Particle Swarm Optimization

Chapter *in* Studies in Computational Intelligence · January 2019

DOI: 10.1007/978-3-319-91341-4_2

CITATIONS
102

READS
242

1 author:

Jagdish Chand Bansal
South Asian University
164 PUBLICATIONS **3,984** CITATIONS

SEE PROFILE

Noname manuscript No.
(will be inserted by the editor)

Particle Swarm Optimization

Jagdish Chand Bansal

the date of receipt and acceptance should be inserted later

Abstract Particle Swarm Optimization (PSO) is a swarm intelligence based numerical optimization algorithm, introduced in 1995 by James Kennedy, a social psychologist, and Russell Eberhart, an electrical engineer. PSO has been improved in many ways since its inception. This chapter provides an introduction to the basic particle swarm optimization algorithm. For better understanding of the algorithm, a worked-out example has also been given.

1 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a swarm intelligent algorithm, inspired from birds' flocking or fish schooling for the solution of nonlinear, nonconvex or combinatorial optimization problems that arise in many science and engineering domains. In [4] Feng et al. used PSO to optimize parameters in radical basis function (RBF) network. In [2] PSO is modified to solve the two-sided U-type assembly line balancing (TUALB) problem. The results obtained by proposed approach showed that designing a two-sided assembly line in U-shaped layout provides shorter lines. Mousavi et al. applied PSO to optimize a supply chain network for a seasonal multi-product inventory system with multiple buyers, multiple vendors and warehouses with limited capacity owned by the vendors [8]. Apart from continuous version of PSO, binary PSO has also been applied extensively. Bansal et al. [1] proposed a modified binary PSO and applied it to solve various forms of knapsack problems. Jain et al. [5] proposed an improved-Binary Particle Swarm Optimization (iBPSO) for cancer classification. The next subsection presents the motivation and general framework of PSO procedure.

Jagdish Chand Bansal
Department of Mathematics, South Asian University, New Delhi, India
E-mail: jcbansal@gmail.com

1.1 Motivation

Many bird species are social and form flocks for various reasons. Flocks may be of different sizes, occur in different seasons and may even be composed of different species that can work well together in a group. More eyes and ears mean increased opportunities to find food and improved chances of detecting a predator in time. Flocks are always beneficial for survival of its members in many ways [10]:

Foraging: A socio-biologist E. O. wilson said that, "In theory at least, individual members of the school (swarm) can profit from the discoveries and previous experience of all other members of the school during the search for food" [11]. If for a group of birds, the food source is the same then some species of birds form flock in a non-competing way. In this way, more birds take advantage of discoveries of other birds about the location of the food.

Protection against Predator: A flock of birds have number of advantages in protecting themselves from the predator:

- More ears and eyes means more chances of spotting a predator or any other potential threat.
- A group of birds may be able to confuse or overwhelm a predator through mobbing or agile flights.
- In case of a group, large availability of prays reduces the danger for any single bird.

Aerodynamics: When birds fly in flocks, they often arrange themselves in specific shapes or formations. Those formations take advantage of the changing wind patterns based on the number of birds in the flock and how each bird's wings create different currents. This allows flying birds to use the surrounding air in the most energy efficient way.

However, the development of PSO requires simulation of some advantages of birds' flock, in order to understand an important property of swarm intelligence and therefore of PSO, it is worth mentioning some disadvantages of the birds' flocking. When birds form flock they also create some risk for them. More ears and more eyes means more wings and more mouths which result more noise and motion. In this situation, more predators can locate the flock causing a constant threat to the birds. A larger flock will also require a greater amount of food which causes more competition for food. This may result in death of some weaker birds of the group. It is important to mention here that PSO does not simulate the disadvantages of the birds' flocking

behavior and therefore, during the search process killing of any individual is not allowed as in Genetic Algorithms where some weaker individuals die out. In PSO, all individuals remain alive and try to make themselves stronger throughout the search process. The improvement in potential solutions in PSO is due to cooperation while in evolutionary algorithms it is due to competition. This concept makes swarm intelligence different from evolutionary algorithms. In short, in evolutionary algorithms a new population is evolved in every generation / iteration while in swarm intelligent algorithms in every generation / iteration individuals make themselves better. Identity of the individual does not change over the iterations.

