mekatronika – Journal of Intelligent Manufacturing & Mechatronics
VOL. 01, ISSUE 01, 45 – 57
DOI: https://doi.org/10.15282/mekatronika.v1i1.155

ORIGINAL ARTICLE

Dual level searching approach for solving multi objective optimisation problems using hybrid particle swarm optimisation and modified adaptive bats sonar algorithm

Nafrizuan Mat Yahya[1,*], Ahmad Razlan Yusoff[1], Azlyna Senawi[2], and M Osman Tokhi[3]

[1]Faculty of Manufacturing Engineering, Universiti Malaysia Pahang, 26600 Pahang, Malaysia.
[2]Faculty of Industrial Sciences and Technology, Universiti Malaysia Pahang, 26600 Pahang, Malaysia.
[3]School of Engineering, London South Bank University, London SE1 0AA, United Kingdom.

ABSTRACT – A dual level searching approach for multi objective optimisation problems using particle swarm optimisation and modified adaptive bats sonar algorithm is presented. The concept of echolocation of a colony of bats to find prey in the modified adaptive bats sonar algorithm is integrated with the established particle swarm optimisation algorithm. The proposed algorithm incorporates advantages of both particle swarm optimisation and modified adaptive bats sonar algorithm approach to handle the complexity of multi objective optimisation problems. These include swarm flight attitude and swarm searching strategy. The performance of the algorithm is verified through several multi objective optimisation benchmark test functions and problem. The acquired results show that the proposed algorithm perform well to produce a reliable Pareto front. The proposed algorithm can thus be an effective method for solving of multi objective optimisation problems.

ARTICLE HISTORY
Received: 30 August 2018
Revised: 24 December 2018
Accepted: 10 January 2019

KEYWORDS
Modified adaptive bats sonar algorithm
Particle swarm optimisation
Multi objective optimisation

Introduction

Multi objective optimisation currently is an essential practice in many bussiness, management and engineering application. Multi objective optimisation problem is multifaceted and solving the problem is to seek compromised solution based on a set of conflicting objectives [1-4]. As there will be no unique solution in a multi objective optimisation problem [5], a set of 'trade-off' solutions, referred as Pareto optimum solutions, compromising the objectives is produced [6-7].

Nowadays, evolutionary algorithms are widely used to deal with multi objective optimisation problem [8-9]. Multi objective evolutionary algorithms are inspired from imitating the successful characteristics of natural phenomena or biological species [4,6].

This paper introduces a new multi objective evolutionary algorithm by integrating the particle swarm optimisation (PSO) algorithm with a modified bats sonar algorithm (MABSA). The proposed algorithm also adopted non-Pareto technique; the weighted sum approach to solve the multi objective optimisation problem. The results from computer simulations on several multi objective optimisation benchmark test functions prove that this new hybrid algorithm can serve as a viable multi objective evolutionary algorithm option for solving multi objective optimisation problems.

The remainder of the paper is organised as follows. Section 2 discusses about the standard PSO algorithm while Section 3 describes on the MABSA approach. Section 4 deals with the hybridisation between PSO and MABSA as a new algorithm for solving multi objective optimisation problems. The computer simulation set up is discussed in Section 5, while the performance of D-PSO-MABSA algorithm on established multi objective optimisation benchmark test functions are discussed in Section 6. Section 7 is presented the results obtained from the performance of D-PSO-MABSA in solving the engineering design problem of a four bar plane truss. The conclusion are finally drawn in Section 8.

Particle Swarm Optimisation (PSO)

Particle swarm optimisation (PSO) is an evolutionary computation technique developed in 1995 which inspired from the social behaviour of a swarm of birds and fishes [10]. PSO has characteristics that are more attractive than the existing evolutionary computation. The characteristics include memory that can be maintained by any individual in the algorithm, build cooperation between the individuals and sharing of information between the individuals [10]. The algorithm has a simple

*CORRESPONDING AUTHOR | Nafrizuan Mat Yahya | ✉ nafrizuanmy@ump.edu.my

theoretical framework, is easy to code into a computer programme and can generate high quality and focused solutions in relatively shorter computation times [11], than other metaheuristic methods.

In PSO, all particles are treated as valueless particles of g-dimensional search space [10]. Each particle will record its current coordinate in the problem space associated with its personal best solution, *pbest*. Meanwhile, the overall best solution and the location obtained so far by any particle in the swarm is labelled as *gbest*.

The concept of PSO involves changing the velocity of every particle toward the *pbest* and *gbest*. For instance, the position of *j*th particle is represented as:

$$x_j = x_{j,1}, x_{j,2}, \dots, x_{j,g} \quad (1)$$

where g is total dimension of the space. The *j*th best previous position represented as:

$$pbest_j = pbest_{j,1}, pbest_{j,2}, \dots, pbest_{j,g} \quad (2)$$

where the best $pbest_j$ among all particles in the swarm is denoted as *gbest*. The velocity of *j*th particle is represented as:

$$v_j = v_{j,1}, v_{j,2}, \dots, v_{j,g} \quad (3)$$

The new velocity and position of each particle at each iteration can be calculated as:

$$v_{j,g}^{(t+1)} = w.v_{j,g}^{(t)} + c_1 * rand(\ ) * \left(pbest_{j,g} - x_{j,g}^{(t)}\right) + c_2 * rand(\ ) * \left(gbest_g - x_{j,g}^{(t)}\right)$$

and

$$x_{j,g}^{(t+1)} = x_{j,g}^{(t)} + x_{j,g}^{(t+1)}, j = 1,2, \dots, n \text{ and } g = 1,2, \dots, m$$

where

$$-V_{max} \le v_{j,g}^{(t)} \le V_{max}\ , \quad (4)$$

n	number of particles in a group
m	number of members in a particle
t	pointer of iterations (generations)
$v_{j,g}^{(t)}$	velocity of particle j at iteration t
V_{max}	maximum velocity
c_1, c_2	acceleration constant
rand ( )	random number between 0 and 1
$pbest_j$	*pbest* of particle *j*
gbest	*gbest* of the swarm

Here, the parameter maximum velocity (V_{max}) determines the resolution (or fineness) in the search space between the current velocity and target velocity [12]. V_{max} is applied to damping the particles velocity to avoid the swarm system explode when the particles' searching process increase with time [13]. So each particle's velocity in every dimension is tied to the V_{max} value [12]. V_{max} value is set at the start of the iteration process and remains constant till iterations end [13].

