Proceedings of the International Conference on Information Technologies (InfoTech-2017)
20-21 September 2017, Bulgaria

Invited paper

COMPARISON OF WILD DOG ALGORITHM, BAT ALGORITHMS AND BIOGEOGRAPHY BASED OPTIMISATION

Dr. Karl O. Jones, Tarek Zaibet and Grégoire Boizanté

Department of Electronics and Electrical Engineering, Liverpool John Moores University, Liverpool
e-mail: k.o.jones@ljmu.ac.uk
United Kingdom

Abstract: Evolutionary Computation (EC) is a diverse and growing research area. Some of the more recent developments within EC are African Wild Dog Algorithm (WDA), Bat Algorithm (BA), and Biogeography Based Optimisation (BBO). These can all be used in optimisation problems. This paper compares the effectiveness of these particular methods on an optimisation problem, in particular tuning a PID controller.
Key words: African Wild Dog Algorithm, Bat Algorithm, Biogeography Based Optimisation.

1. INTRODUCTION

Algorithms that primarily take their principles of operation from processes within nature, such as animal behaviour, come within the field of Evolutionary Computation (EC).

Probably the most widely known method in EC is Genetic Algorithms (GA): based on Darwin's Theory of Evolution. GAs reproduce natural evolutionary procedures on a population representing solutions to a specific problem. GAs have been utilised for PID tuning, producing successful results [1]. Moreover, a range of industrial problems have successfully employed GAs: thus the GA is regarded as an effective optimisation method. Alternative EC schemes exploit agent cooperation, analogous to animal social behaviour, whereby if a single agent is unable to accomplish a task, an accompanying agent might be better placed to complete the task.

Probably the most common industrial controller is the Proportional-Integral-Derivative (PID) controller. Its minimal structure allows for simple implementation, robust performance and application to numerous processes. Appropriate operation of the PID controller is subject to the determination of three parameters, namely proportional gain (K_p), integral action time (T_i) and derivative action time (T_d). Frequently these three parameters are manually tuned using trial and error, which can take a significant amount of time. To surmount this problem, various procedures have been expounded to assist in parameter tuning, such as Ziegler-Nichols [2]. The primary issue of using these procedures is that the solution found only satisfies the defined performance criteria for that approach. Controller tuning can be improved through using optimisation techniques, and especially those based on Evolutionary Computation.

This paper continues from two earlier papers [3][4] which considered the applicability of tuning a PID controller using Bees Algorithm [5], Particle Swarm Optimisation [6], Differential Evolution [7] and the Firefly Algorithm [8].

2. OPTIMISATAION METHODS

2.1 Biogeography-Based Optimisation (BBO)

BBO is an algorithm and metaheuristic inspired by the biogeographic concepts of the evolution of new species, species migration between islands and the extinction of species [9]. BBO optimizes a problem by stochastically refining candidate solutions through a number of iterations, with respect to a defined fitness function. The BBO approach makes no assumptions about the problem, thus it is applicable to a wide class of problems. For example, it has been used for robot controller tuning [10], and for diagnosis of cardiac disease [11]. The operation of BBO is as follows:

Step 1: Generate initial population (N) of candidate solutions (x_k)

Step 2: For each candidate x_k, set emigration probability μ_k proportional to the fitness value for x_k

Step 3: For each x_k, set immigration probability $\lambda_k=1-\mu_k$. $\{z_k\} \leftarrow \{x_k\}$

Step 4: For each individual in z_k
For each independent variable $s \in [1,n]$

Step 5: Use λ_k to probabilistically decide whether to immigrate to z_k

Step 6: If immigrating, then use $\{\mu_k\}$ to probabilistically select the emigrating individual x_j. $z_k(s) \leftarrow x_j(s)$

Step 7: Next independent variable $s \leftarrow s+1$

Step 8: Probabilistically mutate z_k. Next individual $k \leftarrow k+1$

Step 9 . $\{x_k\} \leftarrow \{z_k\}$

Step 10: Repeat Steps 2 to 9 until definite termination conditions are met, such as a pre-defined number of iterations or a failure to make progress for a fixed number of generations.

