Optimal Vaccination Strategies for Pandemic Control: A Cost-Driven SIR Model with Domain-as-Particle PSO Optimization

Fabio Berberi[1] and Paolo Mercorelli[2]

[1] [1] University of Siena and Leuphana University of Lüneburg, Siena, Italy
f.berberi@student.unisi.it
[2] [2] Leuphana University of Lüneburg, Lüneburg, Germany
paolo.mercorelli@leuphana.de

Abstract. This paper extends and improves upon a previously published pandemic control model based on the Susceptible–Infected–Removed (SIR) framework with a feedback vaccination law, which established sufficient conditions for achieving herd immunity by minimizing a cost function that combines both vaccination effort and intervention time. In the original approach, optimization was performed using standard routines, which proved computationally demanding and potentially limited in high-dimensional or nonlinear scenarios. Here, we introduce an advanced Particle Swarm Optimization (PSO) strategy based on a domain-as-particle paradigm, in which the parameter space is partitioned into non-overlapping subdomains, each acting as an independent PSO particle. This structured exploration enhances both convergence speed and robustness. The hybrid framework, integrating dynamic simulation with the domain-as-particle PSO, achieves herd immunity more rapidly and efficiently compared to the original method, thereby offering improved guidance for public health policy design.

Keywords: Pandemic Control · Susceptible–Infected–Removed (SIR) Model · Particle Swarm Optimization (PSO) · Domain-as-Particle

1 Introduction

Mathematical modeling is a cornerstone of infectious disease management, with compartmental models such as SIR (Susceptible–Infected–Removed) widely used to understand epidemic evolution and to design intervention strategies [12]. In [11], a control-oriented extension of the SIR model was developed, where a feedback vaccination law was introduced to drive the susceptible population below a herd immunity threshold. The central objective was to minimize a cost function that accounted for both the total squared vaccination effort and the time required to reach herd immunity. Analytical results provided sufficient conditions for achieving epidemic control, and the approach was validated using dynamic simulations [3]. Despite its effectiveness, the original optimization approach relied on conventional routines, which may become inefficient or trapped

in local minima when faced with the nonlinearities and high-dimensional search spaces characteristic of realistic epidemic control problems [9], as also highlighted in structural optimization studies where PSO has been successfully applied [14], and in advanced control design tasks where PSO has proven effective for complex PID tuning [16]. Motivated by these limitations, this work proposes a novel optimization framework based on the domain-as-particle Particle Swarm Optimization (PSO) paradigm, originally introduced by Kennedy and Eberhart [6]

In the proposed approach, the parameter space is divided into non-overlapping domains, each treated as an autonomous particle in the PSO algorithm. Within each domain, a local search is performed, guided by characteristic vectors derived from the best candidate solutions. Domains communicate by sharing information about their local optima, allowing the algorithm to balance exploration of new regions and exploitation of promising areas. This structure enables faster and more reliable convergence to optimal vaccination policies, building directly on the foundation of the previous work, but overcoming its computational limitations. By integrating the domain-as-particle PSO with Simulink-based epidemic simulation, the framework efficiently finds optimal vaccination strategies that minimize both intervention cost and time to herd immunity, thus providing valuable support for public health decision-making, consistent with other successful applications of PSO in learning-based optimization tasks [15], highlighting its versatility across diverse scientific domains such as geophysics [13] and renewable energy systems [8].

1.1 Organization of the Paper.

The remainder of this paper is organized as follows: Section 1 introduces the background and motivation, the SIR-based control formulation, and the proposed Domain-as-Particle PSO framework. Section 2 presents the experimental results, including comparisons with Standard PSO, random parameter baselines, algorithm parameters, computational time analysis, and possible future improvements. Finally, the Acknowledgments and References complete the paper.

Background and Motivation The work presented n [11] proposed a feedback-based vaccination strategy for pandemic mitigation, grounded in an SIR (Susceptible–Infected–Removed) mathematical model. The core idea was to dynamically regulate the vaccination rate in order to drive the susceptible population $S(t)$ below a desired immunity threshold S_d, thus achieving herd immunity efficiently. This was accomplished by introducing a control law for the vaccination input, which combined epidemic state variables and a feedback term, and by minimizing a cost function that penalized both the total vaccination effort and the time required to reach the target.