Mataric [7] gave the following rules for birds' flocking:

1. **Safe Wandering:** When birds fly they are not allowed to collide with each other and with obstacles.
2. **Dispersion:** Each bird will maintain a minimum distance with any other.
3. **Aggregation:** Each bird will also maintain a maximum distance with any other.
4. **Homing:** All birds will have potential to find a food source or the nest.

All these four rules have not been adopted in the simulation of birds flocking behavior while designing the PSO. In basic PSO model developed by Kennedy and Eberhart, safe wandering and dispersion rules are not followed for the movement of agents. In other words, during the movement in basic PSO agents are allowed to come closer to each other as they can. While aggregation and homing are valid in the PSO model. In PSO, agents have to fly within a particular region so that they can maintain a maximum distance with any other agent. This is equivalent to the fact that throughout the process, search remains within or at the boundaries of the search space. The fourth rule, homing says that any agent in the group may reach to the global optima.

For the development of PSO model, Kennedy and Eberhart followed five fundamental principles which determine whether a group of agents is a swarm or not [12]:

1. Proximity Principle: the population should be able to carry out simple space and time computations.
2. Quality Principle: the population should be able to respond to quality factors in the environment.
3. Diverse Response Principle: the population should not commit its activity along excessively narrow channels.
4. Stability Principle: the population should not change its mode of behaviour every time the environment changes.
5. Adaptability Principle: the population should be able to change its behaviour mode when it is worth the computational price.

1.2 Particle Swarm Optimization Process

Considering these five principles Kennedy and Eberhart developed a PSO model for function optimization. In PSO, the solution is obtained through a random search

equipped with swarm intelligence. In other words, PSO is a swarm intelligent search algorithm. This search is done by a set of randomly generated potential solutions. This collection of potential solutions is known as ***swarm*** and each individual potential solution is known as a ***particle***.

In PSO, the search is influenced by two types of learning by the particles. Each particle learns from other particles and it also learns from its own experience during the movement. The learning from others may be referred as ***social learning*** while the learning from own experience as ***cognitive learning***. As a result from social learning, the particle stores in its memory the best solution visited by any particle of the swarm which we call as ***gbest***. As a result of cognitive learning, the particle stores in its memory the best solution visited so far by itself, called ***pbest***.

Change of the direction and the magnitude in any particle is decided by a factor called ***velocity***. This is the rate of change in the position with respect to the time. With reference to the PSO, time is the iteration. In this way, for PSO, the velocity may be defined as the rate of change in the position with respect to the iteration. Since iteration counter increases by unity, the dimension of the velocity v and the position x becomes the same.

For a $D-$dimensional search space, the i^{th} particle of the swarm at time step t is represented by a D- dimensional vector, $x_i^t = (x_{i1}^t, x_{i2}^t, ..., x_{iD}^t)^T$. The velocity of this particle at time step t is represented by another $D-$dimensional vector $v_i^t = (v_{i1}^t, v_{i2}^t, ..., v_{iD}^t)^T$. The previously best visited position of the i^{th} particle at time step t is denoted as $p_i^t = (p_{i1}^t, p_{i2}^t, ..., p_{iD}^t)^T$. 'g' is the index of the best particle in the swarm. The velocity of the i^{th} particle is updated using the velocity update equation in (1).

Velocity Update Equation:

$$v_{id}^{t+1} = v_{id}^t + c_1 r_1 (p_{id}^t - x_{id}^t) + c_2 r_2 (p_{gd}^t - x_{id}^t) \tag{1}$$

The position is updated using position update equation in (2).