2.2 Bat Algorithm (BA)

The Bat algorithm is a recently developed nature inspired meta-heuristic optimization algorithm, based around on the echolocation behaviour of bats [12]. All species of bats use echolocation to sense distance, and to differentiate between prey (food) and background items. Bats fly randomly with velocity v_i, at position x_i, with a fixed frequency f_{min}, and loudness A_i to search for prey. As the bat searches for its prey, it changes the frequency, loudness and pulse emission rate, *r*, dependant on the proximity of the prey. BA has been applied on a continuous optimization problem by Parpinenli and Lopes [13]. The operation of the Bat Algorithm is:

Step 1: Initialisation. Randomly spread the bats into the solution space.

Step 2: Compute the fitness of each bat and find the current best.

Step 3: Move the bats by generating new solutions by adjusting frequency, velocity and location, using:

$$f_i = f_{min} + (f_{max} - f_{min})\beta \quad (1)$$

$$v_i^t = v_i^{t-1} + \left(\overrightarrow{x^{t-1}} - \overrightarrow{x_*}\right) f_i \quad (2)$$

$$\overrightarrow{v_i^t} = \overrightarrow{x^{t-1}} + v_i^t \quad (3)$$

Step 4: Generate a random number. If it is greater than the fixed pulse emission rate, move the bat by the random walk process.

Step 5: Evaluate the fitness of the bats and update the global near best solution.

Step 6: Check the termination condition to decide whether go back to step 3 or terminate the program and output the near best result.

2.3 African Wild Dog Algorithm (WDA)

WDA is based upon the communal hunting behaviour of African wild dogs [13]. African wild dogs live in packs, dominated by a monogamous breeding pair. They are effective at cooperation and are led by the male leader when hunting. They depend on their sense of sight rather than smell, and pursue their prey in a long, open chase until the prey becomes exhausted. African wild dogs maintain contact with each other in a variety of ways such as voice, smell (olfactory) and posture (body language). They each have a very strong odour, hence they can effortlessly detect other group members at a distance. Pack members vocalize to help coordinate their movements. African wild dog algorithm operates by using an iterative approach to simulate their group hunting behaviour, that is, to find the optimal value. African wild dogs solve the optimization

problem through the steps of initializing dogs' position, competing for the role of lead dog and collaborative moving. The algorithm is as follows:

Step 1: Initialize algorithm parameters. Randomly initialize the wild dog pack such that they occupy as much of the entire search space as possible.

Step 2: Determine the fitness value for each wild dog and hence sort the wild dogs.

Step 3: Collaborative moving: the i^{th} wild dog moves with a certain probability toward wild dog *j* that has a higher fitness value. The new position of wild dog *i* is given by:

$$x_{i,new} = x_i + rand \times (x_i - x_j) \times \left(\frac{ac}{b}\right) \quad (4)$$

where *rand* is a random number between 0 and 1, *a* is the average Euclidean distance of all wild dogs and *b* is the Euclidean distance between wild dog *i* and wild dog *j*, and *c* is the iteration step coefficient:

$$c = 1 - \left(\frac{Iteration\ number}{Max.iterations}\right) \quad (5)$$

Step 4: When hunting in packs, pack members vocalize to help coordinate their movements. Once they locate prey, African wild dogs quickly gather to the lead dog and round up the prey. When solving the objective function value, for this behaviour, generates a random number, r_θ, within the range [0,1]. If r_θ is greater than the pre-set threshold value, θ, then wild dog *i* moves towards the prey. Otherwise, it does not move and goes directly to the next iteration. The updated position is:

$$x_i^{t+1} = \begin{cases} x_i^t & r_m < \theta \\ x_j + rand \times ra & r_m > \theta \end{cases} \quad (6)$$

where r_a is rounding up step length, x_j is the position of the lead dog and x_i^t is the current position of the i^{th} wild dog in the t^{th} iteration. In order to provide more accurate solving, he rounding up step length decreases throughout the iterations:

$$r_a(t) = (x_{imax} - x_{imin}) \times e^{-Maxgen(t)} \quad (7)$$

where *Maxgen(t)* represents the i^{th} iteration.

Step 5: Repeat step 2 to step 4 until the termination criterion is satisfied.

3. APPLICATION RESULTS

The investigation into the efficacy of the methods for controller tuning, was based on establishing suitable optimized PID values for a defined second order system with time delay, given by:

$$G = \frac{7\ e^{-3s}}{7s^2 + 81s + 1} \quad (8)$$

The process was presented with a unit step input, as shown in the Simulink model provided in Figure 1. The presence of the time-delay element poses a particular problem for PID tuning. A selection of cost functions were utilised in this work (Table 1) to provide an appraisal of the action of the optimisation methods. There are alternate

cost functions that might be more effective for controller design; however, this work concentrates not on design but on the exploration of the optimisation algorithms. The result tables present the values of the determined PID parameters from the various algorithms, along with standard transient response criteria for the controlled system, such as the rise time, settling time (±2%) and percentage overshoot (Table 2-Table 5).

