International Journal of Mathematical, Engineering and Management Sciences
Vol. 5, No. 4, 732-743, 2020
https://doi.org/10.33889/IJMEMS.2020.5.4.058

# An Overview of Few Nature Inspired Optimization Techniques and Its Reliability Applications

**Nitin Uniyal**
Department of Mathematics,
University of Petroleum & Energy Studies, Dehradun, India.
E-mail: nuniyal@ddn.upes.ac.in

**Sangeeta Pant**
Department of Mathematics,
University of Petroleum & Energy Studies, Dehradun, India.
*Corresponding author*: pant.sangeet@gmail.com

**Anuj Kumar**
Department of Mathematics,
University of Petroleum & Energy Studies, Dehradun, India.
E-mail: anuj4march@gmail.com

(Received June 21, 2019; Accepted March 15, 2020)

**Abstract**
Optimization has been a hot topic due to its inevitably in the development of new algorithms in almost every applied branch of Mathematics. Despite the broadness of optimization techniques in research fields, there is always an open scope of further refinement. We present here an overview of nature-inspired optimization with a subtle background of fundamentals and classification and their reliability applications. An attempt has been made to exhibit the contrast nature of multi objective optimization as compared to single objective optimization. Though there are various techniques to achieve the optimality in optimization problems but nature inspired algorithms have proved to be very efficient and gained special attention in modern research problems. The purpose of this article is to furnish the foundation of few nature inspired optimization techniques and their reliability applications to an interested researcher.

**Keywords-** Metaheuristics, Grey wolf optimizer, Multi-objective optimization, Reliability optimization

## 1. Introduction
As the term suggests, optimization is an act of moving towards the optimum. In other words, it is a pursuit to the best in the local sense and the method facilitating this, comes under the category of an optimization method. Mathematically, one seeks the extremum (*called optimum*) of a function $f(x_1, x_2, \dots\dots, x_n)$ of several variables in the domain of its definition with simultaneously taking care of the restrictions (*called constraints*) on the variables $x_i, 1 \leq i \leq n$ .

The inevitability of optimization can be understood in the natural learning processes of a child who optimizes the speech from lisping to fluency, writing skills from scribbling to perfection and walking skills from crawling to feet. This act of betterment is widespread in the areas of science, engineering, economics, management, etc. where the numerical data is processed. For instance, a civil engineer is aimed to the best design of an anti-earthquake bridge subject to the approved budget and time constraints. In contrast, a hardware engineer working on computer chips is always keen to minimize the area of the resulting layout accommodating the optimum arrangement of transistors. A well-posed optimization problem has three essential ingredients (Kumar et al., 2018a):

*Variables*: The entities $x_1, x_2, \dots \dots, x_n$ which need to be varied to create different possibilities for the value of $f$.

*Constraints*: The (in)equations involving entities $x_i$ $(1 \le i \le n)$ which restricts their freeness in the specified domain.

*Objective Functions*: A function $f$ assigning a unique value to each different possibility for the $n$ −tuple $(x_1, x_2, \dots \dots, x_n)$ with an objective of getting optimized under the given constraints

The general optimization problem can be described as:

Given $f : D^n \longrightarrow \mathbb{R}$, where $D^n = D_1 \times D_2 \times \dots \times D_n \ni \boldsymbol{x} = (x_1, x_2, \dots, x_n)$.
Find $\boldsymbol{x}^* = (x_1^*, x_2^*, \dots, x_n^*) \in D^n$ satisfying the constraints

$$c_i^{=}(\boldsymbol{x}) = 0, 1 \le i \le n_{c^=} \tag{1}$$

$$c_i^{+}(\boldsymbol{x}) \ge 0,\ 1 \le i \le n_{c^+} \tag{2}$$

$$c_i^{-}(\boldsymbol{x}) \le 0,\ 1 \le i \le n_{c^-} \tag{3}$$

that optimizes (minimize or maximize) the function $f(\boldsymbol{x})$, i.e.,

$$\left.\begin{array}{ll} f(\boldsymbol{x}^*) \le f(\boldsymbol{x}) \text{ (minimization)} & \\ \qquad\qquad \text{or} & \forall \boldsymbol{x} = (x_1, x_2, \dots, x_n) \in D^n \\ f(\boldsymbol{x}^*) \ge f(\boldsymbol{x}) \text{ (maximization)} & \end{array}\right\} \tag{4}$$

where, $n$ is the dimension of the problem which is to optimize. $f(\boldsymbol{x})$ is termed as objective function and its domain $D^n = D_1 \times D_2 \times \dots \times D_n$ is called the *decision variable space*. $D_i$, either continuous or discrete, is the search space of $x_i$, the $i^{th}$ optimization parameter. $D_i$ and $D_j$ are not necessarily identical, in the sense of either type or size. An *n*-dimensional vector of optimization variables $\boldsymbol{x} = (x_1, x_2, \dots, x_n)$ is called the *feasible solution* if it satisfies all the constraints. *search space* can be viewed as the set of all feasible solutions. A feasible solution $\boldsymbol{x}^*$ that minimizes (maximizes) the objective function is called an *optimal solution*. The symbol $c_i^{=}$ is the $i^{th}$ equality constraint function, $n_{c^=}$ is the number of equality constraint functions. $c_i^{+}$ is the $i^{th}$ positive constraint function, $n_{c^+}$ is the number of positive constraint functions. $c_i^{-}$ is the $i^{th}$ negative constraint function, $n_{c^-}$ is the number of negative constraint functions.

