BIO MIMICRY OF BACTERIAL FORAGING OPTIMIZATION ALGORITHM AS DE-NOISING FILTER

Thesis submitted in partial fulfillment of the requirements for the award of the degree of

Master in Electronics and Telecommunication Engineering
Jadavpur University

May 2016

By

SUSHMITA BANDYOPADHYAY

Examination Roll Number: M4ETC1607
Registration Number: 128922 of 2014 - 15

Under the guidance of

Dr. S.S. CHAUDHURI

Department Electronics and Telecommunication Engineering
Jadavpur University
Kolkata-700032

FACULTY OF ENGINEERING AND TECHNOLOGY
JADAVPUR UNIVERSITY

CERTIFICATE

This is to certify that the dissertation entitled “” has been carried out by **SUSHMITA BANDYOPADHYAY** (University Registration No: 128922 of 2014 - 15) under my guidance and supervision and be accepted in partial fulfillment of the requirement for the degree of **Master in Electronics and Telecommunication Engineering**. The research results presented in the thesis have not been included in any other paper submitted for the award of any degree to any other University or Institute.

Dr. S.S. CHAUDHURI
Supervisor
Electronics and Telecommunication Engineering
Jadavpur University

Dr. P. Venkateswaran
Head of the Department
Electronics and Telecommunication Engineering
Jadavpur University

Dr. Sivaji Bandyopadhyay
Dean , FET
Jadavpur University
Kolkata -700032

FACULTY OF ENGINEERING AND TECHNOLOGY
JADAVPUR UNIVERSITY

CERTIFICATE OF APPROVAL

The foregoing thesis is hereby approved as a creditable study of an engineering subject and presented in a manner satisfactory to warrant acceptance as pre-requisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn there in but approve the thesis only for which it is submitted.

-- ------------------------------------

Signature of the Examiner **Signature of the Guide**

FACULTY OF ENGINEERING AND TECHNOLOGY
JADAVPUR UNIVERSITY

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC THESIS

I hereby declare that this thesis titled "**BIO MIMICRY OF BACTERIAL FORAGING OPTIMIZATION ALGORITHM AS DE-NOISING FILTER"** contains literature survey and original research work as part of my Degree of Master of Electronics and Tele Communication.

All information have been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

Name: Sushmita Bandyopadhyay
Examination Roll No: M4ETC1607

Thesis Title: "**BIO MIMICRY OF BACTERIAL FORAGING OPTIMIZATION ALGORITHM AS DE-NOISING FILTER"**

Date:
Place: Kolkata

Signature of the candidate

ACKNOWLEDGEMENT

I take this opportunity to express my profound gratitude and deep regards to my guide **AssociateProfessor Dr.Sheli Sinha Chaudhuri** for her exemplary guidance, monitoring and constant encouragement throughout the work of this project thesis. The help and guidance given by her time to time shall carry me a long way in the journey of life on which I am about to embark.

I would also like to express my special thanks of gratitude to **Prof. Amit Konar** and **HOD (ETCE) Prof. (Dr.) P. Venkateswaran** who gave me the golden opportunity to do this wonderful project.

I am very much thankful to Rimita Lahiri, Mainak Dan, Prosenjit Kumar Mudi, Ranodip Das,Kakali Mudi and Soumik Bhattachaya for being a part of my project by sharing their views, information and suggestions.

The journey would not have possible without the support of the management and all teaching and non-teaching stuffs of Electronics Dept., MCKV Institute of Engineering (MCKVIE), Specially Mr. K.K.Kejriwal (ManagingTrustee- MCKVIE), Dr.Parasar Bandyopadhyay(Director -MCKVIE), Dr.Ashok Kumar Principal-MCKVIE), Associatet professor Dr. Satadal Saha (HOD ECE dept. – MCKVIE) and Sulagna Chakraborty (Lab-instructor- MCKVIE).

I would like to thanks my parents, my family and friends who helped me a lot in finalizing this project report within the limited time frame.

Date:

Place: Kolkata

Sushmita Bandyopadhyay

Exam Roll No: M4ETC1607
Registration Number: 128922 of 2014 - 15

Contents

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1. INTRODUCTION

Nature is the mother of all creatures. It is a great and immense source of inspiration for solving hard and complex problems. What we learn from nature that we try to implement into our problems to solve them. Biological process is an improvement step that strives towards optimization. It always finds the best solution with respect to the objective while maintaining the perfect balance with other components. This idea leads to the inspiration for computing or engineering improvement. Bio-inspired algorithm is a meta heuristics method that truly copies the nature in order to solve optimization problem. Numerous studies had been carried out extensively from the past few decades which resulted in very promising findings. But still the area is not yet explored much. Thus lead to exploration into new areas of application and opportunity. Here we are going to analyze some popular bio-inspired optimization method systematically grouped by the biological field that inspired each and the areas where these algorithms have been most successfully applied and also focused on the principle of each algorithm and their application.

Optimization is a commonly encountered mathematical problem in all engineering disciplines. It literally means finding the best possible/desirable solution. Optimization problems are wide ranging and numerous, hence methods for solving these problems ought to be, an active research topic. Optimization algorithms can be either deterministic or stochastic in nature. Previous methods to solve optimization problems require massive computational efforts, which tend to fail as the problem size increases. This is the motivation for implementing bio inspired stochastic optimization algorithms as computationally efficient alternatives to deterministic approach. Meta-heuristics are based on the iterative improvement of either a population of solutions (as in Evolutionary algorithms, Swarm based algorithms) or a single solution (eg. Tabu Search) and mostly employ randomization and local search to solve a given optimization problem.

1.1.Towards technology through Nature

The real beauty of nature inspired algorithms lies in the fact that it receives its lone inspiration from nature. They have the ability to describe and resolve complex relationships from intrinsically very simple initial conditions and rules with little or no knowledge of the search space Nature is the perfect example for optimization, because if we closely examine each and every features or incident in nature it always find the optimal strategy, still addressing complex interaction among organisms ranging from microorganism to fully fledged human beings, balancing the ecosystem, maintaining diversity, adaptation, physical phenomenon like river formation, forest fire ,cloud, rain .etc..Even though the strategy behind the solution is simple the results are amazing. Nature is the best teacher and its designs and capabilities are extremely enormous and mysterious that researchers are trying to imitate nature in technology. Also the two fields have a much stronger connection since, it seems entirely reasonable that new or constant problems in computer science could have a lot in common with problems nature has encountered and resolved long ago. Thus an easy mapping is possible between nature and technology. Bio inspired computing has come up as a new era in computing encompassing a wide range of applications, covering all most all areas including computer networks, security, robotics, bio medical engineering, control systems ,parallel processing ,data mining, power systems, production engineering and many more. Classical problem solving methodologies involve two branches: Exact methods (logical, mathematical programming) and Heuristics. Heuristic approach seems to be superior in solving hard and complex optimization problems, particularly where the traditional methods fail. BIAs are such heuristics that mimics /imitate the strategy of nature since many biological processes can be thought of as processes of constrained optimization. They make use of many random decisions which classifies them as a special class of randomized algorithms. Formulating a design for bio-inspired algorithms involves choosing a proper representation of problem, evaluating the quality of solution using a fitness function and defining operators so as to produce a new set of solutions. A vast literature exists on bio inspired approaches for solving an impressive array of problems and, more recently, a number of studies have reported on the success of such techniques for solving difficult problems in all key areas of computer science. The two most predominant and successful classes or directions in BIAs involves Evolutionary Algorithms and Swarm based Algorithms which are inspired by the natural evolution and collective behavior in animals respectively. But still, this has been further to enhance a broader view over the domain refined so as to classify the algorithms based on the area of inspiration from nature.