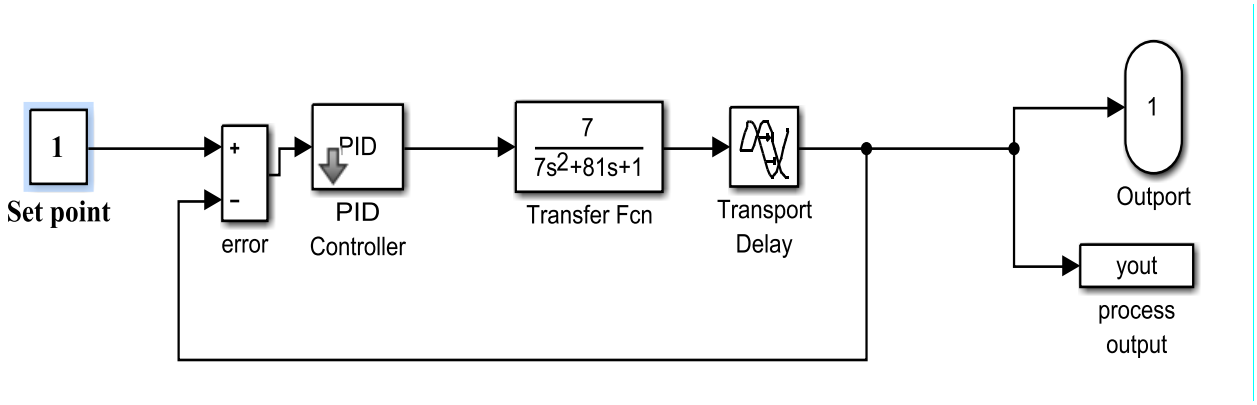

Figure 1 Simulink model of the control system

Table 1. *Cost Functions Employed*

Cost Function:	**Result Table**	**Transient Response**
ISE: Integral Square Error $CF = \int \varepsilon^2 dt$	Table 2	*Figure 2*
IAE: Integral Absolute Error $CF = \int \lvert\varepsilon\rvert dt$	Table 3	*Figure 3*
ITSE: Integral Time Square Error $CF = \int t\varepsilon^2 dt$	Table 4	*Figure 4*
ITAE: Integral Time Absolute Error $CF = \int t\lvert\varepsilon\rvert dt$	Table 5	*Figure 5*

Utilising the Zeigler-Nichols design criteria [2] as a comparison for controller performance (that is, 25% overshoot, an extended settling time and an intermediate rise time) then in all circumstances the BBO and BA tuned PID responses surpass these criteria. In fact, for all four cost functions, BBO and BA determine almost identical PID values. The performance of the WDA tuned controller varies depending on the cost function under consideration. For the ISE cost function, WDA produces PID values comparable to those from BBO and BA. For the ITSE cost function, WDA determines PID values that produce an improved overshoot over the BBO and BA determined values, although the settling time is much longer. Considering ITAE, WDA determines PID values that have a process performance that is similar to that for BBO and BA, however the performance of the IAE tuned PID is significantly below acceptable standards.

Table 2. *Results for the ISE cost function.*

Optimisation Algorithm	PID			Process Response			Fitness value
	Kp	Ti	Td	Overshoot (%)	Settling time (2%)	Rise Time (s)	
BBO	3.959	0.037	5.966	16.61	24.08	2.35	415.96
BA	3.959	0.037	5.966	16.32	24.28	2.35	415.97
WDA	3.965	0.038	6.061	16.53	24.38	2.34	415.97

Table 3. *Results for the IAE cost function.*

Optimisation Algorithm	PID			Process Response			Fitness value
	Kp	Ti	Td	Overshoot (%)	Settling time (2%)	Rise Time (s)	
BBO	3.265	0.029	4.384	5.58	13.96	2.93	957.21
BA	3.264	0.029	4.384	5.57	13.96	2.93	957.23
WDA	3.624	0.038	8.072	10.1	43.52	8.54	1093.10