In the light of qualitative analysis for the calculus of functions, an optimal solution $\boldsymbol{x}^*$ s.t. $\forall \boldsymbol{x} \in D^n, f(\boldsymbol{x}^*) - f(\boldsymbol{x}) \ge 0$ *or* $\le 0$ is called the *global optimum solution*. Such a solution is absolutely the best set of parameters $(x_1^*, x_2^*, \dots, x_n^*) \in D^n$ that optimizes $f$. Hoping straightaway to get such a global winner solution is generally a cumbersome task and this happens in the case of non-linear programming (NLP) (Boyd and Vandenberghe, 2004; Kumar et al., 2017d; Pant et al., 2017a). Nonetheless, expecting a local winner $\boldsymbol{x}^{*^*}$ in some open proper subset $L \subset D^n$ needs to qualify fairly relaxed criteria i.e. $\forall \boldsymbol{x} \in L, f(\boldsymbol{x}^{*^*}) - f(\boldsymbol{x}) \geq\ 0\ or \leq 0.$ Such a solution is termed as a *local optimum solution*. A lot to depends upon the choice of the initial point $x_0 \in L$ while working with a local optimization algorithm but a good converging algorithm is the one, which always converges regardless of this particular choice. Such an algorithm is said to be *globally convergent* in the sense of freeness for the choice of $x_0$ inside $L$ (Bergh, 2001).

## 2. Classification

On the basis of decision variable space, constraints and objective function an optimization problem is classified as follows:

For the given decision variable space $D^n \subseteq \mathbb{R}^n$ (where $\mathbb{R}$ is the set of reals) of real valued programming problem, if in addition $D^n \subseteq \mathbb{Z}^n$ (where $\mathbb{Z}$ is the set of integers) holds then the problem is called as *Integer Programming Problem*. In contrast, if $D^n \subseteq \mathbb{Z}^n$ doesn't hold and $D^n \cap \mathbb{Z}^n \neq \emptyset$ then the problem is called as *Mixed Integer Programming Problem*. Furthermore, if each $D_i, 1 \leq i \leq n$ is finite then the problem is called as *Combinatorial Programming Problem*. The term *combinatorial* has been derived from the fact that the solution is one of all permutations or combinations of the elements of finite sets $D_i$.

An optimization problem is *constrained* or *unconstrained* depending on the presence or absence of the constraints. As long as all the constraints and objective function involves linear polynomials, the problem is called *Linear Programming Problem* (LPP) else it is *Nonlinear Programming Problem* (NLPP). Further, the appearance of posynomials in the constraints and objective function categorizes the problem as *Geometric Programming Problem* (GPP). A polynomial is a function of the form $f(x_1, x_2, \dots, x_n) = \sum_{k=1}^{K} c_k x_1^{a_{1k}} \dots x_n^{a_{nk}}$ where all the coordinates $x_i$ and the coefficients $c_k$ are positive real numbers, and the exponents $a_{ik}$ are real numbers.

Based on the number of objective functions involved, an optimization problem can be classified as *Single /Multi Objective Optimization Problem* (SOOP/MOOP) depending on whether it contains a single or more objective function.

A special class of optimization called *Convex Optimization* deals with the convex nature of objective function to be minimized and related constraints. In such problems due to the convexity of search space and the objective function $f$, local minima is sufficient to guarantee the global minima. It is worth mentioning here the definitions of a convex set and convex function which are as follows:

*Convex set*: A set $S \subseteq \mathbb{R}^n$ is convex if the line segment joining any two points of it lies inside the set $S$ itself. Mathematically, $\forall x, y \in S;\ \ tx + (1-t)y \in S$ where $t \in [0,1]$. (Boyd and Vandenberghe, 2004). However unbounding this parameter $t$ by letting it free on the set of reals, gives rise to an *affine set* where the whole infinite line through any two points of $S$ lies completely inside $S$. Thus for an affine set $S$, the linear combination of $x$ and $y$ i.e. $\forall x, y \in S, \quad t_1 x + t_2 y \in S$ where $t_1 + t_2 = 1$. Hence an affine set is a space in its own regard.

*Convex function:* A real-valued function $f: S \longrightarrow \mathbb{R}$ defined is convex if the domain $S$ is a convex set and the line segment joining any two points on the graph of $f$ lies above or on the graph i.e.

$$f(tx + (1 - t)y) \leq\ tf(x) +\ (1 - t)f(y) \text{ holds for all } x, y \in S \text{ and } t \in [0,1].$$

If the symbol $' \leq '$ is replaced by the symbol of equality $' = '$ then $f$ is an *affine function*.

