OPTIMIZATION METHODS : PARTICLE SWARM OPTIMIZATION

Mohamed Keteb
ENSAE Paris
at King's College London
mohamed.keteb@ensae.fr

Francesca Crucinio
King's College London
Senior Lecturer Mathematics
francesca_romana.crucinio@kcl.ac.uk

ABSTRACT

In this report, we review several well-known methods for the important problem of optimization. We aim to provide both theoretical and empirical perspectives. The first method we will explore is the Particle Swarm Optimization (PSO) method, introduced by James Kennedy and Russell Eberhart in 1955. In the second part of the project, we will delve into another optimization approach using stochastic calculus. Specifically, we will examine Consensus-Based Optimization, developed by René Pinnau et al. in 2017.

***K*eywords** Simulation, Optimization, CBO, PSO

Contents

1 Introduction

Optimization is a crucial problem across many fields, including chemistry, operations research, and machine learning. In machine learning, optimization typically involves minimizing a loss function specific to the problem at hand. For neural networks, after forward propagation, the gradient can be computed, allowing for stochastic gradient descent. This strategy has been highly successful due to its ability to be parallelized. However, what if we could optimize a function without computing gradients or relying on functional analysis tools to determine the local shape of a function? Particle Swarm Optimization (PSO) and Consensus-Based Optimization (CBO), as introduced briefly in the abstract, are methods designed to find the minimum or maximum of a function without using gradients or similar techniques.

In the first part, we will present Particle Swarm Optimization (PSO) from a theoretical perspective. It is worth noting that the theory behind PSO is not as developed as that for Consensus-Based Optimization (CBO). We will also provide some intuition about this method before implementing the algorithm on various functions. At the end, we will attempt to enhance this method by incorporating additional information. Since we are dealing with functions that may not be differentiable, we want to include gradient information, although it is not always available. Therefore, we will use the concept of the subgradient, which we will define later.

****CBO****

The general problem:

In this paper, we focus on the minimization problem. However, without loss of generality, all the theory presented is also valid for the maximization problem.

We want to find,

$$x^\star \in \arg\min_{x\in\mathbb{R}^n} f(x)$$

Where $f \in \mathcal{C}(\mathbb{R}^n, \mathbb{R})$ is some continuous function. Note that g is not necessarily convex, so setting the gradient, if it exists, to zero does not necessarily yield a global minimum, but only a local one.

2 Particule Swarm Optimization (PSO) [1, 2]

2.1 Heuristic

The **PSO** method is a dynamic approach to optimization that can be viewed, as the name suggests, as a swarm of particles interacting with each other. Analogous to the behavior of flocks of birds, **PSO** is inspired by how a group of birds might search for food to maximize their utility. The birds interact with each other until the entire flock converges to the area that maximizes their utility. Birds rely on their own experiences and follow the group to some extent, thus they also consider the overall position of the group. These interactions represent a trade-off that the group must manage to converge as quickly as possible to the optimal area.

The **PSO** algorithm aims to mimic the complex behavior of flocks.

2.2 Current Scheme of PSO

The dynamic system is composed of N particles, $\boldsymbol{x_t^i}$ is the position in $\mathbb{R}^n$ of the i-th particle at time t. Likewise $\boldsymbol{v_t^i}$ is the velocity of the i-th particle at time t. The following system gives the dynamic of the N particles across time.

$$v_{t+1}^i = wv_t^i + c_1 r_1 (pbest_t^i - x_t^i) + c_2 r_2 (gbest_t - x_t^i) \tag{1}$$
$$x_{t+1}^i = x_t^i + v_{t+1}^i \tag{2}$$

At time t, the personal best, $\boldsymbol{pbest_t^i}$, is the best point in $\mathbb{R}^n$ visited so far by the i-th particle, minimizing the objective function $\boldsymbol{g}$, however the global best, $\boldsymbol{gbest_t}$, is the best point visited so far by all particles.

$$pbest_t^i := \arg \min_{u \in \{x_{0:t-1}^i\}} f(u)$$
$$gbest_t := \arg \min_{u \in \{x_{0:t-1}^{1:N}\}} f(u)$$

In the velocity equation (1), two terms contribute to the velocity, aiming to mimic the behavior of flocks.