1.2.Bacterial Foraging Optimization Algorithm

To tackle several complex search problems of real world, scientists have been looking into the nature for years-both as model and as metaphor-for inspiration. Optimization is at the heart of many natural processes like Darwinian evolution, group behavior of social insects and the foraging strategy of other microbial creatures. Natural selection tends to eliminate species with poor foraging strategies and favor the propagation of genes of species with successful foraging behavior, as they are more likely to enjoy reproductive success.

Since a foraging organism or animal takes necessary action to maximize the energy utilized per unit time spent for foraging, considering all the constraints presented by its own physiology such as sensing and cognitive capabilities, environment, the natural foraging strategy can lead to optimization and essentially this idea can be applied to real-world optimization problems. Based on this conception, Passino proposed an optimization technique known as Bacterial Foraging Optimization Algorithm (BFOA). Until date, the algorithm has successfully been applied to real world problems like optimal controller design, harmonic estimation, transmission loss reduction, pattern recognition, controller synthesis for active power filters and power system optimization.

BFOA is a newly added member in the coveted realm of Swarm Intelligence, which also includes powerful optimization techniques like the Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO). On the algorithmic front, several researchers extended the basic BFOA to deal with complex and multi-modal fitness landscapes, dynamical environments and to obtain efficient convergence behavior [18–23]. BFOA has also been hybridized with a few other state-of-the-art evolutionary computing techniques in order to achieve robust and efficient search performances.

An interesting characteristic feature of BFOA is that it has its own local search mechanism through the computational chemotaxis step and reproduction with elimination-dispersion helps in global search. Over certain real-world optimization problems, BFOA has been reported to outperform many powerful optimization algorithms like GA, PSO, etc. in terms of convergence speed and final accuracy, for example. As pointed out by Das et al, unlike PSO and Differential Evolution (DE), the uniqueness of the stability criteria of BFOA remains in the fact that in order to ensure stability of the chemotactic dynamics in BFOA, the step-size parameter must be adjusted (i.e. made adaptive) according to the current location of the bacterium and its current fitness. The efficiency of the algorithm in solving real parameter optimization problems has made it a potential optimization algorithm, worth investing research time these days.

De-noising of image still a concerned for researchers working in this area. It is further challenging in case of medical images mainly images of the internal organs. Various digital filters have been developed and tried by researchers to provide ideal solution in the de-noising of medical images. In the present thesis the authors present a Soft Computing approach to de-noise the images. Bacterial Foraging Optimisation which is a bio-inspired algorithm is used as filter to de-noise images. The performance metrics like MSE and PSNR are calculated which show that Bacterial Foraging Optimisation can act as potential tool for de-noising images.

1.3.Layout of the Work

The thesis is organised in following chapters.
Chapter 1 is the Introduction of the thesis.
Chapter 2 contains an over view of Bio Inspired Optimization Algorithms.
Chapter 3 provides us a brief theoretical idea about Bacterial Foraging Optimization Algorithm.
Chapter 4 explains the proposed idea about BFO Soft Filter for noise removal, the work carried out and the subsequent result observed.
And Finally, Chapter 5 encompasses the Conclusion.

CHAPTER 2

THE BIO- INSPIRED ALGORITHMS

2. Bio-inspired Algorithms

The nature inspired algorithms are flexible and work in changing environment to organize and grow accordingly. These algorithms quality perform in highly complex problems and can even show very good results when problem is not properly defined. These tend to find the best available solution in every changed environment and very good decision maker. Scalability is not a challenge. But in another side these Nature inspired systems are very hard to design as algorithms are inspired from nature and understanding nature fully is complex. Nature inspired systems do not adapt to real world system fully in terms of scalability and performance. Systems can work well in some domain but not in other. As systems are nature inspired, then not having proper knowledge of nature can affect the design of algorithm. So for effective results no ambiguity in data is required. These algorithms have the ability to self learn, self train, self organize and self grow. They can find best optimal solutions to complex problems using simple conditions and rules of nature. The scope in this field is very vast. This field is largely unexplored and therefore no limit on development. This section gives a complete insight on various nature inspired algorithms based on evolutionary computation and swarm intelligence.