## 3. Multi-Objective Optimization

In view of the multi criteria nature of most of the real world problems, one certainly faces an objective function addressing more than one attributes to be optimized in the related optimization problem. For instance, a person $X$ wants to buy a cooling device which can serve his two objectives of maximum comfort and least cost (Figure 1). Among the large range of such devices with an obvious direct proportion between comfort and cost, he will eventually settle down to something which on average fulfill both objectives. He cannot maximize comfort without compromising his penny pincher attitude. It is evident that another customer may select a device based on the same attributes but having values quite different from $X$. This triggers the existence of a number (possibly infinite) of *Pareto optimal* solutions for a MOOP. Applications involving simultaneous optimization of several incommensurable problems like curve fitting (Ahonen et al., 1997), proteins atomic structure determination (Bush et al., 1995), pattern recognition of X-ray diffraction (Paszkowicz, 1996), potential function parameter optimization (Skinner and Broughton, 1995), production scheduling (Swinehart et al., 1996) and design of complex hardware/software systems (Zitzler, 1999) have been posed beautifully.

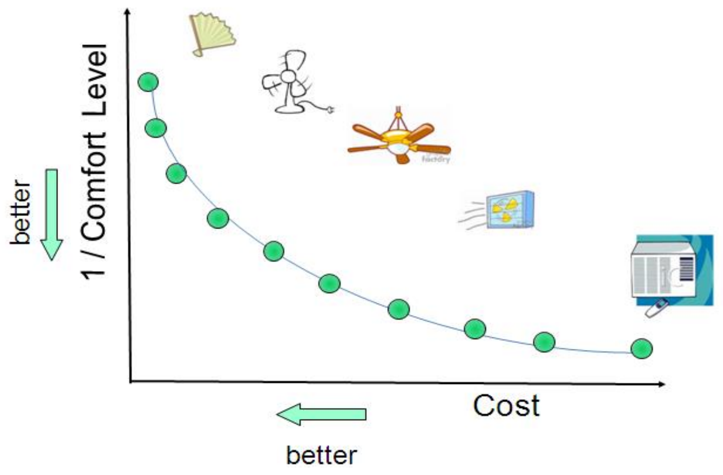


Figure 1. Trade off solutions for cooling system buying DM problem

The general MOOP problem can be defined as:

Given $f : D^n \to \mathbb{R}^k, (k \geq 2)$ and $D^n = D_1 \times D_2 \times \ldots \times D_n \ni \boldsymbol{x} = (x_1, x_2, \ldots, x_n)$

Find $\boldsymbol{x}^* = (x_1^*, x_2^*, \ldots, x_n^*) \in D^n$ which satisfies

$$c_i^{=}(\boldsymbol{x}) = 0, 1 \leq i \leq n_{c^=} \tag{5}$$

$$c_i^{+}(\boldsymbol{x}) \geq 0, \ 1 \leq i \leq n_{c^+} \tag{6}$$

$$c_i^{-}(\boldsymbol{x}) \leq 0, \ 1 \leq i \leq n_{c^-} \tag{7}$$

and minimize

$$f(\boldsymbol{x}) = [f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \dots, f_k(\boldsymbol{x})]^T \in \mathbb{R}^k \tag{8}$$

where, the objective to be maximized (*if any*) is converted into an objective to be minimized by taking its negative. Other symbols are same as earlier we defined in section 1. An *n*-dimensional vector of optimization variables $\boldsymbol{x} = (x_1, x_{2,} \dots, x_n)$ which satisfies all the constraints is called feasible solution and the subset $\boldsymbol{X} \subseteq D^n$ containing all of them is *feasible decision space* or *feasible region* or *search space*. The image of $\boldsymbol{X}$ under $f$ i.e. $f(\boldsymbol{X}) \subseteq \mathbb{R}^k$ is *feasible criterion space*.

Apparently, in multi-objective optimization, a single solution is scarcely the finest for all the objectives simultaneously. This triggers the emergence of paying attention to a refined set $\boldsymbol{X_p} \subseteq \boldsymbol{X}$ of solutions that may be improved further for some $f_i(\boldsymbol{x})$, but only at the cost of degrading at least other component $f_{j\neq i}(\boldsymbol{x})$ of the objective vector function $f(\boldsymbol{x})$. Nevertheless each member of $\boldsymbol{x}^* \in \boldsymbol{X_p}$ dominates (*or* is non dominated by) each member of $\boldsymbol{x} \in \boldsymbol{X} \setminus \boldsymbol{X_p}$ in the sense of functional values.

