10

Improved Dynamic Virtual Bats Algorithm for Identifying a Suspension System Parameters

Ali Osman Topal

Department of Computer Engineering
Epoka University, Tirana, Albania

CONTENTS

10.1 Introduction

Dynamic Virtual Bats Algorithm (DVBA), by Topal and Altun [3], is another meta-heuristic algorithm inspired by bats' ability to manipulate frequency and wavelength of sound waves emitted during their hunt. In DVBA, a role-based search is developed to improve the diversification and intensification capability of the Bat Algorithm. There are only two bats: explorer and exploiter bat. While the explorer bat explores the search space, the exploiter bat makes an intensive search of the locale with the highest probability of locating the desired target. During the search bats exchange roles according to their positions.

Experimental results show that DVBA is suitable for solving most of the low dimensional problems. However, DVBA, similar to other evolutionary algorithms, has some challenging problems. The convergence speed of DVBA is slower than other population-based algorithms like PSO, GA, and BA.

Additionally, in high dimensional multimodal problems, escaping from the local optima traps becomes a difficult task for DVBA. Therefore, accelerating convergence speed and avoiding the local optima have become two important issues in DVBA. To minimize the impact of this weakness, an improved version of DVBA is proposed which accelerates convergence speed and avoids the local optima trap. To achieve both goals, a new search mechanism is proposed for the explorer bat [4]. This new search mechanism improves the search performance and gives DVBA more powerful exploitation/exploration capabilities. The rest of the chapter is organized as follows: in Section 10.2 original DVBA is summarized, in Section 10.3 we present improved DVBA, and in Section 10.4 IDVBA is used for identifying parameters of a suspension system. Finally, the conclusion is presented in Section 10.5.

10.2 Original Dynamic Virtual Bats Algorithm (DVBA)

The dynamic virtual bats algorithm (DVBA), proposed in 2014 for global numerical optimization, is a recently introduced optimization algorithm which imitates the bats' echolocation behavior in nature. When bats search out prey, they burst sound pulses with lower frequency and longer wavelengths so the sound pulses can travel farther. In this long range mode it becomes hard to detect the exact position of the prey; however, it becomes easy to search a large area. When bats detect prey, the pulses will be emitted with higher frequency and shorter wavelengths so that bats are able to update the prey location more often [6, 7]. In DVBA, two bats are used to imitate this hunting behavior. Each bat has its own role in the algorithm and during the search they exchange these roles according to their positions. These bats are referred as explorer bat and exploiter bat. The bat that is in a better position becomes the exploiter; meanwhile the other becomes the explorer. While the exploiter bat increases the intensification of the search around the best solution, the explorer bat will continue to explore other solutions.

In Fig. 10.1 and Fig. 10.2, the hunting strategy of a bat is simulated. The black triangle is the current solution (bat location), the black circles are the visited solutions on the search waves in this iteration. During the search, the explorer bat's search scope gets in its widest shape; the distance between the search waves and the angle between the wave vectors (red dashed arrows) get larger(see Fig. 10.1). On the contrary, if the bat becomes the exploiter bat, its search scope gets its narrowest shape; the distance between the search waves and the angle between the wave vectors get smaller (see Fig. 10.2). The number of the visited solutions is the same for both bats, just the distance between the solutions changes dynamically by using wavelength and frequency.

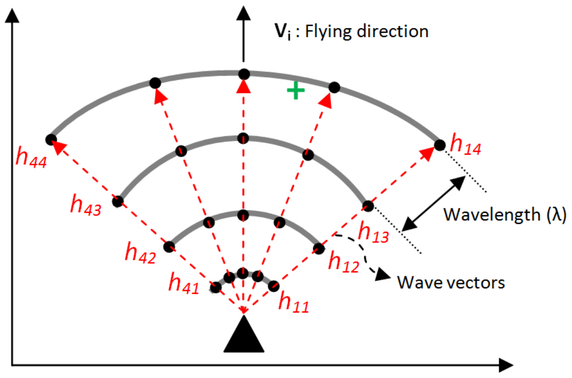

FIGURE 10.1
Exploration: Explorer bat is searching for prey with a wide search scope.

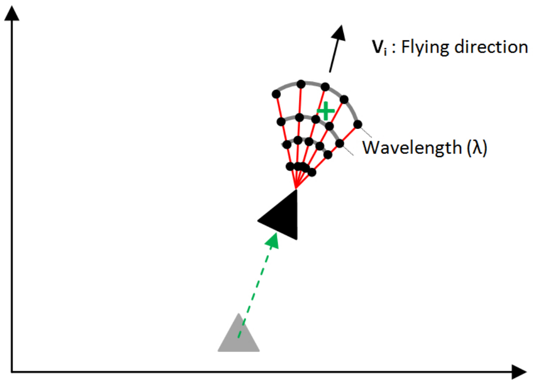

FIGURE 10.2
Exploitation: Exploiter bat is chasing prey with a narrow search space.