Acceleration constant (c_1) and (c_2) are important in determining the motion trajectory of particles [13] and controlling the influence of stochastic components of social and cognitive on overall particle's velocity [14]. [14] divided the constant c_1 as self-confidence factor to represent confidence level in every particle while c_2 is a swarm-confidence factor that represents the confidence level of particles to their neighbourhood. [10,14] had set the value of c_1 and c_2 to 2.0 so that particles will attract to the $pbest$ and $gbest$ position equally. By setting to this value also enables smooth particles trajectory and permits particles to explore far from the target location before being tugged back to the appropriate region.

In general, inertia weight (w) is set in iteration decreasing mode as follows:

$$w = \frac{w_{max} - w_{min}}{iter_{max}} \times iter \quad (5)$$

Here, $iter$ is current iteration while $iter_{max}$ is total number of iteration used. A suitable value of w_{max} is 0.9 while w_{min} is 0.4 [12-13]. This w as suggested by [15] is a mechanism to control the exploration and exploitation abilities in the swarm. The w value will drive the momentum of particles on current velocity influencing a new velocity [14]. So, this parameter will balance between local and global search [12], besides encourage the algorithm to shift from exploration mode to exploitation mode in order to find optimum solution [13]. Figure 1 shows the PSO pseudo code..

Modified Adaptive Bats Sonar Algorithm (MABSA)

A colony of bats is able to construct good communication and sharing information between each other about roast site or foraging area [16]. Echolocation of bats is the ability of bat to produce sound with echo beyond the frequency range of human hearing and used for general orientation and finding prey [17]. Thus, the bats echolocation-based algorithm namely adaptive bats sonar algorithm (ABSA) is developed by [18] was inspired from the echolocation process of bats to search and capture prey. Then, the modified adaptive bats sonar algorithm (MABSA) is

1: Objective function $F(x), x = (x_1, \dots, x_d)^T$
2: Initialise: $number\ of\ iteration\ (MaxIter), number\ of\ particles\ (n), dimension\ (d)$ and $maximum\ velocity\ (V_{max})$
3: **for** $s \leftarrow 1$ **to** n **do**
4: Generate random $position\ (x_d)$ and $velocity\ (v_d)$
5: Evaluate the $fitness\ (F(x))$ for each particle x_d and v_d
6: **end for**
7: Set the $F(x)$ as $pbest$ for each particle
8: Set the $min\ F(x)$ as $gbest$ for the swarm
9: **while** $t \leq MaxIter$ **do**
10: Define the $inertia\ weight\ (w)$ (**Equation 5**)
11: Generate new v_d and x_d of each particles (**Equation 4**)
12: Evaluate the $F(x)$ for each particle v_d and x_d
13: **if** $F(x) \leq pbest$ **then**
14: Assign $F(x)$ as new $pbest$ and its position as new $pbest$ position
15: **else**
16: Remain the previous $pbest$ and its position
17: **end if**
18: **if** $min\ (F(x)) \leq gbest$ **then**
19: Assign $min\ (F(x))$ as new $gbest$ and its position as new $gbest$ position
20: **else**
21: Remain the previous $gbest$ and its position
22: **end if**
23: **end while**
24: Declare the $gbest$ as optimum fitness evaluated and its position as optimum value(s)

Figure 1. Pseudo code of Particle Swarm Optimisation algorithm.

introduced [19] based upon some modification on the ABSA.

The *number of iterations* (*MaxIter*) or generations used in MABSA are kept at 100. This quantity is favourably enough for the bats to explore fully the *d* numbers of search space *dimension* (*Dim*) for the best prey or *global best fitness* (F_{GB}). The chosen value is in line with maximum *MaxIter* used in the PSO algorithm when the algorithm was first introduced by [10].

Inspired by a description of the number of bats in a colony by biologists [20], the *number of bats* (*Bats*) or population in MABSA was selected in the range (700, 1000). By having a larger number of bats, a discovery of the F_{GB} value becomes more resourceful such that there will be a pool of solutions (prey) that can be evaluated to obtain the best ones.

The *beam length* (*L*) is sets as:

$$L = \text{Rand} \times \left(\frac{SS_{size}}{10\% \times Bats}\right) \tag{6}$$

where the solution range (SS_{size}) is the value between the *upper search space* (SS_{Max}) limit and the *lower search space* (SS_{Min}) limit as:

$$SS_{size} = SS_{Max} - SS_{Min} \tag{7}$$

Every *dimension* (*Dim*) has its specific or known as *Dim* constraints. The solution range is divided into micron scale, such as 10% of the overall population of bats in the search space. The percentage is marked as possible search space size of each bat to emit sound without colliding with one another. The random value of *L* is offered to make real variation of beam lengths of each *number of beams* (*NBeam*) at every *Dim* (but stay within the *Dim* constraints) at every iteration. This fixation pushes every bat at each dimension to search in larger perimeter each time with the opportunity to diversify the search tactic during iterations and thus may find the global best solution that may be near to them. Moreover, a *momentum term* (μ) is used in MABSA as a tool to control the risk of convergence to a local optimum, as well as an extra chance to search for optimum solution in a wider range within the SS_{size}. The μ is defined as:

$$L_{new} = L_{old}(1 \pm \mu) \tag{8}$$

where $0 < \mu < 1$.

Altringham et al. [16] have reported that the pulse emission rate grows bit by bit up to 200 per second as the bat keeps updating the location of the object until it catches the prey. This phenomenon is incorporated into the MABSA approach as *beam number increment* (*BNI*). The *BNI* is defined in terms of the *maximum number of beams* ($NBeam_{Max}$) and *minimum number of beams* ($NBeam_{Min}$) as:

$$BNI = \left(\frac{NBeam_{Max} - NBeam_{Min}}{MaxIter}\right) \times iter \quad (9)$$

where $NBeam_{Max} = 200$ and $NBeam_{Min} = 20$.

Thus, $NBeam$ is defined as:

$$NBeam = NBeam_{Min} + BNI \quad (10)$$

The *BNI* method mimics the original pulse rate emitted by the bat as it increases gradually toward the end of the search. As a result, *BNI* will provide a balance between global exploration and local exploitation thus requiring less iteration on average to find a sufficiently optimum solution.

Each *NBeam* with *L* is emitted from the *starting position* (pos_{SP}) with specific angle location. Figure 2 shows angle locations for single batch of beam transmitted by a bat. The MABSA limits the first beam to have θ_m not more than 45° from horizontal axis and the (θ_i) is set as follows:

$$\theta_i = \left(\frac{2\pi - \theta_m}{NBeam}\right) \quad (11)$$

where $\theta_m = rand \leq 0.7854$. By setting θ_i as such, the beams will sweep at random 360° around the bats through iterations such a way that the searching process will neither be too aggressive (overlay a circle) nor to slow (underlay a circle).