Mathematically, $\forall \boldsymbol{x}^* \in \boldsymbol{X_p}$, $\forall \boldsymbol{x} \in \boldsymbol{X} \setminus \boldsymbol{X_p}$; $f_i(\boldsymbol{x}^*) \leq f_i(\boldsymbol{x})$ holds for each $i \in \{1,2,\dots,k\}$ and $f_i(\boldsymbol{x}^*) < f_i(\boldsymbol{x})$ for at least one $j \in \{1,2,\dots,k\}$. This refined set $\boldsymbol{X_p}$ is called as *Pareto optimal solution set* and its range $f(\boldsymbol{X_p})$ is called as *Pareto front or Pareto boundary* (Figure 2). It is notable that any two solutions $\boldsymbol{x_1^*}$, $\boldsymbol{x_2^*} \in \boldsymbol{X_p}$ are incomparable as neither of them can dominate the other in all objectives. The size and shape of Pareto optimal front generally depend upon the interaction among objective functions and their quantity (Deb, 1999). The conflicting nature of objectives result in a Pareto front with a larger span as compared to the case of cooperating objectives

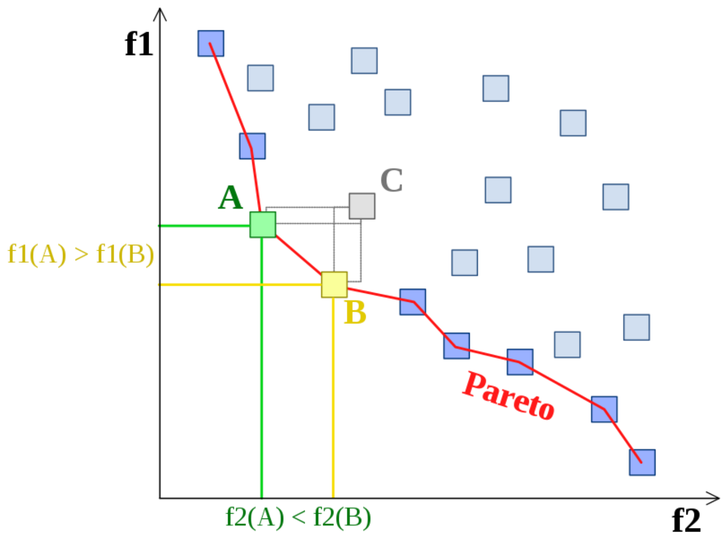


Figure 2. Example of Pareto front (in red), Point C is dominated by both points A and B

## 4. Nature Inspired Optimization Techniques

There are different methods capable of finding solutions to optimization problems but the class which gained popularity is of Meta Heuristics methods. We are reviewing some popular nature inspired Meta Heuristics techniques useful in reliability optimization of complex systems and which in due course of time further laid the foundation to different subsidiary methods.

### 4.1 Ant Colony Optimization (ACO)

The Ant Colony Algorithm was proposed by Dorigo (1992). His source of inspiration for this was the foraging behavior of some ant species. Nature has given them the ability to discover the shortest path between their nest and food source. He found that ants living in a group are well versed to cooperate and find easily the shortest path in search of food, but single ant cannot (Dorigo, 1994). Initially, an ant randomly explores the area surrounding their nest and on locating the food spot, it analyzes food quantity and during its return journey brings some food on it back to the nest. During the return journey, a chemical named *pheromone* dropped by the ant on the ground. The quantity of pheromone deposited, which may depend on the quantity and quality of the food, will guide other ants to the food source. The other ants can sense this chemical in their way of searching food. The rest of the ants choose the way which has higher pheromone. As a result, pheromone accumulates faster in the shorter path and eventually all the ants converge to it. The basic idea of a real ant system can be depicted in Figure 3.

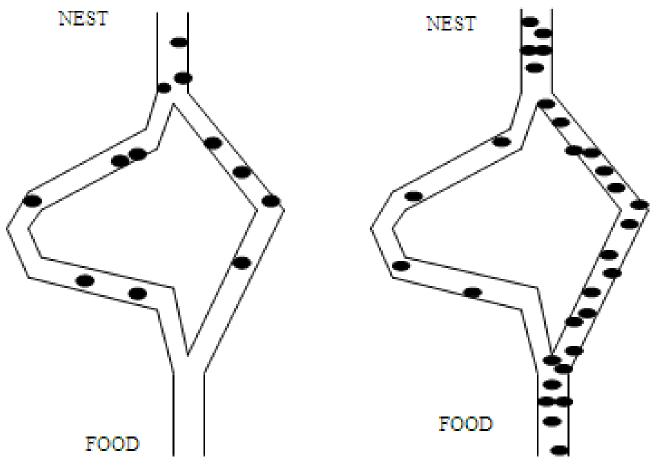


Figure 3. The behavior of real ant movements

Numerous ACO algorithms have been proposed by the researchers, of which the first one, Ant System (AS) (Dorigo, 2006) can be understood in the context of Travelling Salesman Problem. In a network of $n$ cities, where $(i,j)$ represents the edge joining city $i$ to city $j$. Suppose $\tau_{ij}$ is the pheromone associated with the edge $(i,j)$, which is updated by all the $m$ ants involved in building the solution as follows:

$$\tau_{ij} \longleftarrow (1-\rho).\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}\ , \tag{9}$$

where, $\rho$ is the evaporation rate, $\Delta\tau_{ij}^{k}$ is the amount of pheromone laid on edge $(i,j)$ by ant $k$:

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{Q}{L_k} & \text{if ant } k \text{ used edge } (i,j) \text{in its tour} \\ 0 & \text{otherwise,} \end{cases} \tag{10}$$

where, $Q$ is a constant, and $L_k$ is the length of the tour constructed by ant $k$.