2.1. The Taxonomy

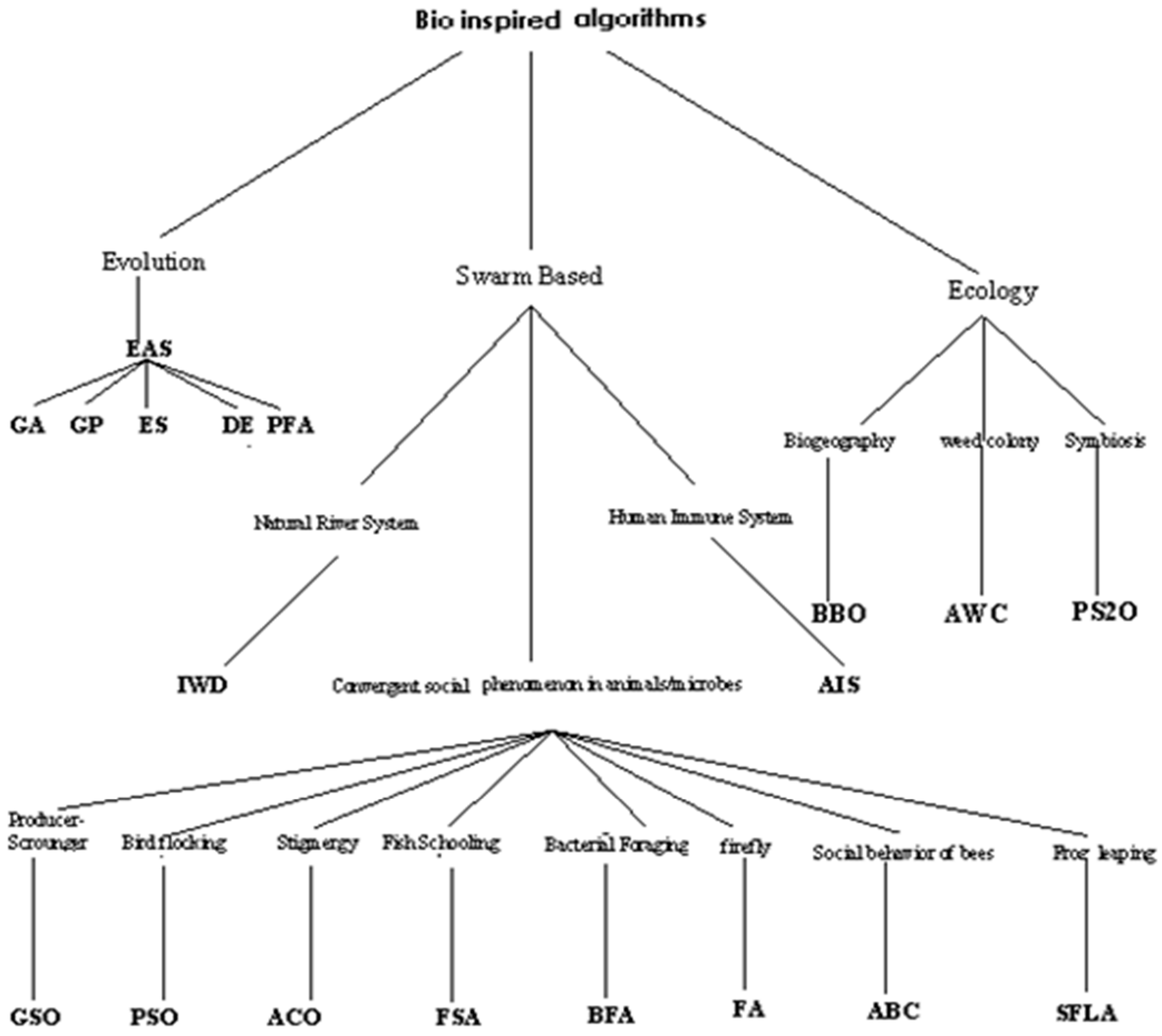


Figure 1
Taxonomy

2.2 Evolutionary Algorithms

Evolutionary computation (EC)[1] is a paradigm in the artificial intelligence territory that aims at benefiting from collective phenomena in adaptive populations of problem solvers utilizing the iterative progress comprising growth ,development, reproduction, selection, and survival as seen in a population . EAs are the most well known, classical and established algorithms among nature inspired algorithms, which is based on the biological evolution n in nature that is being responsible for the design of all living beings on earth, and for the strategies they use to interact with each other. Eas employ this powerful design philosophy to find solutions to hard problems. Eas are non-deterministic algorithms or cost based optimization algorithms.

A family of successful Eas comprises genetic algorithm (GA), genetic programming (GP), Differential Evolution, evolutionary strategy (ES) and most recent Paddy Field Algorithm. The members of the EA family share a great number of features in common. They are all population-based stochastic search algorithms performing with best-to-survive criteria. Each algorithm commences by creating an initial population of feasible solutions, and evolves iteratively from generation to generation towards a best solution. In successive iterations of the algorithm, fitness-based selection takes place within the population of solutions. Better solutions are preferentially selected for survival into the next generation of solutions.

2.2.1. Genetic Algorithm

GA is an evolutionary based stochastic optimization algorithm with a global search potential proposed by Holland in 1975[2]. GAs are among the most successful class of algorithms under Eas which are inspired by the evolutionary ideas of natural selection. They follow the principles of Charles Darwin Theory of survival of the fittest. However, because of its outstanding performance in optimization, GA has been regarded as a function optimizer. Algorithm begins by initializing a population of solution (chromosome). It comprises representation of the problem usually in the form of a bit vector. Then for each chromosome evaluate the fitness using an appropriate fitness function suitable for the problem .Based on this ,the best chromosomes are selected into the mating pool, where they undergo cross over and mutation thus giving new set of solutions(offspring).

The three principal genetic operators in GA involve selection, crossover, and mutation. GA is useful and efficient when:

* The search space is large complex or poorly known.
* No mathematical analysis is available.
* Domain knowledge is scarce to encode to narrow the search space
* For complex or loosely defined problems since it works by its own internal rules.
* Traditional search method fails

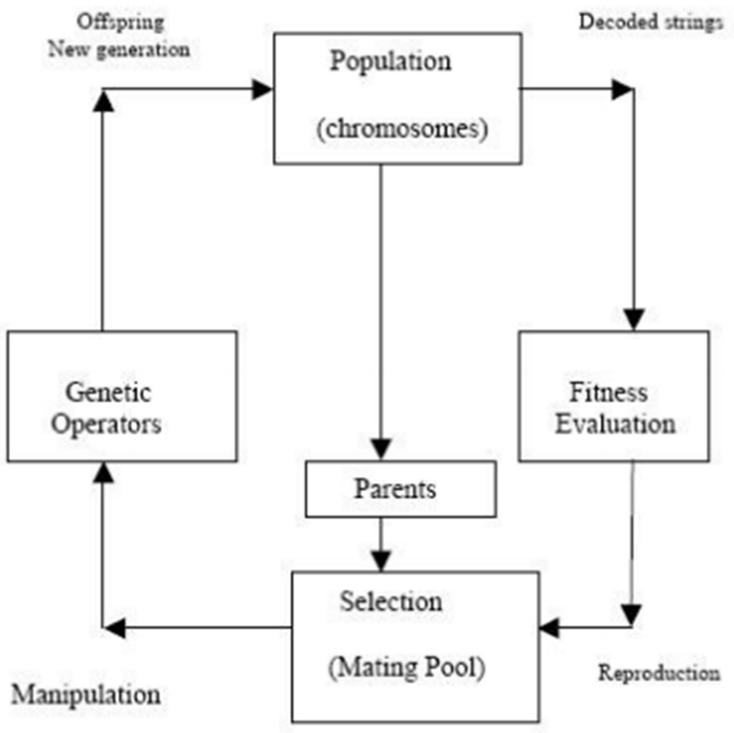


Figure 2
Flow Diagram of GA

Even though GAs can rapidly locate good solutions, for difficult search spaces, it has some disadvantages:
1) GA may have a tendency to converge towards local optima rather than the global optimum of the problem if the fitness function is not defined properly.
2) Operating on dynamic data sets is difficult.
3) For specific optimization problems, and given the same amount of computation time, simpler optimization algorithms may find better solutions than GAs.
4) GAs are not directly suitable for solving constraint optimization problems