2.2.1 The exploration term and The exploitation term

The exploration term or the cognitive component, $\boldsymbol{c_1 r_1 (pbest_t^i - x_t^i)}$, depends on the particle itself. As it contains the best minimizer find so far, this term gives a local research of the best minimizer. Therefore this term mimics the fact that the particle relies on its own experience to find the best point, like birds in a flock. The exploitation term or the social component, $\boldsymbol{c_2 r_2 (gbest_t - x_t^i)}$, depends on all particles. This term mimics the fact that each particle can follow the group to find the best minimizer. The exploration term and

Therefore each particle can rely on the group and on its own experience, there is a trade-off between these two behaviors. we have to choose constants $\boldsymbol{c_1, r_1, c_2}$ and $\boldsymbol{r_2}$ that control this trade-off in order to have better performance of the algorithm.

2.2.2 Parameters of the dynamic

In the cognitive and social component, $\boldsymbol{r_1, r_2} \hookrightarrow \mathcal{U}[0, 1]$ are randomly chosen at each iteration during the evolution of the particles. $\boldsymbol{c_1, c_2} \in [0, 2]$ are user-supplied fixed coefficients that can be viewed as coefficient of trust, N is, as already mentioned, the total number of particles and T, the number of periods in the dynamic.

For instance, if N is very large, it increases the complexity of the algorithm. However, with N large enough, finding the best minimizer improves because we explore the space more extensively, reducing the likelihood of getting stuck in a local minimum, especially for multi-modal functions. On the other hand, if N is not large enough, the candidate minimizer may be far from the true minimum because there is less information shared within the group.

If we set $c_1 = 0,\ c_2 \neq 0$, then individual exploration is eliminated, and each particle has no memory. Intuitively, if a particle becomes trapped in a local minimum, the entire group will converge to this local minimum. On the other hand, if we set $c_1 \neq 0,\ c_2 = 0$, we have no social interaction, and each particle relies solely on its own research, allowing for more exploration of the space. However, it is often more efficient to have $c_1, c_2 \neq 0$ because it combines both behaviors. By finding the right combination of constants, the algorithm can yield a better minimizer

2.2.3 Inertia coefficient and velocity clamping

Often, optimization problems are constrained, and particles must adhere to these constraints. If c_1, c_2 are too large, depending on the problem, the velocity will also become too large. As we can observe in the second dynamic equation, (2), The next position of the particle may not necessarily satisfy the constraints. That is why the velocity at the time $t+1$ in the equation (1) depends on the previous velocity which is controlled by the inertia coefficient $\boldsymbol{w}$. In the first instance, w needs to be sufficiently close to 1 to ensure that the velocity is not too small, which could lead to issues of exploration. On the other hand, we need to set a maximum velocity $\boldsymbol{v_{max}}$ for the particles, which depends on the shape of the constraints .

We simplify the problem and it becomes :

$$x^\star \in \arg\min_{x \in \mathcal{I}} f(x)$$

with $\mathcal{I} \subset \mathbb{R}$ an interval,

In order to prevent the particles from moving excessively beyond the search space, we employ a technique known as velocity clamping, which limits the maximum velocity of each particle.

- if $\mathcal{I} =]-x_{lim}, x_{lim}[$ is symmetric : $v_{max} = k \cdot x_{lim}$
- if $\mathcal{I} =]x_{inf}, x_{sup}[$ is asymmetric : $v_{max} = k \cdot \frac{x_{sup} - x_{inf}}{2}$

With $k \in]0, 1[$ represents a user-supplied velocity clamping factor. If a particle i at time t has a velocity v_t^i larger than v_{max}, then we set it to v_{max}. If v_t^i is lower than $-v_{max}$, we set it to $-v_{max}$.