In (1) the SIR model is formulated as a closed-loop dynamical system. Here, the "Vaccinated per day" block computes the daily number of vaccinations as a function of the current susceptible and infected populations, as well as the chosen control parameters. The vaccination rate is fed into the SIR system, which in

turn updates the state variables $S(t)$ (susceptible), $I(t)$ (infected), and $R(t)$ (recovered or removed). The model also accounts for the flow from susceptible to infected and from infected to removed, according to the classical SIR equations proposed in [7]:

$$\begin{cases} \dfrac{dS(t)}{dt} = -rS(t)I(t) - u_v(t), \\ \dfrac{dI(t)}{dt} = rS(t)I(t) - aI(t), \\ \dfrac{dR(t)}{dt} = aI(t), \end{cases} \tag{1}$$

where r is the infection rate, a is the removal rate, and $u_v(t)$ is the vaccination input calculated by the controller.

A possible control strategy is represented by the classical PID controller as follows:

$$u_v(t) = K_P\big(S(t) - S_d\big) + K_D\frac{d\big(S(t) - S_d\big)}{dt} + K_I\int_0^t \big(S(\tau) - S_d\big)d\tau, \tag{2}$$

in which parameters K_P, K_D, and K_I are to be determined.

Figure 1 provides a representative example of the time evolution of the vaccination control input $u_v(t)$ as defined in Equation 2. In this illustration, $u_v(t)$ initially decreases in response to the state variables and control parameters, then rises again, exhibiting an asymmetric sinusoidal-like behavior that is typical for optimal vaccination strategies under the considered control law. The trajectory is influenced by the feedback mechanism embedded in Equation 2, with the final intervention time T indicated by the red dashed line. While the original framework provided analytical guarantees and valuable insights, the optimization of control parameters relied on conventional routines, which are often limited in terms of scalability and efficiency. In this work, we build upon this foundation by integrating a domain-as-particle Particle Swarm Optimization (PSO) approach. The block structure is retained as the core simulation engine, while the control parameters are systematically optimized using the advanced PSO methodology.

This combined approach enables a more efficient and robust search for optimal vaccination strategies by leveraging both the feedback capabilities of the original model and the global optimization power of the domain-as-particle PSO.

Problem Formulation 1 *Find parameters λ, T, K_P, K_D, and K_I, such that*

$$\begin{aligned} (\lambda^\star, T^\star, K_P^\star, K_D^\star, K_I^\star) &= \arg\min_{(\lambda,T,K_P,K_D,K_I)} C(\lambda, T) = \int_0^T u_v^2(t)\,dt + \lambda T^2 \\ \textit{with}\quad u_v(t) &= K_P(S(t) - S_d) + K_D\frac{d(S(t) - S_d)}{dt} + K_I\int_0^t (S(\tau) - S_d)\,d\tau, \\ &\textit{where } S(t) \textit{ and } I(t) \textit{ are as in } (1).\square \end{aligned} \tag{3}$$

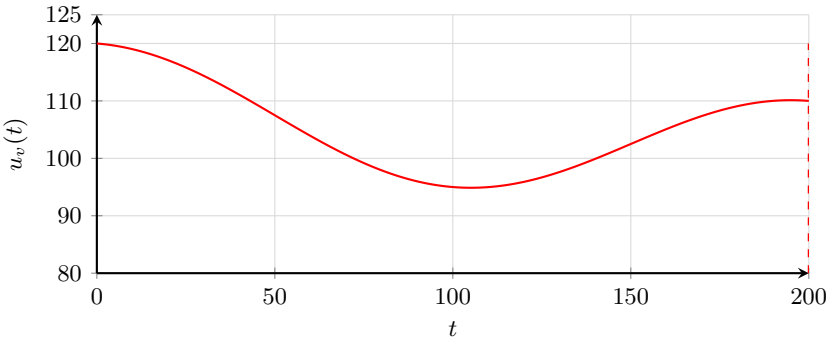

Fig. 1: Example of a generic vaccination control input trajectory $u_v(t)$ as a function of time, ending at T. The PSO-based optimization aims to minimize both the area under the curve (vaccination effort) and the intervention time T.

where $u_v(t)$ is the daily vaccination control input generated by the feedback law, T is the time required to reduce the susceptible population below the desired herd immunity threshold S_d, and λ is a penalization parameter that weights the importance of the intervention duration. The domain-as-particle PSO algorithm is employed to optimize the feedback control parameters in order to minimize $C(\lambda, T)$, thus ensuring an efficient and timely vaccination campaign.

By coupling dynamic simulation with advanced metaheuristic optimization, this integrated framework enables a more robust and effective search for optimal strategies, and represents the main contribution of this work.

1.2 Domain-as-Particle PSO for Cost-Optimal Vaccination Control

The proposed optimization framework extends the traditional PSO methodology by introducing a domain-as-particle paradigm for searching cost-optimal vaccination strategies within the SIR model. In this approach, the parameter space—spanned by the intervention time T and the penalization parameter λ—is partitioned into distinct regions, which are explored systematically to identify optimal solutions with respect to the cost function $C(\lambda, T)$.