A stochastic mechanism involving the probability is followed by an ant $k$ at city $i$ to move towards city $j$, which is given by:

$$p_{ij}^{k} = \frac{\tau_{ij}^{\alpha}.\eta_{ij}^{\beta}}{\sum_{z\in A_i} \tau_{iz}^{\alpha}.\eta_{iz}^{\beta}} \tag{11}$$

where, $A_i$ is the set of all adjacent vertices to vertex $i$ which are not visited yet. $\eta_{ij}$ is the desirability of transition from $i$ to $j$ for ant $k$ and the parameter $\beta \geq 1$ is to control its influence. Typically, $\eta_{ij} = \frac{1}{d_{ij}}$ where $d_{ij}$ is the distance between $i$ and $j$.

## 4.2 Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) is a population-based search algorithm, first proposed by Eberhart and Kennedy (1995). The algorithm mirrors the behavior of animals' societies that don't have any permanent leader among them but they follow temporary leaders in small intervals of time during the quest of food (Pant et al., 2015b). Few classical examples like a flock of birds and school of fish, where the members of the group follow a specific member for a while which has the closest reach to the food source. Thus, the group tends to reach the food source optimally after the finite number of switches at the temporary leader position.

The PSO operates on a population of individuals $N$ $(say)$ called particles, where each particle $i$ $(say)$ represents a potential solution, flying through the search space. The position and velocity i.e. $x^i$ and $v^i$ respectively, are updated according to the relationship between the particles' parameters and the best location, the particle, and the population have found so far. The search is biased toward better regions of space, with the result being a sort of "flocking" toward the best solutions. Let $p^i$ and $f$ respectively denote the personal best position of particle $i$ and the fitness (objective) function. Then the personal best of particle $i$ at $t$ time step is updated as

$$p^i(t+1) = \begin{cases} p^i(t) & \text{if } f\left(x^i(t+1)\right) \geq f\left(p^i(t)\right) \\ x^i(t+1) & \text{if } f\left(x^i(t+1)\right) < f\left(p^i(t)\right) \end{cases} \tag{12}$$

The personal best position is the best position (i.e. best fitness value) of the particle $i$ achieved so far. When smaller neighborhoods are used, the algorithm is generally referred to as an *lbest* PSO. In case the neighborhood is the entire swarm, the best position in the neighborhood is referred to as the global best particle, and the resulting algorithm is referred to as a *gbest* PSO (Kennedy and Eberhart, 1995; Pant et al., 2017b).

For the *gbest* model, the best particle is determined from the entire swarm by selecting the best 'personal best position'. If the position of the global best particle is denoted by the vector $\hat{p}$, then

$$\hat{p} \in \{p^0(t), p^1(t), \dots, p^N(t)\} \text{ s.t. } f(\hat{p}(t) = \min\{f(p^0(t)), f(p^1(t)), \dots, f(p^N(t))\} \tag{13}$$

For the stochastic purpose, the algorithm makes use of two independent random real sequences $< r_1 >$ and $< r_2 >$ in the interval $(0,1)$. The values of $r_1$ and $r_2$ are scaled by two constants $c_1$ and $c_2$ respectively, called acceleration coefficients. These acceleration coefficients weight the stochastic terms $r_1$ and $r_2$ that pull each particle towards *pbest* and *gbest* positions. Thus the velocity and position of particle $i$ are updated using the following equations:

$$v^i(t+1) = v^i(t) + c_1 r_1 [p^i(t) - x^i(t)] + c_2 r_2 [\hat{p}(t) - x^i(t)] \tag{14}$$

$$x^i(t+1) = x^i(t) + v^i(t+1) \tag{15}$$

Stopping criteria depends on the number of iteration or the process may stop if the velocity updates are close to zero. One can measure the quality of the particles using a fitness function and it reflects the optimality of a particular solution (Pant and Singh, 2011; Pant et al., 2015a).

