A Comparison of Particle Swarm optimization and Global African Buffalo Optimization

IOP Conf. Series: Materials Science and Engineering **769** (2020) 012034 doi:10.1088/1757-899X/769/1/012034

A Comparison of Particle Swarm optimization and Global African Buffalo Optimization

Mohammed Adam Kunna Azrag[1*], Tuty Asmawaty Abdul Kadir[1], and Noorlin Mohd Ali[1]

[1] Faculty of Computing, College of Computing and Applied Sciences, Universiti Malaysia Pahang, 26300 Kuantan Pahang Malaysia

Mohammed87kunna@gmail.com

Abstract. The performance of Particle Swarm Optimization (PSO) brings attention to the field of algorithms when deals with different optimization problems. Due to her simple implementation, small consumption, and very effective in finding a solution in many problems, (PSO) becomes well known to the field of algorithms. In addition, the late proposed algorithms mostly are compared to the well-known algorithm such as PSO. Thus, the Global African Buffalo Optimization (GABO) was proposed lately and yet not been compared to the old well-known algorithms in terms of accuracy and time consumption. However, in this paper, a comparison between Particle Swarm Optimization (PSO) and Global African Buffalo Optimization (GABO) algorithms was performed. Five different nonlinear equations with their upper and lower boundaries values were selected as the test optimization functions problem in addition to PSO was applied to real case study. The experimental results illustrated the differences in the performances of both algorithms toward the optimum solution. At the end of the experiments, the PSO algorithm quickly convergence towards the optimum solution using a few particles and iterations rather than GABO. However, the experimental result showed that PSO achieved good results in all the test cases within a short time. In many cases, PSO and GABO are promising optimization methods.

Keywords: PSO; GABO; Convergence; Sphere Function; Rastrigrin function, Griewank Function; Rosenbrock Function; Shubert Function

1. Introduction

Optimization algorithms are applied in many applications such as business activities, engineering, and industrial designs with the aim of either minimizing or maximizing an objective function by methodologically selecting the input values [1]. Lately, nature-inspired optimization algorithms have increasingly been gaining popularity in the field of science and engineering [2]. This development has thrilled many researchers and they have deduced various reasons for it. Some researchers argue that these algorithms are successful because they were developed to replicate some of the most successful concepts in biological, physical, and chemical processes which occur naturally [3, 4]. With this situation, the issue of algorithmic choice always surfaces (since there exists so many to choose from) whenever the need for optimization arises [5]. There is a general understanding that the 'best' algorithm for solving any problem should mainly be selected based on the nature of the problem being faced. The 'No free lunch' optimization theorem reinforced this line of thought [6, 7]. In fact, there is no agreement on the

recommended principles guiding the choice of algorithms when faced with nonlinear large-scale optimization tasks [8].
In addition, the PSO was developed by Kennedy & Eberhart in 1995 with inspiration from the foraging behavior of birds. Different from evolutionary concept of Genetic algorithm (GA) which is based on Darwin's "survival of the fittest" concept, the PSO finds the optimal solution through collaboration between individuals. The study has found that the PSO is suitable for solving complex optimization problems [9,10]. The PSO algorithm has been greatly received and effectively applied to many function optimization and engineering technologies. It could be said that, through the analysis of biological communities, swarm intelligence of other complex behaviors such as their cooperation and competition can often produce some effective solutions to problems [9,10]. In addition, PSO algorithm has been successful in most cases than other algorithms such as GA, Simulating Annealing (SA), and Differential Evolution [11,12,13].
Thus, in this study, the PSO algorithm was studied based on the time consumption and convergence speed in order to reach global optima [9, 10]. Moreover, the Global African Buffalo Optimization (GABO) algorithm was proposed by [14] with inspiration from the global search behavior of the African Buffalo Optimization (ABO) algorithm [15]. The GABO algorithm showed better performances compared to the original ABO when studied with 2 optimization functions problem. However, this study focuses on the comparison between PSO and GABO algorithms by testing 3 global optimization functions based on the computational time and accuracy of each algorithm. The remaining part of this article is arranged as follows: Section two discussed the PSP and GABO methods while Section three explained the experimental process and the results of the study. Section four presented the conclusion of the study.

