Optimization of JenaFox Walking Speed Using Particle Swarm Optimization

Assignment 9
Name: Nicolas Helio Cunha Nabrink

Submission Date: 05.07.2024
required time (hh:mm): 12:00

Introduction

Particle Swarm Optimization (PSO) is a powerful algorithm inspired by the social behavior of birds flocking or fish schooling, and it has been widely used to solve optimization problems in various domains. In this project, we apply PSO to maximize the walking speed of the JenaFox multibody simulation. The simulation uses a neural network controller that adjusts three critical parameters: PEAhip (Passive Elastic Actuation of the hip), AEAhip (Active Elastic Actuation of the hip), and GMh (hip motor gain). These parameters are crucial in determining the efficiency and speed of the walking mechanism.

The primary objective of this project is to implement a PSO algorithm that optimizes these three parameters to achieve the highest possible walking speed. The PSO algorithm will be developed and tested in two phases: a basic implementation and an extended version with velocity limiting. By comparing the swarm behavior and optimization results of both implementations, we aim to understand the impact of velocity limiting on the optimization process and the performance of the JenaFox simulation.

This essay will detail the reasoning behind the choice of optimization parameters, provide a comprehensive overview of the implementation process, assess the optimization performance, visualize the results, and discuss the findings. Through this structured approach, we aim to demonstrate the effectiveness of PSO in optimizing complex systems and provide insights into the behavior and performance of the JenaFox simulation under different optimization scenarios.

Definitions

Table 1 gives an overview to the most important symbols, constants and variables

Symbol	Property	Value	Unit
N	Number of particles	25	-
n	Number of dimensions	3	-
ϕ_1	Cognitive coefficient	-	-
ϕ_2	Social coefficient	-	-
$v_{\max}$	Maximum velocity	3.0	m/s
σ	Neighborhood size	3	-
w	Inertia weighting	0.4	-
x	Position of particles	-	-
v	Velocity of particles	-	-
b	Best known position of particles	-	-
$f\,x$	Objective function value	-	-
$f\,b$	Best objective function value	-	-
x_i	Position of particle i	-	-
v_i	Velocity of particle i	-	-
b_i	Best position of particle i	-	-
h_i	Best position in the neighborhood of particle i	-	-

Table 1: Definitions of Symbols and Properties

Approach and Implementation

The implementation of the Particle Swarm Optimization (PSO) algorithm was carried out in accordance with the guidelines provided in Simon 2013. Section 11 of the book outlined a structured approach to PSO, including the initialization of particles, the updating of velocities and positions, and the incorporation of various coefficients to influence particle movement.

Implementation Steps

1. **Initialization**: The swarm was initialized with $N = 25$ particles, each having $n = 3$ dimensions representing the parameters PEAhip, AEAhip, and GMh. The positions x were randomly distributed within a specified range using the `unifrnd` function, and the initial velocities v were set to zero. The initial best known positions b and their corresponding function values $f\,b$ were also determined.

2. **Velocity Update**: The velocity of each particle was updated using the formula:

   $$v_i(t+1) = w \cdot v_i(t) + \phi_1 \cdot (b_i - x_i) + \phi_2 \cdot (h_i - x_i)$$

   where w is the inertia weighting, ϕ_1 and ϕ_2 are random vectors influenced by the cognitive and social coefficients respectively, and h_i represents the best position in the neighborhood of particle i. The implementation followed the guidelines from Section 11.1 of Simon 2013 for this update mechanism.

3. **Position Update**: The new position of each particle was computed as:

   $$x_i(t+1) = x_i(t) + v_i(t+1)$$

   Additionally, constraints were applied to ensure GMh remained within the specified bounds (0 to 2).

4. **Velocity Limiting**: To prevent particles from moving too rapidly and potentially missing optimal regions, velocity limiting was implemented:

   $$\text{if } \|v_i\| > v_{\max},\ v_i = \left(\frac{v_i}{\|v_i\|}\right) \cdot v_{\max}$$

   This was done in accordance with Section 11.2 of Simon 2013.

5. **Evaluation**: The objective function was evaluated using the `simFox` function, which provided the forward velocity of the JenaFox simulation as the output. The objective was to maximize this velocity, thus the fitness values were inverted for minimization within the PSO framework.

6. **Dynamic Plotting**: The implementation included dynamic plotting to visualize the best velocity over iterations and the positions of particles in the 3D parameter space. This provided a clear view of the optimization progress and particle movement.

7. **Optimization Performance**: The performance was assessed in terms of runtime, utilizing the department's computer pool to handle computationally intensive tasks. The results were saved using the `save` command, ensuring data persistence.

