarm Optimization (PSO) is a population-based method inspired by biological swarms, where particles navigate the search space by tracking their best positions and exchanging information across various topologies [1, 5]. Gray-box approaches enhance standard PSO by incorporating partial problem structure knowledge via Variable Interaction Graphs (VIGs), which capture correlations among variables [2, 12].

We present VIGPSO—a PSO variant that dynamically constructs a VIG during optimization. Our hypothesis is that dynamically learned variable interactions can (1) accelerate convergence, (2) lower objective values ($p < 0.05$), and (3) improve robustness, especially in high-dimensional, non-separable spaces [4].

2 Background and Related Work

Incorporating fitness landscape insights can enhance stochastic search methods like PSO. Towers *et al.* [11] used neural networks to estimate fitness landscapes and predict ruggedness, enabling adaptive parameter selection. Chicano *et al.* [2] extended these ideas with the Dynastic Potential Crossover operator for pseudo-Boolean problems. They construct Variable Interaction Graphs (VIGs) using either co-occurrence patterns or Fourier analysis, later transforming these graphs into chordal structures and clique trees—a process that works best under sparse connectivity.

The original PSO algorithm by Kennedy and Eberhart [5] updates particle velocities by combining inertia, cognitive, and social components. Building on this framework, Tin'os *et al.* [10] demonstrated that dynamic VIGs can be learned during optimization using a simple criterion based on fitness differences, successfully detecting over 97% of true variable interactions without extra evaluations.

Detecting variable interactions is also crucial in cooperative co-evolutionary algorithms. For example, Omidvar *et al.* [7] proposed differential grouping, which infers interactions by examining fitness variations when variables are perturbed. While earlier variants such as GDG, XDG, DDG, ODG, and RDG rely on predetermined interactions, dynamic methods update groupings as the search progresses [13].

3 Optimization Strategy

Our approach leverages learned variable interactions to modify particles' continuous trajectories via a dynamically created VIG in a black-box setting. Unlike the ILS algorithm by Tin'os *et al.* [10], which used VIGs for discrete changes, we begin (Steps 1–2 of Algorithm 1) by initializing a zero-weighted adjacency matrix G to record correlations between dimensions. In each PSO iteration, standard position and velocity updates are computed and used to calculate pairwise Pearson correlations between dimensions [14].

Algorithm 1 VIGPSO

1: Initialize PSO parameters: particles, velocities, *pbest*, *gbest*,τ_1, τ_2,
2: Initialize empty Variable Interaction Graph (VIG) matrix $G \leftarrow 0$
3: **for** $t = 1$ to $max\_iterations$ **do**
4: Compute inertial weight $\omega_{curr} \leftarrow \omega(1 - 0.6prog)$
5: Store current positions as X_{old}
6: **for** each particle p **do**
7: **for** each dimension d **do**
8: Compute standard PSO velocity v_s
9: Retrieve connected dimensions $\mathcal{N}$ from G
10: **if** $\mathcal{N}$ is not empty **then**
11: Get weights: $w_n \leftarrow G_{d,n}$ for $n \in \mathcal{N}$
12: Normalize weights: $w_n \leftarrow w_n / \sum_{n \in \mathcal{N}} w_n$
13: $v_{\text{vig}} \leftarrow \sum_{n \in \mathcal{N}} w_n v_n$
14: $\alpha \leftarrow 0.3(1 - e^{-2t/t_{max}})$
15: $v' \leftarrow (1 - \alpha)v_s + \alpha v_{\text{vig}}$
16: **else**
17: $v' \leftarrow v_s$
18: **end if**
19: Clip velocity v' to bounds
20: **end for**
21: Update position $x \leftarrow x + v'$
22: Update *pbest* and *gbest* if improved
23: **end for**
24: **if** t mod $update\_interval = 0$ **then**
25: Compute particle movement $\Delta X \leftarrow X - X_{old}$
26: **for** each pair of dimensions i, j **do**
27: Compute correlation $\rho \leftarrow corr(\Delta X_i, \Delta X_j)$
28: **if** $|\rho| > \tau_1$ **then**
29: $G_{i,j} \leftarrow |\rho|$
30: **else if** $|\rho| < \tau_2$ **then**
31: $G_{i,j} \leftarrow 0$
32: **end if**
33: **end for**
34: **end if**
35: **end for**
36: **return** *gbest*

Weak correlations below a pruning threshold τ_2 are removed (Steps 24–31 of Algorithm 1) to retain only meaningful interactions.