2. Methodology

2.1. Particle Swarm Optimization

The PSO was first presented by Kennedy & Eberhart [9, 10] as an optimization framework based on inspiration from the social life pattern and movement dynamics of birds, fish, and insects. In the PSO, the swarm is modelled by the particles and each particle has a velocity and position within a multidimensional space. It is used to find the best global position and its information on the best neighbour. When the particle in the swarm stochastically and independently search for their destination, they exchange information with each other, causing their movement towards the optimal point from different directions and paths. Consequently, they will be able to explore a wide search area, giving the best probability of establishing the global optimum [9, 10]. When compared to the other algorithms, one major advantage of PSO is that it is easy to implement because it does not require crossover, decoding, or encoding like the GA. Furthermore, its computational time is cost-efficient when compared to the other algorithms because it uses a small number of parameters. PSO is simple in both numerical and theoretic implementation; its simplicity, robustness, and stability made it suitable for application in several fields, such as electric motor, image processing, and automatic systems [16, 17, 18, 19]. Basically, the PSO is comprised of a swarm of "n" particles whose position individually represents a tentative solution to the fitness function within the dimensional search space. The updating of the particles' position is dependent on the following three factors, which are the particles' inertia weight ω, the local best position $p(t)$, and the global best position $G(t)$. These factors are calculated within the velocity and position of each particle using the following equations:

$$v_i(t+1) = \omega v_i + c_1 r_1\big(p(t) - x_i(t)\big) + c_2 r_2(G(t) - x_i(t))$$

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

Where $v_i(t+1)$ is the velocity of particle i at iteration t, ω is the inertia weight equal to 1, $c_1 c_2$ are the acceleration constant, $r_1 r_2$ are a random number between (0,1), $p(t)$ is the local best position already found by particle i until iteration t, $G(t)$ is the global best position found in the entire particle i until iteration t, and $x_i(t)$ is the particle position i at iteration t.

2.2. Global African Buffalo Optimization

With the ability of the buffalos to recognize the global best position in a standard ABO algorithm at step 3 [13], the buffalos were made to perform a research of the best position found by the ABO buffalos to ensure that the global best position are the same and as close enough to the global optimum solution as possible. This can be seen in step 4. With this idea, some modifications of the standard ABO algorithm are proposed. The modified algorithm selected the best position which has already been found by the ABO buffalos and then set them as the best position to be tested by considering the lower and the upper values of the herds' best ($\boldsymbol{bgmax}$). After this, the algorithm then starts to search for the optimum global solution. In step 5 if the global optimum solution was found, then ABO move to step 6 if not, it repeats steps 2-5. But Step 6 in GABO is asking to ensure the global best is reached if not repeat step 2-6 (This is the primary difference between ABO and GABO). Step 7 outputs the optimum values which means the best minimum values. In this way, the buffalos can search for more domains. The Pseudocode of the proposed GABO is presented in Figure 1:

1. Initialize the buffalos randomly to nodes within the solution space;
2. Update the exploitation of the buffalos using:
   $mk' = mk + lp1(bg - wk) + lp2(bp.k - wk)$, Where mk and wk represents the exploitation and exploration moves, respectively of the k[th] buffalo (k=1,2,…,N), $lp1$ and $lp2$ are learning factors, bg is the herds best fitness, and bp is the individual best buffalos locations.
3. Update the exploration fitness of the buffalos using the equation below:
   $mk' = (wk + mk)$.
4. Set the best position bp to be the new dimension of the global best position $bgmax$.
5. Is the global best position found and $bgmax$ is updating? Yes, go to step 6. If no repeat step 1 to 5.
6. If the stopping condition is not reached, revert to step 2, else, proceed to step 6.
7. Result the best solution.

Figure 1: Pseudocode of GABO

The performance of GABO was greatly improved with the introduction of the search for the optimum global solution of the best position that has already been found by buffalos into the original version of ABO. This improvement was evident through the experimental study carried out on the benchmark problem of Sphere and Rosenbrock functions. To further illustrate the relevance of this study, the results of the experiments using the two non-linear tested functions were discussed and reported [9].

3. Experimental Setup

For comparison, five nonlinear functions and one case study was investigated.