## 4.3 Grey Wolf Optimizer (GWO)

Grey Wolf optimization algorithm (GWO) proposed by Mirjalili et al. (2014) is inspired by the socio-hunting behaviour of a wild animal named Grey wolves. They present its multi-objective version in the year 2016 (Mirjalili et al., 2016). This algorithm is mathematically modeled on the basis of hunting and encircling behavior of grey wolves during their hunting of prey (Figure 4). This algorithm is governed by the following equations:

$$\vec{D} = \left| \vec{C} . \vec{X}_P(t) - \vec{X}(t) \right| \tag{16}$$

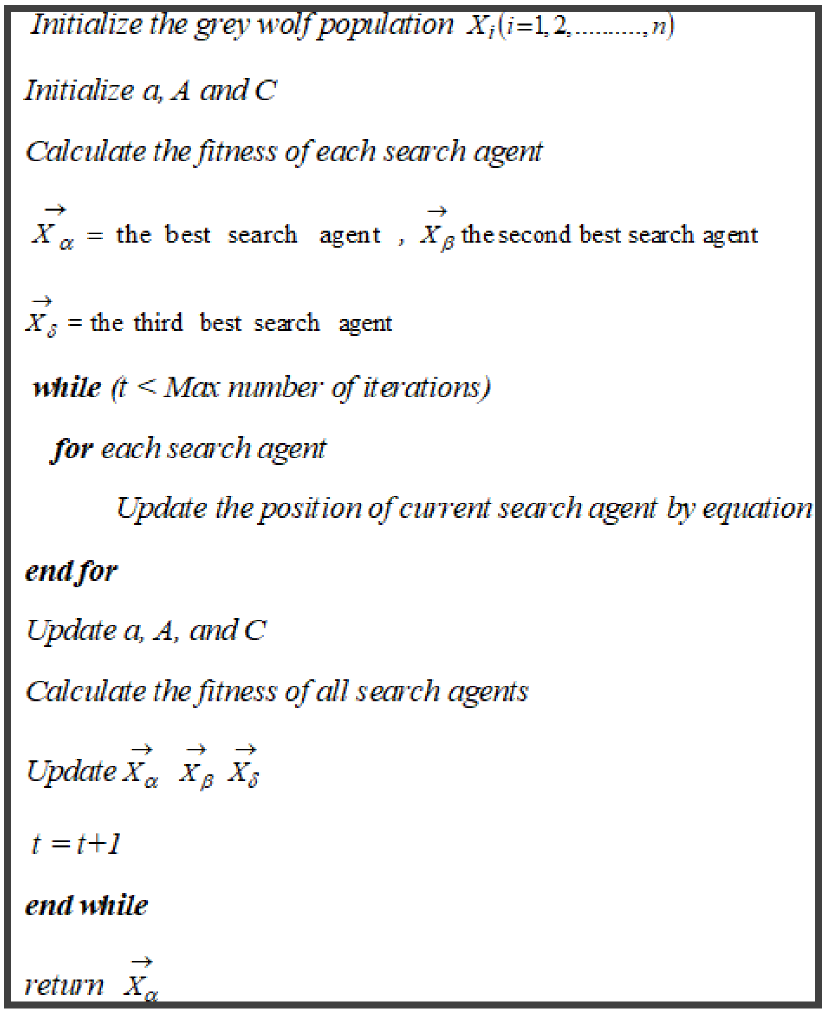
$$\vec{X}(t+1) = \vec{X}_P(t) - \vec{A} . \vec{D} \tag{17}$$

Here, the position vector of the pray is denoted by $\vec{X}_P$ while $\vec{X}$ gives the position vector of a grey wolf. The current iteration is denoted by '$t$' and the coefficient vectors $\vec{A}$ & $\vec{C}$ can be calculated by the help of the below mentioned equations:

$$\vec{A} = 2\vec{a} . \vec{r}_1 - \vec{a} \tag{18}$$

$$\vec{C} = 2 . \vec{r}_2 \tag{19}$$

where, $r_1$ and $r_2$ are random vectors in $[0,1]$ and component of $\vec{a}$ are linearly decreasing from 2 to 0 over the course of iterations.



```
Initialize the grey wolf population $X_i (i = 1, 2, .........., n)$
Initialize a, A and C
Calculate the fitness of each search agent
$\vec{X}_\alpha$ = the best search agent , $\vec{X}_\beta$ the second best search agent
$\vec{X}_\delta$ = the third best search agent
 while (t < Max number of iterations)
   for each search agent
         Update the position of current search agent by equation
end for
Update a, A, and C
Calculate the fitness of all search agents
Update $\vec{X}_\alpha$ $\vec{X}_\beta$ $\vec{X}_\delta$
t = t+1
end while
return $\vec{X}_\alpha$
```

Figure 4. GWO algorithm pseudo code

## 5. Reliability Optimization

For releasing the need for industry, a product designer must satisfy different criteria associated with product development. These are specific to maximizing product reliability and minimizing product cost. At the same time, he/she has to ensure the comfort of consumers and enhancing the functional safety of the product (Kumar and Singh, 2008; Kumar et al., 2017c).

Various researchers of the field of reliability engineering have applied the concepts of nature inspired optimization techniques to their reliability optimization problems (Table 1). Recently, the reliability-cost optimization of the life support system in space capsule by using a very recent metaheuristic named Multi-Objective Grey Wolf Optimizer (MOGWO) approach has been done by Kumar et al. (2019b). The efficiency of MOGWO in optimizing the reliability-cost of life support system has also been demonstrated by comparing its results with a very popular swarm based optimization technique named multi-objective particle swarm optimization. Kumar et al. (2019a) also presented a framework based on Grey Wolf Optimizer for technical specifications optimization of residual heat removal system of a nuclear power plant safety system.