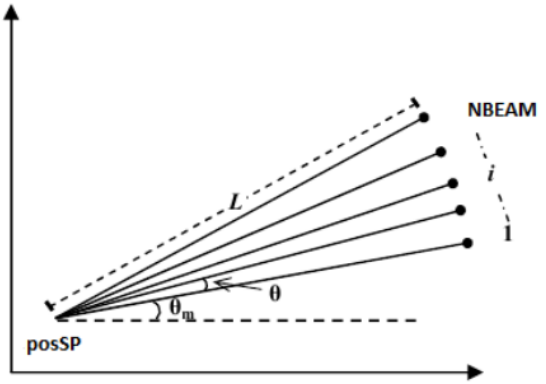

Figure 2. Single batch of beams transmitted by a bat [19].

The level of best fitness solution found in the algorithm has been raised up to four stages in the developed MABSA. The main target is mentioned before; F_{GB}, while another three levels are *starting position fitness* (F_{SP}), *local best fitness* (F_{LB}) and *regional best fitness* (F_{RB}). Meanwhile, there are three levels of best solution found by the algorithm in PSO [14]. The levels are *personal best* (*pb*) which is the best solution for every particle, *local best* (*lb*) which is the neighbourhoods best solution and *global best* (*gb*) is the global best solution of among the *pb*. These three levels are similar to F_{LB}, F_{RB} and F_{GB} of MABSA respectively.

In PSO, the *lb* improve the overall performance of algorithm where the individual *lb* influenced the performance of immediate neighbours [21-22]. Ultimately, the neighbourhoods preserve swarm diversity by hindering the flow of information through the network [23]. This move prevents the particles reach to the global best particle immediately or trap in a local optimum but allows them to explore larger search space [22-23]. This beneficial element inspired to an existence of F_{RB} which functioning as neighbourhoods best solution-MABSA version. As an addition, F_{RB} also forms the main link between F_{LB} and F_{GB} values. So F_{RB} act as leverage instrument to balance finely between exploration (diversification) and exploitation (intensification) processes of the algorithm and so to help the algorithm escape from premature convergence.

The initialisation of these levels will help the MABSA refine the search for the solution by a colony of bats in the search space in each step and leave out bad solutions immediately. As a result, the algorithm takes less time to converge to the optimum solution. In point of fact, [21] mentioned that many type of research shown that communication between individuals within a group is important where the performance of the group in overall is affected by the structure of the social network. Besides, [22] argued that the distribution of information via distant acquaintances be crucial, such that it posses information that a colleague might not. In conjunction to that, the four levels of the best solution created in MABSA ideally match with the information transfer mechanisms practised by a colony of bats as explored by [16]. These are intentional signalling match to F_{SP}, local enhancement match to F_{LB}, social facilitation match to F_{RB} and imitative learning match to F_{GB}.

The *end point position* (pos_i) for each transmitted beam in MABSA is calculated as:

$$pos_i = \alpha \times pos_{SP} + \beta \times L(cos[\theta_m + (i-1)\theta])^{\omega}, \quad i = 1, \ldots, N \quad (12)$$

where pos_{SP} is beam's *starting position*. In the above equation, there are two random variables and one constant. The first random variable is called *position adaptability factor* (α) where $0 < \alpha < 1$. This factor is included to make sure that every bat is able to adapt to the new pos_{SP} faster as derived from the previous pos_{SP}, *local best position* (pos_{LB}) , *regional best position* (pos_{RB}) and *global best position* (pos_{GB}). This factor has the same characteristic as random walk method. The second random variable is *collision avoidance factor* (β) where $0 < \beta < 1$. The factor is essential to avoid the beams from overlapping or incidentally colliding with other bats' beam as every bat has produced a number of beams from new pos_{SP} simultaneously.

The only constant in this equation is *beam-tuning constant* (ω) which is equal to 2. This constant also can be considered as acceleration constant. The function of this constant is to strengthen β such that ω will divert the angle of transmited beam to the new angle in the designated search space. The value 2 is selected because it will give a good balance. If a very high value is selected, it will destroy the influence of the beam angle such that the orientation of new bat position will be catastrophic. A smaller value, on the other hand, will not make any significant change to the angle of transmitted beam.

The MABSA is also equipped with bounce back strategy. This will confirm that every pos_i achieved by each bat during the iterations is worth considering as possible optimum pos_{GB} for the algorithm. When each beam is transmitted from every bat, it will be verified to ensure that the pos_i of the transmitted beam does not fall beyond SS_{Max} or below SS_{Min}. If the pos_i reaches outside SS_{Size}, the transmitted beam will be diverted automatically to new location inside the labelled SS_{Size} using one of the following equations:

$$pos_i = \alpha \times SS_{Max} - \tau, \; i = 1, \dots, N \quad (13a)$$

$$pos_i = \alpha \times SS_{Max} + \tau, \; i = 1, \dots, N \quad (13b)$$

These equations contain *bounce back repositioning factor* (τ) where the value is $0 < \tau < 1$. This factor is to help the bats to relocate a beam transmission to a new beams' end point from the maximum or minimum search space. This factor will avoid overwriting other bats' beam end points. The *bounce back repositioning factor* is the fastest contigency action of bats to swing to newly transmitted beam's end point after hitting the designated search space boundaries. This strategy helped to reduce much time to spend to consider the previous factors (which are: *position adaptability factor*, *collision avoidance factor* or *beam-tuning constant*) as normal bats do.

The reciprocal altruism characteristic has further been incorporated into MABSA to strengthen the procedure of colony searching for the best solution. This reciprocal altruism behaviour widely runs through a colony of bats as reported by many researchers in bats ecology [16, 24-25]. By inserting this behaviour into the algorithm, a member of the colony will disseminate and sharing the location of the best fitness found so far to other bats. As a result, all bats will fly to the best prey ever found when the search process comes to an end. The adoption of this real prey hunting behaviour of the colony of bats into the algorithm is symbolised by two levels of arithmetic mean.