3.1. The Nonlinear functions

The Sphere function is the first function as described by Eq. 1:

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

Where $x = [x1, x2, \ldots, xn]$ is an n-dimensional real-valued vector.
The *Rosenbrock* function is the second function as described by Eq. 2:

$$f_1(x) = \sum_{i=1}^{n} (100(x_i - x_i^2)^2 + (x_i - 1)^2) \quad (2)$$

The generalized *Rastrigrin* function is the third function as described by Eq. 3:

$$f_2(X) = \sum_{i=1}^{n}(X_i^2 - 10\cos(2\pi X_i) + 10) \quad (3)$$

The fourth function is *Griewank* function described by equation (4):

$$f_3(X) = \frac{1}{4000}\sum_{i=1}^{n} X_i^2 - \prod_{i=1}^{n}\cos\left(\frac{X_i}{\sqrt{i}}\right) + 1 \quad (4)$$

The fifth function is *Shubert* function described by equation (5):

$$f_4(x) = (\sum_{i=1}^{5} i\cos((i+1)x_1 + i)(\sum_{i=1}^{5} i\cos((i+1)x_2 + i \quad (5)$$

As recommended by [9], the selection of the upper and lower as the original values was done to facilitate the benchmarking. The initialization of the lower and upper values of the two function is presented in Table 1.

Table 1: The lower and upper values

Function	The lower and upper values
$f(x)$	[-10, 10]
$f_1(x)$	[-5, 10]
$f_2(X)$	[-5.12, 5.12]
$f_3(X)$	[-100,100]
$f_4(X)$	[-10,10]

Studying the PSO and GABO algorithms, different population sizes were used for each function and these population sizes are 10, 20, and 30 particles and buffalos. The maximum number of iteration is set to be 30, 60, and 90 with the dimension is 5 in both algorithms. To get the mean global best position, each algorithm was evaluated 5 times.

3.2. Case study

Parameter estimation of the kinetics in metabolic model is difficult task due to this kinetics are reported from different laboratories in different condition, many pathways, and ordinary differential equations (ODE) consuming huge time. Thus, the main metabolic model of *Escherichia Coli* (*E. Coli*) formulated by [20, 22] was used as a case study due to its large scale kinetic parameters and the estimation may can be performed. This model contains 5 pathways with 172 kinetic parameters in addition to metabolites and enzymes.

4. Result

4.1. The functions result

The PSO global best position presented a perfect result when compared with the result of GABO. Moreover, as shown in Table 3, 4, 6, 7, 9, and 10, PSO presented a faster convergence speed towards the optimum values compared to GABO. However, PSO's convergence was quick in both functions. PSO required only 0.23 sec to achieve the global best position while 0.95 sec was required by GABO to achieve the same feat in *Sphere* function. In *Rosenbrock* function, PSO required only 0.35 sec to achieve the global best position while GABO required 1.03 sec. In *Rastrigrin* function, PSO required

0.41 sec to achieve the global best position while GABO required 0.92 sec. These comparisons are depicted below. In addition, the PSO algorithm searching wide area in all cases than GABO with 3100 and 2709 iteration respectively.
Table 2 presents the testing of the *Sphere* function using PSO & GABO on a 5-dimensional space with (10, 20, 30) particles and buffalos, and (30, 60, 90) iterations. Table 2 showed that PSO achieved a good global best position. In Table 3 & 4 below, the time consumption of *Sphere* function by PSO and GABO is presented. The PSO searching 3100 times in the *Sphere* function within 0.23s means that searching large space with a short time to ensure the optimum solution is given. While GABO searching 2709 times in the Sphere function within 2.42s.

Table 2: PSO & GABO results for *Sphere* function

Particles & buffalos size	Dimension	Iteration	PSO global best position	GABO global best position
10	5	30	2.2569e-5	1.2630e-4
		60	3.2074e-6	2.2053e-4
		90	2.7257e-7	2.0382e-5
20	5	30	5.2174e-7	2.2846e-4
		60	3.5328e-8	2.0258e-5
		90	2.0172e-9	3.0124e-3
30	5	30	3.2538e-9	2.5402e-5
		60	4.2860e-10	3.0245e-4
		90	4.2584e-11	3.5682e-6

Table 3: Time consumption of *Sphere* function by PSO

Function name	Calls	Total time	Self-time
PSO	1	1.169s	0.23s
Sphere	3100	1.146s	1.146s

Table 4: Time consumption of *Sphere* function by GABO

Function name	Calls	Total time	Self-time
GABO	1	3.37s	0.95s
Sphere	2709	2.42s	2.42s

Table 5 presented the testing of the *Rosenbrock* function using PSO & GABO. The parameters used are: 5-dimensional space with 10, 20, 30 particles & buffalos, and 30, 60, 90 iterations. There was improvement in the global best position of PSO compared to that of GABO (Table 5). Furthermore, PSO presented a faster convergence speed toward the optimum compared to GABO (Tables 6 & 7). However, it was observed that the PSO convergences quickly as it achieved the global best position within 0.035 sec while 1.03 sec was required for GABO to achieve a similar feat (Tables 6 & 7). Finally, the total time performance speed of the PSO and GABO algorithm is 1.607s and 2.59s respectively with 90 iterations.