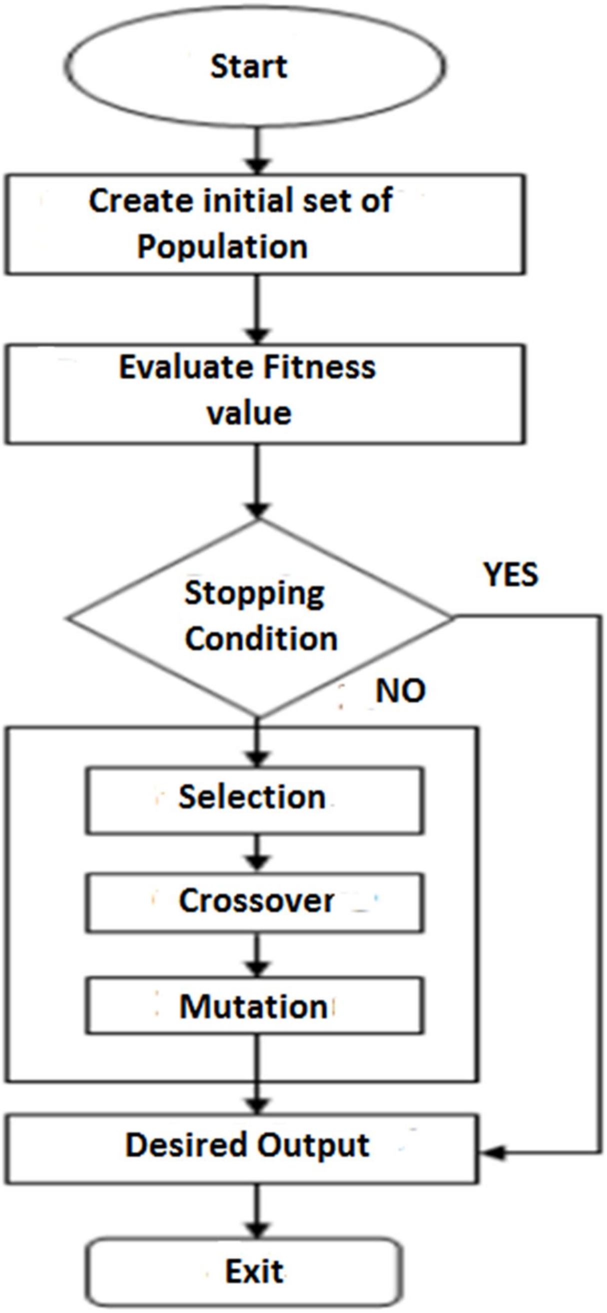


Figure 3
Flow Chart for Genetic Algorithm

2.2.2. Genetic Programming

Proposed by Koza in 1992[3],GP being an extension to Genetic algorithms differs from the latter in terms of representation of the solution.GP represent an indirect encoding of a potential solution (in the form of a tree),in which search is applied to the solution directly , and a solution could be a computer program. The second fundamental difference is in the variable-length representation adopted by GP in contrast with the fixed length encoding in GA. The population in GP generates diversity not only in the values of the genes but also in the structure of the individuals.

Hence GP resembles evolution of a population of computer programs. The four steps in Genetic programming involve:

1)Generate an initial population of computer programs comprising the functions and terminals.

2) Execute each program in the population and assign it a fitness value according to how well it solves the problem.

3) Create a new population of computer programs.

i) Copy the best existing programs

ii) Create new computer programs by mutation

iii) Create new computer programs by crossover (sexual reproduction).

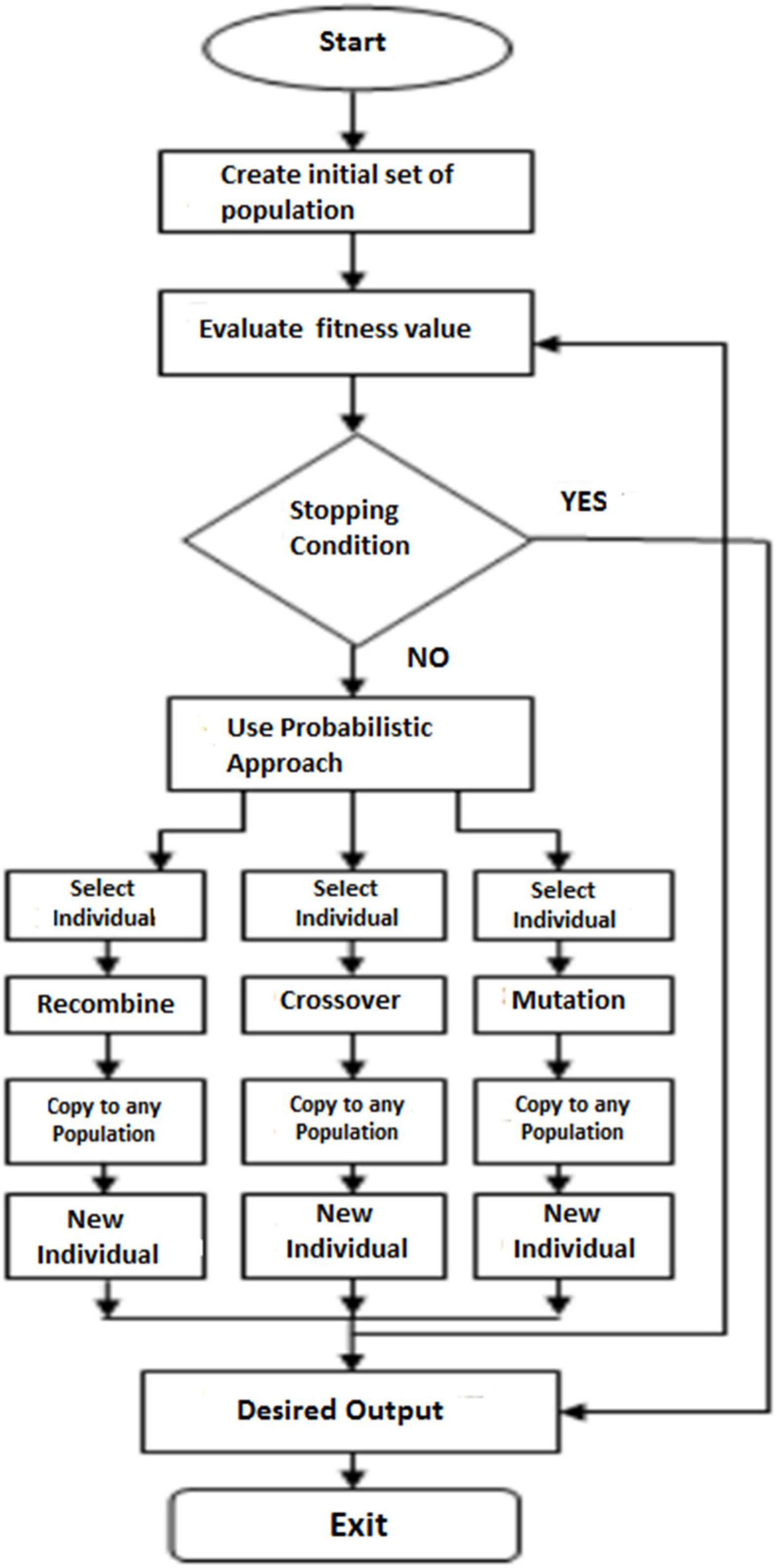


Figure 4
Flow chart for Genetic Programming

2.2.3. Evolution Strategies

Evolution Strategies was developed by three students (Bienert, Rechenberg, Schwefel) at the Technical University in Berlin in 1964[4] in an effort to robotically optimize an aerodynamic design problem. Evolution Strategies is a global optimization algorithm inspired by the theory of adaptation and evolution by means of natural selection. Specifically, the technique is inspired by macro-level or the species-level process of evolution (phenotype, hereditary, variation) unlike genetic algorithm which deals with micro or genomic level (genome, chromosomes, genes, alleles). A very important feature of ES is the utilization of self-adaptive mechanisms for controlling the application of mutation. These mechanisms are aimed at optimizing the progress of the search by evolving not only the solutions for the problem being considered, but also some parameters for mutating these solutions. Some common Selection and Sampling schemes in ES are as follows:

(1+1)-ES:

This is a simple selection mechanism in which works by creating one real-valued vector of object variables from its parent by applying mutation with an identical standard deviation to each object variable. Then, the resulting individual is evaluated and compared to its parent, and the better survives to become a parent of the next generation, while the other is discarded.