2.3 The PSO algorithm

Having introduced **PSO** work-frame, we can now describe the algorithm. We describe the algorithm for a constrained optimization problem, with $\mathcal{I}$ the constrained space.

Algorithm 1 PSO

Initialise c_1, c_2, T, k and N particles, $x_0^i \hookrightarrow \mathcal{U}(\mathcal{I})$ for $i = 1 \dots N$
for $t = 1$ **to** T **do**
    Update individual and global best fitnesses and positions along with r_1, r_2
    Update individual position and velocity (clamped or not)
end for
Return $gbest_T$

This algorithm does not have a specific stopping criterion. It stops when the number of iterations T is reached, or we can use a variant that stops when there are no further updates of the global best minimizer.

2.4 Implementing PSO Github

In this part, we implement the algorithm on python and we test it on four tricky functions that we can find in the paper of Consensus-Based Optimization with personal best.

All of those functions are differentiable and non-convex, then a method that do not rely on the setting to Zero the gradient is pretty interesting.

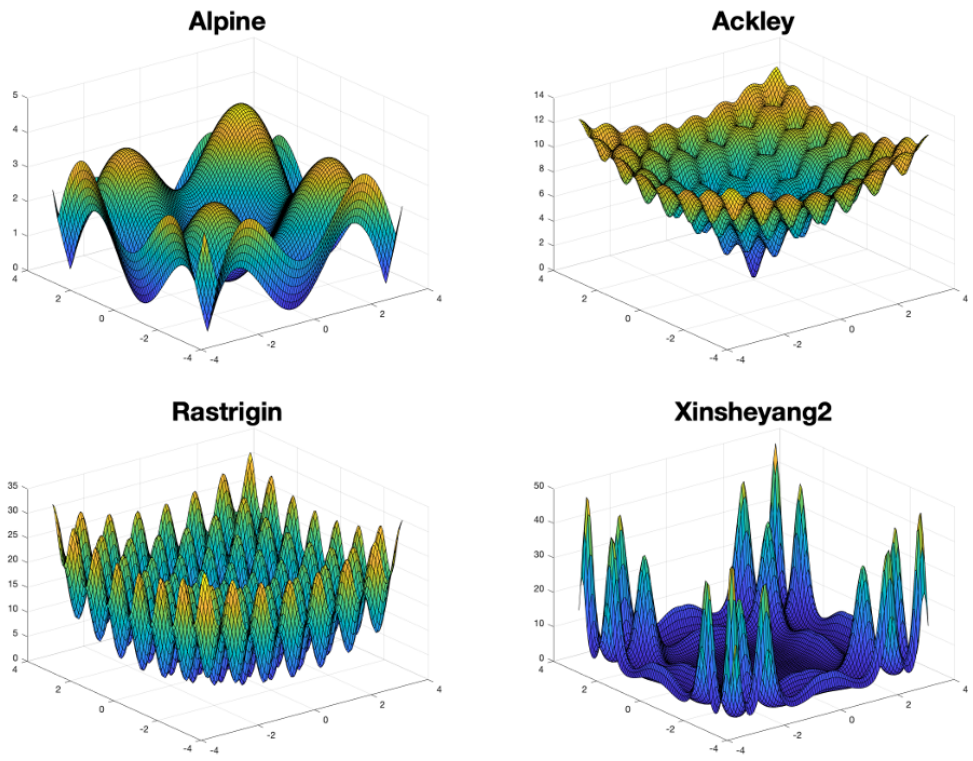

Figure 1: Four test functions

Test functions [5]

$\forall x \in \mathbb{R}^d$,

- Alpine : $f(x) = \sum_{i=1}^{d} |x_i \sin(x_i) + 0.1x_i|$
- Ackley : $f(x) = -20\exp(-0.2\sqrt{\frac{1}{d}\|x\|_2^2}) - \exp(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)) + 20 + \exp(1)$
- Rastrigin : $f(x) = 10d + \sum_{i=1}^{d} x_i^2 - 10\cos(2\pi x_i)$
- Xinsheyang2 : $f(x) = \sum_{i=1}^{d} |x_i| \exp(-\sum_{i=1}^{d} \sin(x_i^2))$

3 Variants of PSO [3]

PSO (Particle Swarm Optimization) is a meta-heuristic algorithm among many others. While numerous optimization algorithms have been developed, they often share a common drawback: the number of parameters to tune can be extensive. In contrast, PSO requires tuning only three parameters. Consequently, many variants of PSO have been proposed.