Table 5: PSO & GABO results for *Rosenbrock* function

Particles & buffalos size	Dimension	Iteration	PSO global best position	GABO global best position
10	5	30	0	0.402
		60	0	0.375
		90	0	0.324
20	5	30	0	0.321
		60	0	0.284
		90	0	0.247
30	5	30	0	0.235
		60	0	0.215
		90	0	0.145

Table 6: Time consumption of *Rosenbrock* by PSO

Function name	Calls	Total time	Self-time
PSO	1	1.607s	0.035s
Rosenbrock	3100	1.572s	1.572s

Table 7: Time consumption of *Rosenbrock* function by GABO

Function name	Calls	Total time	Self-time
GABO	1	2.59s	1.03s
Rosenbrock	2709	1.56s	1.56s

In Table 8 below, the *Rastrigrin* function was tested using PSO and GABO. The parameters used are: 5-dimensional space with 10, 20, 30 particles & buffalos, and 30, 60, 90 iterations. It was discovered that the global best position of PSO improved better than the result of GABO as seen below in Table 5. Moreover, PSO's convergence towards the optimum was faster compared to GABO as presented in Tables 9 & 10. PSO took only 0.041 sec to reach the best global position while GABO required 0.92 sec to achieve the same feat (Tables 9 & 10). Finally, the total time performance speed of the PSO and GABO algorithm is 1.184s and 2.74s respectively with 90 iterations.

Table 8: PSO & GABO results for *Rastrigrin* function

Particles & buffalos size	Dimension	Iteration	PSO global best position	GABO global best position
10	5	30	2.3891e-2	1.0325
		60	2.2357e-4	1.0213
		90	3.6824e-2	1.0105
20	5	30	3.0217e-3	1.0023
		60	4.2017e-2	0.9982
		90	4.8170e-4	0.9407
30	5	30	1.0843e-5	2.5402e-5
		60	3.0281e-6	3.0245e-4
		90	4.9107e-7	3.5682e-6

Table 9: Time consumption of *Rastrigrin* by PSO

Function name	Calls	Total time	Self-time
PSO	1	1.184s	0.041s
Rastrigrin	3100	1.143s	1.143s

Table 10: Time consumption of *Rastrigrin* function by GABO

Function name	Calls	Total time	Self-time
GABO	1	2.74s	0.92s
Rastrigrin	2709	1.82s	1.82s

In Table 11 below, the *Griewank* function was tested using PSO and GABO. The parameters used are: 5-dimensional space with 10, 20, 30 particles & buffalos, and 30, 60, 90 iterations. It was discovered that the global best position of PSO improved better than the result of GABO as seen below in Table 5. PSO's convergence towards the optimum was faster compared to GABO (Tables 12 & 13). PSO required 0.051 sec to achieve the best global position while GABO took 0.71 sec to achieve the same feat (Tables 12 & 13). The total time performance speed of the PSO and GABO algorithm is 1.702 sec and 2.64 sec, respectively with 90 iterations.

Table 11: PSO & GABO results for *Griewank* function

Particles & buffalos size	Dimension	Iteration	PSO global best position	GABO global best position
10	5	30	0.00230	0.903
		60	0.00120	0.725
		90	0.00020	0.587
20	5	30	0.00040	0.504
		60	0.00051	0.358
		90	0.00064	0.179
30	5	30	1.4027e-3	0.165
		60	1.0004e-4	0.084
		90	2.2048e-5	0.067

Table 12: Time consumption of *Griewank* by PSO

Function name	Calls	Total time	Self-time
PSO	1	1.702s	0.051s
Griewank	3100	1.651s	1.651s

Table 13: Time consumption of *Griewank* function by GABO

Function name	Calls	Total time	Self-time
GABO	1	2.64s	0.71s
Griewank	2709	1.93s	1.93s

In Table 14 below, the *Shubert* function was tested using PSO and GABO. The parameters used are: 5-dimensional space with 10, 20, 30 particles & buffalos, and 30, 60, 90 iterations. It was discovered that the global best position of PSO improved better than the result of GABO as seen below in Table 5. Moreover, PSO's convergence towards the optimum was faster compared to GABO as presented in Tables 15 and 16. It took PSO 0.055 sec to achieve the global best position while GABO took 0.86 sec (Tables 15 & 16). The total time performance speed of the PSO and GABO algorithm is 1.326s and 2.79s respectively with 90 iterations.