The wavelength and the distance between the solutions are proportional. The frequency and the angle between the wave vectors are inversely proportional.

The algorithm determines the best solution in the bat's search scope. If it is better than the current location (solution), the bat will fly to the better solution, decrease the wavelength and increase the frequency for the next iteration. These changes are targeted to increase the intensification of the search. Unless there is no better solution than the current solution in the search scope, the bat will stay on it, turn around randomly and keep scanning its nearby surrounding space. It will keep spinning in this position and expanding its search scope until it finds a better solution. Bats also exchange roles according to their position.

10.3 Improved Dynamic Virtual Bats Algorithm (IDVBA)

10.3.1 The weakness of DVBA

In DVBA, the explorer bat's search scope size is limited by the wavelength which might not be large enough to detect better solutions near its surrounding space. Thus, it is very likely that the explorer bat will be trapped in a local optima. In addition, the exploiter bat's search scope size becomes very small during the exploitation and it moves very slowly. Therefore it might need too much time to reach the global optima. These problems in DVBA have been eradicated by introducing probabilistic selection restart techniques in IDVBA.

10.3.2 Improved Dynamic Virtual Bats Algorithm (IDVBA)

To improve search performance and give DVBA more powerful search capabilities, two probabilistic selections are introduced: R - random flying probability and C - convergence probability in [4]. If the explorer bat is stuck in large local minima, it chooses to fly away from the trap randomly with a probability R related to number of unsuccessful attempts. In addition, it chooses to fly near to the exploiter bat with a probability C related to the number of escapes attempted from the traps. R and C are calculated as follows:

$$R_i^{t+1} = R_i^t[1 - 1/(\gamma_r \times trial_i)], \tag{10.1}$$

$$C_i^{t+1} = C_i^t[1 - 1/(\gamma_c \times rfly_i)], \tag{10.2}$$

where γ_c and γ_r are constant. $trial_i$ and $rfly_i$ denote the number of unsuccessful attempts and the number of random restarts, respectively. Obviously, the higher the $trial_i$ is, the greater the probability that the explorer bat might fly away from the trap to a random solution in the search space. As the unsuccessful attempts increase, the random flying probability R_i decreases and the possibility of $rand() < R_i$ being true (line 21 in Algorithm 13) increases. This can help the explorer bat to escape from the local optima trap rapidly. However, the explorer bat should not leave the trap without exploring the nearby surrounding space. Thus, γ_r should be chosen carefully.

The convergence probability C gives the possibility to the explorer bat to converge with the exploiter bat, instead of exploring a random position. Thus, the exploitation speed will be increased rapidly around the best position. Time after time the explorer bat visits the exploiter bat to speed up the exploitation process and then flies away randomly to keep up the exploration process. This also increases the exploitation capability of IDVBA. As shown in Eq. 10.2, the

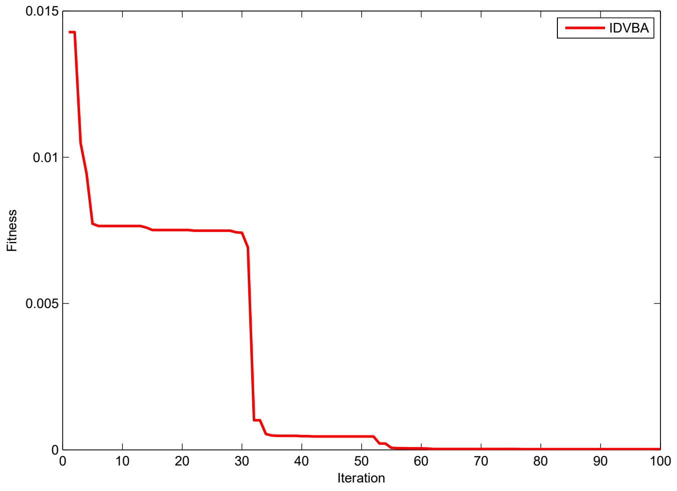

FIGURE 10.3
The convergence characteristic of IDVBA on 2D Rastrigin function.

convergence probability C is inversely proportional to the number of random restarts $rfly_i$ done by the explorer bat. Thus, increasing random restarts will increase the probability of $rand() > C$ that allows the explorer bat to visit the exploiter bat more often (line 30-31 in Algorithm 13).