Position Update Equation:

$$x_{id}^{t+1} = x_{id}^t + v_{id}^{t+1} \tag{2}$$

where $d = 1, 2, ..., D$ represents the dimension and $i = 1, 2, ..., S$ represents the particle index. S is the size of the swarm and c_1 and c_2 are constants, called cognitive and social scaling parameters, respectively or simply acceleration coefficients. r_1, r_2 are random numbers in the range $[0, 1]$ drawn from a uniform distribution. It appears from equations (1) and (2) that every particle's each dimension is updated independently from the others. The only link between the dimensions of the problem space is introduced via the objective function, i.e., through the locations of the best positions found so far *gbest* and *pbest* [9]. Equations (1) and (2) define the basic version of PSO algorithm. An algorithmic approach of PSO procedure is given in algorithm 1:

Algorithm 1: Basic Particle Swarm Optimization

```
Create and Initialize a D-dimensional swarm, S and corresponding velocity
  vectors ;
for t= 1 to the maximum bound on the number of iterations do
    for i=1 to S do
        for d=1 to D do
            Apply the velocity update equation 1;
            Apply position update equation 2;
        end
        Compute fitness of updated position;
        If needed, update historical information for pbest and gbest;
    end
    Terminate if gbest meets problem requirements;
end
```

1.3 Understanding Update Equations

The right hand side in the velocity update equation (1), consists of three terms [3] :

1. The previous velocity v, which can be thought of as a momentum term and serves as a memory of the previous direction of movement. This term prevents the particle from drastically changing direction.
2. The second term is known as the cognitive or egoistic component. Due to this component, the current position of a is attracted towards its personal best position. In this way, throughout the search process, a particle remembers its best position and thus prohibits itself from wandering.
   Here, it should be noted that $(p_{id} - x_{id})$ (superscript t is dropped just for simplicity) is a vector whose direction is from x_{id} to p_{id} which results the attraction of current position towards the particle's best position. This order of x_{id} and p_{id} must be maintained for attraction of current position towards the particle's best position. If we write the second term using vector $(x_{id} - p_{id})$ then the current position will repel from the particle's best position.
3. The third term is called social component and is responsible for sharing information throughout the swarm. Because of this term a particle is attracted towards the best particle of the swarm, i.e. each particle learns from others in the swarm.
   Again the same reason stands here also to keep the order of x_{id} and p_{gd} in the vector $(p_{gd} - x_{id})$.

It is clear that cognitive scaling parameter c_1 regulates the maximum step size in the direction of the personal best position of that particle while social scaling parameter c_2 regulates the maximum step size in the direction of global best particle. Figure 1 presents a typical geometric illustration of a particle's movement in a 2-Dimensional space