Table 14: PSO & GABO results for *Shubert* function

Particles & buffalos size	Dimension	Iteration	PSO global best position	GABO global best position
10	5	30	0.0231	0.9237
		60	0.0095	0.8021
		90	0.0082	0.7292
20	5	30	0.0078	0.7023
		60	0.0064	0.6087
		90	0.0052	0.5742
30	5	30	0.0045	0.5247
		60	0.0031	0.4213
		90	0.0022	0.3214

Table 15: Time consumption of *Shubert* by PSO

Function name	Calls	Total time	Self-time
PSO	1	1.326s	0.055s
Shubert	3100	1.271s	1.271s

Table 16: Time consumption of *Shubert* function by GABO

Function name	Calls	Total time	Self-time
GABO	1	2.79s	0.86s
Shubert	2709	1.93s	1.93s

4.2. The estimation result

As it can be seen in the results function benchmark above, the PSO algorithm performed better than GABO algorithm in short time and searching wide area when testing five nonlinear functions. Thus, PSO was used to estimate 7 kinetic parameters of the main metabolic model of *E. coli* [20], were 8 metabolites of 12 experimental data [21] are optimized and described in Table 17 & 18 respectively. However, PSO algorithm achieved 29.36% distance minimization [22].

Table 17: Kinetic parameters estimation by PSO

Kinetics	**Original**	**lower**	**Upper**	**Kinetic estimation by PSO**
v_{max}^{pyk}	1.085	0.9	1.34	1.032
n_{pk}	3	2.5	3.25	2.647
icdh	24.421	23.9	24.6	24.306
k_{icdh}^{f}	289800	289799.4	289800.7	289799.65
$k_{icdhnadp}^{d}$	0.006	0.004	0.04	0.0372
$k_{icdhnadp}^{m}$	0.017	0.009	0.05	0.0482
v_{max}^{icl}	3.8315	3.3315	4.1	3.594

In Table 18 below, 8 metabolites of (Glc, $G6P$, $F6P$, PEP, $6PG$, $Ru5P$, $Xu5P$, and $E4P$) were optimized well and the rest of the others metabolites are not optimized due to the nonlinearity of the model, the

pathways, there is sum metabolites was lumped, and many kinetics reported or estimated from different laboratories in different conditions.

Table 18: The metabolites optimization

Metabolites	**Chassagnole**	**Kadir simulation**	**PSO Optimization**
Glc	0.0556	0.12203	0.1155
$G6P$	3.48	0.12989	0.21931
$F6P$	0.6	0.021457	0.022598
FDP	0.272	1.5186	2.5257
PEP	2.67	1.5076	1.9136
PYR	2.67	2.8279	3.1883
$6PG$	0.808	0.017854	0.01876
$Ru5P$	0.111	0.021398	0.022488
$Xu5P$	0.138	0.026516	0.027911
$S7P$	0.276	0.00473	0.0473
$R5P$	0.398	0.076388	0.027912
$E4P$	0.098	0.27837	0.003424
Distance minimization	0		29.36%

5. Conclusion

The performance of PSO and GABO algorithms were compared in this study based on selected test functions. The best position of the ABO served as the new search dimension for GABO when searching for the optimum values. PSO was studied to investigate the efficiency of the studied methods; five functions were employed, and one case study was investigated. The experimental outcome pointed towards PSO having a quicker convergence compared to the GABO. Because PSO works better than GABO, the kinetic parameters estimation was studied for PSO based on the main metabolic model of *E. coli*. Thus, PSO algorithm estimated 7 kinetic parameters and optimized 8 metabolites of 12 experimental data sets. Nevertheless, PSO was observed to perform accurately but requires more investigations on more complex multimodal, separable, and non-separable functions with GABO.

ACKNOWLEDGMENT
The authors would like to thank to Malaysia Ministry of Education for funding this work (grant number: FRGS/1/2018/ICT02/UMP/02/8: RDU 190181), all reviewers for the supportive comments and Universiti Malaysia Pahang for the support.

References

[1] Cheng, S., Shi, Y., & Qin, Q. (2015). Experimental Study on Boundary Constraints Handling in Particle Swarm Optimization from a Population Diversity Perspective. In Emerging Research on Swarm Intelligence and Algorithm Optimization (pp. 99-127). IGI Global.
[2] Rauff, J. (2015). Nature-Inspired Optimization Algorithms. Mathematics and Computer Education, 49(3), 208. Fusion, 57(5), 054007.

