Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Migration: An Efficient Explorer Operator to Guide Swarm Evolution

Wan Yi
Jilin University
Lili He
Jilin University
Yanchun Liang
Jilin University
Renchu Guan
Jilin University
Xiaosong Han (hanxiaosong@jlu.edu.cn)
Jilin University

Research Article

Keywords: Particle swarm optimization, Wasserstein Auto-encoders, Salmon migration, Fast Convergence

Posted Date: October 21st, 2022

DOI: https://doi.org/10.21203/rs.3.rs-2183682/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License. Read Full License

Abstract

Particle Swarm Optimization with Migration (MPSO) is proposed to solve the issue that PSO will come across unbearable time cost problem when dealing with High-dimension, Expensive and Black-box objective function tasks. Migration operator is inspired by the migration of Salmon. Salmon will start a dangerous journey from the ocean to the home rivers for reproduction. The process of the entire behavior is similar with the reduction and recovery of dimension. Therefore, we design the Migration operator where a pre-trained Wasserstein Auto-encoders (WAE) is applied to simulates the migration behavior to accelerate the process of evolution in PSO, and we use Least-Squares Regression in lower space to product better generation. In comparison with famous baselines methods in some benchmark functions, MPSO converges more faster and more accurate which show the great potential of migration operation.

1. Introduction

Particle Swarm Optimization (PSO) is a typical bionic swarm intelligence algorithm designed by simulating the predatory behavior of birds and fishes (Kennedy and Eberhart 1995). Since PSO was proposed, due to its simple and effective characteristics, it was quickly applied to many large-scale optimization problems. In the optimization process, each particle has two characteristics, position vector and velocity vector, and each particle will follow its own and the swarm's historical experience to pursue a better position. The classic PSO has been paid much attention by the academic community and applied by industry effectively because of its easy implementation, high precision, and fast convergence.

In the academic field, PSO is being continuously improved, and it is a core algorithm in swarm intelligence. For example, Liu Y formulated four rules by introducing the mechanism for survival of the fittest, which simulated the competition among the swarms. This was an improved feature selection method that gradually achieves higher generalization ability by performing kernel parameter optimization and feature selection at the same time (Liu et al. 2011). Sierra M R proposed a new Multi-Objective Particle Swarm Optimizer, which was based on Pareto dominance and the use of a crowding factor to filter out the list of available leaders (Sierra and Coello 2005). Sun L proposed a cooperative particle swarm optimizer with statistical variable interdependence learning, which was a statistical model proposed to explore the interdependencies among variables (Sun et al. 2012). Compared with other recent collaborative optimization algorithms, CPSO-SL solved the separability and inseparability problems and proves that CPSO-SL was highly competitive, even when all other PSO-based methods fail. Recently, meta-heuristic algorithms are becoming powerful methods for solving NP problems. Vahideh Hayyolalam (Vahideh and Ali 2020) proposed Black Widow Optimization Algorithm (BWO), which inspired by the unique mating behavior of black widow spiders and proved numerous advantages in different aspects such as early convergence and achieving optimized fitness value compared to other algorithms. In Section 4, we also take other classic meta-heuristic algorithms into comparison, such as ABC (Artificial Bee Colony algorithm) (Karaboga 2005) as well as BBO (Biogeography-based optimization algorithm) (Simon 2008).

In the industry, Moayedi H aimed to optimize Adaptive Neuro-Fuzzy Inferences System (ANFIS) with PSO (Moayedi et al 2020), comparing PSO-ANFIS and ANFIS, the parameters optimized by PSO-ANFIS were obviously better than those optimized by ANFIS. Duan H proposed an improved PSO model for solving the optimal formation reconfiguration control problem for multiple Uninhabited Combat Air Vehicles (UCAVs) (Duan et al 2008), the mutation strategy and a special mutation-escape operator were adopted in the improved PSO model to make particles explore the search space more efficiently.

Although PSO converges faster than other evolutionary computation methods, it is still not efficient enough for some complex engineering problems, especially when the problem is High-dimension with Expensive Black-box (HEB) objective function. Thus, it is still important to improve PSO's convergence speed, and avoid falling into the local optimum at the same time. In improved PSO, parameter settings, swarm topology, and particle updating strategy play important roles in determining the search results to a large extent, because they determine the velocity and direction of particles in the search space. Shi and Eberhart introduced interval weights into the speed update equation, provided parameter selection criteria (Shi et al 1998), and proposed a fuzzy adaptive method for interval weight adjustment (Shi et al 2001). The experimental results illustrated that the fuzzy adaptive PSO was a promising optimization method, which was especially useful for optimization problems with a dynamic environment. Recently, Xia X (Xia et al 2020) transplanted the multi-exemplar and forgetting ability to PSO, and proposes an expanded PSO,

called XPSO, which expanded the “social-learning” part of each particle from one exemplar to two exemplars, learn from both the locally and globally best exemplars and assigns different forgetting abilities to different particles. The results show XPSO was benefit to improve the accuracy of the solution and speed up convergence. Shi and Eberhart (Shi et al 1998) introduced a new parameter, called inertia weight, into the original particle swarm optimizer. Simulation experiments show that this new parameter has a significant and effective impact on the particle swarm optimizer. Cui (Cui et al 2017) has proposed a new particle generation method with PCA, and the experiments prove that the new particle can guild the swarm convergence quickly.

Above all, researchers have proposed many methods to improve PSO, which can be roughly divided into the following categories: parameters adjustment, velocity update strategy improvement, prediction of high-quality particles leading swarm.

Briefly, the essential contributions of this paper are as follows:

1. Inspired by Salmon migration, we propose a novel operator named Migration to extend the searching space and help the swarm fast convergence.
2. We demonstrate the performance by conducting Several benchmark experiments functions and our method-MPSO gains better speed performance with acceptable or none accuracy decrease in most experiments.

The rest part of this paper is organized as follows. Section 2 is an introduction to related work. Section 3 is a description of the proposed Migration Particle Swarm Optimization (MPSO). Numeric experiments are in Section 4. Section 5 discussed MPSO parameter setting. Finally, conclusions are in Section 6.

2. Literature Review

In this section, the Salmon migration process, GPAM-PSO and Wasserstein Auto-Encoders will be briefly introduced.

2.1 Salmon migration

In the BBC documentary “Natural World”, there is an episode dedicated to the most famous ecological phenomenon on the west coast of North America-Salmon migration. Migration can be observed in many kinds of Salmon, including Oncorhynchus keta, sockeye, Coho Salmon, king Salmon, etc. Salmon (Jonsson et al 2009, Crozier et al 2008) is an incredible fish. They were born in freshwater, but spent most of their adult time in the ocean. As Fig. 1 presented, Salmon are anadromous, which means their lives will begin in the fresh water (Called Fry, Fingerling or Smolt according to their size), following with the migration to ocean as adults (Called Salmon), and finally return to their freshwater home stream (Because their eggs cannot survive in the seawater) to reproduce. Main reasons of the migration are the huge population and limited resources, which drive Salmon swim to the ocean for food and survival. However, the process of migration is difficult for Salmon. During the returning process, Salmon will go upstream, swimming towards the birthplace fearlessly, and jump over the small waterfalls and levees without preying. Of course, the most dangerous condition is far more than that, the Salmon migration season coincides with the end of the bear’s hibernation period. So, in the returning process, there will be many bears waiting for hunting on the Salmon migration route. After suffering all kinds of difficulties, Salmon can return to their birthplace to lay eggs. Therefore, only the strongest and smartest Salmon can survive to their birthplace to lay eggs, which is in line with the principle of survival of the fittest in nature. After a long journey, Salmon will have a huge change when it gets close to the spawning ground, and the spawning site will be selected in a shallow water area with a very turbulent sandy bottom which is named redd.

2.2 GPAM-PSO

In order to generate high-quality particles leading the population forward, Cui and Li (Cui et al 2017) combined Principal Component Analysis (PCA) and PSO to predict the global optimum based on extracting the features reflecting the evolutionary trend. The Globally-optimal prediction-based adaptive mutation particle swarm optimization (GPAM-PSO) algorithm consists of three modules: (1) population velocity and position updating module (M1); (2) globally-optimal -prediction based searching module (M2); (3) adaptive mutation strategy module (M3). In module M2, PCA is used to reduce particles’ dimension, and a quadratic polynomial is fitted according to the particles in low-dimensional space and their objective values. Then, a high-quality

particle can be obtained from the polynomial's peak or trough. Finally, the generated particle is restored to the original space by the PCA transformation matrix. Experiments show that this method can help PSO to converge quickly. However, after many tests, when the best position is not 0, the effect is limited.