In Fig. 10.3 the convergence characteristic of IDVBA is shown. Here, IDVBA is trying to optimize the multi-model Rastrigin function in 100 iterations. It can be easily seen, when the explorer bat visited the exploiter bat. As is expected, both bats collaborate to scrutinize the best found solution together, which increases the convergence speed rapidly during the visit.

Algorithm 13 IDVBA pseudo code. f_{gbest} is the global best solution and d is the number of dimensions. The code we discuss in the text is in boldface.

```
1:  Objective function $f(x)$, $x = (x_1, ..., x_d)^T$
2:  Initialize the bat population $x_i (i = 1, 2)$ and $v_i$
3:  Initialize wavelength $\lambda_i$ and frequency $f_i$
4:  Define the parameters: $\beta$, the scope width variable $b$
5:  Initialize the number of the waves(j) and search points(k)
6:  Find $f_{gbest}$ based on the bats' starting positions
7:  $\underline{C_i = R_i = 1.0}$
8:  while ($t <$ Max number of iterations) do
9:      for each bat do
10:         Create a sound waves scope
11:         Evaluate the solutions on the waves
12:         Choose the best solution on the waves, $h_*$
```

```
13:         if $f(h_*)$ is better than $f(x_i)$ then
14:             Move to $h_*$, update $x_i$
15:             Change $v_i$ towards the better position
16:             Decrease $\lambda_i$ and increase $freq_i$
17:             $\underline{trial_i = 0}$
18:         end if
19:         if $f(x_i)$ is not better than the best solution $f_{gbest}$ then
20:             $\underline{\textit{Calculate the random flying probability } R_i \textit{ by } Eq.10.1}$
21:             if $\underline{rand() < R_i}$ then
22:                 $\overline{\text{Change the direction randomly}}$
23:                 $\underline{trial_i = trial_i + 1}$
24:             else
25:                 $\underline{\textit{Restart the search from a random position}}$
26:                 $\underline{\textit{Reset } R_i \textit{ and } trial_i}$
27:                 $\underline{rfly_i = rfly_i + 1}$
28:                 $\underline{\textit{Calculate the } C_i \textit{ by } Eq.10.2}$
29:             end if
30:             if $\underline{rand() > C_i}$ then
31:                 $\underline{\textit{Produce a new solution around the exploiter bat.}}$
32:                 $\underline{\textit{Reset } C_i \textit{ and } rfly_i}$
33:             end if
34:             Increase $\lambda_i$ and decrease $freq_i$
35:         end if
36:         if $f(x_i)$ is the best found solution then
37:             Minimize $\lambda_i$ and maximize $freq_i$
38:             Change the direction randomly
39:             $\underline{\textit{Reset } rfly_i \textit{ and } trial_i}$
40:         end if
41:     end for
42:     Rank the bats and find the current best $x_{gbest}$
43: end while
```

In our experiments, we set $\gamma_r = 12$ and $\gamma_c = 10$. By using these values, in a 100 iterations search, the explorer bat is given the chance to fly randomly 6 to 8 times and also, it is given the chance to visit the exploiter bat 3 to 6 times. IDVBA is given as in Algorithm 13. The differences from DVBA are shown in underlined font.

10.4 Application of IDVBA for identifying a suspension system

In this section IDVBA is applied to identify parameters of a suspension system. A quarter-car model was presented in response to the vertical force for the suspension system [2] as shown in Figure 10.4.

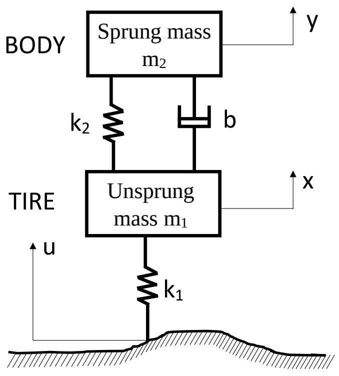

FIGURE 10.4
A quarter-car model of suspension system.

In this figure, m_1 is the mass of the tire, k_1 is the tire stiffness, m_2 is the mass of the vehicle, k_2 is the suspension stiffness, and b is the damping coefficient. u is the road displacement which is our input and y is the vertical displacement of sprung mass m_1 and that is our output.