The VIG is then integrated into the PSO velocity update using an adaptive weighting scheme (Steps 8–17 of Algorithm 1). For each dimension d, the new velocity is computed as:

$$\begin{aligned} v'_d &= (1 - \alpha)v_s + \alpha v_{\text{vig}} \\ v_{\text{vig}} &= \frac{\sum_{i \in \mathcal{N}_d} w_i v_i}{\sum i \in \mathcal{N}_d w_i} \end{aligned}$$

where v_s is the standard velocity update, w_i is the VIG edge weight for neighbors $\mathcal{N}_d$, and α is an adaptive weight that increases with iterations. Alternative communication topologies (e.g., ring or star) can replace the fully connected *gbest* topology, and velocity clipping is applied to prevent explosions [3].

The weight α, which determines the relative contribution of v_{vig}, evolves as optimization proceeds according to:

$$\alpha = 0.3(1 - e^{-2prog})$$

where $prog = t/max\_iterations$, ensuring a smooth transition from exploration to exploitation. Thus, VIGPSO enhances standard PSO by gradually increasing the influence of learned variable interactions over time.

4 Experimental Design

We tested the effectiveness of our approach using benchmark functions specifically chosen to represent different types of variable interactions and separability characteristics. Three categories of functions were selected to help understand how the VIG adaptation performed under different optimization scenarios. For consistency, all functions were evaluated within the bounds $[-5, 5]$ for all dimensions. For fully separable functions where variables could be optimized independently, we used the Sphere function:

$$f(x) = \sum_{i=1}^{n} x_i^2,$$

the Sum Squares function:

$$f(x) = \sum_{i=1}^{n} i x_i^2,$$

and Schwefel 2.22:

$$f(x) = \sum_{i=1}^{n} |x_i| + \prod_{i=1}^{n} |x_i|$$

For partially separable functions, where groups variables interacted while the groups remained independent of each other, we used the Dixon-Price function:

$$f(x) = (x_1 - 1)^2 + \sum_{i=2}^{n} i(2x_i^2 - x_{i-1})^2,$$

and the Rastrigin function:

$$f(x) = 10n + \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i)].$$

For fully non-separable functions, where all variables interacted in complex ways, we used the Rosenbrock function:

$$f(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2],$$

the Griewank function:

$$f(x) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}),$$

and the Alpine function:

$$f(x) = \sum_{i=1}^{n} |x_i \sin(x_i) + 0.1x_i|.$$

Performance was evaluated on dimensions 10, 30, 50, and 1000 using 50 particles over 300 iterations, with 100 independent runs per configuration. The evaluation metrics included final objective value, convergence curves, robustness across function types, and

Table 1: Statistical Comparison of VIGPSO vs Standard PSO

Type	Function: p-value; Winner
Separable	**Sphere, SumSquares, Schwefel 2.22**: All < 0.001 (VIGPSO)
Partially Separable	**Dixon-Price**: All < 0.001 (VIGPSO). **Rastrigin**: 10: 0.0047 (PSO), 30: n.s., 50 & 1000: < 0.001 (VIGPSO)
Non-Separable	**Rosenbrock**: All < 0.001 (VIGPSO). **Griewank**: 10: n.s., 30/50/1000: < 0.001 (VIGPSO). **Alpine**: 10: 0.0021 (VIGPSO), 30: n.s., 50 & 1000: < 0.001 (VIGPSO).

Statistical significance determined using Mann-Whitney U test with α = 0.05. P-values are rounded to three decimal places.

statistical significance (Mann-Whitney U test, two-sided, with α = 0.05).