2.3 Wasserstein Auto-Encoders

The Auto-encoder is a single hidden layer unsupervised learning neural network (Rumelhart et al 1989), which can be used for high-dimensional complex data processing. Auto-encoder (Chen et al 2012, Hornik et al 1989) pre-trains through a common BP neural network, with the goal of making the input value equal to the output value. AE can be regarded as the compression coding of the input data, the hidden layer outputs the encoded vector, and the output layer is the decoding process. And multiple AEs are combined for multi-stage data compression, which is named as Stack Auto-encoder (shown as Fig. 2). AEs are well applied in many researching areas such as speech recognition (Vincent et al 2010, Alam et al 2020), recommendation systems (Zhang et al 2020)]and so on. WAE (Wasserstein Auto-Encoders) is a novel auto-encoder algorithm (ITolstikhin et al 2018). WAE minimized the penalty form of the Wasserstein distance between the model distribution and the target distribution. WAE approached generative modeling from the optimal transport (OT) point of view, which, as ITolstikhin claimed, can avoid the "mode collapse" problem and train easier with an encoder. It aims at minimizing $OT\, W_c(P_X, P_G)$ between the true (but unknown) data distribution P_X and a latent variable model *PG* specified by the prior distribution P_Z of latent codes *Z* and the generative model $P_G\ (X|\ Z)$of the data points *X* given *Z* (as shown in Fig. 2). Moreover, Wasserstein Distance can cover more cases than other divergences (such as calculation between two distributions without overlap, where *KL* is mean less and *JS* become a constant) and provide available gradient. WAE obtains some good characteristics: stable training, code architecture and good latent manifold structure. More importantly, it is able to generate higher quality output. Hence, WAE is selected to simulate Salmon migration process.

3. Mpso

In this section, Particle Swarm Optimization with Migration (MPSO) is described in detail. Inspired by the behavior of Salmon migration, the habit of Salmon is similar to spatial transformation. Furthermore, WAE is able to simulate the process of spatial transformation. After Salmon migration, the fish lay eggs in freshwater. Least-Squares Regression is employed to simulate the behavior to generate new particles. due to the survival of the fittest in nature, the newly generated particles are theoretically high-quality. These particles speed up the convergence and can also guide the swarm to jump out of local extremes. The workflow of MPSO is shown as Fig. 3.

3.1 Migration

The Migration operator is combined by three steps, including: (1) Homing; (2) Spawning; (3) Leaving. WAE is selected as the dimensional transformation tool to imitate the Homing and Leaving steps (Fig. 4). The first half layers are encoders, which is used to extract particle's key information from origin-dimensional to low-dimensional space, and the latter half layers are decoders, which are used to restore low-dimensional particles to the aforementioned origin low-dimensional particles. It can be seen from Fig. 2 that AEs is a coupling of multiple Linear layers, which is similar to the Salmon migration in the natural world. The Salmon will go upstream and jump continuously during the homing, just like the particles keep the dominant information after the process of dimensional transformation. The Homing step is implemented by the encoder of WAE. Likes the strongest Salmon could survive during the migration, the high-quality information of particles could be retained to new space after encoding. Then, after "hatching" new particles in lower-dimensional space, we will return them to the original swarm for selection. This step is called "Leaving" in migration, simulated by the decoder of WAE. To keep the size of swarm, a roulette is conducted for selection. In other words, the fitness values of the particles in the original swarm are used as the probability of roulette, and then the selected particle will be replaced by the newly generated particle which will accelerate the process of evolution. During Migration, WAE will train with a few epochs by least squares regression. The formula for least squares method fitting (or say the loss function of WAE) is as follow.

$$loss = \sum_{i=1}^{n} \left(y_i - y_{pre}\right)^2$$

1

It is the same as mean square loss, where y_i is the actual value and y_{pre} is the predicted value.

3.2 Spawning

After homing, the excellent Salmon could survive in their hometown. In order to produce better offspring, the strongest Salmon pairs. In MPSO, some high-quality particles are selected to obtain better particles in lower-dimensional space. The female Salmon will find proper riffle to build the red (spawning ground) to lay her roe. The redd is benefit greatly to hatch eggs. In MPSO, a quadric surface according to particles' fitness values represents the redd.

In the process of fitting surface, the particle selection strategy is still roulette, that is, each particle is given a probability of being selected according to the fitness, and finally the particle with high fitness is most likely selected. The benefit of quadric surface is to find extremum easily. If it is a maximum optimization problem, the parabolic opens down, and the parabolic opens up conversely. Then the crests or troughs could be obtained as high-quality particles in the low-dimensional space. This process simulates the process of Salmon hatching. As the iteration increasing, the particles are quickly gathering at the optimal value, shown as Fig. 5. The particle swarm is moving from dispersion to aggregation, and gradually approaches the global optimal solution with the optimal position. It is worth noting that when the optimization goal is a maximum value, the opening direction of the parabola should be downward. The new born fishes will leave their hometown to the ocean. In MPSO, the new generation particles will return to the original space according to the decoder part of the trained Wasserstein auto-encoders network.

3.3 Effectiveness of the new generated particle

In PSO, high-quality particles play a key role to lead the swarm evolvement. An experiment is designed to prove that the new generated particles are superior to the particles of the classic PSO(CPSO). In order to measure the fitness value trend of the swarm, the decline rate is defined as Eq. (2).

$$R\left(x_i\right) = \frac{h\left(x_i\right) - h\left(x_{i+1}\right)}{h\left(x_i\right)}$$

2

where h is the best fitness value in i th iteration. During the experiment, Migration is conducted every 10 iterations in the first 50 iterations. The result is shown as Fig. 6. It can be seen, comparing with the CPSO, the decline rate of MPSO changes sharply at the Migration point. This phenomenon prove that Migration is able to help the swarm convergent rapidly.

4. Numerical Experiments

In this section, 9 benchmark functions (as shown in Table 1) are employed to compare MPSO with CPSO, GMAP-PSO, ABC, BBO and BWO.

4.1 Experiment Setting of MPSO

In MPSO, the weight of WAE is randomly initial by Gaussian distribution, and we set the training epoch as 10 in all benchmark functions. the migration operator is used every 10 iterations in the first 50 iterations.

4.2 Benchmark Functions

Benchmark1-6(Liang et al 2013, Meng et al 2016, Dixon et al 1978) are single-modal functions, and the others are multi-modal functions (Floudas et al 2013, De et al 2020), The objective is to find the minimum value in the domain of these functions. The

minimum positions of all the benchmark functions are origin.

Table 1
The Benchmark Functions

	No.	Function name	Function formula	Domain
Single-model	1	Sphere function	$f(x)=\sum_{i=1}^{n}{x_i}^2$	[-100, 100]
	2	Quardirc function	$f(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i}x_j\right)^2$	[-100, 100]
	3	Rosenborck function	$f(x)=\sum_{i=1}^{n-1}\left[100\left(x_{i+1}-{x_i}^2\right)^2+\left(x_i-1\right)^2\right]$	[-100, 100]
	4	High Conditioned function	$f(x)=\sum_{i=1}^{n}\left(10^6\right)^{\frac{i-1}{n-1}}{x_i}^2$	[-100, 100]
	5	Bent Cigar function	$f(x)={x_1}^2+10^6\sum_{i=2}^{n}{x_i}^2$	[-100, 100]
	6	Step function	$f(x)=\sum_{i=1}^{n-1}\left(\lfloor x_i+0.5\rfloor\right)^2$	[-100, 100]
Multi-model	7	Schwefel function	$f(x)=418.9829\times n+\sum_{i=1}^{n}x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$	[-500, 500]
	8	Ackley function	$f(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}{x_i}^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos\left(2\pi x_i\right)\right)+20+e$	[-32, 32]
	9	Generalized Griewank function	$f(x)=\frac{1}{4000}\sum_{i=1}^{n}{x_i}^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	[-600, 600]

4.3 Comparison of MPSO and CPSO

In this section, the MPSO and CPSO is compared, which is named Experiment 1 in this paper. During the experiments, the max iteration and optimization parameters were set as 200 and 30 dimensions. Each benchmark function was run 10 times to obtain the average effect. The specific result is shown in Table 2.