(μ +λ)-ES:

Here μ parents are selected from the current generation and generate λ offspring, through some recombination and /or mutation operators. Out of the union of parents and offspring (μ + λ), the best μ kept for next generation. It inherently incorporates elitism.

(μ, λ)-ES:

Currently used variant is (μ , λ)-ES .Here μ parents selected from the current generation and used to generate λ offspring (with λ >= μ) and only the best μ offspring individuals form the next generation discarding the parents completely. This does not incorporate elitism.

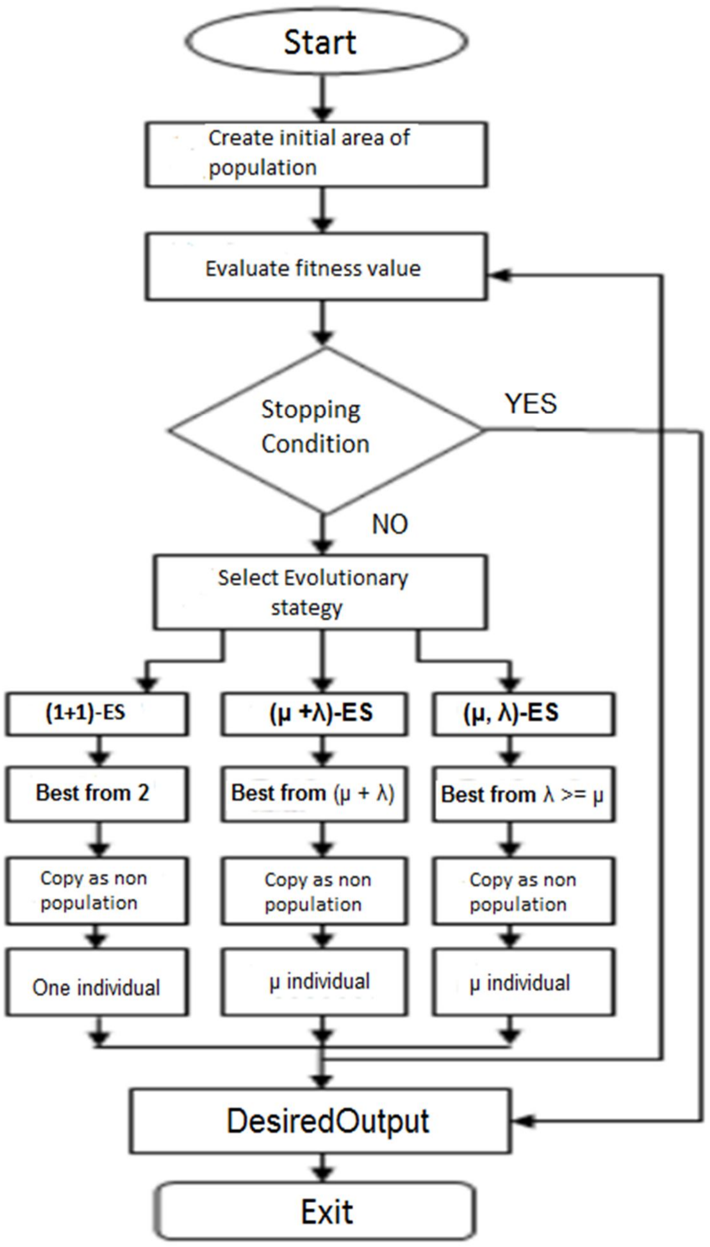


Figure 5
Flow Chart for Evolution Strategies

2.2.4. Differential Evolution

Another paradigm in EA family is differential evolution (DE) proposed by Storn and Price in 1995[5]. DE is similar to GAs since populations of individuals are used to search for an optimal solution. The main difference between Gas and DE is that, in GAs, mutation is the result of small perturbations to the genes of an individual while in DE mutation is the result of arithmetic combinations of individuals. At the beginning of the evolution process, the mutation operator of DE favors exploration. As evolution progresses, the mutation operator favors exploitation. Hence, DE automatically adapts the mutation increments to the best value based on the stage of the evolutionary process. Mutation in DE is therefore not based on a predefined probability density function.

Advantages:

*DE is easy to implement, requires little parameter tuning

* Exhibits fast convergence

* It is generally considered as a reliable, accurate, robust and fast optimization technique.

Limitations:

* According to Krink et al. (2004), noise may adversely affect the performance of DE due to its greedy nature.

*Also the user has to find the best values for the problem-dependent control parameters used in DE and this is a time consuming task.
A self-adaptive DE (SDE) algorithm can eliminates the need for manual tuning of control parameters

2.2.5. Paddy Field Algorithm

Recent algorithm Proposed by **Premaratne et al** in 2009 [6], which operates on a reproductive principle dependant on proximity to the global solution and population density similar to plant populations .Unlike evolutionary algorithms ,it does not involve combined behavior nor crossover between individuals instead it uses pollination and dispersal.PFA constitutes five basic steps.

Sowing:

The algorithm operates by initially scattering seeds (initial population p0) at random in an uneven field.

Selection:

Here the best plants are selected based on a threshold method so as to selectively weed out unfavorable solutions and also controls the population.

Seeding:

In this stage each plant develops a number of seeds proportional to its health. The seeds that drop into the most favorable places (most fertile soil, best drainage, soil moisture etc.) tend to grow to be the best plants (taller) and produce more number of seeds. The highest plant of the population would correspond to the location of the optimum conditions and the plant‘s fitness is determined by a fitness function.

Pollination:

For seed propagation pollination is a major factor either via animals or through wind. High population density would increase the chance of pollination for pollen carried by the wind

Dispersion:

In order to prevent getting stuck in local minima, the seeds of each plant are dispersed .Depending on the status of the land it will grow into new plants and continue the cycle.

As per no free lunch rule, the PFA only has a lower computational cost. Since the PFA doesn‘t have crossover, the optimum solution can be migrated to reach the optimum solution.