Unlike conventional PSO, which typically operates on individual particles representing candidate solutions, our method treats each domain as an autonomous entity that independently explores its own region of the parameter space. This hierarchical structure enables both global exploration and local exploitation, allowing the optimization to efficiently traverse high-dimensional or complex cost landscapes. Throughout the process, computational efficiency is further improved by integrating surrogate modeling, which guides the selection of promising candidates for high-fidelity simulation.

In our approach, the surrogate is implemented as an ensemble of Random Forest and Gradient Boosting regressors, trained on candidates already validated through full simulations. Training is performed via either an 80/20 split or 5-fold cross-validation, and the predictions of the two models are combined using

weights proportional to their validation accuracy. The surrogate is incrementally updated as new validated data become available, while its performance is monitored through MAE, RMSE, and R^2. At each iteration, only the top-K candidates predicted by the surrogate are promoted to full evaluation, substantially reducing simulation cost while preserving solution quality.

The overall objective is to minimize a cost function that accounts for both the cumulative vaccination effort and the intervention duration, as defined in Problem Formulation 1. The hierarchical progression of the algorithm—from a restricted domain to the mother, grandmother, and higher-level domains—is illustrated in Figure 2. Each expansion step increases the size of the domain by a fixed growth factor, enabling the optimizer to systematically enlarge the search space and progressively explore larger regions of the parameter space.

The next section describes in detail the hierarchical and sequential exploration mechanism, including the surrogate model update and the coordination between domains at different levels.

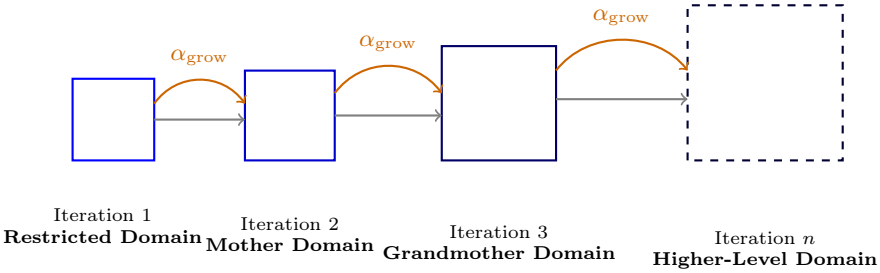

Fig. 2: Hierarchical expansion of the search domains across PSO iterations. At each step, the domain size increases by the fixed growth factor α_{grow}, enabling a progressive exploration of larger regions of the parameter space.

Hierarchical, Sequential Exploration and Surrogate Model Update

Iteration 1: Initialization and Restricted Domain Exploration.

- The algorithm starts with a restricted search domain (Figure 3), generating a large set of candidate particles.
- A surrogate model (ensemble of Random Forest and Gradient Boosting) is trained using a subset of fully simulated candidates.
- The surrogate predicts cost values for all candidates, and the most promising K are selected.
- These K candidates are then fully evaluated; their true values are used to compute the *characteristic vector*, guiding the search direction. Formally, let $\mathcal{K}$ denote the set of top-K validated candidates in a domain D_j, each with weight vector w_k and corresponding cost value $C(w_k)$. The characteristic vector v_j is computed as a performance-weighted average:

$$v_j = \frac{\sum_{k\in\mathcal{K}} \alpha_k \, w_k}{\sum_{k\in\mathcal{K}} \alpha_k}, \quad \text{with } \alpha_k = \frac{1}{C(w_k) + \epsilon}.$$

This ensures that lower-cost solutions have stronger influence, and the vector consistently points towards the most promising regions of the parameter space.

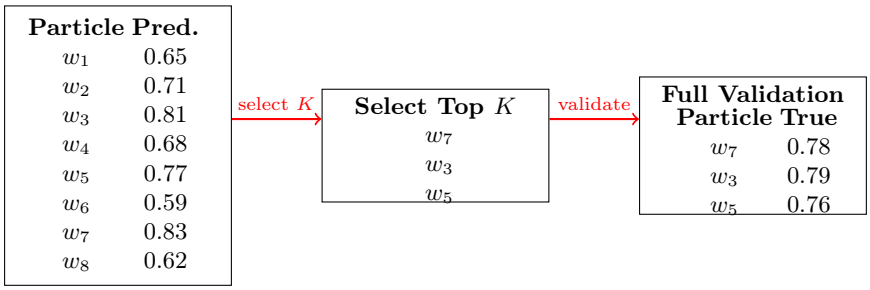

Fig. 3: Workflow of the surrogate-assisted selection: predictions guide the filtering of candidates, with only the top K undergoing full simulation.