Table 2
Comparison of MPSO and CPSO

Functions	Measure	CPSO	MPSO
Sphere	Mean	19.6929	1.79841
	Min	4.317605	0.588388
	10th iteration	23973.79	2429.987
Quardirc	Mean	239.7223	30.51135
	Min	64.65025	13.19043
	10th iteration	303981.4	41128.6
Rosenbrock	Mean	56.2539	18.72129
	Min	13.82219	3.547897
	10th iteration	530392.4	117775.5
High Conditioned	Mean	10217.76	2978.26
	Min	1743.238	1255.0957
	10th iteration	1.3E + 08	48867458
Schwefel	Mean	68.1541	1.78156
	Min	1.477865	0.627021
	10th iteration	4375.315	4046.86
Step	Mean	11.6	0.5
	Min	1	0
	10th iteration	26904.6	2099.2
Bent Cigar	Mean	9494269	403797.5
	Min	3909514	1214564
	10th iteration	2.32E + 10	1.72E + 08
Ackley	Mean	8.403727	1.999783
	Min	3.210841	1.076549
	10th iteration	18.98047	15.51756
Generalized Griewank	Mean	1.158092	0.239174
	Min	1.034249	0.152213
	10th iteration	206.0291	2.013736

In Table 2, ‘Mean’ represents the average best fitness value of 10-time runs. ‘Min’ is the best fitness value among 10 runs. ‘10th-iteration’ is the average best fitness value of the tenth iteration of ten-time runs, and it represents the result of the first migration. It can be seen that MPSO is completely superior to CPSO. Especially the average fitness value after the tenth iteration, under the effect of the migration operator, the convergence speed of MPSO is improved greatly. The evolution curves are drawn as Fig. 7.

From Fig. 7, the MPSO curves of all benchmark functions are under the CPSO curves, and this would imply that MPSO convergent faster. The MPSO curves have obviously jagged at the migration points, which can prove to be the migration operator contributes a lot to the overall downward trend.

4.4 Comparison of MPSO and GPAM-PSO

In this section, MPSO is compared with GPAM-PSO built by our group, which is named Experiment 2. In order to compare the benchmark functions whose extreme points are non-zero points, an offset is added on the original functions to a certain extent. Then, the functions' extreme points are converted from 0 to 50 for comparison. Table 3 shows the result.

Due to the characteristics of benchmark functions, some functions cannot be changed, so 7 benchmark functions are selected for comparison. As shown in Fig. 8 and Table 3, whether it is a unimodal function or a multimodal function, the effect of the MPSO is significantly better than GPAM-PSO. Especially in terms of declining speed, once migration operator is conducted, the fitness drops significantly. According to the final result, it can be seen that the effect of MPSO is also significantly better than GAMP-PSO.

Table 3
Comparison of MPSO and GPAM-PSO.

Functions	Measure	GPAM-PSO	MPSO
Sphere	Mean	8.879313	6.217305
	Min	4.0196355	1.106688
	10th iteration	16437.54	14250
Quardirc	Mean	143.3448	111.4815
	Min	20.53691	18.95066
	10th iteration	240720.9	207051.9
Rosenbrock	Mean	1014.922	212.4618
	Min	219.1546	64.00719
	10th iteration	322736006	20940387
High Conditioned	Mean	5798.778	2978.26
	Min	2171.948	560.2115
	10th iteration	1.6E + 08	48867458
Schwefel	Mean	24.2952	1.78156
	Min	1.280917	0.627021
	10th iteration	4609.926	4046.86
Step	Mean	6.6	0.5
	Min	0	0
	10th iteration	16473.3	2099.2
Bent Cigar	Mean	6809128	403797.5
	Min	1221979.73	1214564
	10th iteration	1.74E + 10	1.72E + 08

4.5 Comparison of MPSO, ABC, BBO and BWO

In this experiment, we compared MPSO with several famous baselines: ABC (Artificial Bee Colony algorithm), BBO (Biogeography-based optimization algorithm), BWO (Black Widow Optimization Algorithm)

For MPSO, we set optimization parameters, max iteration and population size as 50,150, 200. For other baselines, we present the same results according to their papers.

The result show in Table 4, MPSO perform better at single model benchmark and close to other SOTA method in most multi-model benchmark.

Table 4
Comparison of MPSO, ABC, BBO and BWO

Functions	Measure	MPSO	ABC	BBO	BWO
Sphere	Mean	4.11E − 56	2.17E-01	4.94E-06	4.25E − 39
	Min	1.24E − 65	4.76E-02	3.57E-06	2.57E − 44
Quardirc	Mean	1.09E − 27	9.02E + 00	4.49E + 00	4.17E + 00
	Min	0.00E + 00	7.48E + 00	3.68E + 00	3.63E + 00
Rosenbrock	Mean	2.01E − 04	3.18E + 02	2.19E + 03	1.13E + 02
	Min	0.00E + 00	1.00E + 02	5.32E + 02	2.21E + 01
Schwefel	Mean	1.10E + 04	7.21E + 03	1.09E + 04	1.02E + 04
	Min	1.09E + 04	6.36E + 03	8.33E + 03	8.57E + 03
Step	Mean	4.67E + 00	1.31E + 02	9.77E + 00	3.33E − 02
	Min	0.00E + 00	7.70E + 01	2.00E + 00	0.00E + 00
Ackley	Mean	1.80E + 01	1.15E + 01	2.48E-01	3.70E − 14
	Min	1.69E + 01	9.19E + 00	8.20E-03	2.93E − 14

5. Parametric Analysis

In view of time consumption, once the Migration operator is conducted, the time cost will increase accordingly, since Migration is quite complex. Therefore, it is better to control the migration usage frequency. Then, it is worth discussing the operator in three aspects, migration running position, frequency and cooldown time.

5.1 Position and frequency of Migration

The aim of this section is to find conducting Migration at which evolution stage influences the PSO most. Firstly, the entire evolution process is divided into three stages, the initial, middle, and final stages. In order to discover the best Migration position, Migration are conducted at the different stages with different frequency. Then the final optimization results and the changing trend of each MPSO are compared to determine the most reasonable position. A single-modal (Rosenbrock function) and a multi-modal function (Generalized Griewank function) are selected for the experiment. Test1, Migration operator is conducted once at 5th, 10th, 50th, 100th, and 150th iteration relatively. As shown in Fig. 9, it can be seen that running migration operator at the 5th iteration reaches the best performance. Test2, the Migration operator is conducted twice at initial, final and both stages. From Fig. 10, it shows that the effect is the best when the Migration is running at the 5th and 10th iteration. Test3, the Migration operator is conducted 5 times at different positions. As shown in Fig. 11, the same conclusion can be drawn as above that the Migration conducted at the initial stage is able to help PSO convergence rapidly most. Finally, different Migration frequency at the initial stage is compared as Fig. 12. It can be seen that the final result of the twice Migration conduction is the best, however the 5 times Migration convergent fast at initial stage. Thus, if an optimization result is wanted quickly, it is better to increase the Migration frequency. Optimization speed and precision are actually contradictory, and different strategies are adopted to balance accuracy and speed based on the real problems. Nevertheless, based on a large number of experiments, running the Migration 5 times at initial stage is highly recommended, if there is little background knowledge.

5.2 Cooldown time of Migration

In former experiments, it is discovered that if the Migration operator is conducted every iteration, the optimal fitness will remain almost unchanged in the next few iterations. The reason is that the new particles obtained by Migration are high-quality particles, and there is a certain distance from the original particle swarm. The particle swarm needs to be updated for several iterations under the leadership of the new particle to stabilize. These iterations are named as cooldown time. It is necessary to analyze the length of cooldown time. It has been proved that Migration conducted at the initial stage every 5 iterations reach the best performance from above experiments. Test4, the Migration interval is extended to 10 iterations, and the result is shown as Fig. 13. Regarding the benchmark function, 5 iterations' cooldown is better. However, the best cooldown time depends on the optimization problems actually. Although the length of the cooldown time is uncertain, it is not recommended to run the Migration operator one after the other during the evolution.

6. Conclusion

In this paper, PSO with Migration is proposed to simulate Salmon migration. Salmon migration is simulated with the idea of WAE, and spawning is simulated with the method of fitting a surface. Experimental results show that the Migration operator is able to improve the convergence speed greatly in the early stage. Also, the Migration is a generous operation, and it could be fused into Genetic Algorithm, Ant Colony Optimization, and other swarm intelligence algorithms.

Declarations

ACKNOWLEDGMENT

The authors are grateful for the support of the National Key Research and Development Program of China No.2021ZD0112501 and No.2021ZD0112502, the National Natural Science Foundation of China No.61972174 and No.62172187, the Science and Technology Planning Project of Jilin Province No.20220201145GX, No.20200708112YY and No.20220601112FG, the Science and Technology Planning Project of Guangdong Province No.2020A0505100018, Guangdong Universities' Innovation Team Project No.2021KCXTD015 and Guangdong Key Disciplines Project No.2021ZDJS138.