For every bat, the arithmetic mean evaluate the balancing point between pos_{SP}, pos_{LB} and pos_{RB} in current iteration (t) with pos_{GB} of the latest F_{GB} to be appoint as a new pos_{SP} for next iteration (t+1). The first level of arithmetic mean involves measuring of central tendency between pos_{SP}, pos_{LB} and pos_{RB} of each bat for current iteration only. Next, the second level of arithmetic mean find the central tendency between the position value resulted from the first level of arithmetic mean and pos_{GB}. As results, during new iteration, every bat will start transmit a set of new beams from the pos_{SP} which has been specified after considering (or sharing) the balancing point of the positions of all four level of best fitness solution; F_{SP}, F_{LB}, F_{RB} and F_{GB}. The two levels of arithmetic mean is expressed as follows:

$$pos_{SP}(t+1) = \frac{\frac{pos_{SP}(t) + pos_{LB}(t) + pos_{RB}(t)}{3}}{2} + pos_{GB} \quad (14)$$

The overall steps of the MABSA are presented as the pseudo code in Figure 3.

A Dual-Particle Swarm Optimisation-Modified Adaptive Bats Sonar Algorithm (D-PSO-MABSA)

A dual level search strategy is adopted through integration of the two algorithm for getting the Pareto optimum set of the problem considered as shown in Figure 4. This hybrid algorithm is named dual-particle swarm optimisation-modified adaptive bats sonar algoritm (D-PSO-MABSA). The D-PSO-MABSA algorithm uses the weighted sum approach to combine all objectives into a single objective. The weights are generated randomly from a uniform distribution. By doing so, the Pareto optimum set can be acquired efficiently as well as Pareto front would be estimated appropriately.

Here, the dual level searching process means that at every time to obtain Pareto optimum point, there are always two levels of search. During the first level, PSO acts as a global search agent of the algorithm with its embedded global (exploration) and local (exploitation) search components. As an explorer, the PSO is first to discover and mark a potential location of a solution in the compound of designated search space. The PSO will run according to its standard algorithmic procedures such as locating new velocity and position to obtain the *pbest* and *gbest*.

In the second level search process, the optimum solutions obtained by the PSO are used to initialise the starting positions of the population in the MABSA. The MABSA considered as a local search agent of the developed algorithm also has its global search (diversification component) and local search (intensification component). Here, MABSA works as a follower to find the optimum solutions starting from the prospective location previously marked by the PSO within the designated search space.

1: Objective function $F(x), x = (x_1, \ldots, x_d)^T$
2: Initialise: $Bats$, $MaxIter$, Dim, SS_{Size}, $NBeam_{Max}$ and $NBeam_{Min}$
3: **for** $n \leftarrow 1$ **to** $Bats$ **do**
4: **for** $d \leftarrow 1$ **to** Dim **do**
5: Generate random pos_{SP}
6: Evaluate F_{SP} value for $F(pos_{SP})$
7: **end for**
8: **end for**
9: Assign the most optimum value as F_{GB} and its position as pos_{GB}
10: **while** $t \leq MaxIter$ **do**
11: Define $NBeam$ to transmit by using BNI (**Equation 9** and **Equation 10**)
12: **for** $n \leftarrow 1$ **to** $Bats$ **do**
13: **for** $N \leftarrow 1$ **to** $NBeam$ **do**
14: **for** $d \leftarrow 1$ **to** Dim **do**
15: Set L and limit μ (**Equation 6** and **Equation 8**)
16: **end for**
17: **end for**
18: Generate random θ_m and θ (**Equation 11**)
19: Transmit $NBeam$ starting from pos_{SP}
20: **for** $N \leftarrow 1$ **to** $NBeam$ **do**
21: **for** $d \leftarrow 1$ **to** Dim **do**
22: Determine pos_i for each transmitted beam (**Equation 12**)
23: Verify pos_i for each transmitted beam within SS_{Size}
24: **if** $pos_i \geq SS_{Max}$ **then**
25: Update pos_i (**Equation 13a**)
26: **end if**
27: **if** $pos_i \leq SS_{Min}$ **then**
28: Update pos_i (**Equation 13b**)
29: **end if**
30: **end for**
31: Evaluate F_i value for $F(pos_i)$
32: Assign the optimum value of F_i as F_{LB} and its position as pos_{LB}
33: **if** $F_{LB} \leq F_{SP}$ **then**
34: Assign F_{LB} as F_{RB} and pos_{LB} as pos_{RB}
35: **else**
36: Assign F_{SP} as F_{RB} and pos_{SP} as pos_{RB}
37: **end if**
38: **end for**
39: **end for**
40: Select the optimum value among F_{RB} as current F_{GB} and its pos_{RB} as current pos_{GB}
41: **if** current $F_{GB} \leq$ previous F_{GB} **then**
42: Update current F_{GB} as new F_{GB} and current pos_{GB} as new pos_{GB}
43: **else**
44: Retain previous F_{GB} and pos_{GB}
45: **end if**
46: **for** $n \leftarrow 1$ **to** $Bats$ **do**
47: Determine new pos_{SP} using (**Equation 14**)
48: Evaluate new F_{SP} value for $f(x)$
49: **end for**
50: **end while**
51: Declare F_{GB} as optimum fitness evaluated and pos_{GB} as its optimum value(s)

Figure 3. Pseudo code of Modified Adaptive Bats Sonar Algorithm.