Formulas Used

- Velocity update:

  $$v_i(t+1) = w \cdot v_i(t) + \phi_1 \cdot (b_i - x_i) + \phi_2 \cdot (h_i - x_i)$$

- Position update:

$$x_i(t+1) = x_i(t) + v_i(t+1)$$

- Velocity limiting:

$$\text{if } \|v_i\| > v_{\max},\ v_i = \left(\frac{v_i}{\|v_i\|}\right) \cdot v_{\max}$$

By adhering to these steps and formulas, the PSO algorithm was successfully implemented, balancing exploration and exploitation through the cognitive and social components, while ensuring stability with inertia weighting and velocity limiting. The dynamic plots and performance assessments provided valuable insights into the optimization process and outcomes.

Parameter Selection

The parameters for the Particle Swarm Optimization (PSO) algorithm were chosen based on the guidelines provided in Simon 2013, as well as considerations from previous assignments. This subsection details the reasoning behind each parameter choice:

Cognitive and Social Coefficients

The cognitive coefficient (ϕ_1) and the social coefficient (ϕ_2) were both set using uniformly distributed random values between 0 and 2:

$$\phi_1, \phi_2 \sim \text{Unif}(0, 2)$$

According to the PSO literature and guidelines, setting ϕ_1 and ϕ_2 in this range helps balance exploration and exploitation. The cognitive component (ϕ_1) encourages particles to explore their own best-known positions, while the social component (ϕ_2) encourages them to move towards the global best positions found by their neighbors. Using a uniform distribution ensures that these influences vary dynamically during the optimization process, potentially improving convergence to an optimal solution.

Inertia Weighting

The inertia weighting (w) was set to 0.4:

$$w = 0.4$$

Inertia weighting plays a crucial role in balancing the exploration and exploitation abilities of the swarm. A lower value of w reduces the influence of previous velocities, thus enhancing the algorithm's ability to explore new regions of the search space. The value of 0.4 was chosen based on the recommendation in Section 11.3.1 of Simon 2013, which suggests that lower inertia weights can help in fine-tuning the search process near optimal solutions. Additionally, similar values are often recommended in the literature to prevent premature convergence while maintaining a reasonable level of exploration.

Maximum Velocity

The maximum velocity ($v_{\max}$) was set to 3:

$$v_{\max} = 3$$

This parameter constrains the maximum step size a particle can take in a single iteration. Setting $v_{\max}$ to 3, around 10% the size of the search space, helps ensure that particles do not move too quickly through the search space, which could cause them to overshoot optimal regions Simon 2013. It is important to note that this limiter is applied to every dimension of the search space. A more controlled movement allows for a more thorough search and fine-tuning near the best solutions found.

Initial Position

The initial positions of the particles were set using a uniform distribution within the range of -15 to 15:

$$x \sim \text{Unif}(-15, 15)$$

This range was chosen to ensure that the initial positions cover a broad area of the search space. The choice of this range was influenced by results from a previous assignment, where the optimal values for PEAhip and AEAhip were found to be approximately 5 and -10, respectively. Including these values within the initial search interval increases the likelihood that the particles will explore regions near previously known good solutions, potentially accelerating convergence to an optimal solution.

Number of Particles and Maximum Iterations

The number of particles (N) was set to 30 and the maximum number of iterations was set to 2. These values were chosen to balance the algorithm's runtime and performance, considering the limited processing hardware capacity. A larger number of particles and iterations could potentially lead to better optimization results but would require significantly more computational resources and time. By choosing $N = 25$ and 30 iterations, the algorithm runs efficiently within the available hardware constraints while still exploring the search space adequately.

Number of Neighbors

The number of neighbor particles (σ) was set to 3. The number of neighbors small is proven to help avoiding local minima and providing better global behaviour Simon 2013.

By carefully selecting these parameters, the PSO algorithm was configured to effectively balance exploration and exploitation, ensuring a robust search process that can adapt dynamically to the characteristics of the optimization landscape. These choices were guided by both theoretical insights from Simon 2013 and practical considerations from past experience.

Results

In this section, the results of the Particle Swarm Optimization (PSO) algorithm are presented. The performance of the optimization process is illustrated through two main visualizations: a plot of the best velocity achieved in each iteration and a scatter plot depicting the swarm behavior. These visualizations are provided for both trials - one with a velocity limiter applied and one without. The comparison between these two trials highlights the impact of the velocity limiting constraint on the optimization process and the overall swarm dynamics.

Results without Velocity Limiter

The results of the PSO trial without a velocity limiter are presented in this subsection. The following graphs provide insights into the optimization process and the swarm behavior.