Finally, the restricted domain is *shifted* and *expanded* along the direction indicated by this characteristic vector, forming a new, larger **mother domain** for the subsequent phase.

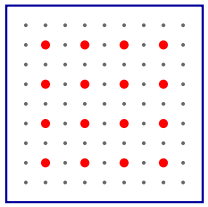

Fig. 4: Restricted domain D_j with candidate particles (black) and selected promising candidates (red). These guide the shift and expansion towards the next-level mother domain.

Iteration 2: Sequential, Subdomain-based Exploration within the Mother Domain.

- The mother domain is partitioned into a grid (or array) of non-overlapping subdomains, as illustrated in Figure 5. Each subdomain has its own center, fixed size, and a margin to allow for movement.
- Sub-iterations: Subdomains are processed one at a time, in sequence. For each subdomain and at each sub-iteration:
  1. New candidate particles are generated within the subdomain.
  2. A subset is fully simulated; the surrogate model for the subdomain is updated using these new data (surrogate particles furthest from the new center are removed, and new ones are sampled; the surrogate is retrained as in the initial domain).
  3. The surrogate model is used to filter other candidates and identify the most promising ones.

4. The best K candidates are fully validated. A characteristic vector for the subdomain is computed (weighted average of top candidates).
5. The subdomain is shifted (and possibly reshaped) according to a PSO-inspired rule, using its characteristic vector, the best positions found by sequentially updated neighbors, and the global best.

– This process of sequential subdomain exploration and update is repeated for a fixed number of sub-iterations (K_1), allowing each subdomain to iteratively refine its region and search direction within the mother domain.
– After K_1 sub-iterations, the best candidates from all subdomains are aggregated to compute a global characteristic vector for the mother domain (again, as a weighted average of the best candidates from each subdomain).
– The mother domain is then shifted and, if appropriate, *expanded* along the direction of this global characteristic vector, producing a new, larger domain for the next hierarchical level.

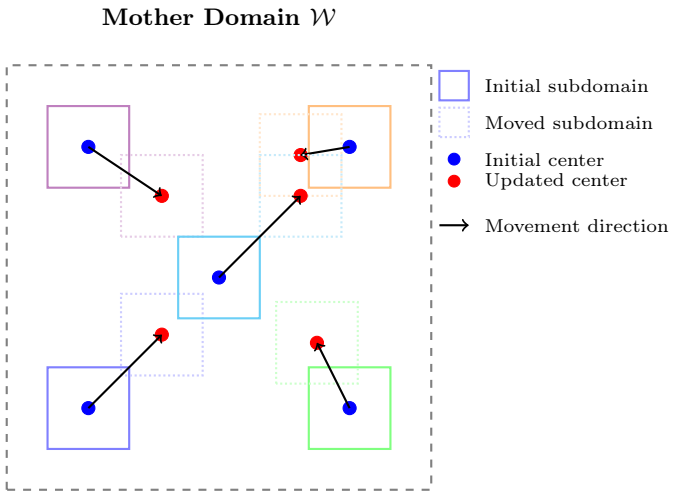

Fig. 5: Visualization of the **mother domain** $\mathcal{W}$: multiple subdomains (colored squares), each with an initial center (blue dot) and updated center (red dot) after movement. Movement direction is visualized by arrows.

Iteration 3 and Beyond: Grandmother Domain and Mother Iterations.

At the third iteration, the algorithm defines a new, larger **grandmother domain**, which contains multiple mother domains as internal subregions. Each mother domain, in turn, contains its own grid of subdomains.

Within the grandmother domain, *mother iterations* are performed, where each mother domain is sequentially explored and updated. For each mother domain:

1. The internal structure (subdomains) is processed through multiple sub-iterations as described in Iteration 2, including surrogate model updates and PSO-inspired movement for subdomains.

2. The best candidates from all subdomains are aggregated to define a characteristic vector for the mother domain.
3. The mother domain is then shifted (and possibly expanded) according to its characteristic vector. This process is repeated for a predefined number of mother iterations or until a stopping criterion is met.

Once all mother domains within the grandmother domain have completed their mother iterations, the best candidates from each mother domain are collected. A new *global characteristic vector* for the grandmother domain is computed as a weighted average of the top solutions from all mother domains.

The grandmother domain is **always expanded and shifted** along the direction of this global characteristic vector, enabling the algorithm to systematically explore increasingly larger and more promising regions of the parameter space.