Besides the simplicity of dealing with the parameters of standard PSO, many hybrid PSO algorithms have been developed by incorporating other well-known meta-heuristic algorithms.

3.1 Tuning the parameters

3.1.1 The inertia w

The first parameter we can control is the inertia weight. This parameter was introduced to regulate velocity, preventing particles from flying everywhere and missing the optimal solution. The inertia weight can be a fixed parameter; studies have shown that the best performance is achieved when w is in the range $[0.8, 1.2]$. Another possibility is to randomize w, similar to r_1, r_2, at each step as follows:

$$w = \frac{1 + \mathcal{U}[0,1]}{2} \in [0.5, 1]$$

The strategy or behavior we aim to achieve is for the swarm to extensively explore the majority of the search space at the beginning. This extensive exploration provides valuable information for the second part of the run, where the focus shifts to exploitation to find the optimal solution.

However, this kind of behavior might not be achieved with a static inertia, so w needs to be time-varying. Consequently, another PSO variant has been proposed with the following update for the inertia:

$$w_t = (w_{max} - w_{min})\frac{T - t}{T} + w_{min}$$

As we decrease the inertia, we achieve the desired behavior. At the beginning, when w is large, the velocity increases, allowing particles to explore the search space freely. As w decreases, the focus shifts towards better exploitation. Experiments show that better performance is achieved if the PSO run starts with ww at 0.9 and decreases to 0.4.This version of the time-varying inertia weight method, according to the formula, is called linearly-varying inertia weight (LVIW). LVIW is one of the most common, if not the most common, time-varying techniques that have been widely used.

Another possibility, given that we are dealing with a dynamic algorithm based on sharing information within the particle swarm, is to set an adaptive inertia weight. The idea is to take into account the improvement in the fitness of each particle. Specifically, we can consider the proportion $P_s(t)$ of particles that have successfully lowered their personal best at time t. w is then updated as follows,

$$w_t = (w_{max} - w_{min})P_s(t) + w_{min},\ P_s(t) \in [0, 1]$$

As P_s decreases over time—initially, it's easier to improve the personal best, but it becomes more challenging as time goes by. The formula shows then that w will also decrease as intended.

3.1.2 Acceleration coefficients c_1, c_2

The behavior of the swarm of particles is also influenced by the coefficients c_1 and c_2. When c_1 is large, particles may explore widely across the search space. Similarly, a relatively large c_2 can lead to premature convergence as particles tend to gather together. Therefore, it is beneficial to increase c_1 and decrease c_2 over time. A time-varying acceleration coefficient has been proposed, and the coefficients are updated as follows :

$$c_1(t) = (c_{1f} - c_{1i}) \cdot \frac{t}{T} + c_{1i} \searrow$$
$$c_2(t) = (c_{2f} - c_{2i}) \cdot \frac{t}{T} + c_{2i} \nearrow$$

with $c_{1f} = 0.5,\ c_{1i} = 2.5,\ c_{2f} = 2.5$, and $c_{2i} = 0.5$,

3.2 Neighborhood topology in PSO

The standard PSO algorithm does not impose any specific topology for the particles. In fact, each particle considers all other particles as its neighbors and incorporates this information into the global best. This topology is known as a star topology.

We can introduce a new variant of PSO by changing the neighborhood topology. The proposed new topology is the ring topology. For instance, in this topology, the ith particle considers the $(i+1)$th and $(i-1)$th particles as its neighbors to update the global best.

3.3 Other variants of PSO

Many PSO variants have been developed, often achieving better performance but at the expense of higher computational costs. These new PSO variants are typically implemented in relatively low-dimensional spaces, where the PSO run performs numerous function evaluations.