Table 1. Applications of metaheuristics to reliability optimization

| Model | Optimization Techniques | Algorithm Description | Source |
|---|---|---|---|
| Redundancy Allocation | Nature Inspired Metaheuristic | Ant Colony Optimizer (ACO) | Zafiropoulos and Dialynas (2007) |
| | | Simulated Annealing (SA) | Atiqullah and Rao (1993); Wattanapongsakorn and Lavitan (2004) |
| | | Genetic Algorithm (GA) | Coit and Smith (1996); Deeter and Smith (1997) |
| | | Tabu Search (TS) | Kulturel-Konak et al. (2003) |
| Reliability Allocation | Nature Inspired Metaheuristic | Particle Swarm Optimization (PSO) | Hodgson (2002); Pant et al. (2015a) |
| | | Cuckoo Search Algorithm (CSA) | Kumar et al. (2017b) |
| | | Ant Colony Optimizer (ACO) | Shelokar et al. (2002) |
| | | Grey Wolf Optimizer (GWO) | Kumar et al. (2017a); Kumar et al. (2019a); Kumar et al. (2019b) |
| Redundancy – Reliability Allocation | Nature Inspired Metaheuristic | Evolutionary Algorithm (EA) | Ramirez-Rosado and Bernal-Agustín (2001) |
| | | Genetic Algorithm (GA) | Huang et al. (2006) |

## 6. Conclusion

In the last two decades, nature inspired optimization algorithms have witnessed an increasing interest amongst the community of reliability researches. In this article, mathematical background related to optimization and various aspects associated with nature inspired optimization algorithms in the context of reliability optimization are discussed which is beneficial for the researcher in the field of reliability optimization.

**Conflict of Interest**
The authors confirm that there is no conflict of interest to declare for this publication.

**Acknowledgments**
Support from the University of Petroleum & Energy Studies (UPES), Dehradun for conducting this work is gratefully acknowledged. The authors are also thankful to anonymous reviewers for their suggestions to improve this paper.

## References

Ahonen, H., de Souza Júnior, P.A., & Garg, V.K. (1997). A genetic algorithm for fitting lorentzian line shapes in Mössbauer spectra. *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, *124*(4), 633-638.

Atiqullah, M.M., & Rao, S.S. (1993). Reliability optimization of communication networks using simulated annealing. *Microelectronics Reliability*, *33*(9), 1303-1319.

Bergh, F. (2001). *An analysis of particle swarm optimizers*. Ph.D. Thesis, University of Pretoria.

Boyd, S.P., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press, United Kingdom.

Bush, T.S., Catlow, C.R.A., & Battle, P.D. (1995). Evolutionary programming techniques for predicting inorganic crystal structures. *Journal of Materials Chemistry*, *5*(8), 1269-1272.

Coit, D.W., & Smith, A.E. (1996). Reliability optimization of series-parallel systems using a genetic algorithm. *IEEE Transactions on Reliability*, *45*(2), 254-260.

Deb, K. (1999). Multi-objective genetic algorithms: problem difficulties and construction of test problems. *Evolutionary Computation*, *7*(3), 205-230.

Deeter, D.L., & Smith, A.E. (1997). Heuristic optimization of network design considering all-terminal reliability. In *Annual Reliability and Maintainability Symposium* (pp. 194-199). IEEE. Philadelphia, USA.

Dorigo, M. (1992). *Optimization, learning and natural algorithms*. Ph.D. Thesis, Politecnico di Milano.

Dorigo, M. (1994). Learning by probabilistic Boolean networks. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)* (Vol. 2, pp. 887-891). IEEE. Orlando, USA.

Dorigo, M., Birattari, M., & Stutzle, T. (2006). Artificial ants as a computational intelligence technique. *IEEE Computational Intelligence Magazine*, *1*(4), 28-39.

Eberhart, R., & Kennedy, J. (1995). A new optimizer using particle swarm theory. In *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science* (pp. 39-43). IEEE. Nagoya, Japan.

Hodgson, R.J.W. (2002). Particle swarm optimization applied to the atomic cluster optimization problem. In *Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation* (pp. 68-73). New York, USA.

Huang, H.Z., Qu, J., & Zuo, M.J. (2006). A new method of system reliability multi-objective optimization using genetic algorithms. In *RAMS'06. Annual Reliability and Maintainability Symposium, 2006* (pp. 278-283). IEEE. Newport Beach, USA.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95-International Conference on Neural Networks* (Vol. 4, pp. 1942-1948). IEEE. Perth, Australia.

Kulturel-Konak, S., Smith, A.E., & Coit, D.W. (2003). Efficiently solving the redundancy allocation problem using Tabu search. *IIE Transactions*, *35*(6), 515-526.

Kumar, A., & Singh, S.B. (2008). Reliability analysis of an n-unit parallel standby system under imperfect switching using copula. *Computer Modeling and New Technologies*, *12*(1), 47-55.