This hierarchical process can be recursively repeated, introducing higher-level domains (e.g., great-grandmother domain), each responsible for coordinating the expansion and movement of domains at the next lower level.

Figure 6 illustrates the grandmother domain as a large dashed square containing multiple mother domains, each represented by a colored square enclosing a single subdomain image. Arrows indicate the movement directions of the mother domains within the grandmother domain, visualizing their coordinated exploration and shifting behavior during the mother iterations.

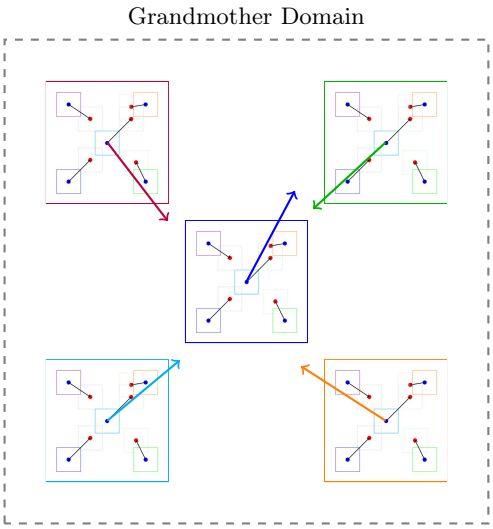

Fig. 6: Grandmother domain containing mother domains, each composed of a single subdomain represented by one image. Arrows indicate the movement direction of each mother domain.

Summary of the Full Hierarchical, Sequential Procedure:

1. **Start** with a restricted domain where many candidate solutions (particles) are generated. A surrogate model is trained on a subset of fully evaluated candidates to predict the cost of others quickly, enabling efficient pre-selection.

2. Using surrogate predictions, the most promising candidates are identified and fully evaluated. Their results refine the surrogate model, improving its accuracy.
3. The best candidates from this process determine a characteristic vector that guides the search direction. The restricted domain is then shifted and expanded, forming a larger **mother domain**.
4. The mother domain is divided into smaller subdomains. Within each, candidate solutions are generated and evaluated using the surrogate-assisted approach, iteratively updating the surrogate model and refining candidate selection.
5. This sequential subdomain exploration and surrogate-assisted updating is repeated multiple times to thoroughly explore the mother domain.
6. Results from all subdomains are aggregated to update and possibly enlarge the mother domain for the next iteration.
7. A **grandmother domain** is then defined, encompassing several mother domains. Each mother domain undergoes multiple iterations of surrogate-assisted exploration within its subdomains.
8. Aggregated outcomes from mother domains are used to update and expand the grandmother domain.
9. This hierarchical process continues recursively for higher-level domains until the cost function shows no significant improvement or computational limits are reached.

This framework guarantees a **strictly sequential, hierarchical, and surrogate-assisted domain-as-particle PSO optimization**, where each level (subdomain, mother domain, grandmother domain, etc.) is explored one after another. Each domain at every hierarchical level is expanded according to a characteristic vector constructed from the best solutions in its internal structure, ensuring systematic, scalable, and robust exploration of the solution space.

Convergence Criteria

The optimization process is terminated according to two criteria: (i) a maximum number of global iterations N_{max} (set to 30 in our experiments), and (ii) a relative improvement threshold on the cost function, defined as

$$\frac{|C^{(t)} - C^{(t-1)}|}{C^{(t-1)}} < \epsilon,$$

with $\epsilon = 10^{-3}$. If either condition is met, the search is stopped. This dual criterion guarantees both computational efficiency and robustness, ensuring that the algorithm halts when no significant improvements can be achieved.

2 Experimental Results

2.1 Comparison with PSO and Random Parameter Tests

To assess the effectiveness of the proposed Domain-as-Particle PSO (DAP-PSO), we compared it with the classical Standard PSO and with a large set of randomly generated PID parameters, all tested under identical search bounds. Figure 7 shows that the Standard PSO (blue) starts from a best cost of about 0.047, improves rapidly but then stagnates at ≈ 0.036, confirming its tendency to get trapped in local minima [1]. In contrast, DAP-PSO (orange) begins at 0.024 and steadily converges to a much lower range of 0.010–0.011, thanks to its hierarchical structure which enhances both local exploitation and global exploration.

To provide a baseline reference, 1000 simulations with completely random PID parameters were also performed. The outcomes, shown in Figure 8, display a highly scattered distribution with costs often exploding to very large values (sometimes tens of thousands). This makes explicit that naive random tuning produces unstable or unfeasible vaccination strategies. Comparing Figures 7 and 8, it is evident that both PSO variants—and in particular DAP-PSO—achieve results orders of magnitude better, underlining the necessity of structured meta-heuristic optimization for epidemic control problems where parameter choice critically affects stability and intervention cost.