The dynamic equations of motion for the displacements x and y are shown as follows:

$$m_1 \frac{d^2x}{dt^2} + b(\frac{dx}{dt} - \frac{dy}{dt}) + k_1(x - u) + k_2(x - y) = 0 \tag{10.3}$$

$$m_2 \frac{d^2y}{dt^2} + b(\frac{dy}{dt} - \frac{dx}{dt}) + k_2(y - x) = 0 \tag{10.4}$$

Our aim is to identify/estimate the parameters of a quarter-car suspension system. The basic idea of parameter estimation is to compare the real system output with the estimated model output for the same input [5]. We check how well the estimated model response fits the actual system response. To test the estimated model system we will use excitation signals that correspond to a realistic excitation of the system in real life. Consequently, in order to estimate the system parameters, an impulse signal is given as the excitation signal to a real and estimated model as shown in Figure 10.5 [2]. Then, the outputs are given as inputs to generate the objective function f, where the fitness will be calculated. We used the sum of squares error between the real and estimated model responses as objective function:

$$f = \sum_{n=0}^{N} (y[n] - y'[n])^2 \tag{10.5}$$

where, $y[n]$ and $y'[n]$ are real and estimated values in each sample, respectively, and N is number of samples. IDVBA will try to minimize the sum of squares

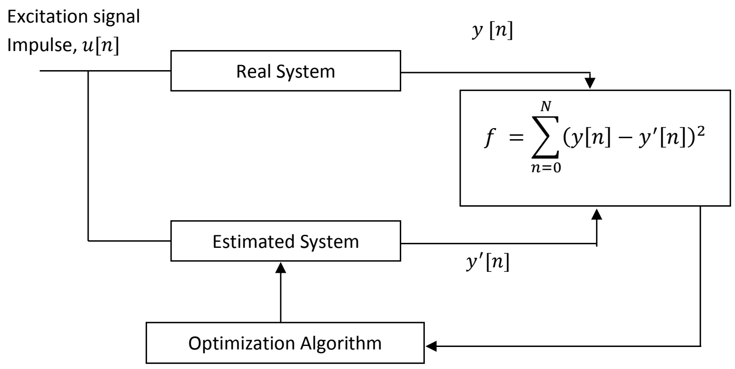

FIGURE 10.5
The estimation process.

error f by changing the parameters(m_1, m_2, k_1, k_2, and b) of the estimated model.

The quarter-car's nominal parameters are summarized as follows [1]:

$m_1 = 26kg$,
$m_2 = 253kg$,
$k_1 = 90000N/m$,
$k_2 = 12000N/m$,
$b = 1500N.sec/m$

The estimated model's parameters ranges are set as follows [1]:

$20 \leq m_1 \leq 30$,
$200 \leq m_2 \leq 300$,
$8500 \leq k_1 \leq 90000$,
$10000 \leq k_2 \leq 15000$,
$1200 \leq b \leq 1700$.

In order to test the efficiency of IDVBA, it is compared with the original DVBA. We used Equation 10.5 as objective function. In Equation 10.5, previously mentioned nominal parameters are used to find the displacement (y) of the real system for an impulse input signal. For the comparison, we set N=500 samples and iterations 100. The algorithms parameters are set as follows:

number of wave vectors: j=5,
number of search points: k=6,

number of search points on the search scope: $jxk = 30$ for each bat
β: 100,
scope wideness variable: b=20
And also, for IDVBA we set $\gamma_r = 12$ and $\gamma_c = 10$.

In optimization algorithms, the step size plays a very important role in terms of convergence speed and accuracy. If it is very small, accuracy can increase, however the convergence speed might slow down. In DVBA, the step size ρ is used to change the distance between the search points on the bat's search scope as shown in Figure 10.1. To decide the step size, we should pay attention to the size of the search space. If the parameters of the optimization problem have different ranges, like the quarter-car's parameters, different step sizes should be used for each parameter. For example, if we decide the step size based on k_1, it would be too big for m_1. While k_1 moves slowly, m_1 will just jump from its maximum to minimum values. So, ρ has different values for each parameter.

The test results are shown in Table 10.1 in terms of the best fitness values (BFV), the worst fitness values (WFV), the mean, and the standard deviation (STDEV) of the results found over the 20 independent runs by each algorithm. The best values are typed in bold.

Furthermore, Figure 10.6 illustrates the convergence speed of these algorithms for 50 iterations. The horizontal axis of the graph is the number of iterations, and the vertical axis is the mean of function values.

In Table 10.2, the parameters that are estimated by IDVBA, DVBA, and the actual parameters are shown.

TABLE 10.1
Comparison of DVBA and IDVBA identification of a quarter-car suspension system by using 100 iterations. BFV, WFT, Mean, and STDEV" indicate the best fitness value, the worst fitness value, the mean, and standard deviation of the results found over the 20 independent runs by each algorithm.