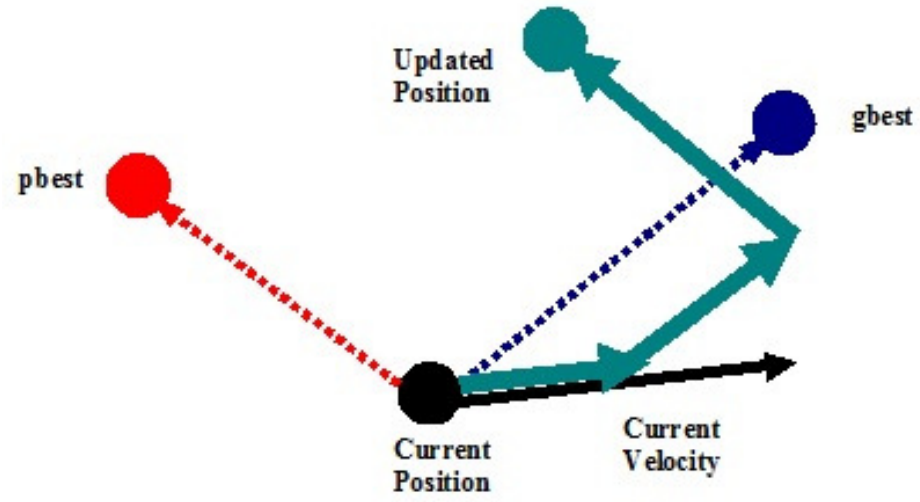

Fig. 1: Geometric Illustration of Particle's Movement in PSO Process

From the update equations, it is also clear that the PSO design of Kennedy and Eberhart follows five basic principals of PSO, described in section 1.1. In the PSO process, calculations are carried out over a series of time steps in a D-dimensional space. Population at any time step, follows the direction guided by gbest and pbest, i.e. the population is responding to the quality factors and thus the quality principal is adhered to. Because of uniformly distributed random numbers r_1 and r_2 in the velocity update equation, a random allocation of current position between pbest and gbest justifies the diverse response principle. Principle of stability is also justified in PSO process because no particle of the swarm moves randomly but only when it receives a better information from gbest. The swarm changes when gbest changes and therefore adaptability principle is adhered to.

2 Particle Swarm Optimization Parameters

The convergence speed and the ability of finding optimal solution of any population based algorithm is greatly influenced by the choice of its parameters. Usually, a general recommendation for the setting of parameters of these algorithms is not possible as it is highly dependent upon the problem parameters. However, theoretical and/or experimental studies have been carried out to recommend the generic range for parameter values. Likewise other population based search algorithms, tuning of parameters for a generic version of PSO has always been a challenging task due the presence of stochastic factors r_1 and r_2 in the search procedure. The basic version of

PSO enjoys the luxury of very few parameters. This chapter discusses parameters of only the basic version of PSO introduced in [6].

One radical parameter is the swarm size which is often set empirically on the basis of the number of decision variables in the problem and problem complexity. In general, 20-50 particles are recommended.

Another parameters are scaling factors, c_1 and c_2. As mentioned earlier, these parameters decide the step size of the particle for the next iteration. In other words, c_1 and c_2 determine the speed of particles. In basic version of PSO, $c_1 = c_2 = 2$ were chosen. With this choice, particle's speed increases without control which is good for faster convergence rate but harmful for better exploitation of the search space. If we set $c_1 = c_2 > 0$ then particles will attract towards the average of pbest and gbest. $c_1 > c_2$ setting will be beneficial for multimodal problems while $c_2 > c_1$ will be beneficial for unimodal problems. Small values of c_1 and c_2 will provide smooth particle trajectories during the search procedure while larger values of c_1 and c_2 will be responsible for abrupt movements with more acceleration. Adaptive acceleration coefficients have also been proposed by the researchers [**?**].

Stopping criterion is also a parameter not only for PSO but for any population based meta-heuristic algorithm. Popular stopping criteria are usually based on maximum number of function evaluations or iterations which are proportional to the time taken by the algorithm and acceptable error. A more efficient stoping criteria is based on the available search capacity of the algorithm. If an algorithm does not improve the solution with a significant amount upto a certain number of iterations, search should be stopped.

3 A Worked-Out Example

In this section, a numerical example is explained for better understanding of the working of PSO. For simplicity, following sphere function in two dimension is considered to minimize using PSO.

$$Minf(x_1, x_2) = x_1^2 + x_2^2; \text{ where } x_1, x_2 \in (-5, 5).$$

First, we generate swarm of size 5, randomly using uniform distribution in the range (-5, 5):
The position matrix x =

x_{ij}	1	2
1	4.7059	-0.7824
2	4.5717	4.1574
3	-0.1462	2.9221
4	3.0028	4.5949
5	-3.5811	1.5574

As mentioned in the PSO Algorithm 1, initialization of velocity vectors is also required at this stage. Velocity corresponding to a particle is initialized in the range

$[-V_{max}, +V_{max}]$. Here V_{max}, the maximum velocity bound is a PSO parameter and is usually set $V_{max} = X_{max}$.