2.3. Swarm Intelligence

Swarm Intelligence (Kennedy and Eberhart, 2001[7]) is a recent and emerging paradigm in bio inspired computing for implementing adaptive systems. In this sense, it is an extension of EC. While Eas are based on **genetic adaptation** of organisms SI is based on collective **social behavior** of organisms. As per definitions in literature, Swarm Intelligent encompasses the implementation of collective intelligence of groups of simple agents that are based on the behavior of real world insect swarms, as a problem solving tool. The word —swarm comes from the irregular movements of the particles in the problem space.SI has been developed alongside with Eas. Some most well-known strategies in this area are discussed below. These trajectory tracking algorithms being inspired by the collective behavior of animals, exhibit decentralized, self-organized patterns in the foraging process.

2.3.1 Swarm Intelligence Principles*:*

SI can be described by considering five fundamental principles.

Principle: the population should be able to carry out simple space and time computations.

Quality Principle: the population should be able to respond to quality factors in the environment.

Diverse Response Principle**:** the population should not commit its activity along excessively narrow channels.

Stability Principle**:** the population should not change its mode of behavior every time the environment changes.

Adaptability Principle: the population should be able to change its behavior mode when it is worth the computational price.

2.3.1.1 Particle Swarm Optimization

Particle swarm optimization (PSO) is a computational intelligence oriented, stochastic, population-based global optimization technique proposed by Kennedy and Eberhart in 1995[8]. It is inspired by the social behavior of bird flocking searching for food. PSO has been extensively applied to many engineering optimization areas due to its unique searching mechanism, simple concept, computational efficiency, and easy implementation. In PSO, the term ―particles refers to population members which are mass-less and volume-less (or with an arbitrarily small mass or volume) and are subject to velocities and accelerations towards a better mode of behavior. Each particle in the swarm represents a solution in a high-dimensional space with four vectors, its current position, best position found so far, the best position found by its neighborhood so far and its velocity and adjusts its position in the search space based on the best position reached by itself (pbest) and on the best position reached by its neighborhood (gbest) during the search process. In each iteration, each particle updates its position and velocity as follows: where**,** represents Particle position represents Particle velocity represents Best "remembered" position represents cognitive and social parameters, are random numbers between 0 and 1

Steps in PSO algorithm can be briefed as below:

1) Initialize the swarm by assigning a random position in the problem space to each particle.
2) Evaluate the fitness function for each particle.
3) For each individual particle, compare the particle‘s fitness value with its pbest . If the current value is better than the pbest value, then set this value as the pbest and the current particle‘s position, xi, as pi.
4) Identify the particle that has the best fitness value. The value of its fitness function is identified as guest and its position as pg.
5) Update the velocities and positions of all the particles using (1) and (2).
6) Repeat steps 2–5 until a stopping criterion is met (e.g., maximum number of iterations or a sufficiently good fitness value).

Advantages over Genetic Algorithm:
(a) PSO is easier to implement and there are fewer parameters to adjust.
(b) PSO has a more effective memory capability than GA.
© PSO is more efficient in maintaining the diversity of the swarm, since all the particles use the information related to the most successful particle in order to improve themselves, whereas in Genetic algorithm, the worse solutions are discarded and only the new ones are saved; i.e. in GA the population evolve around a subset of the best individuals. There are many similarities between the PSO and Eas. Both of them initialize solutions and update generations, while the PSO has no evolution operators as does the latter. In a PSO, particles try to reach the optimum by following the current global optimum instead of using evolutionary operators, such as mutation and crossover. It is claimed that the PSO, in addition to continuous functions, has been showing stability and convergence in a multidimensional complex space also. (Clerc and Kennedy, 2002).

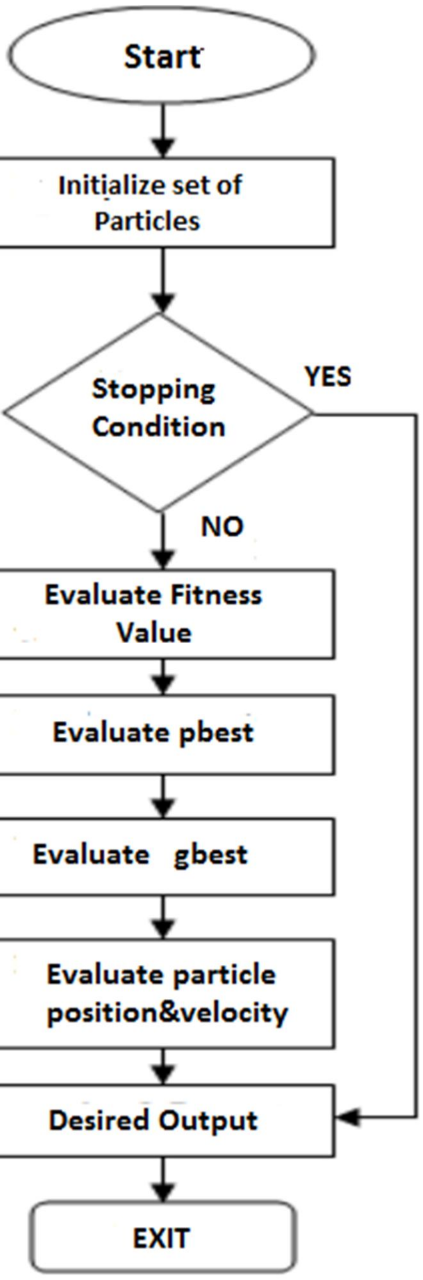


Figure 6
Flow Chart for Particle Swarm Optimization

2.3.1.2. Ant Colony Optimization

ACO is among the most successful swarm based algorithms proposed by Dorigo& Di Caro in 1999 [9] .It is a meta heuristic inspired by the foraging behavior of ants in the wild, and moreover, the phenomena known as stigmergy, term introduced by Grasse in 1959.Stigmergy refers to the indirect communication amongst a self-organizing emergent system via individuals modifying their local environment. The most interesting aspect of the collaborative behavior of several ant species is their ability to find shortest paths between the ants' nest and the food sources by tracing pheromone trails Then, ants choose the path to follow by a probabilistic decision biased by the amount of pheromone: the stronger the pheromone trail, the higher its desirability. Because ants in turn deposit pheromone on the path they are following, this behavior results in a *self-reinforcing process* leading to the formation of paths marked by high pheromone concentration. By modeling and simulating ant foraging behavior, brood sorting, nest building and self-assembling, etc. algorithms can be developed that could be used for complex, combinatorial optimization problems.

The first ant algorithm, named __Ant System (AS), was developed in the nineties by Dorigo et al. (1996) and tested successfully on the well known benchmark Travelling Salesman Problem. The ACO meta heuristic was developed (Dorigo& Di Caro, 1999;) to generalize, the overall method of solving combinatorial problems by approximate solutions based on the generic behavior of natural ants.

ACO is structured into **three** main functions as follows:

AntSolutionsConstruct: This function performs the solution construction process where the artificial ants move through adjacent states of a problem according to a transition rule, iteratively building solutions.