Objective function $F(x) = [F_1(x), F_2(x), \ldots, F_N(x)]^T$, $x = (x_1, \ldots, x_d)^T$
Initialise: $Bats$, $MaxIter$, Dim, SS_{Size}, $NBeam_{Max}$, $NBeam_{Min}$, n, V_{max} and d
for $j \leftarrow 1$ **to** N (points on Pareto set) **do**
  Generate K weights ($w_k \geq 0$)
  for $s \leftarrow 1$ **to** n **do**
    Generate random *position* (x_d) and *velocity* (x_d)
    Evaluate the $fitness$ ($F(x)$) for each particle x_d and v_d
  end for
  Set the $F(x)$ as $pbest$ for each particle
  Set the min $F(x)$ as $gbest$ for the swarm
  while $t \leq MaxIter$ **do**
    Define the *inertia weight* (w) (**Equation 5**)
    Generate new v_d and x_d of each particles (**Equation 4**)
    Evaluate the $F(x)$ for each particle v_d and x_d
    if $F(x) \leq pbest$ **then**
      Assign $F(x)$ as new $pbest$ and its position as new $pbest$ position
    else
      Remain the previous $pbest$ and its position
    end if
    if min ($F(x)$) $\leq gbest$ **then**
      Assign min ($F(x)$) as new $gbest$ and its position as new $gbest$ position
    else
      Remain the previous $gbest$ and its position
    end if
  end while
  Assign $pbest$ as F_{SP}; its position as pos_{SP} and $gbest$ as F_{GB}; its position as pos_{GB}
  while $t \leq MaxIter$ **do**
    Define $NBeam$ to transmit by using BNI (**Equation 9** and **Equation 10**)
    for $n \leftarrow 1$ **to** $Bats$ **do**
    for $N \leftarrow 1$ **to** $NBeam$ **do**
    for $d \leftarrow 1$ **to** Dim **do**
      Set L and limit μ (**Equation 6** and **Equation 8**)
    end for
    end for
    Generate random θ_m and θ (**Equation 11**)
    Transmit $NBeam$ starting from pos_{SP}
    for $N \leftarrow 1$ **to** $NBeam$ **do**
    for $d \leftarrow 1$ **to** Dim **do**
      Determine pos_i for each transmitted beam (**Equation 12**)
      Verify pos_i for each transmitted beam within SS_{Size}
      if $pos_i \geq SS_{Max}$ **then**
        Update pos_i (**Equation 13a**)
      end if
      if $pos_i \leq SS_{Min}$ **then**
        Update pos_i (**Equation 13b**)
      end if
      end for
      Evaluate F_i value for $F(pos_i)$
      Assign the optimum value of F_i as F_{LB} and its position as pos_{LB}
      if $F_{LB} \leq F_{SP}$ **then**
        Assign F_{LB} as F_{RB} and pos_{LB} as pos_{RB}
      else
        Assign F_{SP} as F_{RB} and pos_{SP} as pos_{RB}
      end if
    end for
    end for
    Select the optimum value among F_{RB} as current F_{GB} and its pos_{RB} as current pos_{GB}
    if current $F_{GB} \leq$ previous F_{GB} **then**
      Update current F_{GB} as new F_{GB} and current pos_{GB} as new pos_{GB}
    else
      Retain previous F_{GB} and pos_{GB}
    end if
    for $n \leftarrow 1$ **to** $Bats$ **do**
      Determine new pos_{SP} using (**Equation 14**)
      Evaluate new F_{SP} value for $f(x)$
    end for
  end while
end for
Declare F_{GB} as optimum fitness evaluated and pos_{GB} as its optimum value(s)

Figure 4. Pseudo code of Dual-Particle Swarm Optimization-Modified Adaptive Bats Sonar Algorithm.

The MABSA first set the number of individuals in the population randomly between 700-1000 bats at every iteration. The value has been inspired from the real population of bats in a colony. Then, PSO will follow suit although the standard PSO algorithm has 100-200 number of particles only. The equivalence of population size between PSO and MABSA is crucial to a smooth phase transition of the final solution found by the PSO and inherited by MABSA during the algorithm runs. Thus, the population size criterion will act as a look-alike handshaking or acknowledgement procedure of the dual level search process.

MABSA proceeds through its normal search procedure in transmitting the sound beams by bats into the dedicated search space to get pos_{LB} and F_{LB} and finally pos_{RB} and F_{RB}. This operation runs until the specified maximum iterations. As in the original MABSA, the pos_{GB} with its F_{GB} resulting from the overall iterations will be declared as the best optimum solution to the problem studied. The optimum solution thus obtained is considered as one Pareto optimum point. The developed algorithm will repeatedly run until the total number of Pareto optimum points are obtained to get a complete set of Pareto.

There are two factors are considered to set PSO as global search agent and MABSA as local search agent. These factors are inspired by the real behaviour of both swarm groups. As noted, PSO is represented based on a swarm of birds flying in search of food while MABSA is based on a colony of bats flying for capturing preys. The factors are swarm flight attitude and swarm searchings strategy.

The first factor is the flight attitude of the swarm. A good global search agent has a capability of viewing and monitoring the search space from the highest position. The broad perspective from the higher ground makes it easier for the agent to mark possible areas within the search space containing potential solutions that would be a true exploration process in swarm intelligence. A local search agent is, on the other hand, needed to verify the location of potential solutions found by a global search agent. To be a good local search agent, the agent must have the ability to observe and inspect the solutions from stone's throw view. This exploitation process should be put after the exploration process so that the solutions developed by a global agent could be validated properly by the local search agent. In reality, the bar-headed goose that is a family of birds can fly to the highest up to 6437 m [26]. Meanwhile, according to research by [27], bats only fly less than 10 m above the sea level. These facts have enthused PSO to be defined as global search agent while MABSA as local search agent.

Looking at the proposed swarm searching strategy, there is a distinct line between the searching strategy of PSO and MABSA. In the PSO, the algorithm utilises the velocity and positioning of particles to evaluate the obtained solution whereas MABSA depends on the transmission and positioning of sound beams. In the real world, birds can fly with a velocity between 20 to 30 mph [28]. With this fast speed, the searching process of PSO may miss locations of good solutions on their way towards other possible target solutions. Moreover, the velocity of particles in PSO itself makes the particle or bird to move in a single line thus not covering a broad search area at one time. The sound beams transmitted in MABSA are multi line able to disperse and sweep a large search envelope. Thus, the issue of missing good solutions in a smaller area of designated search space does not arise. Hence, the sequence of searching process as applied in any good swarm intelligence method is followed here where coarse searching (diversification) is done first by PSO followed by fine searching (intensification) by MABSA. In this context, labelling PSO as global search agent and MABSA as local search agent in the proposed hybrid algorithm D-PSO-MABSA is reasonable choice given their characteristics.

Computer Simulation Set Up

The proposed D-PSO-MABSA algorithm was coded using MATLAB software version *Matlab®R2013a*. Computer simulations of the D-PSO-MABSA algorithm on several multi objective optimisation benchmark test functions and an engineering design problem were performed on *Intel®Core™ i5* processor of 2400 CPU @ 3.10*GHz* with 4.00*GB RAM*.

The computer simulation are divided into two parts. The objective of the first part is to show the superior performance of the D-PSO-MABSA algorithm with established multi objective optimisation benchmark test functions. The two test functions namely, Zitzler-Deb-Thiele's (ZDT) 1 and Schaffer function 1 are involved.

The second part is to test the performance of D-PSO-MABSA algorithm on an engineering design problem. A four bar plane truss problem is selected as a platform for the proposed algorithm. The problem is run several different suite of Pareto points.