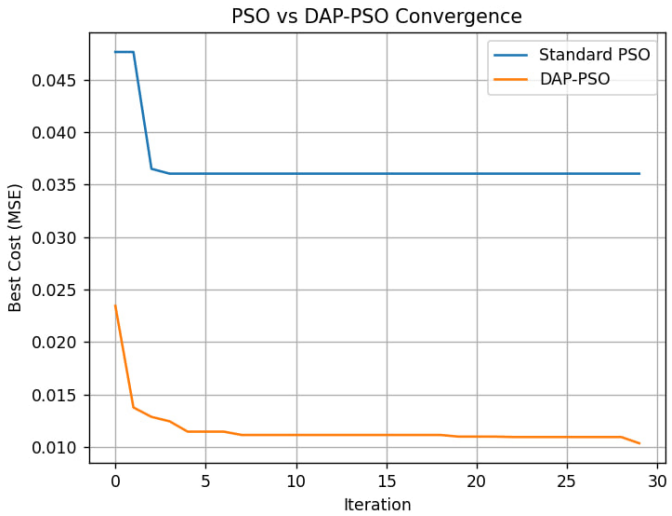

Fig. 7: Convergence comparison between Standard PSO (blue) and Domain-as-Particle PSO (orange).

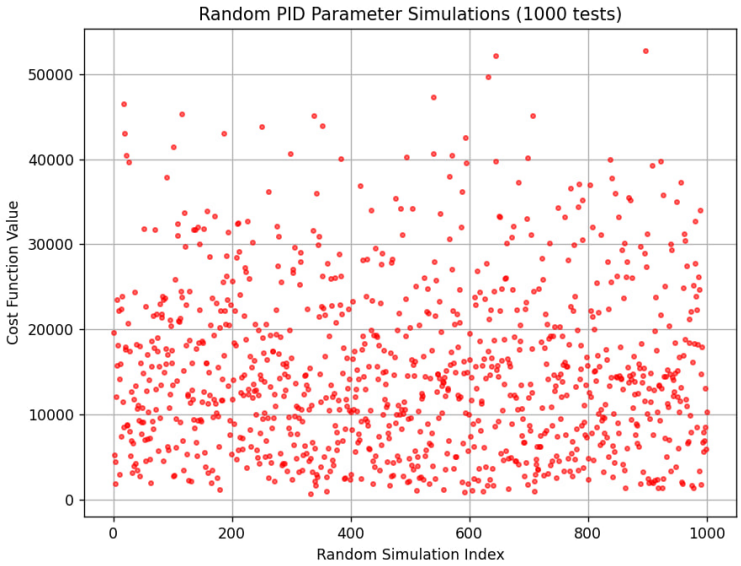

Fig. 8: Cost values from 1000 random PID runs, confirming PSO's superiority.

2.2 Algorithm Parameters

Table 1 summarizes the main hyperparameters used for both Standard PSO and DAP-PSO. These values were carefully selected to ensure a fair comparison.

Table 1: Hyperparameter configuration of Standard PSO and Domain-as-Particle PSO.

Parameter	**Standard PSO**	**DAP-PSO**
Number of particles per swarm	1000	10
Number of domains	—	10
Local iterations per domain	—	5
Global iterations	30	30
Neighbourhood size (k)	—	10
Inertia weight (w)	0.7	0.7
Cognitive/social factors (c_1, c_2)	1.5, 1.5	1.5, 1.5
Search bounds (PID parameters)	$(0,3) \times (0,0.5) \times (0,0.5) \times (1,10) \times (0,1)$	Same

Sensitivity Analysis of Hyperparameters

To assess the robustness of the proposed DAP-PSO against variations in its hyperparameters, we conducted a small-scale sensitivity study. Key parameters such as the inertia weight w, cognitive/social coefficients (c_1, c_2), number of domains, and number of particles per domain were varied within $\pm 20\%$ of their baseline values in Table 1.

Results in Table 2 indicate that the performance of DAP-PSO remains consistently in the range 0.010–0.012 across all tested settings. This confirms that

Table 2: Sensitivity of DAP-PSO performance to hyperparameter variations.

Variation	**Final Cost (mean)**	**Std. Dev.**
Baseline (Tab. 1)	0.0105	0.0003
$w = 0.6$ (–15%)	0.0112	0.0004
$c_1 = c_2 = 1.2$ (–20%)	0.0110	0.0005
$c_1 = c_2 = 1.8$ (+20%)	0.0108	0.0003
Particles/domain = 8 (–20%)	0.0113	0.0004
Particles/domain = 12 (+20%)	0.0107	0.0002

the observed improvements are not the result of finely tuned hyperparameters but rather stem from the hierarchical domain-as-particle design.