However, PSO variants are rarely tested on real engineering problems. Rigorous statistical analyses that provide conclusive proof of the new algorithms' efficiency, such as by demonstrating their velocity of convergence and confidence intervals, have not yet been conducted.

One of the PSO variants is the multi-swarm PSO. This new meta-heuristic is based on increased collaboration between particles. To achieve this, we split the swarm into two sub-swarms: one is responsible for exploring the search space, while the other focuses on exploitation. By communicating with each other, the two sub-swarms may perform better than a standard PSO.

In the same vein of cooperation between particles, a new PSO variant called Cooperative Particle Swarm Optimization (CPSO) has been developed, utilizing the cooperation concept from genetic algorithms.

Another way to create cooperation is to establish hegemonic relationships between particles. The total number of particles is divided into slave swarms and a master swarm. The slave swarms handle the exploration task and communicate with the master swarm, allowing the master to progress slowly to prevent convergence from being stuck in a local minimum.

3.4 Hybridization

Hybridization involves combining the PSO algorithm with another algorithm to create a new algorithm that performs better than either algorithm individually.

with : **GA DE GSA SA**

4 The Challenge of the project

Even though PSO can perform very well, there are some weaknesses that researchers have aimed to address through modifications to the coefficients and the meta-heuristic itself.

One of the crucial problems of PSO is premature convergence. When dealing with multi-modal functions, the algorithm can easily get stuck in a local minimum. This weakness arises from a lack of diversity within the swarm of particles.

A difficulty arises from controlling the coefficients at each iteration, especially when the inertia is not static. In fact, there is no guarantee that the best combination of parameters is achieved. Additionally, much work is needed in controlling the velocity to prevent particles from flying far away from the search space, which is often constrained in real-world and engineering problems.

To mitigate the drawbacks of PSO and address new challenges, consider the following points for developing an improved PSO variant :

- Hybridize with some algorithms or come up with a new meta-heuristic to improve the convergence.
- Study the behavior in a high-dimensional and low-dimensional search space.
- Statistical analysis of the convergence.

5 Convergence of PSO with martingale and Markov chain theory : details of the proofs [4]

In this part we try to give all the proofs of the propositions, the next step would be to extend those propositions to other variants of PSO.

The problem :

$$x^\star \in \arg\min_{x \in K} f(x)$$

With $K \subset \mathbb{R}^d$ a compact space.

5.1 The particle swarm as a Markov Chain

If we combine (1) and (2), we can recognise an expression of a Markov chain set up.

$$\forall i \in \{1, \dots, m\},\ x_{t+1}^i = x_t^i + w(x_t^i - x_{t-1}^i) + c_1 r_1 (pbest_t^i - x_t^i) + c_2 r_2 (gbest_t - x_t^i) \tag{3}$$

With m the number of particles in the swarm.

In this section we want to understand the behavior of the particles by modeling each particle as a Markov chain, however we notice that in the dynamic relation (3) that the position of each particles depends on the two previous positions. So the state space of the random process has to be bigger. Which leads us to the following definition.

Proposition

$\forall i \in \{1, \dots, m\}$,

$$\xi_t^i = (x_t^i, x_{t-1}^i, pbest_t^i, gbest_t) \in \mathbb{R}^{4d} \tag{4}$$

(ξ_t) is a Markov chain and we denote $\mathcal{S}$ the state space

Proof. Let's determine the behavior of ξ_{t+1}^i, for $i = 1, \dots, m$,

$$\xi_{t+1}^i = (x_{t+1}^i, x_t^i, pbest_{t+1}^i, gbest_{t+1})$$

We compute the first component of ξ_{t+1}^i with the vector ξ_t^i only and the dynamic relation (3). The second component x_t^i is also provided by ξ_t^i without any computation. We compute $pbest_{t+1}$ as follows :

$$pbest_{t+1} = \max\{f(x_{t+1}^i),\ pbest_t\}$$

Only ξ_t^i is needed, the last component is the global best at time $t+1$ which is obtained as follows :

$$gbest_{t+1} = \max_{j=1,\dots,m} \{f(x_{t+1}^j),\ gbest_t\}$$

Therefore $gbest_{t+1}$ depends on the present states $(x_{t+1}^j,\ j \neq i)$ of the other particles. Each present state of the particle $j \neq i$ depends only on $\xi_t^j = (x_t^j, x_{t-1}^j, pbest_t^j, gbest_t)$ through (3), then we need only the former ξ_t^i to have $gbest_t$ which is common to all $(\xi_t^j)_{j=1,\dots,m}$. We have shown that,