Performance of D-PSO-MABSA Algorithm on Established Multi Objective Optimisation Benchmark Test Functions

Zitzler-Deb-Thiele's function (ZDT 1)

This function was among the well-known benchmark test function used to evaluate an algorithm developed for solving the multi objective optimisation problem. The function constitutes an unconstrained problem and has a convex Pareto front [29]. The function is defined as:

Minimise

$$F_1(x) = x_1$$

and

$$F_2(x) = \left(1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i\right)\left(1 - \sqrt{\frac{F_1}{g}}\right) \quad (15)$$

where

$0 \leq x_i \leq 1$
$1 \leq i \leq 20$

Table 1 shown 15 Pareto optimum point tabulated in terms of F_1 and F_2. The values of w_1 and w_2 are recorded to show linear increasing and decreasing in weighted sum values respectively. The search for each single Pareto optimum point was conducted over 100 iterations of D-PSO-MABSA algorithm. Figure 5 shows the Pareto optimum set of ZDT 1 function. It is noted that the proposed algorithm achieved a set of Pareto optimum points each comprising a non-dominated solution. Moreover, the set of non-dominated solutions successfully formed convex Pareto front as expected witht the result obtained by [29].

Table 1. ZDT 1 function test results.

w_1	w_2	F_1	F_2
0.0667	0.9333	1.0000	0.0000
0.1333	0.8667	0.9999	0.0000
0.2000	0.8000	0.9999	0.0000
0.2667	0.7333	1.0000	0.0000
0.3333	0.6667	1.0000	0.0000
0.4000	0.6000	0.5625	0.2500
0.4667	0.5333	0.3265	0.4286
0.5333	0.4667	0.1914	0.5625
0.6000	0.4000	0.1110	0.6668
0.6667	0.3333	0.0625	0.7500
0.7333	0.2667	0.0330	0.8183
0.8000	0.2000	0.0156	0.8750
0.8667	0.1333	0.0059	0.9229
0.9333	0.0667	0.0012	0.9652
1.0000	0.0000	0.0003	0.9838

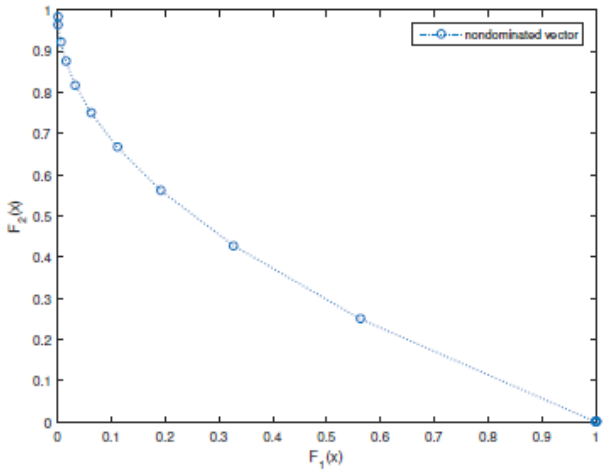

Figure 5. Pareto front for ZDT 1 function.

Schaffer function 1

This function has been used by [30] to evaluate their algorithm; Pareto archived evaluation strategy (PAES) in solving the multi objective optimisation problem. The function constitutes an unconstrained problem, has a convex Pareto front and is defined as:

Minimise

$$F_1(x) = x^2$$

and

$$F_2(x) = (x-2)^2 \quad (16)$$

where

$-10 \leq x_i \leq 10$
$1 \leq i \leq 20$

In this case study, the D-PSO-MABSA is applied to find 30 Pareto optimum points. Table 2 shows the results of F_1 and F_2 after using the values of w_1 and w_2 accordingly. The developed algorithm was run over 100 iterations for the search of each Pareto optimum point.

Table 2. Schaffer function 1 test results.

w_1	w_2	F_1	F_2
0.0333	0.9667	3.7357	0.0045
0.0667	0.9333	3.4837	0.0178
0.1000	0.9000	3.2401	0.0400
0.1333	0.8667	3.0054	0.0710
0.1667	0.8333	2.7762	0.1114
0.2000	0.8000	2.5624	0.1594
0.2333	0.7667	2.3514	0.2177
0.2667	0.7333	2.1495	0.2850
0.3000	0.7000	1.9604	0.3598
0.3333	0.6667	1.7755	0.4456
0.3667	0.6333	1.6085	0.5355
0.4000	0.6000	1.4365	0.6423
0.4333	0.5667	1.2856	0.7502
0.4667	0.5333	1.1383	0.8706
0.5000	0.5000	1.0000	1.0000
0.5333	0.4667	0.8709	1.1380
0.5667	0.4333	0.7511	1.2845
0.6000	0.4000	0.6402	1.4397
0.6333	0.3667	0.5361	1.6074
0.6667	0.3333	0.4454	1.7759
0.7000	0.3000	0.3592	1.9618
0.7333	0.2667	0.2847	2.1505
0.7667	0.2333	0.2154	2.3589
0.8000	0.2000	0.1603	2.5588
0.8333	0.1667	0.1130	2.7685
0.8667	0.1333	0.0712	3.0042
0.9000	0.1000	0.0407	3.2338
0.9333	0.0067	0.0183	3.4771
0.9667	0.0333	0.0046	3.7341
1.0000	0.0000	0.0000	3.9985

As noted in Figure 6, the developed algorithm performed well with the Scheffer function 1; the Pareto optimum points obtained were non-dominated solutions and formed a smooth Pareto front. The results thus obtained match those reported by [29] particularly when considering the values of both objective functions F_1 and F_2 as shown in Figure 7.

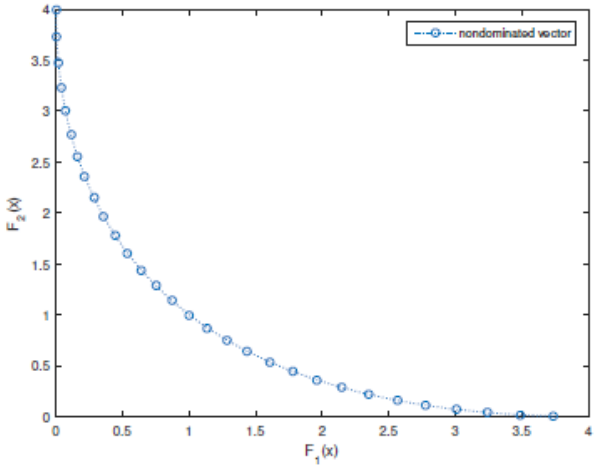

Figure 6. Pareto front for Schaffer function 1.