References

1. Kennedy J, Eberhart R. Particle swarm optimization[C]//Proceedings of ICNN'95-International Conference on Neural Networks. IEEE, 1995, 4, 1942–1948.
2. Liu Y, Wang G, Chen H, et al. An improved particle swarm optimization for feature selection[J]. Journal of Bionic Engineering, 2011, 8(2), 191–200.
3. Sierra M R, Coello C A C. Improving PSO-based multi-objective optimization using crowding, mutation and-dominance[C]//International conference on evolutionary multi-criterion optimization. Springer, Berlin, Heidelberg, 2005, 505–519.
4. Sun L, Yoshida S, Cheng X, et al. A cooperative particle swarm opti-mizer with statistical variable interdependence learning[J]. Information Sciences, 2012, 186(1), 20–39.
5. Vahideh Hayyolalam and Ali Asghar Pourhaji Kazem. Black Widow Optimization Algorithm: A novel meta-heuristic approach for solving engineering optimization problems[J]. Engineering Applications of Artificial Intelligence, 2020, 87
6. Karaboga, D.. An idea based on honey bee swarm for numerical optimization. Technical report-tr06, Erciyes university, engineering faculty, computer engineering department. (2005) (Vol. 200, pp. 1–10)
7. Simon, D. Biogeography-based optimization. IEEE transactions on evolutionary computation, (2008). 12(6), 702–713.
8. Moayedi H, Raftari M, Sharifi A, et al. Optimization of ANFIS with GA and PSO estimating ratio in driven piles[J]. Engineering with Computers, 2020, 36(1), 227–238.

9. Duan H, Ma G, Luo D. Optimal formation reconfiguration control of multiple UCAVs using improved particle swarm optimization[J]. Journal of Bionic Engineering, 2008, 5(4), 340–347.
10. Shi Y, Eberhart R C. Parameter selection in particle swarm op-timization[C]//International conference on evolutionary programming. Springer, Berlin, Heidelberg, 1998, 591–600.
11. Shi Y, Eberhart R C. Particle swarm optimization with fuzzy adaptive inertia weight[C]//Proceedings of the workshop on particle swarm opti-mization. Indianapolis, IN: Purdue School of Engineering and Technol-ogy, IUPUI, 2001, 2001.
12. Xia X, Gui L, He G, et al. An expanded particle swarm optimization based on multi-exemplar and forgetting ability[J]. Information Sciences, 2020, 508, 105–120.
13. Shi Y, Eberhart R. A modified particle swarm optimizer[C]//1998 IEEE international conference on evolutionary computation proceedings. IEEE world congress on computational intelligence (Cat. No. 98TH8360). IEEE, 1998, 69–73.
14. Cui Q, Li Q, Li G, et al. Globally-optimal prediction-based adaptive mutation particle swarm optimization[J]. Information Sciences, 2017, 418, 186–217.
15. Jonsson B, Jonsson N. A review of the likely effects of climate change on anadromous Atlantic salmon Salmo salar and brown trout Salmo trutta, with particular reference to water temperature and flow[J]. Journal of fish biology, 2009, 75(10), 2381–2447.
16. Crozier L G, Zabel R W, Hamlet A F. Predicting differential effects of climate change at the population level with life-cycle models of spring Chinook salmon[J]. Global Change Biology, 2008, 14(2), 236–249.
17. Shlens J. A tutorial on principal component analysis[J]. arXiv preprint arXiv:1404.1100, 2014.
18. Japkowicz N, Hanson S J, Gluck M A. Nonlinear autoassociation is not equivalent to PCA[J]. Neural computation, 2000, 12(3), 531–545.
19. Hinton G E, Salakhutdinov R R. Reducing the dimensionality of data with neural networks[J]. science, 2006, 313(5786), 504–507.
20. Rumelhart D E, Hinton G E, Williams R J. Learning representations by back-propagating errors[J]. nature, 1986, 323(6088), 533–536.
21. Chen M, Xu Z, Weinberger K, et al. Marginalized denoising autoen-coders for domain adaptation[J]. arXiv preprint arXiv:1206.4683, 2012.
22. Hornik K, Stinchcombe M, White H. Multilayer feedforward networks are universal approximators[J]. Neural networks, 1989, 2(5), 359–366.
23. Vincent P, Larochelle H, Lajoie I, et al. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion[J]. Journal of machine learning research, 2010, 11(12).
24. Alam F F, Rahman T, Shehu A. Evaluating Autoencoder-Based Fea-turization and Supervised Learning for Protein Decoy Selection[J]. Molecules, 2020, 25(5), 1146.
25. Zhang G, Liu Y, Jin X. A survey of autoencoder-based recommender systems[J]. Frontiers of Computer Science, 2020, 1–21.
26. ITolstikhin I, Bousquet O, Gelly S, et al. Wasserstein auto-encoders[J]. arXiv preprint arXiv:1711.01558, 2017.
27. Liang J J, Qu B Y, Suganthan P N. Problem definitions and evaluation criteria for the CEC 2014 special session and competition on sin-gle objective real-parameter numerical optimization[J]. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, 2013, 635.
28. Meng A, Li Z, Yin H, et al. Accelerating particle swarm optimization using crisscross search[J]. Information Sciences, 2016, 329: 52–72.
29. Dixon, L. C. W., Szego, G. P. (1978). The global optimization problem: an introduction. Towards global optimization, 2, 1–15.
30. Floudas, C. A., Pardalos, P. M., Adjiman, C., Esposito, W. R., Gümüs, Z. H., Harding, S. T., ... & Schweiger, C. A. (2013). Handbook of test problems in local and global optimization (Vol. 33). Springer Science & Business Media.http://www.optima.amp.i.kyotou.ac.jp/member/student/hedar/Hedarfiles/TestGO.htm
31. De Jong K. Evolutionary computation: a unified ap-proach[C]//Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion. 2020, 327–342.