A parameter sensitivity analysis was performed for each of the 32 configurations (8 functions × 4 dimensions) via grid search over the following parameters: inertial weight 0.4, 0.5, 0.6, 0.8, cognitive and social learning factors 1.0, 1.5, 2.0, 2.5, correlation thresholds 0.3, 0.5, 0.7, pruning thresholds 0.3, 0.5, 0.7, and update intervals 5, 10, 15. Each combination was evaluated over 100 iterations to select the best configuration, ensuring a fair comparison between standard PSO and VIGPSO.

Tuning results revealed that VIGPSO generally performs best with lower inertial weights ($\omega = 0.4$) and higher social learning factors, while standard PSO favors moderate inertial weights (ω = 0.6) with higher cognitive factors.

The time complexity of VIGPSO is $O(TSd^2)$—with T iterations, S particles, and d dimensions—compared to standard PSO's $O(TSd)$ due to: (i) per-particle VIG influence computations requiring $O(d^2)$ operations per particle (each dimension sums contributions from up to $d-1$ others) and (ii) VIG updates costing $O(Sd^2)$ per iteration. Although this overhead is significant in high dimensions, the improved performance justifies the trade-off; the VIG update interval parameter helps control the frequency of these updates without altering the asymptotic complexity.

5 Results and Discussion

As mentioned above, the benchmark experiments were conducted across eight test functions, categorized by their separability characteristics. The results of the Mann-Whitney U test are presented in Table 1. The column labeled "Lower Obj." identifies the algorithm that returned the lower objective value (on average). A dash ("–") indicates no statistically significant difference in performance.

Figure 1 shows the convergence curves for both algorithms at 1000 dimensions. The solid line represents the mean global best fitness over 100 independent runs, and the shaded area indicates one standard deviation from the mean.

Figure 2 shows the distribution of final fitness values achieved by both algorithms for the 1000-dimensional experiments. The boxplot illustrates the median, quartiles, and outliers of the final solutions from 100 independent runs.

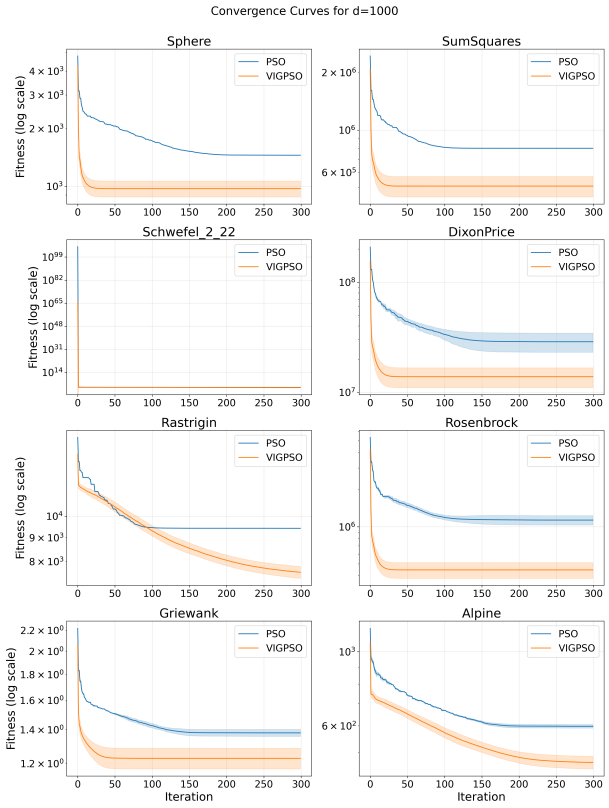

Figure 1: Mean fitness and standard deviation bands in 1000 dimensions.

The results across three performance criteria were very strong. VIGPSO achieved statistically significant objective improvements ($p < 0.05$) in 20 out of 24 test configurations (Table 1), particularly in higher dimensions. Three of the remaining four configurations were statistically equivalent, and standard PSO was best in only one 10-dimensional case. As dimensionality increased, VIGPSO's performance consistently surpassed that of PSO.