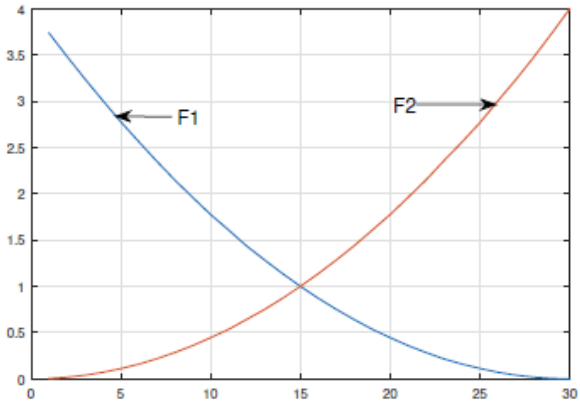

Figure 7. Plot of separated F_1 and F_2 of Schaffer function 1.

Performance of D-PSO-MABSA in Solving the Engineering Design Problem of a Four Bar Plane Truss

This multi objective engineering design problem was considered by [31] after developed by Stadler and Dauer in 1992. The problem is to design a four bar plane truss as shown in Figure 8.

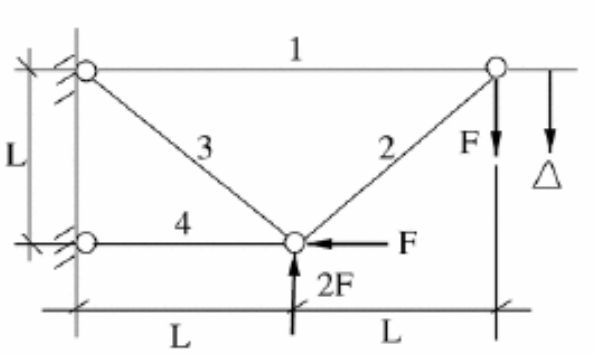

Figure 8. A four bar plane truss [31].

The design has two objectives, namely to minimise the volume of the truss (F_1) and at the same time to minimise its joint displacement, $\Delta(F_2)$. This can be expressed as:

Minimise

$$F_1(x) = L\left(2x_1 + \sqrt{2}x_2 + \sqrt{x_3} + x_4\right)$$

and

$$F_2(x) = \frac{FL}{E}\left(\frac{2}{x_1} + \frac{2\sqrt{2}}{x_2} - \frac{2\sqrt{2}}{x_3} + \frac{2}{x_4}\right)$$

subject to

$$\left(\frac{F}{\sigma}\right) \le x_1 \le 3\left(\frac{F}{\sigma}\right)$$

$$\sqrt{2}\left(\frac{F}{\sigma}\right) \le x_2 \le 3\left(\frac{F}{\sigma}\right)$$

$$\sqrt{2}\left(\frac{F}{\sigma}\right) \le x_3 \le 3\left(\frac{F}{\sigma}\right) \qquad (17)$$

$$\left(\frac{F}{\sigma}\right) \le x_4 \le 3\left(\frac{F}{\sigma}\right)$$

where

$$F = 10kN$$
$$E = 2 \times \frac{10^5 kN}{cm^2}$$
$$L = 200cm$$
$$\sigma = \frac{10kN}{cm^2}$$

It is expected that the non-dominated solutions forming the Pareto front will be as shown in Figure 9.

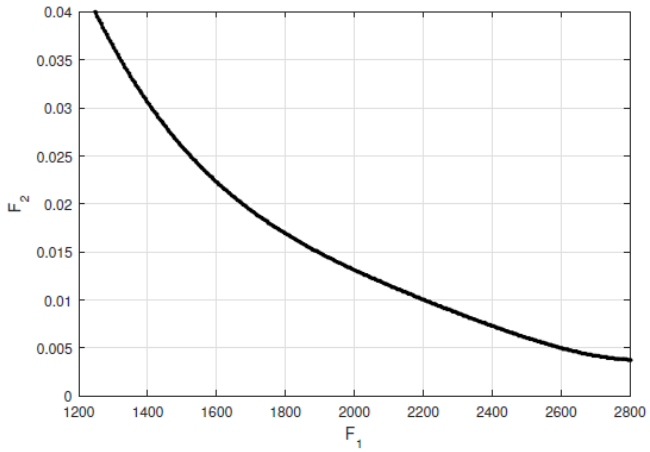

Figure 9. The true (or global) Pareto front of a four bar plane truss problem [31].

To show the ability of the developed D-PSO-MABSA algorithm to find the trade-off solutions of the problem, five dissimilar number of Pareto optimum sets were used. The sets adopted were 40, 100, 500, 1000 and 4000. Figure 10a to 10e shows the results for the different number of Pareto optimum sets respectively.

Referring to the Figure 10a, when a set of 40 Pareto points is used, there were six non-dominated solutions produced by the algorithm. These six points were non-dominated solutions that formed a Pareto front as a basis to relate to the two objectives studied. With the number of Pareto points increased to 100, as shown in Figure 10b, there were seven non-dominated solutions forming the Pareto front approximately similar to that reported by [31].

After the number of Pareto points had been increased to 500 and 1000 as in Figure 10c and Figure 10d respectively, both cases resulted in a few of non-dominated vectors besides the huge amount of dominated vectors. Nevertheless, these small groups of non-dominated vectors successfully resulted in a Pareto front that connected the true relationship between both objectives to minimise the volume and minimise joint displacement of the truss.

However, when 4000 Pareto points were considered as shown in Figure 10e, the solutions concentrated more toward the centre of the designated search space. Here also, the non-dominated solutions appear to be significantly clearer. These non-dominated solutions formed a Pareto front similar to that reported by [31]. Indeed, the value of F_1 here was smaller as compared to the reference figure while the value of F_2 remained similar.

To conclude, the D-PSO-MABSA algorithm performed well to optimise the design of a four bar plane truss. The performance was proven by the ability of the developed algorithm to result in a Pareto front from non-dominated solutions with any number of Pareto optimum solution considered. These Pareto fronts provided good compromise solutions of minimising two different objectives named the volume and the joint displacement of the truss.

Conclusion

This paper has introduced a hybridisation of particle swarm optimisation with a modified adaptive bats sonar algorithm to solve multi objective optimisation problems with weighted sum method as an approach. A dual level searching for multi objective optimisation problem using PSO and MABSA has been proposed. The proposed approach includes two factors to justify the relevance of this hybridisation strategy. The factors are swarm flight attitude and swarm searching strategy.