Kumar, A., Pant, S., & Ram, M. (2017a). System reliability optimization using gray wolf optimizer algorithm. *Quality and Reliability Engineering International*, *33*(7), 1327-1335.

Kumar, A., Pant, S., & Ram, M. (2018a). Complex system reliability analysis and optimization. In: Ram, M., Davim, J.P. (eds) *Advanced Mathematical Techniques in Science and Engineering*, River Publisher, pp.185-199.

Kumar, A., Pant, S., & Ram, M. (2019a). Gray wolf optimizer approach to the reliability-cost optimization of residual heat removal system of a nuclear power plant safety system. *Quality and Reliability Engineering International*, *35*(7), 2228-2239.

Kumar, A., Pant, S., & Singh, S.B. (2017b). Reliability optimization of complex systems using cuckoo search algorithm. In: Ram, M., Davim, J.P. (eds) *Mathematical Concepts and Applications in Mechanical Engineering and Mechatronics*. IGI Global, pp. 94-110.

Kumar, A., Pant, S., & Singh, S.B. (2017c). Availability and cost analysis of an engineering system involving subsystems in series configuration. *International Journal of Quality & Reliability Management*, *34*(6), 879-894.

Kumar, A., Pant, S., Ram, M., & Chaube, S. (2019b). Multi-objective grey wolf optimizer approach to the reliability-cost optimization of life support system in space capsule. *International Journal of System Assurance Engineering and Management*, *10*(2), 276-284.

Kumar, A., Pant, S., Ram, M., & Singh, S.B. (2017d). On solving complex reliability optimization problem using multi-objective particle swarm optimization. In: Ram, M., Davim, J.P. (eds) *Mathematics Applied to Engineering*. Elsevier, Amsterdam, pp. 115-131.

Mirjalili, S., Mirjalili, S.M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software*, *69*, 46-61.

Mirjalili, S., Saremi, S., Mirjalili, S.M., & Coelho, L.D.S. (2016). Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization. *Expert Systems with Applications*, *47*, 106-119.

Pant S., Kumar A., Ram M. (2017a). Reliability optimization: a particle swarm approach. In: Ram, M., Davim, J.P. (eds) *Advances in Reliability and System Engineering*. Springer, Cham, pp. 163-187.

Pant, S., & Singh, S.B. (2011). Particle swarm optimization to reliability optimization in complex system. In *2011 IEEE International Conference on Quality and Reliability* (pp. 211-215). IEEE. Bangkok, Thailand.

Pant, S., Anand, D., Kishor, A., & Singh, S.B. (2015a). A particle swarm algorithm for optimization of complex system reliability. *International Journal of Performability Engineering*, *11*(1), 33-42.

Pant, S., Kumar, A., Bhan, S., & Ram, M. (2017b). A modified particle swarm optimization algorithm for nonlinear optimization. *Nonlinear Studies*, *24*(1), 127-138.

Pant, S., Kumar, A., Kishor, A., Anand, D., & Singh, S.B. (2015b, September). Application of a multi-objective particle article swarm optimization technique to solve reliability optimization problem. In *2015 1st International Conference on Next Generation Computing Technologies (NGCT)* (pp. 1004-1007). IEEE. Dehradun, India.

Paszkowicz, W. (1996). Application of the smooth genetic algorithm for indexing powder patterns–tests for the orthorhombic system. In *Materials Science Forum* (Vol. 228, pp. 19-24). Trans Tech Publications Ltd. doi.org/10.4028/www.scientific.net/msf.

Ramírez-Rosado, I.J., & Bernal-Agustín, J.L. (2001). Reliability and costs optimization for distribution networks expansion using an evolutionary algorithm. *IEEE Transactions on Power Systems*, *16*(1), 111-118.

Shelokar, P.S., Jayaraman, V.K., & Kulkarni, B.D. (2002). Ant algorithm for single and multiobjective reliability optimization problems. *Quality and Reliability Engineering International*, *18*(6), 497-514.

Skinner, A.J., & Broughton, J.Q. (1995). Neural networks in computational materials science: training algorithms. *Modelling and Simulation in Materials Science and Engineering*, *3*(3), 371.

Swinehart, K., Yasin, M., & Guimaraes, E. (1996). Applying an analytical approach to shop–floor scheduling: a case study. *International Journal of Materials and Product Technology*, *11*(1-2), 98-107.

Wattanapongsakorn, N., & Levitan, S.P. (2004). Reliability optimization models for embedded systems with multiple applications. *IEEE Transactions on Reliability*, *53*(3), 406-416.

Zafiropoulos, E.P., & Dialynas, E.N. (2007). Methodology for the optimal component selection of electronic devices under reliability and cost constraints. *Quality and Reliability Engineering International*, *23*(8), 885-897.

Zitzler, E. (1999). *Evolutionary algorithms for multiobjective optimization: methods and applications*. Ph.D. Thesis, ETH Zurich, Switzerland.