$\forall i \in \{1, \dots, m\}$

$$\xi_{t+1}^i \,|\, \xi_t^i, \dots, \xi_1^i = \xi_{t+1}^i \,|\, \xi_t^i$$

■

Likewise all the swarm behaves as a Markov chain,

Proposition

The swarm at time t is represented as,

$$\zeta_t = (\xi_t^1, \dots, \xi_t^m) \tag{5}$$

(ζ_t) is a Markov chain.

Proof. We want to show that ζ_t determines ζ_{t+1}, each component of ζ_{t+1} is determined by ξ_t^i then ζ_t. We can also say that ζ_t is a vector of Markov chains then it is a Markov chain. ■

In the first proof we can notice that there is some interdependence between the particles through the global best, this dependence is controlled with the following proposition.

Definition

Let $\mathcal{F}_{n,k}$ denote the $\sigma - algebra$ generated by $\xi_k^1, \dots, \xi_k^n$, for $n = 1, \dots, m$.

$$\varphi(k) = \sup_{n=1,\dots,m-1} \sup \left\{ |P(B|A) - P(B)|,\ A \in \mathcal{F}_{n,k},\ B \in \sigma(\xi_{n+1}^k) \right\}$$

The transition kernel $P(\zeta_{k+1} \,|\, \zeta_k)$ and it's independent counterpart verifies the following inequality,

Proposition

$\forall k \geq 1$,

$$\left| P(\zeta_{k+1} \,|\, \zeta_k) - \prod_{i=1}^{m} P(\xi_{k+1}^i \,|\xi_k^i) \right| \leq 2^{m-1} \varphi(k+1)$$

Proof. ■

$P(\zeta_{k+1} \,|\, \zeta_k)$ is a transition kernel then the inequality has to be verified for $A_1, \dots, A_m \in \mathcal{B}(\mathcal{S})$ and $x_1, x_m \in \mathcal{S}$,

$$\begin{aligned} P(\zeta_{k+1} \in A_1 \times \cdots \times A_m \,|\, \zeta_k = x_1, \dots, x_m) &= P(\xi_{k+1}^1 \in A_1 \,|\, \xi_k^1 = x_1, \dots, \xi_{k+1}^m \in A_m \,|\, \xi_k^m = x_m) \\ &= P(\tilde{A_1}, \dots, \tilde{A_m}) \\ &= P(\tilde{A_1}) P(\tilde{A_2} \,|\, \tilde{A_1}) \dots P(\tilde{A_m} \,|\, \tilde{A_1} \dots \tilde{A_m}) \end{aligned}$$

We notice that the position of the particle i at time $t+1$, ξ_t^i depends on the position of the other particles, so here we touch the idea of the modelisation. In fact, the entire swarm behaves like a Markov chain, the position of the swarm at time t depends only depends only on its position at time $t-1$. We define then, the following Markov chain process :

$$\zeta_t = (\xi_t^1, \ldots, \xi_t^N) \in \mathbb{R}^{4Nd}$$

Proposition 1

(ζ_t) is a Markov chain.

5.2 A close optimal swarm state

We define $\mathcal{M}$ the optimal particle state of the random process ξ_t^i that describes the particle i as : Let, $\mathcal{A}$ the set of the solution of the minimizing problem,

$$\mathcal{M} = \mathbb{R}^d \times \mathbb{R}^d \times \{g^\star\} \times \{g^\star\}$$

with $g^\star \in \mathcal{A}$, which means when the personal best and the global best reach the same theoretical solution of the problem it's then an optimal state. $\mathcal{M}$ is obviously included in the space state of the random process, ξ_t^i.