The proficiency of the developed algorithm to solve the multi objective optimisation problem has been examined through two different sets of computer simulation tests. The first test was about the performance of the algorithm on established multi objective optimisation benchmark test functions. The second test was to show the capability of the algorithm to solve an engineering design problem which is a four bar plane truss. The computer simulation results have proved the ability of the D-PSO-MABSA algorithm to solve a multi objective optimisation problem. The application of the proposed algorithm to solve practical multi objective optimisation problem in engineering context will be considered in future works.

References

[1] J. Castro-Gutierrez, D. Landa-Silva, J. M. Pérez, "Improved dynamic lexicographic ordering for multiobjective optimisation," in *Parallel Problem Solving from Nature, PPSN XI*, Springer, 2010, pp. 31–40.

[2] D. Cvetkovic and I. C. Parmee, "Evolutionary design and multi–objective optimisation," in *Sixth European Congress on Intelligent Technique and Soft Computing*, Aachen, Germany, 1998, pp.397–401.

[3] I. P Stanimirovic, "Compendious lexicographic method for multi-objective optimization," *Series Mathematics Informatics*, vol. 27, pp. 55–66, 2012.

[4] X.-S.Yang, "Bat algorithm for multi-objective optimisation". *International Journal of Bio-Inspired Computation*, vol. 3 (5), pp. 267–274, 2011.

[5] P. Ngatchou, A. Zarei, M. A. El-Sharkawi, "Pareto multi objective optimization," in *13th International Conference on Intelligent Systems Application to Power Systems*, Virginia, USA, 2005, pp.84–91.

[6] C. A. C. Coello, "Evolutionary multi-objective optimization: a historical view of the field," *IEEE Computational Intelligence Magazine*, vol. 1 (1), pp. 28–36, 2006.

[7] A. Zhou, B. –Y. Qu, H. Li, S. –Z. Zhao, P. N. Suganthan, Q. Zhang, "Multiobjective evolutionary algorithms: A survey of the state of the art," *Swarm and Evolutionary Computation*, vol. 1(1), pp. 32–49, 2011.

[8] D. F. Jones, S. K. Mirrazavi, M. Tamiz, "Multi-objective meta-heuristics: An overview of the current state of-the-art," *European Journal of Operational Research* vol. 137 (1), pp. 1–9, 2002.

[9] J. Moore and R. Chapman, "Application of particle swarm to multiobjective optimization," Dept. of Comp. Sc. and Soft. Eng., Auburn University, USA, Tech Rep. 1999.

[10] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *IEEE International Conference on Neural Network IV*, Perth, Australia, 1995, pp. 1942–1948.

[11] C.-C. Kao, C.-W. Chuang, R.-F. Fung, "The self-tuning PID control in a slider–crank mechanism system by applying particle swarm optimization approach," *Mechatronics* vol. 16 (8), pp. 513–522. 2006.

[12] R. C. Eberhart and Y. Shi, "Particle swarm optimization: developments, applications and resources," in *2001 Congress on Evolutionary Computation*, Seoul, South Korea, 2001, pp. 81–86.

[13] J. Kennedy, J. F. Kennedy, R. C. Eberhart, Y. Shi, *Swarm Intelligence*, New York: Morgan Kaufmann, 2001.

[14] A. P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, New York: John Wiley & Sons, 2005.

[15] Y. Shi, and R. Eberhart, R. "A modified particle swarm optimizer," in *1998 IEEE International Conference on Evolutionary Computation and World Congress on Computational Intelligence*, Alaska, USA, 1998, pp. 69–73.

[16] J. D. Altringham, L. Hammond, T. McOwat, "Bats: biology and Behaviour, London: The Oxford University Press, 1996.

[17] M. Airas, "Echolocation in bats," in *The postgrad seminar course of HUT Acoustics Laboratory on spatial sound perception and reproduction*, Helsinki, Finland, 2003, pp. 1–25.

[18] N. M. Yahya, M. O. Tokhi, H. A. Kasdirin, "A new bats echolocation-based algorithm for single objective optimisation," *Evolutionary Intelligence*, vol. 9 (1), pp. 1–20, 2016.

[19] N. M. Yahya and M. O. Tokhi, "A modified bats echolocation-based algorithm for solving constrained optimisation problems," *International Journal of Bio-Inspired* Computation, vol. 10(1), pp. 12–23, 2017.

[20] S. L. Voigt-Heucke, M. Taborsky, K. D. Dechmann, "A dual function of echolocation: bats use echolocation calls to identify familiar and unfamiliar individuals," *Animal Behaviour*, vol. 80 (1), pp. 59–67, 2010.

[21] J. Kennedy, "Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance, " in *1999 Congress on Evolutionary Computation*, Indianapolis, USA, 1999, pp. 1931–1938.

[22] J. Kennedy, and R. Mendes, "Population structure and particle swarm performance," in *2002 Congress on Evolutionary Computation*, Honolulu, USA, 2002, pp. 1671–1676.

[23] E. S. Peer, F. van den Bergh, A. P. Engelbrecht, "Using neighbourhoods with the guaranteed convergence PSO," in *2003 IEEE Swarm Intelligence Symposium*, Indianapolis, USA, 2003, pp. 235–242.

[24] L. K. DeNault, and D. A. McFarlane, "Reciprocal altruism between male vampire bats, desmodus rotundus," *Animal Behaviour*, vol. 49 (3), pp. 855–856, 1995.

[25] G. S. Wilkinson, "Reciprocal altruism in bats and other mammals," *Ethology and Sociobiology,* vol. 9 (2), pp. 85–100, 1988.

[26] K. Than, (2011). *Highest flying bird found; can scale himalaya* [Online]. Available: http://news.nationalgeographic.com/news/2011/06/110610-highest-flying-birds-geese-himalaya-mountainsanimals/,

[27] I. Ahlén, H. J. Baagøe, L. Bach, "Behavior of scandinavian bats during migration and foraging at sea," *Journal of Mammalogy*, vol. 90 (6), pp. 1318–1323, 2009.

[28] P. R. Ehrlich, D. S. Dobkin, D. Wheye, (1988). *How fast and high do birds fly?* [Online]. Available: http://web.stanford.edu/group/stanfordbirds/text/essays/HowFast.html

[29] E. Zitzler, K. Deb, L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary Computation*, vol. 8 (2), pp. 173–195, 2000.

[30] J. Knowles, and D. Corne, "The pareto archived evolution strategy: A new baseline algorithm for pareto multiobjective optimisation," in 1999 Congress on Evolutionary Computation, Washington, USA, 1999.

[31] C. A. C. Coello, "A short tutorial on evolutionary multiobjective optimization," in *Evolutionary Multi-Criterion Optimization*, Springer, 2001, pp. 21-40.