	BFV	**WFV**	**Mean**	**STDEV**
DVBA	3.2593e-06	5.2664e-04	1.1746e-04	2.2907e-04
IDVBA	**2.3480e-09**	**1.9909e-05**	**6.8526e-06**	**9.1657e-06**

TABLE 10.2
Comparison of DVBA and IDVBA on identification of quarter-car suspension system parameters with the real parameters by using 100 iterations.

	DVBA	IDVBA	Actual Values
m_1	27.73	24.72	26.00
m_2	232.74	257.29	253.00
k_1	86187.95	89931.03	90000.00
k_2	11005.63	12220.11	12000.00
b	1367.52	1531.63	1500.00

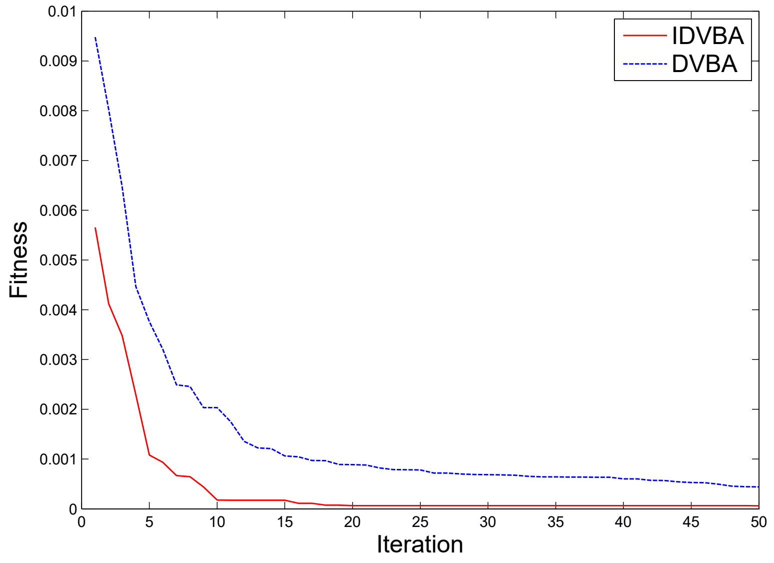

FIGURE 10.6
The comparision of convergence speed of DVBA and IDVBA.

As reported in Table 10.1, IDVBA performs better for identification of suspension parameters. From Figure 10.6, it is explicitly observed that IDVBA found effective results with the fastest convergence speed, which means that IDVBA using the improved search mechanism is more competent in tackling complex problems.

Since, we used only one impulse signal as excitation signal, we can say that the estimation is pretty close to the actual system parameters. To increase the accuracy of the estimated parameters, a series of impulses with random amplitudes, like Gaussian band-limited white noise can be used as an excitation signal to the real and the modeled system. It will take more time to compute, but give more chances for the algorithm to increase accuracy.

10.5 Conclusions

In this chapter, the original DVBA and the improved DVBA are presented. We used a quarter-car model system to show the performance of the algorithms in system identification. We can say that slightly changing the explorer bat's behaviour in DVBA, increased the convergence speed and the accuracy of the algorithm. The results show that IDVBA is superior to the original DVBA and it is a promising optimization algorithm for system identification. As future work, there might be some improvements on the exploiter bat to increase the detailed search as well.

References

1. H.Peng, R.Strathearn, A.G.Ulsoy, "A novel active suspension design technique-simulation and experimental results" in *Proc. of American Control Conference, IEEE*, vol. 1, 1997, pp. 709-713.
2. A.Alfi, and M.M.Fateh, "Parameter identification based on a modified PSO applied to suspension system" in *Journal of Software Engineering & Applications, JSEA*, 2010, vol.3, pp.221-229.
3. A.O.Topal , O.Altun, "Dynamic virtual bats algorithm (DVBA) for global numerical optimization" in *Proc. of the IEEE International Conference on Congress on Intelligent Networking and Collaborative Systems (INCoS)*, 2014, pp. 320-327 .
4. A.O.Topal, Y.E.Yildiz, M.Ozkul, "Improved dynamic virtual bats algorithm for global numerical optimization" *Proc. of the World Congress on Engineering and Computer Science.* Vol. 1, pp.462-467, 2017.
5. S.Lajqi, J.Gugler, N.Lajqi, A.Shala, R.Likaj, "Possible experimental method to determine the suspension parameters in a simplified model of a passenger car" *International Journal of Automotive Technology*, vol. 13(4), pp. 615-621, 2012.
6. C.Chandrasekar, "An optimized approach of modified bat algorithm to record deduplication." *International Journal of Computer Applications* 62.1, 2013.
7. M.Airas, "Echolocation in bats." *Proc. of Spatial Sound Perception and Reproduction.* The postgrad seminar course of HUT Acoustics Laboratory, pp. 1-25, 2003.