Definition 3.1: Giving a set of state $\mathcal{C}$ and (X_n) a random process, and P the probability transition of the random process. We say that $\mathcal{C}$ is closed if :

$$\forall i \in \mathcal{C}, j \notin \mathcal{C},\ P_{ij} = 0$$

Namely, we cannot reach another state outside of $\mathcal{C}$, it's an absorbing state.

Proposition 2

$\mathcal{M}$ is a closed state of the random process (ξ_t^i).

Now we can define the optimal swarm state Γ which is a the optimal state of the whole swarm process (ζ_t). In a simple way

$$\zeta_t = (\xi_t^1, \ldots, \xi_t^N) \in \Gamma \iff \exists\, \xi_t^i \in \mathcal{M}$$

Which means the swarm is in an optimal state if any particle is in an optimal state. Therefore, the social term might drive the swarm to the optimal particle and then an optimal solution of the problem.

Proposition 3

Γ is a closed state of the random process (ζ_t).

The proofs of the closure of the optimal states or due to the fact that the personal best and the general best tend to approach a minimizer when they are equal to a solution of the problem then the personal best and the general become static.

5.3 Condition of convergence with martingale theory

Let's consider the following random process $(F(\zeta_t))_t$ with F such that, $F(\zeta_t) = g(gbest_t)$. By the compactness of K, $F(\zeta_t)$ is bounded and by construction $(F(\zeta_t))_t$ is a decreasing process, therefore we have the following preposition,

Proposition 4

$(F(\zeta_t))_t$ is a super-martingale that converges almost surely.

The limit a.s of $(F(\zeta_t))_t$ it is not necessarily $g(g^\star)$ but, there exists a random variable ζ^∞ such that,

$$\zeta_t \xrightarrow[t\to\infty]{a.s} \zeta^\infty$$

Now we want to know what are the conditions such that, $\zeta^\infty \in \Gamma\ a.s$

6 Kernalize The PSO

7 PSO with subgradient

7.1 Subgradient and optimization

Definition 7.1 : We say $g \in \mathbb{R}^n$ is a subgradient of $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ at $x \in \mathbf{dom}(f)$, if for all $z \in \mathbf{dom}(f)$,

$$f(z) \geq f(x) + g^T(z - x)$$

and we call, subdifferential the set $\partial f(x)$ of all the subgradient of f at x.

8 Conclusion

Your conclusion here

Acknowledgments

This was was supported in part by.....

Resources

thesis on PSO
https://repository.up.ac.za/bitstream/handle/2263/24297/00thesis.pdf

kernelized CBO https://timroith.github.io/2022/11/07/PCBO.html

A Course in Interacting Particle Systems https://arxiv.org/abs/1703.10007

Thesis CBO

https://mediatum.ub.tum.de/doc/1647263/bmmw81e4a9756ymq6onm5i6qi.Philippe%20S%C3%BCnnen%20final.pdf

More about convergence :

https://repository.up.ac.za/bitstream/handle/2263/17262/VanDenBergh_Convergence(2010).pdf;sequence=1

https://www.igi-global.com/gateway/article/full-text-pdf/328092&riu=true
Connection PSO SDEs :

https://arxiv.org/abs/2108.00393

References

[1] James Blondin. Particle swarm optimization: A tutorial. 2009.

[2] James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks*, volume 4, pages 1942–1948. IEEE, 1995.

[3] TAREQ M. SHAMI et al. Particle swarm optimization: A comprehensive survey. *IEEE*, 2021.

[4] Guosong Yu Gang Xu. On convergence analysis of particle swarm optimization algorithm. *Journal of Computational and Applied Mathematics*, pages 65–73, 2017.

[5] Claudia Totzeck and Marie-Therese Wolfram. Consensus-based global optimization with personal best. *arXiv preprint arXiv:2005.07084*, 2020.