Therefore, in this example velocity vector is generated uniformly in the range $[-5,5]$.
The velocity matrix V =

v_{ij}	1	2
1	4.0579	-2.215
2	-3.7301	0.4688
3	4.1338	4.5751
4	1.3236	4.6489
5	-4.0246	-3.4239

Next step is the objective function evaluation for the current position matrix. The fitness is usually the value of the objective function in the optimization problem being solved. A solution with better objective function value represents a better fit solution. Since the considered problem is a minimization problem, we will consider a solution better fit if it has small objective function value. Substituting $x_{11} = 4.7059$ and $x_{12} = -0.7824$ in the objective function $f = x_1^2 + x_2^2$, we get 22.7576. Similarly, calculating objective function value for other position vectors, we get the following initial fitness matrix: It can be observed that the minimum of these 5 objective function values

f_1	22.7576
f_2	38.1844
f_3	8.5600
f_4	30.1299
f_5	15.2497

corresponding to 5 particles is 8.5600. Therefore, the most fit solution of this swarm is $x_3 = (-0.1462, 2.9221)$ which we call "*gbest*". Since this is the first iteration, no previous iteration exists for the comparison and therefore every particle's current position is also the "*pbest*" position.

Now we proceed to the next iteration using PSO update equations. It should be noted here that all calculations for velocity and position update are carried out component wise.

Let us consider to update the first particle $x_1 = (4.7059, -0.7824)$. First we will update its first component $x_{11} = 4.7059$. The velocity component corresponding to x_{11} is $v_{11} = 4.0579$. Therefore, we will apply velocity update equation to v_{11} as follows: (considering $c_1 = c_2 = 2$ and $r_1 = 0.34, r_2 = 0.86$)

$$\begin{aligned} v_{11} &= 4.0579 + 2 \times 0.34 \times (4.7059 - 4.7059) + 2 \times 0.86 \times (-0.1462 - 4.7059) \\ &= -4.2877 \end{aligned}$$

Since the updated velocity component -4.2877 lies in the range $[-5,5]$, we accept the value for updating position. In case of updated velocity component value goes

beyond the pre-specified range, we will consider the nearest boundary value. For example, suppose the updated velocity component is -5.8345 then we will set it to -5 because the maximum bound of velocity on this side is -5. Now if the component is 6.8976, the updated velocity will be set equal to 5 because of the similar reason.

Now the position update equation for x_{11} is

$$\begin{aligned} x_{11} &= 4.7059 + (-4.2877) \\ &= 0.4182 \end{aligned}$$

Since the updated solution component lies in the search space $(-5,5)$, we accept the solution. If the updated position does not lie within the given search space, there are many methods suggested by researchers to deal with the situation, some of them will be discussed in the chapter 2. For this example, we will randomly re-initialize the particle if the updated value falls outside the search space boundary.

Similarly, we will update the second component $x_{12} = -0.7824$. To update this, we will first apply velocity update equation on $v_{12} = -2.215$.

$$\begin{aligned} v_{12} &= -2.215 + 2 \times 0.47 \times (-0.7824 - (-0.7824)) + 2 \times 0.91 \times (2.9221 - (-0.7824)) \\ &= 4.5272 \end{aligned}$$

Updated $v_{12} = 4.5272$ is in the range $[-5,5]$ and therefore we will use this value to update x_{12}.

$$\begin{aligned} x_{12} &= -0.7824 + 4.5272 \\ &= 3.7448 \end{aligned}$$

Updated x_{12} is again within the search space $(-5,5)$ so we accept the solution.

Thus the first particle after applying the PSO update equations becomes:
$x_1 = (0.4182, 3.7448)$

We update all the particles using the same procedure.