Table 4. *Results for the ITSE cost function.*

Optimisation Algorithm	PID			Process Response			Fitness value
	Kp	Ti	Td	Overshoot (%)	Settling time (2%)	Rise Time (s)	
BBO	3.684	0.034	5.303	12.24	17.35	2.53	924.99
BA	3.684	0.034	5.303	12.24	17.35	2.53	924.99
WDA	3.784	0.039	6.916	8.69	30.37	2.45	970.89

Table 5. *Results for the ITAE cost function.*

Optimisation Algorithm	PID			Process Response			Fitness value
	Kp	Ti	Td	Overshoot (%)	Settling time (2%)	Rise Time (s)	
BBO	3.023	0.028	3.702	3.07	13.42	3.34	1518.46
BA	3.022	0.028	3.700	3.06	13.41	3.34	1518.46
WDA	2.362	0.025	1.491	3.57	14.74	4.60	2008.96

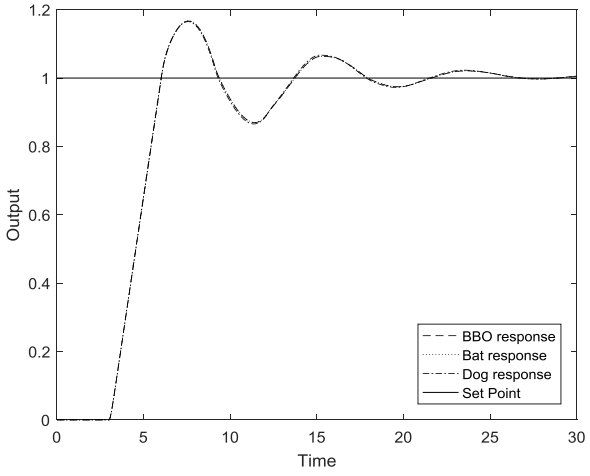

Figure 2. Responses for ISE cost function.

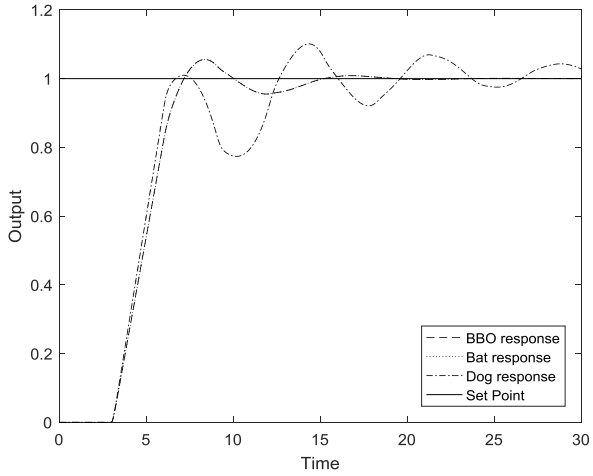

Figure 3. Responses for IAE cost function.

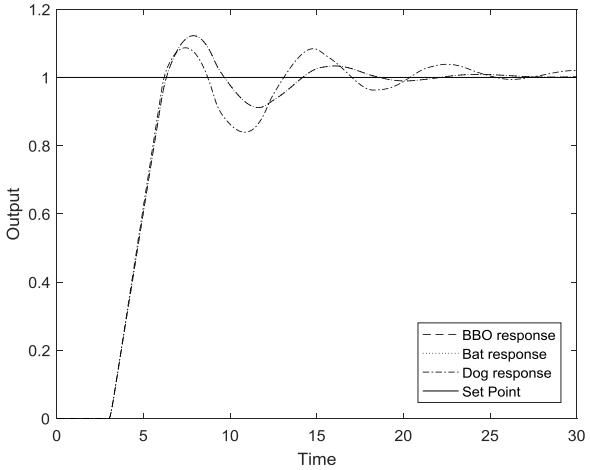

Figure 4. Responses for ITSE cost function.

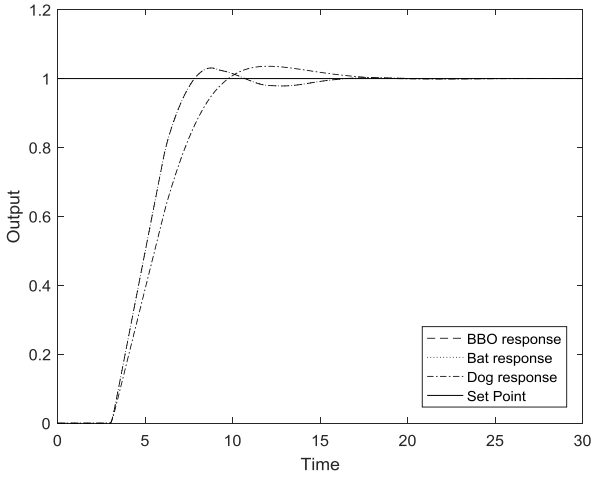

Figure 5. Responses for the ITAE cost function.