For fully separable functions (Sphere, Sum Squares, and Schwefel 2.22), VIGPSO generally converged faster with less variance, although some functions like Schwefel 2.22 exhibited notable fluctuations. For partially separable functions (Dixon-Price and Rastrigin), VIGPSO showed mixed results: it significantly improved Dixon-Price across all dimensions ($p < 0.001$), while Rastrigin favored PSO at $d = 10$ ($p = 0.005$), was equivalent at $d = 30$ ($p = 0.468$), and was outperformed by VIGPSO at $d = 50$ ($p = 0.027$). At 1000 dimensions, VIGPSO outperformed PSO across all runs ($p < 0.001$), highlighting its scalability.

For non-separable functions (Rosenbrock, Griewank, and Alpine), VIGPSO significantly improved results for Rosenbrock ($p < 0.001$). Griewank showed no difference at $d = 10$ ($p = 0.691$) but favored VIGPSO at $d = 30$ and $d = 50$ ($p < 0.001$) with strong performance continuing at 1000 dimensions, while Alpine also favored VIGPSO at $d = 10$ ($p = 0.002$), $d = 50$, and $d = 1000$ ($p < 0.001$). These findings emphasize VIGPSO's enhanced capacity to leverage variable interactions as the search space expands.

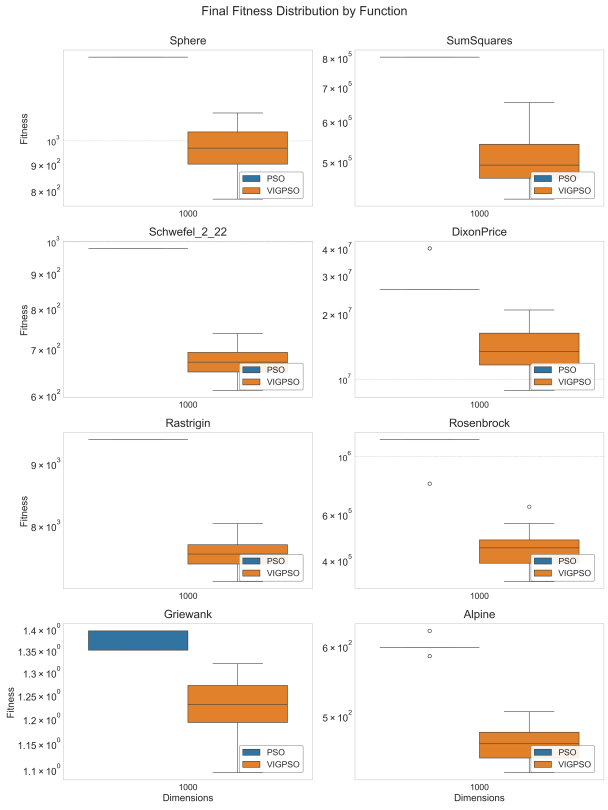

Figure 2: Distribution of final fitness values achieved by both algorithms across all test functions at 1000 dimensions.

The robustness analysis for the 1000-dimensional experiments (Figure 2) indicates that in several test configurations, VIGPSO exhibits wider interquartile ranges. This increased variability may be attributed to its adaptive variable interaction graph, which promotes diverse exploration paths across runs.

Although VIGPSO incurs additional computational overhead for maintaining and updating its variable interaction graph, this cost is justified by its improved convergence and solution quality—even on separable functions, where its velocity update mechanism appears to impart extra diversity and implicit momentum effects.

6 Conclusion

The results demonstrate VIGPSO's effectiveness in black-box optimization, with statistically significant improvements in 28 of 32 configurations and consistent performance across separable and non-separable problems, especially at higher dimensions.

Although VIGPSO improved solution quality, it also introduced greater variance in lower dimensions, likely due to its adaptive learning mechanism. This trade-off appears acceptable for complex problems where superior solutions are critical.

Future work will explore adaptive mechanisms to modulate learned interactions by dimensionality, alternative correlation metrics beyond Pearson correlation, and extensions to constrained problems. The success of variable interaction learning suggests incorporation into advanced PSO variants such as Comprehensive Learning PSO (CLPSO) [6] and adaptation for cooperative co-evolutionary methods. For instance, VIGPSO could refine subpopulation grouping in differential grouping-based CCEA methods [7], enhance factored evolutionary algorithms (FEA) [9] where overlapping factors capture additional interactions, and integrate with dynamic factor approaches [8] to directly tie adaptive VIG construction to improved group definitions.