Pheromone Update: performs pheromone trail updates. This may involve updating the pheromone trails once complete solutions have been built, or updating after each iteration. In addition to pheromone trail reinforcement, ACO also includes pheromone trail evaporation. Evaporation of the pheromone trials helps ants to forget bad solutions that were learned early in the algorithm run.

DeamonActions: is an optional step in the algorithm which involves applying additional updates from a global perspective (for this no natural counterpart exists). This may include applying additional pheromone reinforcement to the best solution generated (known as offline pheromone trail update). An alternative approach, called the ant colony system (ACS) has been introduced by Dorigo and Gambardella (1997) to improve the performance of ant system. It is based on four modifications of ant system: a different transition rule, a different pheromone trail update rule, the use of local updates of pheromone trail to favor exploration, and the use of candidate list to restrict the choice.

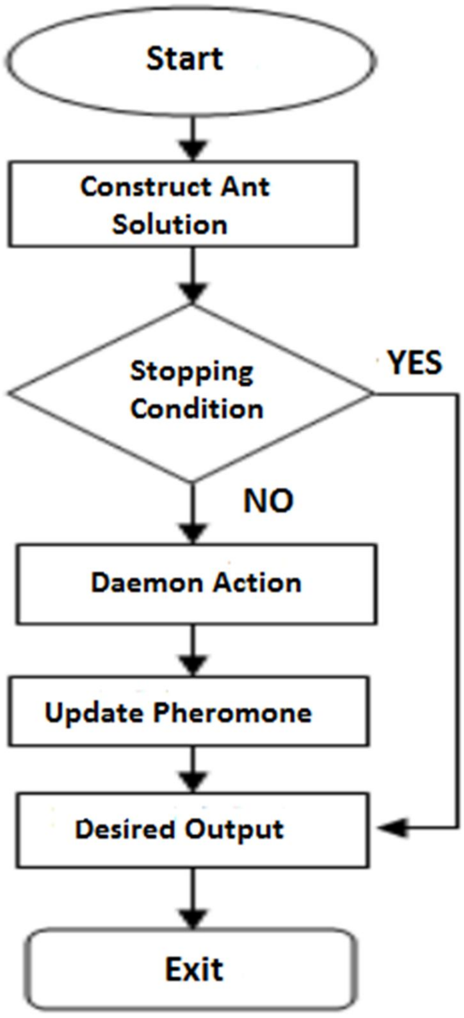


Figure 7
Flow chart for Ant Colony Optimization

2.3.1.3. Artificial Bee Colony Algorithm (ABC)

Based on the behavior of the bees in nature, various swarm intelligence algorithms are available. These algorithms are classified into two; foraging behavior and mating behavior. Examples of algorithms simulating the foraging behavior of the bees include the Artificial Bee Colony (ABC) ,the Virtual Bee algorithm proposed by Yang , the Bee Colony Optimization algorithm proposed by Teodorovic and Dell'Orco , the BeeHive algorithm proposed by Wedde et al., the Bee Swarm Optimization algorithm proposed by Drias et al. and the Bees algorithm proposed by Pham et al. An individual entity (e.g., a bee in a bee colony) exhibit a simple set of behavior policies (e.g., migration, replication, death), but a group of entities (e.g., a bee colony) shows complex emergent behavior with useful properties such as scalability and adaptability. Artificial Bee Colony is a predominant algorithm simulating the intelligent foraging behavior of a honeybee swarm, proposed by Karaboga and Basturk [10].In ABC algorithm, the colony of artificial bees contains three groups of bees: *employed bees, onlookers and scouts.*

A bee waiting on the dance area for making a decision to choose a food source is called onlooker and one going to the food source visited by it before is named employed bee. The other kind of bee is scout bee that carries out random search for discovering new sources. The position of a food source represents a possible solution to the optimization problem and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution. A swarm of virtual bees is generated and started to move randomly in two-dimensional search space. Bees interact when they find some target nectar and the solution of the problem is obtained from the intensity of these bee interactions. A randomly distributed initial population solutions (xi=1,2…D) is being dispread over the D dimensional problem space. An employed bee produces a modification on the position (solution) in her memory depending on the local information (visual information) and tests the nectar amount (fitness value) of the new source (new solution). Provided that the nectar amount of the new one is higher than that of the previous one, the bee memorizes the new position and forgets the old one. After all employed bees complete the search process; they share the nectar information of the food sources and their position information with the onlooker bees on the dance area. In the next phase **Reproduction**, based on the probability value associated with the food source, **Pi**,the artificial onlooker bee chooses a food source Where, N is the number of food sources (that is the number of employed bees), *fit* © is the fitness value of the solution © which is proportional to the nectar amount of the food source in the position i. In the last phase, **Replacement of bee and Selection**, *if* a position can not be improved further through a predetermined number of cycles, then that food source is assumed to be abandoned. The value of predetermined number of cycles is an important control parameter of the ABC algorithm, which is called —*limit* for abandonment. After each candidate source position is produced and then evaluated by the artificial bee, its performance is compared with that of its old one. If the new food has an equal or better nectar than the old source, it is replaces the old one in the memory. Otherwise, the old one is retained in the memory. The local search performance of ABC algorithm depends on neighborhood search and greedy selection mechanisms performed by employed and onlooker bees. The global search performance of the algorithm depends on random search process performed by scouts and neighbor solution production mechanism performed by employed and onlooker bees