Figures

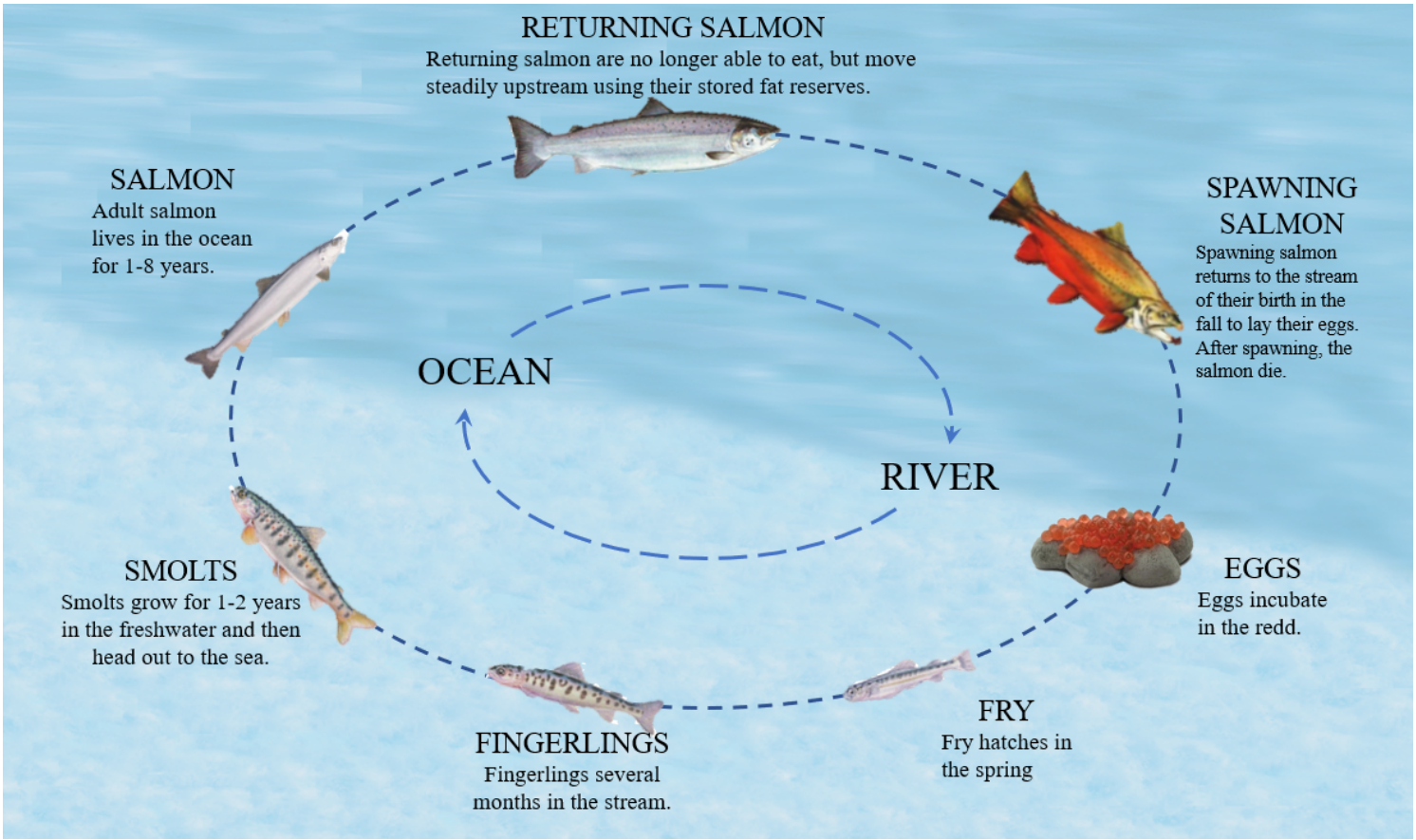


Figure 1

Salmon migration route map

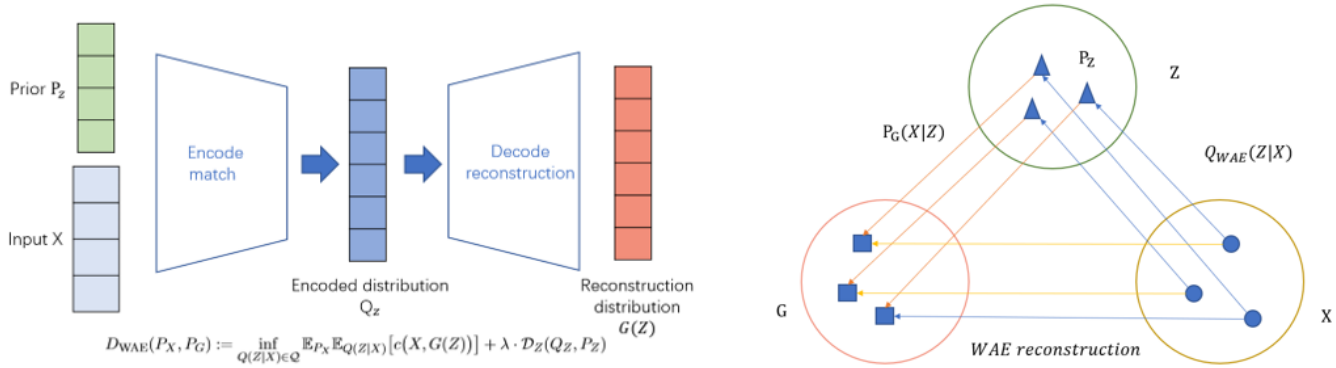


Fig. 2. From (ITolstikhin et al 2018) The loss of WAE is constructed by two parts: 1) the OT cost between X and G, described as the Mathematical expectation of distance in distribution of X and marginal distribution of $Q(Z|X)$ 2) D_Z is an arbitrary divergence between Q_Z and P_Z.

Figure 2

See above image for figure legend.