Best Velocity per Iteration

Figure 1 shows the best velocity achieved by the swarm in each iteration. The PSO algorithm successfully improved the walking speed, achieving a maximum velocity of 0.90007 m/s. The corresponding parameter values at this best velocity are:

- PEAhip = 4.8803
- AEAhip = -8.6124
- GMh = 1.7167

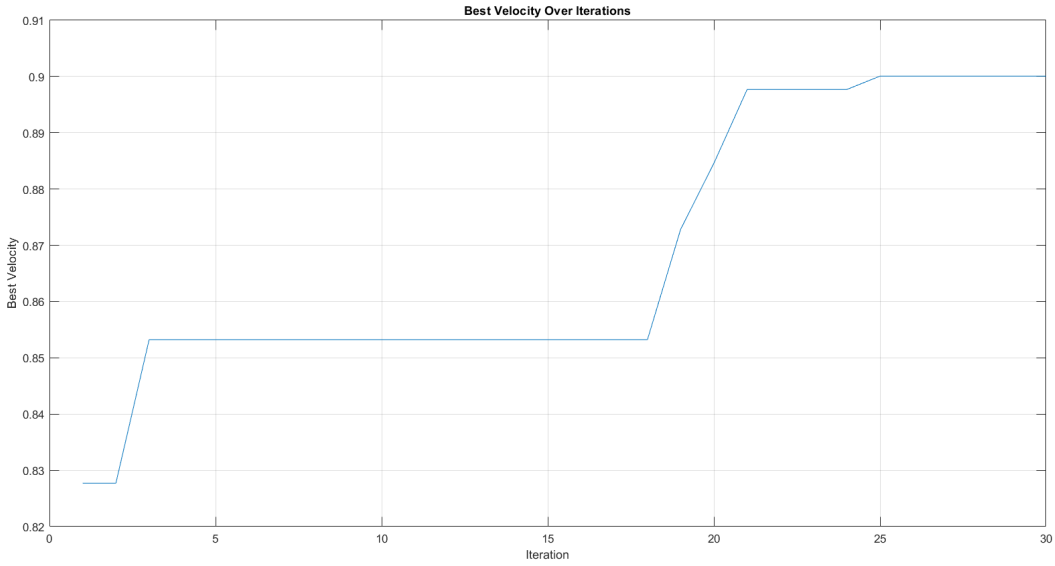

Figure 1: Best Velocity per Iteration for the Trial without Velocity Limiter

Swarm Behavior

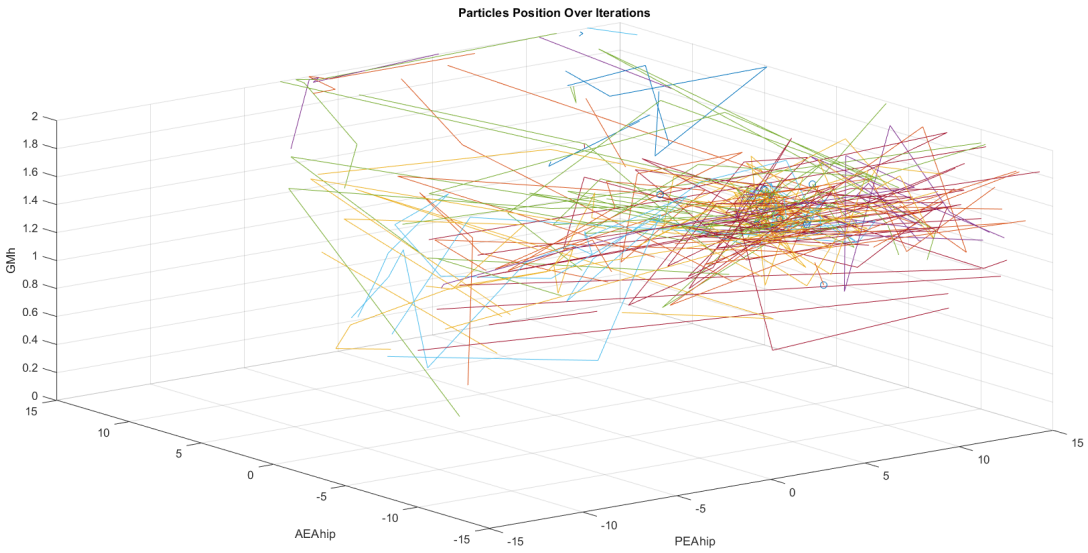

Figure 2: Swarm Behavior for the Trial without Velocity Limiter

Figure 2 presents a scatter plot illustrating the movement of the particles throughout the iterations. The particles start from their initial random positions and gradually move towards the region of the optimal solution. Over the iterations, the particles get concentrated near the final solution position, demonstrating the convergence of the swarm.