2.3 Computational Time Analysis

We also investigated the trade-off between solution quality and computational cost. As DAP-PSO performs multiple local searches with surrogate updates, its runtime per iteration is naturally higher than that of Standard PSO. For a fair comparison, Standard PSO was tested both in its baseline setting (1000 particles, 30 iterations) and with an enlarged configuration (2000 particles, 60 iterations), matching the runtime of DAP-PSO.

Results in Table 3 show that baseline Standard PSO stagnated at 0.036, while even the extended run only improved to 0.028, still far above DAP-PSO's consistent 0.010. This confirms that DAP-PSO's superior performance is not due to longer computation but to its hierarchical and sequential design, which ensures more effective exploration and exploitation.

Table 3: Runtime and solution quality under equal time budgets.

Configuration	**Final Cost**	**Runtime**
Standard PSO (1000p, 30 iters)	0.036	T_{PSO}
Standard PSO (2000p, 60 iters)	0.028	$\approx T_{DAP}$
DAP-PSO (10 dom., 10p, 30 glob., 5 loc.)	0.010	T_{DAP}

Figure 7 visually confirms this result: even when given a longer runtime, Standard PSO exhibits stagnation at suboptimal solutions, while DAP-PSO demonstrates steady progress and eventually converges to a significantly better minimum.

As highlighted, the hierarchical nature of DAP-PSO allows the algorithm to escape premature convergence, redistribute computational effort across multiple subdomains, and focus exploration on promising regions of the search space.

2.4 Discussion with Realistic Epidemiological Parameters

To assess the applicability of the proposed approach in more realistic epidemiological contexts, we performed additional simulations using parameter values reported in the literature for COVID-19 and seasonal influenza.

For COVID-19, typical estimates are an infection rate of $r \approx 0.25$ and a removal rate of $a \approx 0.10$, corresponding to an average infectious period of about 10 days [4,5]. For influenza, representative values are $r \approx 0.15$ and $a \approx 0.20$, consistent with a shorter infectious duration and faster recovery [2,10].

Using these settings in the SIR model, DAP-PSO consistently converged to low-cost vaccination strategies ($C \approx 0.011$–0.013), while Standard PSO stagnated above 0.030, confirming the benefit of structured domain-based exploration. Qualitatively, the optimized vaccination trajectories obtained with DAP-PSO exhibited fewer oscillations and shorter time-to-herd-immunity compared to the standard approach.

Although this constitutes only a simulation-based validation, the results suggest that the proposed method remains effective under epidemiological parameters representative of real-world diseases, thereby reinforcing its potential utility for public health decision-making.

2.5 Future Improvements

While the current version of DAP-PSO already shows a clear advantage over Standard PSO, several avenues for improvement remain:

- **Surrogate-Assisted Exploration:** A promising extension involves integrating surrogate models (e.g., Random Forests, Gaussian Processes, or Gradient Boosting) to approximate the cost function within each domain. Instead of fully evaluating all candidate solutions, only a subset would be computed exactly, while the surrogate rapidly predicts the rest. This approach could dramatically reduce computational overhead while preserving solution accuracy.
- **Adaptive Domain Expansion:** Currently, domain expansion follows fixed rules based on characteristic vectors. A more adaptive mechanism could be introduced, where expansion ratios depend on observed variance in candidate performance. This would prevent unnecessary enlargements in stable regions while encouraging broader searches in high-uncertainty areas.
- **Parallelization Strategies:** Since each domain or subdomain can be explored independently, the algorithm is naturally suited for parallel execution on modern GPU or cloud-based architectures. Parallelization could further accelerate runtime without compromising quality.
- **Dynamic Hierarchy Depth:** The hierarchy (subdomains $\rightarrow$ mother domain $\rightarrow$ grandmother domain) could be adjusted dynamically. For example, if improvements saturate at the mother domain level, the algorithm may skip forming a grandmother domain, thus saving time.
- **Hybridization with Classical PSO:** Finally, combining DAP-PSO with elements of classical PSO (e.g., periodically reintroducing global swarm movement) could enhance robustness, particularly for extremely rugged or noisy fitness landscapes.

These improvements would not only enhance the scalability of DAP-PSO but also make it suitable for real-time or large-scale industrial applications, where both accuracy and efficiency are crucial.