References

[1] Tim Blackwell and James Kennedy. 2019. Impact of Communication Topology in Particle Swarm Optimization. *IEEE Transactions on Evolutionary Computation* 23, 4 (2019), 689–702. doi:10.1109/TEVC.2018.2880894

[2] Francisco Chicano, Gabriela Ochoa, L. Darrell Whitley, and Renato Tinós. 2022. Dynastic Potential Crossover Operator. *Evolutionary Computation* 30, 3 (09 2022), 409–446. doi:10.1162/evco_a_00305 arXiv:https://direct.mit.edu/evco/article-pdf/30/3/409/2040916/evco_a_00305.pdf

[3] Eberhart and Yuhui Shi. 2001. Particle swarm optimization: developments, applications and resources. In *Proceedings of the Congress on Evolutionary Computation*, Vol. 1. 81–86 vol. 1. doi:10.1109/CEC.2001.934374

[4] Tim Hendtlass. 2009. Particle Swarm Optimisation and high dimensional problem spaces. In *IEEE Congress on Evolutionary Computation*. 1988–1994. doi:10.1109/CEC.2009.4983184

[5] James Kennedy and Russell Eberhart. 1995. Particle swarm optimization. In *Proceedings of International Conference on Neural Networks*, Vol. 4. IEEE, 1942–1948.

[6] J.J. Liang, A.K. Qin, P.N. Suganthan, and S. Baskar. 2006. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Transactions on Evolutionary Computation* 10, 3 (2006), 281–295. doi:10.1109/TEVC.2005.857610

[7] Mohammad Nabi Omidvar, Xiaodong Li, Yi Mei, and Xin Yao. 2014. Cooperative Co-Evolution With Differential Grouping for Large Scale Optimization. *IEEE Transactions on Evolutionary Computation* 18, 3 (2014), 378–393. doi:10.1109/TEVC.2013.2281543

[8] Shehzad Qureshi and John W. Sheppard. 2016. Dynamic sampling in training artificial neural networks with overlapping swarm intelligence. In *2016 IEEE Congress on Evolutionary Computation (CEC)*. 440–446. doi:10.1109/CEC.2016.7743827

[9] Shane Strasser, John Sheppard, Nathan Fortier, and Rollie Goodman. 2017. Factored Evolutionary Algorithms. *IEEE Transactions on Evolutionary Computation* 21, 2 (2017), 281–293. doi:10.1109/TEVC.2016.2601922

[10] Renato Tinós, Michal Witold Przewozniczek, and Darrell Whitley. 2022. Iterated Local Search with Perturbation Based on Variables Interaction for Pseudo-Boolean Optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Boston, MA, USA). Association for Computing Machinery, New York, NY, USA, 296–304. doi:10.1145/3512290.3528716

[11] Sebastian Towers, Jessica James, Harrison Steel, and Idris Kempf. 2024. Learning-Based Estimation of Fitness Landscape Ruggedness for Directed Evolution. *bioRxiv 2024.02.28.582468* (2024). doi:10.1101/2024.02.28.582468

[12] L Darrell Whitley, Francisco Chicano, and Brian W Goldman. 2016. Gray box optimization for Mk landscapes (NK landscapes and MAX-kSAT). *Evolutionary computation* 24, 3 (2016), 491–519.

[13] Shuai Wu, Zhitao Zou, and Wei Fang. 2018. A Dynamic Global Differential Grouping for Large-Scale Black-Box Optimization. In *Advances in Swarm Intelligence*, Ying Tan, Yuhui Shi, and Qirong Tang (Eds.). Springer International Publishing, Cham, 593–603.

[14] Maoqing Zhang, Wuzhao Li, Liang Zhang, Hao Jin, Yashuang Mu, and Lei Wang. 2023. A Pearson correlation-based adaptive variable grouping method for large-scale multi-objective optimization. *Information Sciences* 639 (2023), 118737. doi:10.1016/j.ins.2023.02.055