4. CLOSING COMMENTS

Regarding the transient responses of the controlled system, two of the three EC algorithms (BBO and BA) are able to determine PID values the give process transient responses that are an improvement over the classical Zeigler-Nichols response. BBO and BA produce very similar PID values for all cost functions, suggesting that their approaches to determining a solution is similar. The outcome for the WDA is somewhat mixed, in that its performance seems linked to the cost function employed.

The power of Evolutionary Computation approaches comes from the parallel nature of their search capabilities. Techniques such as those considered in this work, have presented themselves as successful solutions for optimisation problems. In spite of this, it should be noted that these algorithms are not a universal solution, despite their apparent robustness. Each approach has a range of control parameters that the user must set, and it is the apt selection of these parameters that is fundamental to the algorithm's success. Furthermore, one problem remains for all algorithms, specifically the proper selection of the cost function: an issue with all optimisation techniques.

REFERENCES

[1] Herrero, J.M., *et al. (***2002**). Optimal PID tuning with Genetic Algorithms for Non-Linear process Models. IFAC 1^{5th} Triennial World Congress, Barcelona, Spain.

[2] Ziegler, J.G. and Nichols, N.B. (**1942**). Optimum settlings for automatic controllers. ASME Transactions, (Vol. 64), pp. 759-768.

[3] Jones, K.O. and Bouffet, A. (**2008**). Comparison of Bees Algorithm, Ant Colony Optimisation, and Particle Swarm Optimisation for PID Controller Tuning. 9th International Conference on Computer Systems and Technologies. pp. IIIA.9-1(6).

[4] Jones, K.O. and Boizanté, G. (**2011**). Comparison of Firefly Algorithm Optimisation, Particle

Swarm Optimisation and Differential Evolution. 12th International Conference on Computer Systems and Technologies. pp. 191-197.

[5] Pham D.T., Ghanbarzadeh A., Koç E., Otri S., Rahim S., and Zaidi M. (**2006**). The Bees Algorithm, A Novel Tool for Complex Optimisation Problems. Proc 2nd Virtual International Conference on Intelligent Production Machines and Systems. Elsevier (Oxford), pp.454-459.

[6] Kennedy, J. and R.C. Eberhart. (**2001**). Swarm Intelligence. Morgan Kaufman Publishers.

[7] Storn, R. and K. Price. (**1995**). Differential Evolution – A Simple and Efficient Adaptive Scheme for Global Optimisation over Continuous Spaces. Technical Report TR-95-012, ICSI. Available via ftp://ftp.icsi.berkeley.edu/pub/techreports/1995/tr-95012.ps.z

[8] Yang, X.S. (**2008**). *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, UK.

[9] Simon, D. (**2008**). Biogeography-Based Optimisation. IEEE Transactions on Evolutionary Computation, Vol. 12, No. 6, pp. 702-713.

[10] Thomas, G. , Lozovyy, P. and Simon, D. (**2011**). Fuzzy Robot Controller Tuning with Biogeography-Based Optimization. 24th International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems, Syracuse, New York, pp. 319-327.

[11] Ovreiu, M. and Simon, D. (**2010**). Biogeography-based optimization of neuro-fuzzy system parameters for diagnosis of cardiac disease. Genetic and Evolutionary Computation Conference, Portland, Oregon, pp. 1235-1242.

[12] Yang, X.S. (**2010**). A new mataheuristic bat inspired algorithm. In *Nature Inspired Cooperative Strategies for Optimization*. Eds. González, J.R., Pelta, D.A., Cruz, C., Terrazas, G and Krasnogor, N. Vol. 284. Pp. 65-74.

[13] Parpinelli, R. and Lopes, H. (**2011**). New inspirations in swarm intelligence: a survey. International Journal of Bio-Inspired Computation. Vol. 3 Issue 1, pp. 1-16.

[14] Buttar A.S., Goel A. K. and Kumar S. (**2010**). Wild Intelligence: A Novel Intelligence as Dog Group Wild Chase & Hunt Drive (DGWCHD). *Indian Patent Office Journal,* Issue No.14, pp 7873.