Second Particle:

$$\begin{aligned} v_{21} &= -3.7301 + 2 \times 0.34 \times (4.5717 - 4.5717) + 2 \times 0.86 \times (-0.1462 - 4.5717) \\ &= -11.8449 \\ x_{21} &= 4.5717 + (-11.8449) \\ &= -7.2732 \end{aligned}$$

Since the updated value of x_{21} is out of the search space, we re-initialize this x_{21} in the range $(-5,5)$. Let $x_{21} = 3.4913$.

$$\begin{aligned} v_{22} &= 0.4688 + 2 \times 0.12 \times (4.1574 - 4.1574) + 2 \times 0.06 \times (2.9221 - 4.1574) \\ &= 0.3206 \\ x_{22} &= 4.1574 + 0.3206 \\ &= 4.4780 \end{aligned}$$

Thus the second particle after PSO updating becomes:
$x_2 = (3.4913, 4.4780)$

Third Particle:

$$v_{31} = 4.1338 + 2 \times 0.69 \times (-0.1462 - (-0.1462)) + 2 \times 0.34 \times (-0.1462 - (-0.1462))$$
$$= 4.1338$$
$$x_{31} = -0.1462 + 4.1338$$
$$= 3.9876$$
$$v_{32} = 4.5751 + 2 \times 0.69 \times (2.9221 - 2.9221) + 2 \times 0.34 \times (2.9221 - 2.9221)$$
$$= 4.5751$$
$$x_{32} = 2.9221 + 4.5751$$
$$= 7.4972 \text{(exceeding the search space bounds.)}$$
$$x_{32} = \text{random value in the range (-5, 5)}$$
$$= 4.3399$$

Thus the updated third particle is $(3.9876, 4.3399)$.

Fourth Particle:

$$v_{41} = 1.3236 + 2 \times 0.18 \times (3.0028 - 3.0028) + 2 \times 0.23 \times (-0.1462 - 3.0028)$$
$$= -0.1249$$
$$x_{41} = 3.0028 + (-0.1249)$$
$$= 2.8779$$
$$v_{42} = 4.6489 + 2 \times 0.61 \times (4.5949 - 4.5949) + 2 \times 0.94 \times (2.9221 - 4.5949)$$
$$= 1.5040$$
$$x_{42} = 4.5949 + 1.5040$$
$$= 6.0989 \text{(falls outside the search space bounds.)}$$
$$x_{42} = \text{random value in the range (-5, 5)}$$
$$= 2.5774$$

Updated fourth particle is $(2.8779, 2.5774)$.

Fifth Particle:

$$v_{51} = -4.0246 + 2 \times 0.09 \times (-3.5811 - (-3.5811)) + 2 \times 0.39 \times (-0.1462 - (-3.5811))$$
$$= -1.3454$$
$$x_{51} = -3.5811 + (-1.3454)$$
$$= -4.9265$$
$$v_{52} = -3.4239 + 2 \times 0.65 \times (1.5574 - 1.5574) + 2 \times 0.10 \times (2.9221 - 1.5574)$$
$$= -3.1510$$
$$x_{52} = 1.5574 + (-4.0246)$$
$$= -2.4672$$

Updated fifth particle is $(-4.9265, -2.4672)$.

Therefore, after this initial iteration, the updated velocity matrix v is shown in Table 1, the updated position matrix x and the fitness matrix are shown in Table 2 and Table 3, respectively.

Table 1: Updated velocity matrix

v_{ij}	1	2
1	-4.2877	4.5272
2	-11.8449	0.3206
3	4.1338	4.5751
4	-0.1249	1.504
5	-1.3454	-3.151

Table 2: Updated position matrix

x_{ij}	1	2
1	0.4182	3.7448
2	3.4913	4.4780
3	3.9876	4.3399
4	2.8779	2.5774
5	-4.9265	-2.4672

Table 3: Updated fitness values

f_1	14.1984
f_2	32.2416
f_3	34.7356
f_4	14.9252
f_5	30.3574

Clearly, it can be seen that the minimum objective function value is 14.1984 which corresponds to the first particle. Therefore, *gbest* for the updated swarm is the first particle x_1.