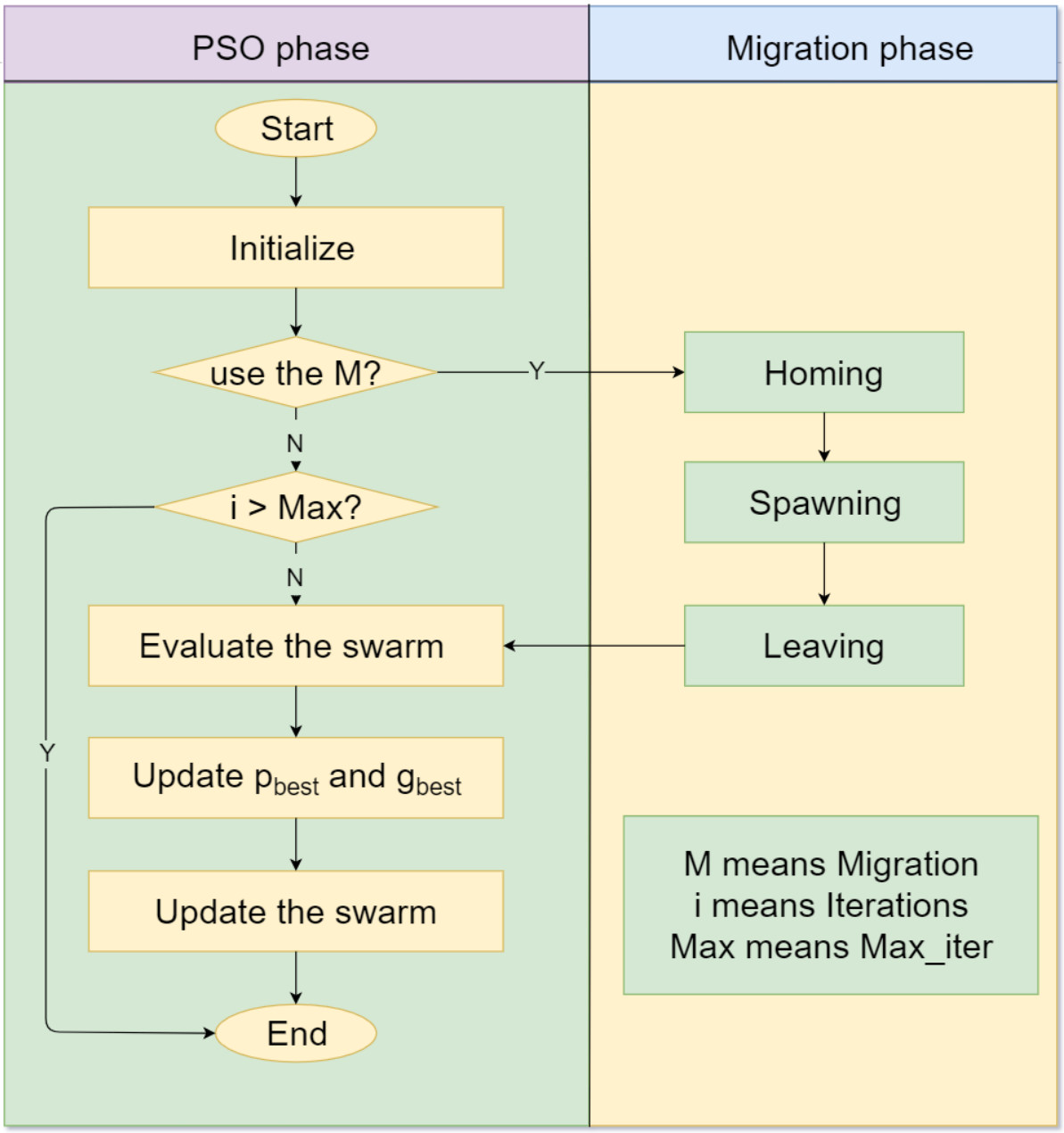


Figure 3

MPSO flowchart

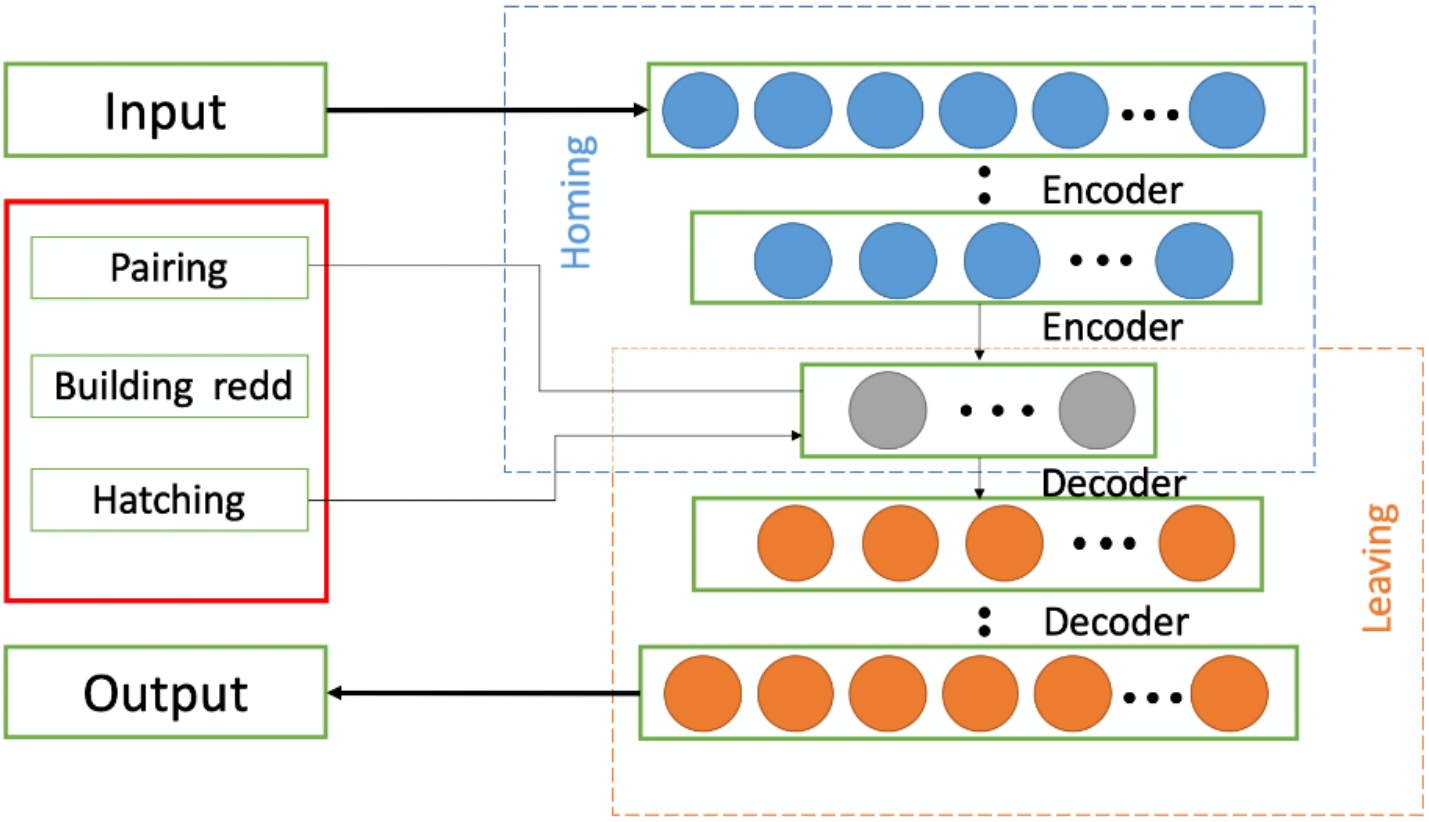


Figure 4

Network structure diagram

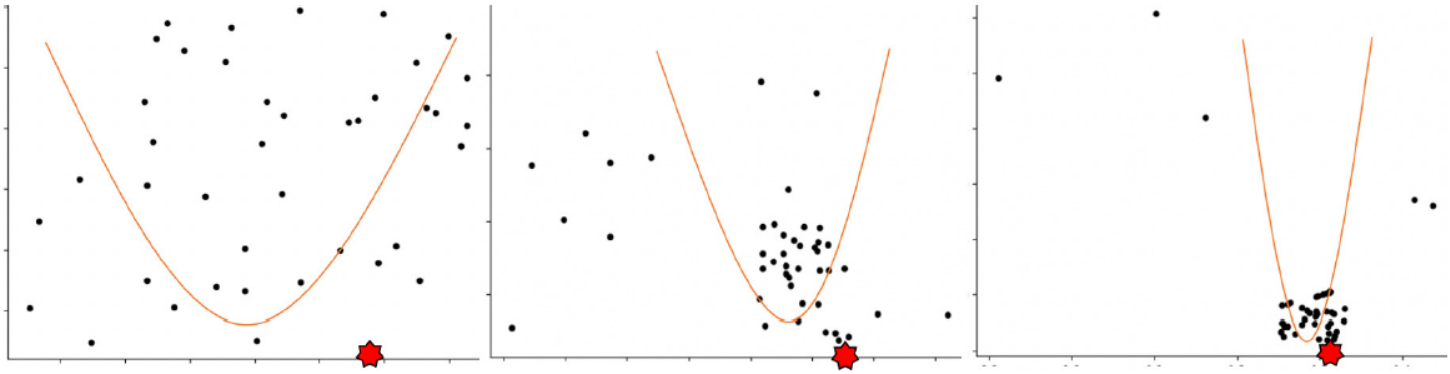


Figure 5

Evolutionary relationship between the predicted optimum (show by the Extremum points of the curve) and the global optimum (show as star point in the bottom). As the iteration increasing, the particles (show as little dots) are quickly gathering at the optimal value