[3] Gandomi, A. H., & Alavi, A. H. (2012). Krill herd: a new bio-inspired optimization algorithm. Communications in Nonlinear Science and Numerical Simulation, 17(12), 4831-4845.
[4] Priami, C. (2009). Algorithmic systems biology. Communications of the ACM, 52(5), 80-88.
[5] Huang, H.-J., & Lam, W. H. (2002). Modeling and solving the dynamic user equilibrium route and departure time choice problem in network with queues. Transportation Research Part B: Methodological, 36(3), 253-273.
[6] Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. IEEE transactions on evolutionary computation, 1(1), 67-82.
[7] Xu, H., Caramanis, C., & Mannor, S. (2012). Sparse algorithms are not stable: A no-freelunch theorem. IEEE transactions on pattern analysis and machine intelligence, 34(1), 187-193.
[8] Ellison, C. L., Finn, J., Qin, H., & Tang, W. M. (2015). Development of variational guiding center algorithms for parallel calculations in experimental magnetic equilibria. Plasma Physics and Controlled.
[9] Shi, Y., & Eberhart, R. C. (1999). Empirical study of particle swarm optimization. Paper presented at the Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on.
[10] Eberhart, Russell, and James Kennedy. "Particle swarm optimization." Proceedings of the IEEE international conference on neural networks. Vol. 4. 1995.
[11] Ou, C., & Lin, W. (2006, June). Comparison between PSO and GA for parameters optimization of PID controller. In *2006 International conference on mechatronics and automation* (pp. 2471-2475). IEEE.
[12] Wang, B., Wu, Z., & Zhao, Z. (2010). Performance comparison of GA, PSO, and DE approaches in estimating low atmospheric refractivity profiles. *Wuhan University Journal of Natural Sciences*, *15*(5), 433-439.
[13] Parwekar, P., Rodda, S., & Mounika, S. V. (2018). Comparison between Genetic Algorithm and PSO for Wireless Sensor Networks. In *Smart Computing and Informatics* (pp. 403-411). Springer, Singapore.
[14] Azrag, M. A. K., Kadir, T. A., Odili, J. B., & Essam, M. H. A. (2017). A Global African Buffalo Optimization. International Journal of Software Engineering & Computer Sciences (IJSECS), 3, 138-145.
[15] Odili, J. B., & Kahar, M. N. M. (2015a). African Buffalo Optimization (ABO): a New MetaHeuristic Algorithm. Journal of Advanced & Applied Sciences, 03(03), 101-106.
[16] Singh, Narinder, Sharandeep Singh, S. B. Singh, and Shelly Arora. "Half mean particle swarm optimization algorithm." Int. J. Sci. Eng. Res 3, no. 8 (2012): 1-9." International Journal of Scientific and Engineering Research (2012): 1-9.
[17] Mukherjee, V., & Ghoshal, S. P. "Intelligent particle swarm optimized fuzzy PID controller for AVR system." Electric Power Systems Research (2007): 1689-1698.
[18] Feng, D., Wenkang, S., Liangzhou, C., Yong, D., & Zhenfu, Z. "Infrared image segmentation with 2-D maximum entropy method based on particle swarm optimization (PSO)." Pattern Recognition Letters (2005): 597–603.
[19] Ghamisi, Pedram, Micael S. Couceiro, Fernando ML Martins, and Jon Atli Benediktsson. "Multilevel image segmentation based on fractional-order Darwinian particle swarm optimization." IEEE Transactions on Geoscience and Remote sensing. IEEE, 2014. 2382-2394.
[20] Kadir, T. A. A., Mannan, A. A., Kierzek, A. M., McFadden, J., & Shimizu, K. (2010). Modeling and simulation of the main metabolism in Escherichia coli and its several single-gene knockout mutants with experimental verification. *Microbial cell factories*, *9*(1), 88.
[21] Chassagnole, Christophe, Naruemol Noisommit-Rizzi, Joachim W. Schmid, Klaus Mauch, and Matthias Reuss. "Dynamic modeling of the central carbon metabolism of Escherichia coli." *Biotechnology and bioengineering* 79, no. 1 (2002): 53-73.
[22] Mohammed Adam Kunna Azrag, Tuty Asmawaty Abdul Kadir, Muhammad Nomani Kabir, and

Aqeel S. Jaber, "Large-Scale Kinetic Parameters Estimation of Metabolic Model of Escherichia Coli," *International Journal of Machine Learning and Computing* vol. 9, no. 2, pp. 160-167, 2019.