Now we compare this *gbest* with the previous *gbest*, obviously updated gbest is not better than the previous one so for the carrying out the next iteration, we consider the gbest of previous iteration $(-0.1462, 2.9221)$.

Now for each particle, we observe the selection of *pbest*. It should be noted that *gbest* is for the whole swarm and *pbest* is for a particular particle.

For the first particle:
Fitness in the previous swarm = 22.7576
Fitness in the current swarm = 14.1984

Clearly, the fitness of current swarm is better than that of its previous, so we set $pbest_1 = (0.4182, 3.7448)$. On the other hand, if the fitness of current swarm would not be better than that of its previous then the current *pbest* and old *pbest* would be the same.

Similarly,
for the second particle: $pbest_2 = (3.4913, 4.4780)$;
for the third particle: pbest3 = (2.2534, 3.1379);
for the fourth particle: pbest4 = (1.6400, 1.3202) and
for the fifth particle: pbest5 = (2.2668, 2.0009).
The same procedure is continued until the termination criterion is attained.

As a final note to the chapter, "PSO is a dynamic population of active, interactive agents with no inherent intelligence ". In PSO each individual teaches its neighbor, each individual learns from its neighbors. During the search procedure, potential solutions make better than random guesses using *Collaborative Trial and Error* strategies. These guesses are better than random search because they are informed by social learning. Since its inception, PSO has seen many changes which made it a strong candidate for numerical optimization. Researchers have applied PSO to almost all kind of problems where a numerical optimization technique is expected to work.

Particle Swarm Optimization is quite flexible for modifications according to the problem requirements. Therefore, even after 23 years of its invention, there is enough scope to modify PSO and apply it to new complex optimization problems.

References

1. Jagdish Chand Bansal and Kusum Deep. A modified binary particle swarm optimization for knapsack problems. *Applied Mathematics and Computation*, 218(22):11042–11061, 2012.
2. Yılmaz Delice, Emel Kızılkaya Aydoğan, Uğur Özcan, and Mehmet Sıtkı İlkay. Balancing two-sided u-type assembly lines using modified particle swarm optimization algorithm. *4OR*, 15(1):37–66, 2017.
3. Andries P Engelbrecht. *Computational intelligence: an introduction*. Wiley. com, 2007.
4. Jingwei Feng, Fengchun Tian, Pengfei Jia, Qinghua He, Yue Shen, and Shu Fan. Improving the performance of electronic nose for wound infection detection using orthogonal signal correction and particle swarm optimization. *Sensor Review*, 34(4):389–395, 2014.
5. Indu Jain, Vinod Kumar Jain, and Renu Jain. Correlation feature selection based improved-binary particle swarm optimization for gene selection and cancer classification. *Applied Soft Computing*, 62:203–215, 2018.
6. Kennedy James and Eberhart Russell. Particle swarm optimization. In *Proceedings of 1995 IEEE International Conference on Neural Networks*, pages 1942–1948, 1995.
7. Maja J Mataric. Interaction and intelligent behavior. Technical report, DTIC Document, 1994.

8. Seyed Mohsen Mousavi, Ardeshir Bahreininejad, S Nurmaya Musa, and Farazila Yusof. A modified particle swarm optimization for solving the integrated location and inventory control problems in a two-echelon supply chain network. *Journal of intelligent manufacturing*, 28(1):191–206, 2017.
9. Ioan Cristian Trelea. The particle swarm optimization algorithm: convergence analysis and parameter selection. *Information processing letters*, 85(6):317–325, 2003.
10. webpage. http://birding.about.com/od/birdbehavior/a/why-birds-flock.htm.
11. Edward Wilson. 0.(1975) sociobiology: The new synthesis, 1980.
12. Bin Yang. *Modified particle swarm optimizers and their application to robust design and structural optimization*. PhD thesis, Mu?nchen, Techn. Univ., Diss., 2009, 2009.