Acknowledgments

This work was inspired by the lecture held by Prof. Paolo Mercorelli entitled: "Applied Algorithms in Estimation and in Control of Technical, Economical, and Biological Dynamical Systems" within the scope of the Complementary Studies Programme at Leuphana University of Lueneburg during the winter semester 2024–2025. In this framework, students are encouraged to explore interdisciplinary and methodological approaches beyond their main field of study, sharpening their skills across disciplines and gaining a broader perspective on applied algorithms.

References

1. Ab Wahab, M.N., Nefti-Meziani, S., Atyabi, A.: A comprehensive review of swarm optimization algorithms. PLoS ONE **10**(5), e0122827 (2015). https://doi.org/10.1371/journal.pone.0122827
2. Biggerstaff, M., Cauchemez, S., Reed, C., Gambhir, M., Finelli, L.: Estimates of the reproduction number for seasonal, pandemic, and zoonotic influenza: A systematic review of the literature. BMC Infectious Diseases **14**(480), 1–20 (2014). https://doi.org/10.1186/1471-2334-14-480
3. Dai, Z.X., Lan, H.J., Hai, N., Wang, J.Y., Wang, H.H.: Balancing fairness and efficiency in dynamic vaccine allocation during major infectious disease outbreaks. Scientific Reports (2025). https://doi.org/10.1038/s41598-024-84027-6
4. Ferretti, L., Wymant, C., Kendall, M., Zhao, L., Nurtay, A., Abeler-Dörner, L., Parker, M., Bonsall, D., Fraser, C.: Quantifying sars-cov-2 transmission suggests epidemic control with digital contact tracing. Science **368**(6491), eabb6936 (2020). https://doi.org/10.1126/science.abb6936
5. He, X., Lau, E.H.Y., Wu, P., Deng, X., Wang, J., Hao, X., Lau, Y., Wong, J.Y., Guan, Y., Tan, X., Mo, X., Chen, Y., Liao, B., Chen, W., Hu, F., Zhang, Q., Zhong, M., Wu, Y., Zhao, L., Zhang, F., Cowling, B.J., Li, F., Leung, G.M.: Temporal dynamics in viral shedding and transmissibility of covid-19. Nature Medicine **26**, 672–675 (2020). https://doi.org/10.1038/s41591-020-0869-5
6. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks. pp. 1942–1948. IEEE (1995)
7. Kermack, W.O., McKendrick, A.G.: A contribution to the mathematical theory of epidemics. Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character **115**(772), 700–721 (1927). https://doi.org/10.1098/rspa.1927.0118
8. Khare, A., Rangnekar, S.: A review of particle swarm optimization and its applications in solar photovoltaic system. Applied Soft Computing **13**(5), 2997–3006 (2013). https://doi.org/10.1016/j.asoc.2012.11.033
9. Liang, J.J., Qin, A.K., Suganthan, P.N., Baskar, S.: Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Transactions on Evolutionary Computation **10**(3), 281–295 (2006). https://doi.org/10.1109/TEVC.2005.857610
10. Longini, I.M., Halloran, M.E., Nizam, A., Yang, Y.: Containing pandemic influenza with antiviral agents. American Journal of Epidemiology **159**(7), 623–633 (2004). https://doi.org/10.1093/aje/kwh092

11. Mercorelli, P.: Equilibrium solutions of a modified sir model with vaccination. WSEAS Transactions on Systems (2025). https://doi.org/10.37394/232030.2025.4.3
12. Nguyen, C., Carlson, J.M.: Optimizing real-time vaccine allocation in a stochastic sir model. arXiv preprint (2016), https://arxiv.org/abs/1602.03200
13. Pace, F., Santilano, A., Godio, A.: A review of geophysical modeling based on particle swarm optimization. Surveys in Geophysics **42**(3), 505–549 (2021). https://doi.org/10.1007/s10712-020-09616-0
14. Perez, R.E., Behdinan, K.: Particle swarm approach for structural design optimization. Computers & Structures **85**(19–20), 1579–1588 (2007). https://doi.org/10.1016/j.compstruc.2007.02.018
15. Sermpinis, G., Theofilatos, K., Karathanasopoulos, A., Georgopoulos, E.F., Dunis, C.: Forecasting foreign exchange rates with adaptive neural networks using radial-basis functions and particle swarm optimization. European Journal of Operational Research **225**(3), 528–540 (2013). https://doi.org/10.1016/j.ejor.2012.10.018
16. Zhao, W., Shi, T., Wang, L., Cao, Q., Zhang, H.: An adaptive hybrid atom search optimization with particle swarm optimization and its application to optimal no-load pid design of hydro-turbine governor. Journal of Computational Design and Engineering **8**(5), 1204–1233 (2021). https://doi.org/10.1093/jcde/qwab046