The total elapsed time for this simulation was 3,301 seconds, which is approximately 55 minutes. This runtime reflects the computational effort required for the PSO algorithm to explore the search space and converge to an optimal solution without the velocity limiting constraint.

Results with Velocity Limiter

The results of the PSO trial with a velocity limiter of 3 m/s are presented in this subsection. The following graphs provide insights into the optimization process and the swarm behavior with the applied velocity constraint.

Best Velocity per Iteration

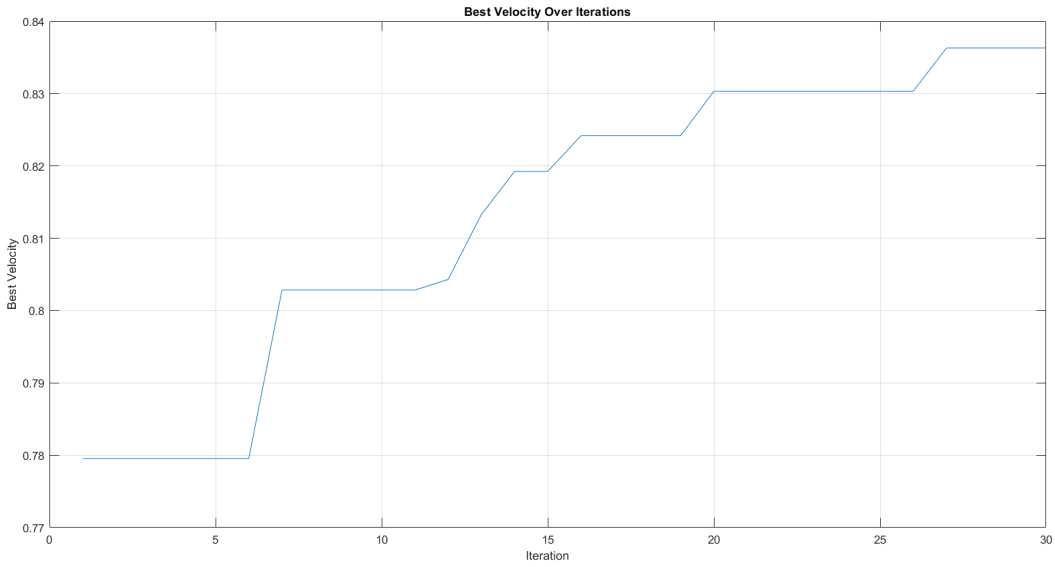

Figure 3: Best Velocity per Iteration for the Trial with Velocity Limiter

Figure 3 shows the best velocity achieved by the swarm in each iteration. The PSO algorithm, with the velocity limiter applied, achieved a maximum velocity of 0.8363 m/s. The corresponding parameter values at this best velocity are:

- PEAhip = 7.3253
- AEAhip = -11.9883
- GMh = 1.68895

Swarm Behavior

Figure 4 presents a scatter plot illustrating the movement of the particles throughout the iterations. The particles start from their initial random positions and gradually move towards the region of the optimal solution. Over the iterations, the particles get concentrated near 3 different minimum solutions, demonstrating the convergence of the swarm.

The total elapsed time for this simulation was 4,585.2 seconds, which is approximately 76 minutes. This runtime reflects the computational effort required for the PSO algorithm to explore the search space and converge to an optimal solution with the velocity limiting constraint applied.