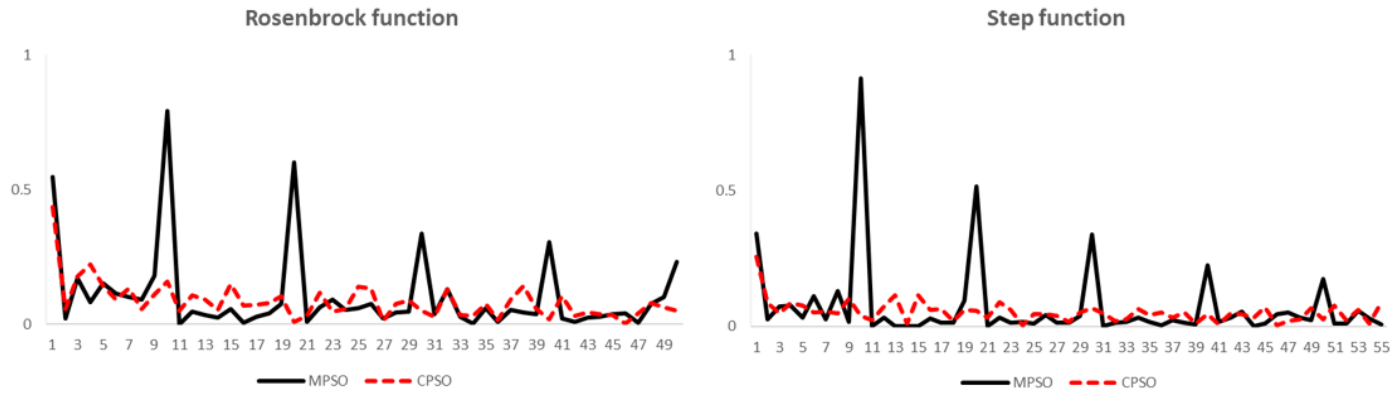


Figure 6

Comparison of fitness decline rate

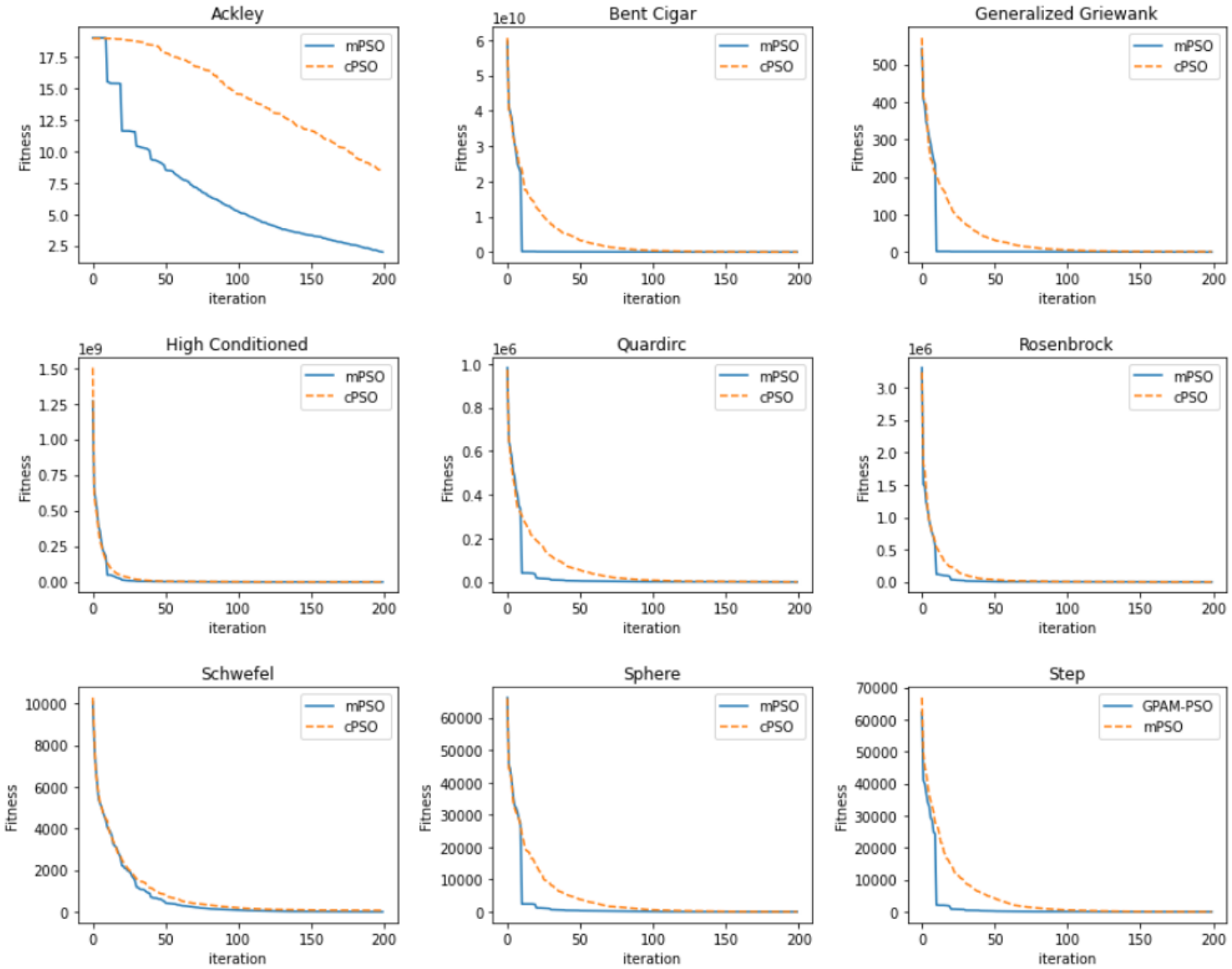


Figure 7

Evolution Curves Comparison of MPSO and CPSO

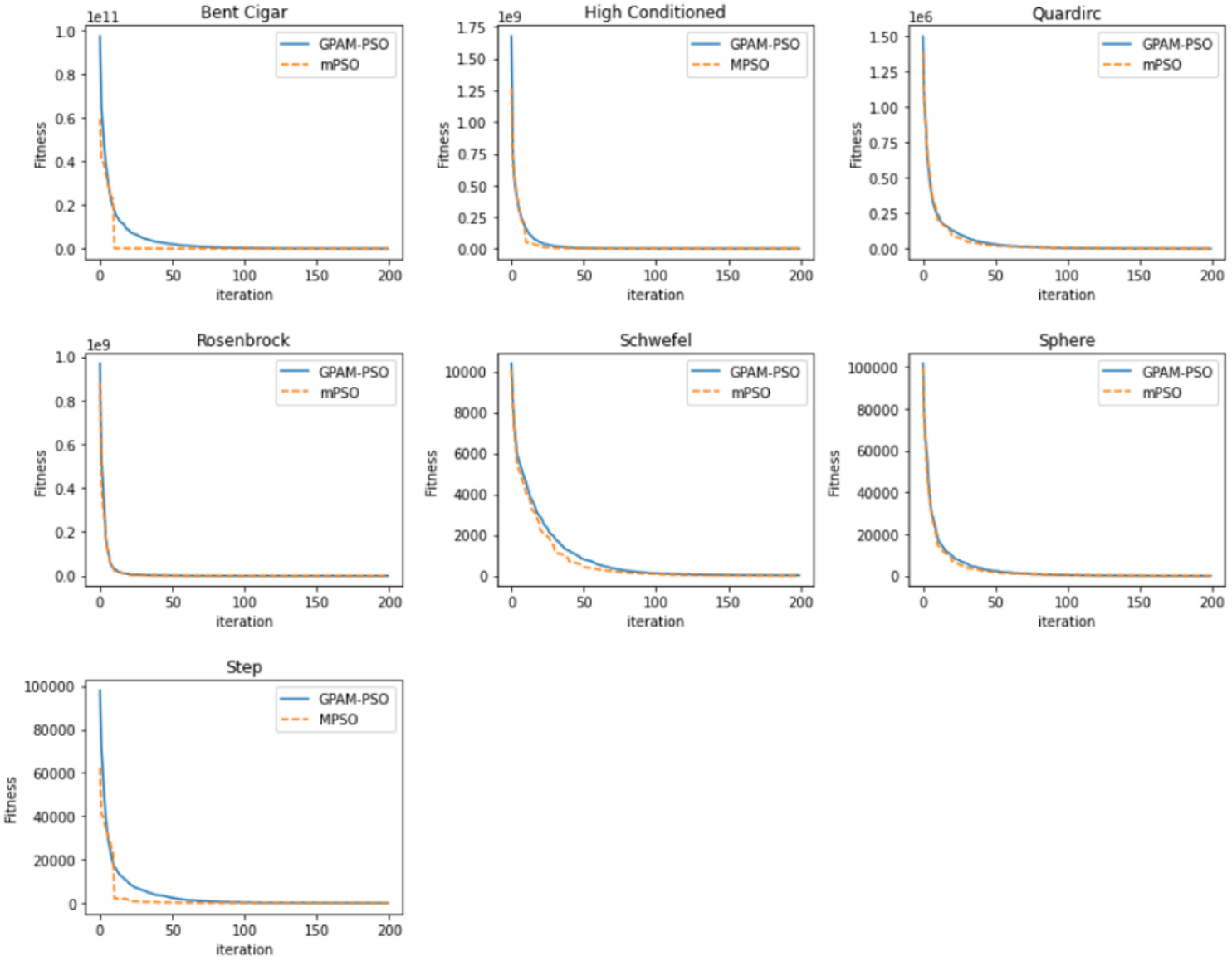


Figure 8

Comparison of the fitness decline speed of different types of functions

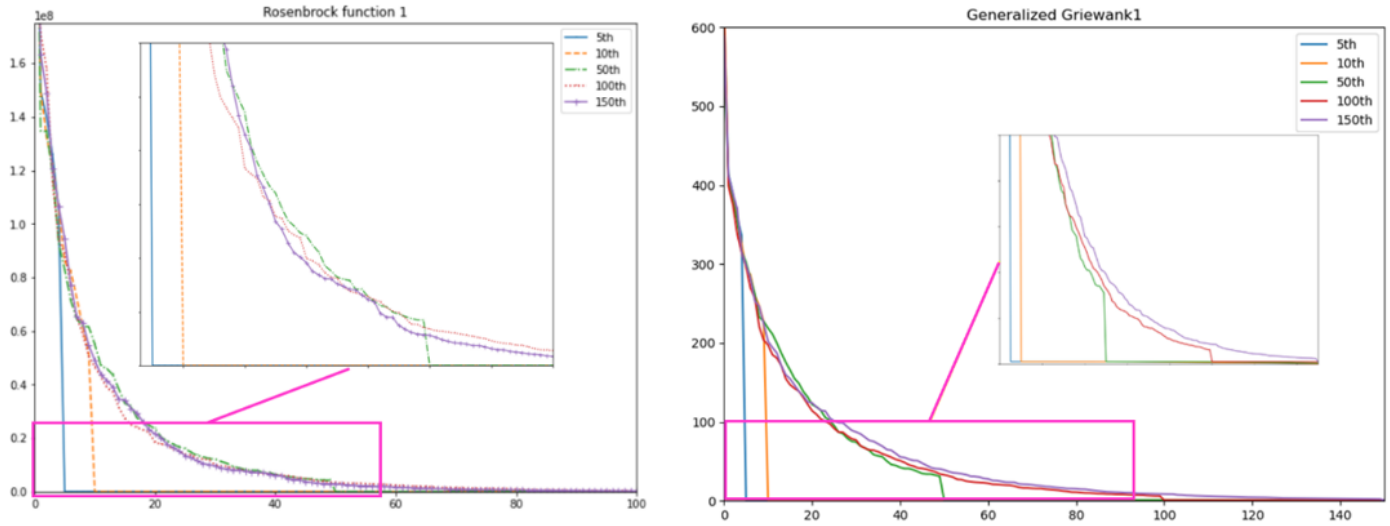


Figure 9

Evolution curves of Test1: Impact of migration iteration. We also enlarge the important part of the figure shown by the yellow (or purple as followed) box. Rosenbrock is showing in left while Generalized Griewank1 is showing in right. (The following figures use the same setting)

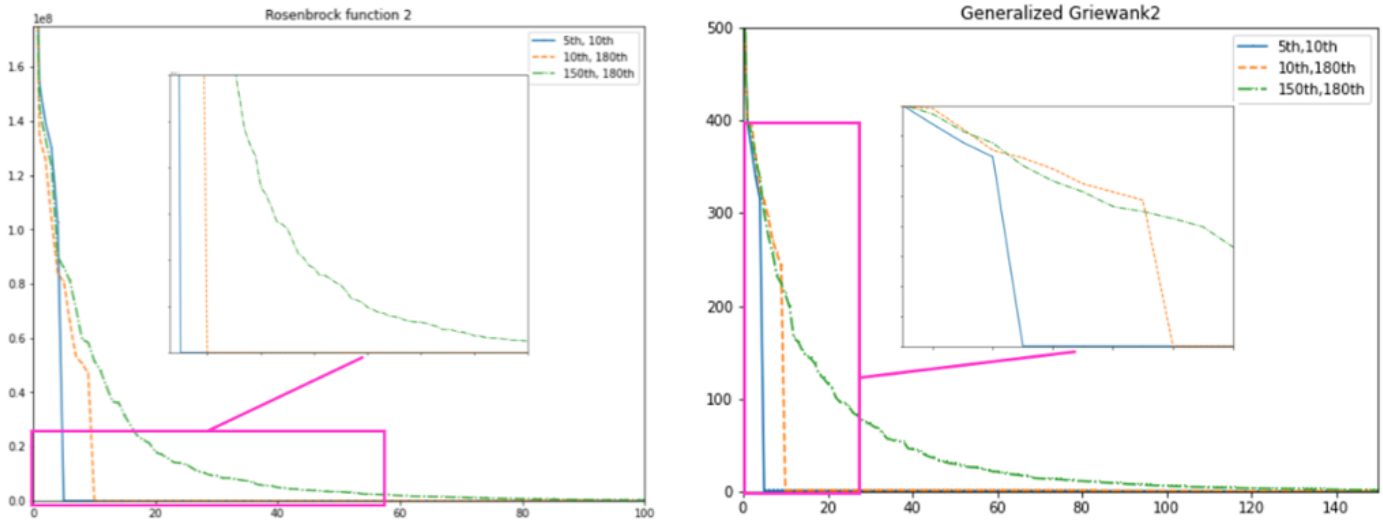


Figure 10

Evolution curves of Test2: Ablation Experiment on stages that apply migration operation.

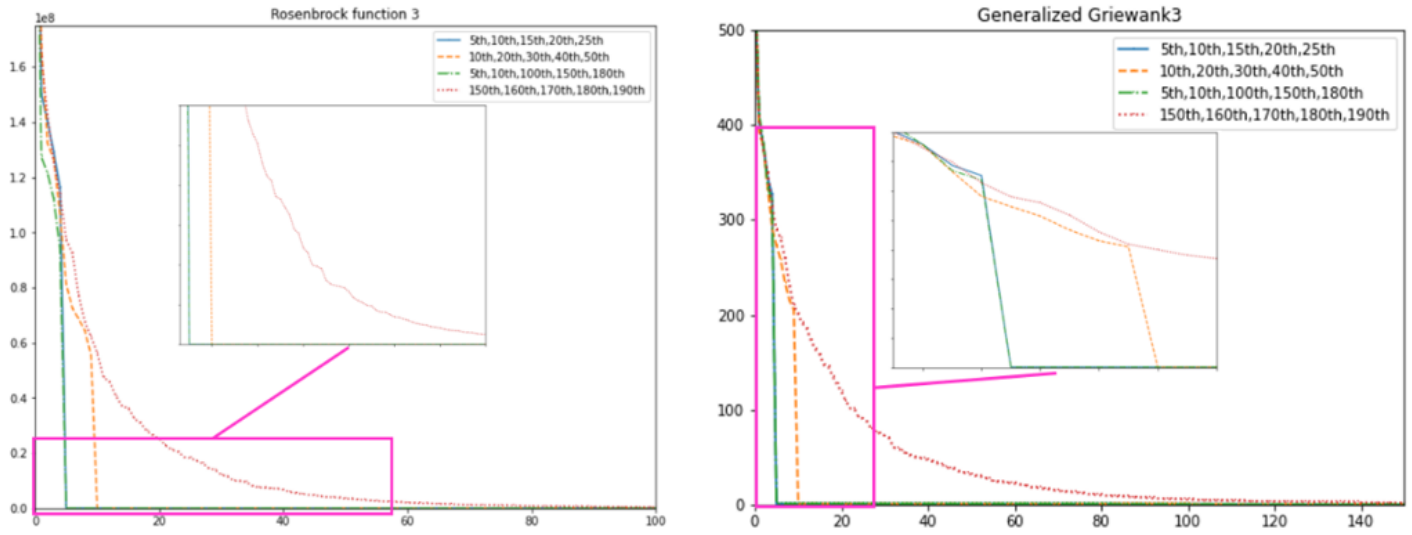


Figure 11

Evolution curves of Test3: The Migration operator is conducted 5 times at different positions.

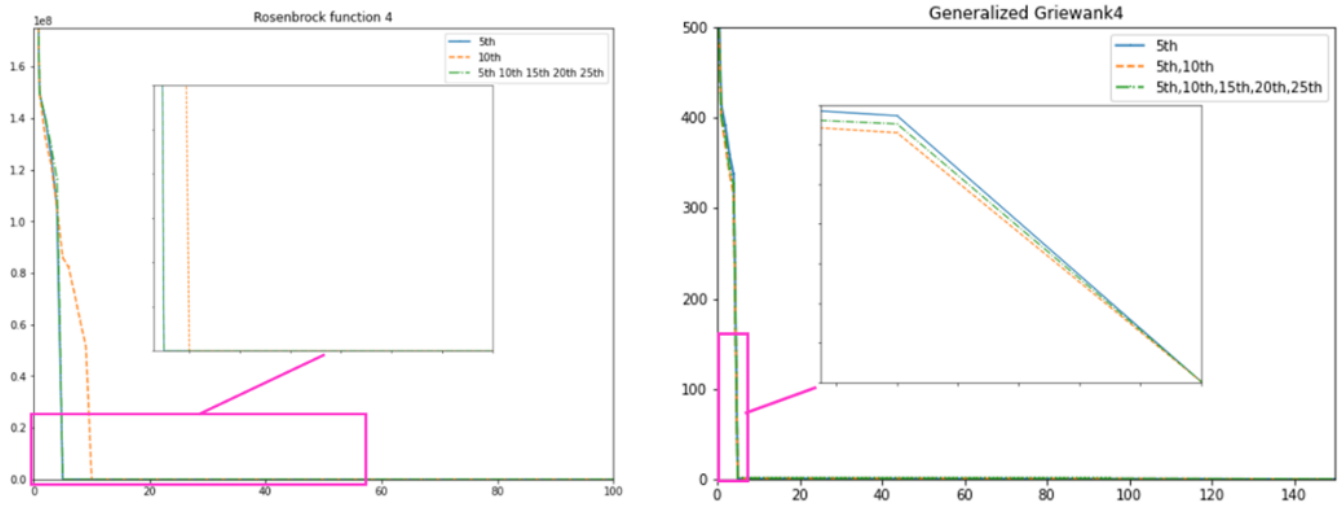


Figure 12

Comparison chart for best results: Different Migration frequency at the initial stage is compared.

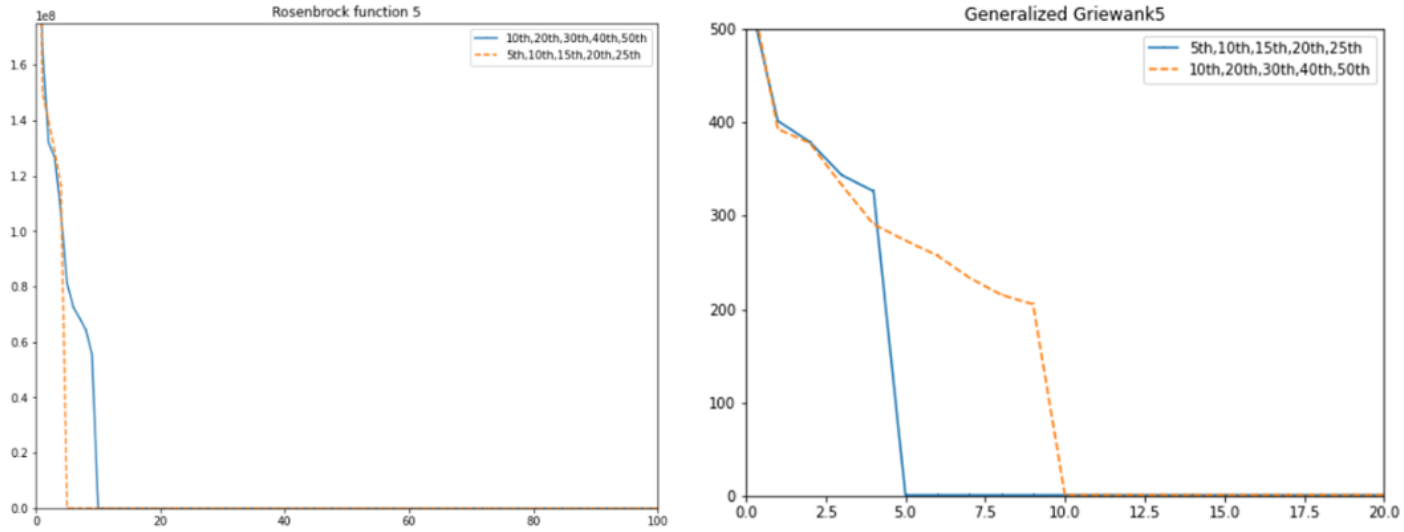


Figure 13

Evolution curves of Test4: The impact that extend migration interval to 10 iterations