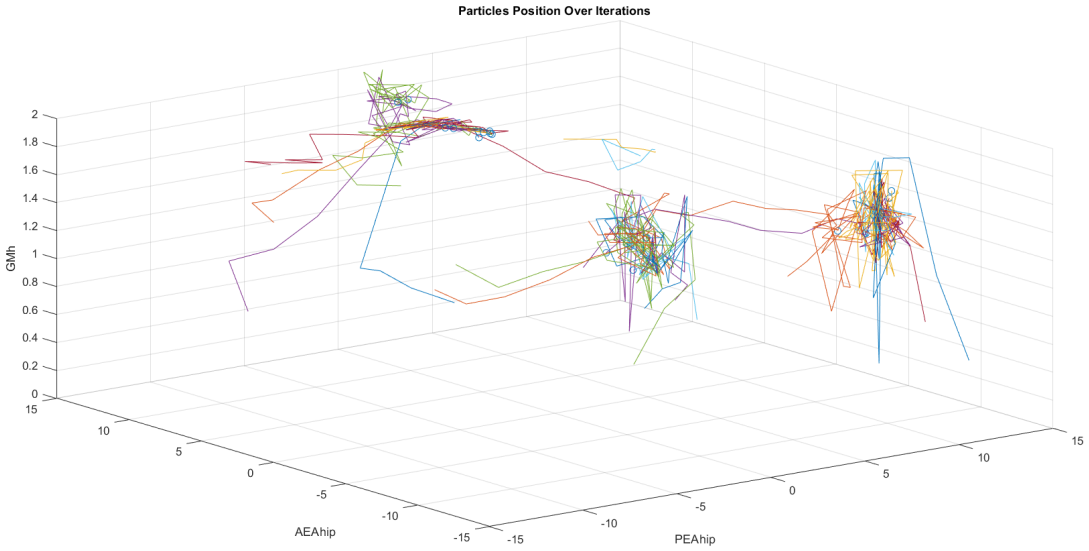

Figure 4: Swarm Behavior for the Trial with Velocity Limiter

Discussion

The Particle Swarm Optimization (PSO) algorithm was implemented and tested to maximize the walking speed of the JenaFox multibody simulation by optimizing three parameters: PEAhip, AEAhip, and GMh. The simulations were conducted with and without a velocity limiting constraint, and the results were analyzed and compared.

The computation for both trials took a significant amount of time, exceeding one hour even with a robust hardware configuration featuring an Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz 2.59GHz and 16GB RAM. This runtime highlights the computational intensity of the PSO algorithm, particularly when applied to complex simulations like JenaFox. Ideally, the number of iterations should be increased to at least 100 to ensure a thorough exploration of the search space and to enhance the likelihood of converging to a global optimum. However, due to practical constraints and the scope of this project, the number of iterations was limited.

The results of the trial without the velocity limiter showed a best objective (walking speed) of 0.90007 m/s with the following parameter values: PEAhip = 4.8803, AEAhip = -8.6124, and GMh = 1.7167. In comparison, the trial with the velocity limiter applied (3 m/s) resulted in a best objective of 0.8363 m/s with parameter values: PEAhip = 7.3253, AEAhip = -11.9883, and GMh = 1.68895. Both achieved the goal of overcoming 5m of distance in the simulation.

The simulation with the velocity limiter demonstrated a smoother convergence towards the optimal solution, as evidenced by the more gradual and consistent improvement in the best velocity per iteration. However, this smoother convergence required more iterations to achieve results comparable to the trial without the velocity limiter. The velocity limiter helps in preventing excessive and abrupt changes in particle velocities, promoting a more stable optimization process, but it also slows down the convergence rate.

Given the limited number of iterations in this study, the initial positions of the particles had a substantial impact on the final results. The PSO algorithm's performance and the quality of the obtained solutions were heavily influenced by the initial random distribution of the particles. This is particularly evident in cases where multiple trials with the same configuration yielded different outcomes due to variations in initial positions.

The swarm behavior graphs provide an interesting visualization of the particles' movement and convergence. Figure 2 shows that, in the trial without the velocity limiter, all particles converged to a single region. This indicates that the algorithm found a dominant optimal solution. In contrast, Figure

4 for the trial with the velocity limiter reveals three distinct regions where the particles concentrated. This suggests the presence of multiple local minima, highlighting the impact of the velocity constraint on the swarm's exploration and exploitation balance.

For future implementations, increasing the number of iterations would likely lead to more reliable and optimized results. Additionally, experimenting with different swarm sizes (N) and adjusting other PSO parameters could further improve the algorithm's performance. To mitigate the impact of initial particle positions, multiple runs of the algorithm could be averaged to obtain a more consistent and representative outcome.

In conclusion, the PSO algorithm proved effective in optimizing the walking speed of the JenaFox simulation, with clear differences observed between trials with and without velocity limiting. While the computational intensity and the influence of initial conditions were notable challenges, the results provide valuable insights into the behavior and performance of PSO in this context.

References

Simon, D. (2013). *Evolutionary optimization algorithms*. John Wiley & Sons.

AI usage declaration

tools used

ChatGPT

used prompts

1-I have to write an essay based on the following problem: (problem copied from moodle) The first part of my essay is problem statement and introduction, please write that for me.

2-I implemented the solution following the steps: (...) Write a text defining the approach and implementation for my essay based on those steps

3- My results show the following conclusions: (...) write it in a paragraph form

4-Now I need you to write a discussion for that essay, I want you to note the following points in it: (...)

comments

It was very useful to write paragraphs when given bullet points of what I wanted to include