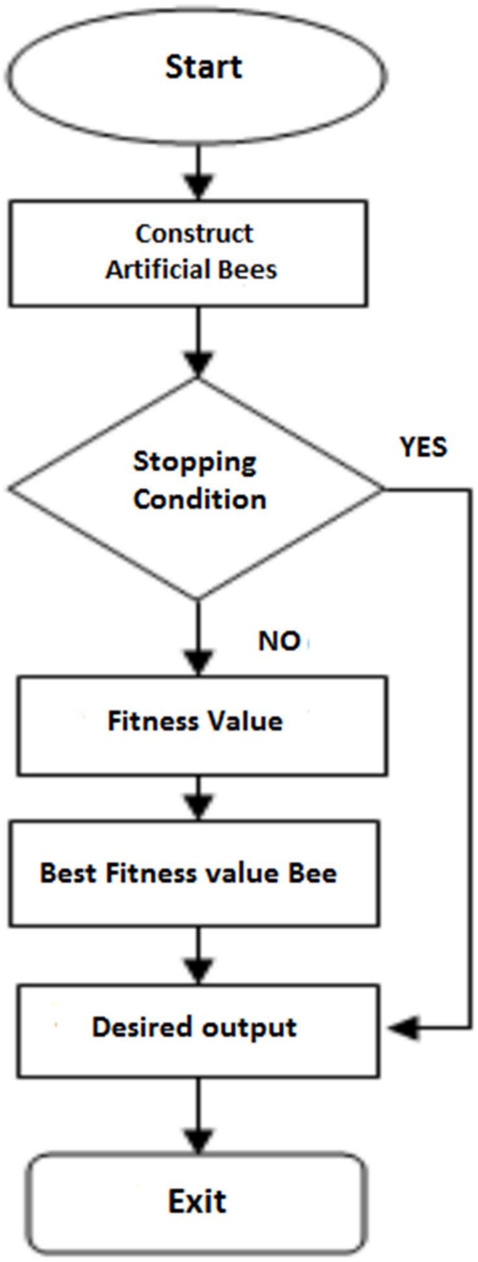


Figure 8
Flow Chart for Artificial Bee Colony Algorithm

2.3.1.4. Fish Swarm Algorithm

The fish swarm algorithm (FSA) is a new population-based/swarm intelligent evolutionary computation technique proposed by Li et al. [11] in 2002 which is inspired by the natural schooling behavior of fish.FSA presents a strong ability to avoid local minimums in order to achieve global optimization. A fish is represented by its D-dimensional position Xi = (x1, x2, . . .,xk, . . ., xD), and food satisfaction for the fish is represented as Fsi. The relationship between two fish is denoted by their Euclidean distance dij = ||Xi –Xj||. FSA imitates three typical behaviors, defined as ―searching for food, ―swarming in response to a threat, and ―following to increase the chance of achieving a successful result.

Searching is a random search adopted by fish in search of food, with a tendency towards food concentration. The objective is to minimize FS (food satisfaction).

Swarming: aims in satisfying food intake needs, entertaining swarm members and attracting new swarm members. A fish located at Xi has neighbors within its visual. Xc identifies the center position of those neighbors and is used to describe the attributes of the entire neighboring swarm. If the swarm center has greater concentration of food than is available at the fish‘s current position Xi (i.e., FSc<Fsi), and if the swarm (Xc) is not overly crowded (ns/n < δ), the fish will move from Xi to next Xi+1, toward Xc .

Following behavior implies when a fish locates food, neighboring individuals follow. Within a fish‘s visual, certain fish will be perceived as finding a greater amount of food than others, and this fish will naturally try to follow the best one(Xmin) in order to increase satisfaction(i.e., gain relatively more food[Fsmin<Fsi] and less crowding[nf/n < δ]). Nf represents number of fish within the visual of Xmin. Three major

Parameters involved in FSA include visual distance (visual), maximum step length (step), and a crowd factor. FSA effectiveness seems primarily influenced by the former two (visual and step).

2.3.1.5. Bacterial Foraging Optimization Algorithm

This has been evolving as a new and promising branch in Bio inspired Algorithms that can bridge the gap between microbiology and engineering. These classes of algorithms inherit the characteristics of bacterial foraging patterns such as chemo taxis, metabolism, reproduction and quorum sensing. The complex and organized activities exhibited in bacterial foraging patterns inspire a new approach to solve complex optimization problems. The Bacterial Foraging Optimization Algorithm (BF0) was introduced by Passino in 2002[13].

Foraging is a social phenomenon of a bacterial colony rather than an individual behavior. BFOA consists of three principal mechanisms namely, chemo taxis, reproduction, and elimination-dispersal.

Chemotaxis(cell movement) is the activity of bacteria gathering to nutrient-rich areas in a spontaneous fashion; in this context, a cell-to-cell communication mechanism is established to simulate the biological behavior of bacterial movement (swim/tumble).

Reproduction comes from the concept of natural selection; under this procedure, only the best adapted bacteria tend to survive and transmit their genetic characters to succeeding generations, while the less adapted ones tend to perish.

Elimination-dispersal events randomly select parts of the bacteria population to diminish and disperse into random positions in the environment; this way the algorithm ensures the diversity of the species, and prevents getting trapped to local optima, improving global search ability.

2.3.1.6. Firefly algorithm

Firefly algorithm proposed by Yang [15] can be considered as an unconventional swarm-based heuristic algorithm for constrained optimization tasks inspired by the flashing behavior of fireflies. The algorithm constitutes a population-based iterative procedure with numerous agents (perceived as fire flies) concurrently solving a considered optimization problem. Agents communicate with each other via bioluminescent glowing which enables them to explore cost function space more effectively than in standard distributed random search. Intelligence optimization technique is based on the assumption that solution of an optimization problem can be perceived as agent (fire fly) which glows proportionally to its quality in a considered problem setting. Consequently each brighter fire fly attracts its partners (regardless of their sex), which makes the search space being explored more efficiently. The firefly algorithm has three particular idealized rules which are based on some of the basic flashing characteristics of real fireflies.

They are the following:

1) All fireflies are unisex and they will move towards more attractive and brighter ones regardless of their sex.

2) The degree of attractiveness of a firefly is proportional to its brightness. Also the brightness may decrease as the distance from the other fire flies increases due to the fact that the air absorbs light. If there is not a brighter or more attractive fire fly than a particular one it will then move randomly.

3) The brightness or light intensity of a fire fly is determined by the value of the objective function of a given problem.

Advantage:

Mainly uses real random numbers and is based on the global communication among the swarm particles (ie the firefly), hence more effective in multi objective optimization.

2.3.1.7. Group search optimizer

The group search optimizer (GSO), was investigated at the University of Liverpool (He et al., 2006)[16].It is a population based optimization algorithm, which adopts the producer–scrounger (PS) model metaphorically for designing optimum searching strategies, inspired by animal foraging behavior. Similar to the PSO, the population of the GSO is called a **group** and each individual in the population is called a **member**. In the search space, each member knows its own position, its head angle and a head direction, which can be calculated from the head angle via a polar to Cartesian co-ordinate transformation.

A group constitutes **three types of members**: producers, scroungers and rangers.

Producers**:** perform producing strategies, searching for food.

Scroungers**:** perform scrounging strategies, joining resources uncovered by others.

Rangers**:** perform random walk motions and will be dispersed from their current positions. The producer will find the best point with the best resource (fitness value). If the best point has a better fitness than its current position (in minimization problem as an example), then it will fly to this point. Or it will stay in its current position and turn its head to a new randomly generated angle. If the producer cannot find a better area after a number of iterations, it will turn its head back to zero degree. For scroungers, area copying is adopted, which is the commonest scrounging behavior in sparrows .Random walks are employed by the rangers. If a scrounger (or ranger) finds a better location than the current producer and other scroungers, in the next searching iteration it will switch to be a producer, and all the other members, including the producer in the previous searching iteration, will perform scrounging strategies. It is also assumed that the producer and the scroungers do not differ in their relevant phenotypic characteristics. Therefore, they can switch between the two roles.

2.3.1.8. Shuffled frog Leaping Algorithm

Proposed by Muzaffar Eusuff and Kevin Lansey in 2003[17].Shuffled frog-leaping algorithm (SFLA) is a population-based cooperative meta-heuristic algorithm with efficient mathematical function and global search capability. The SFLA is a search metaphor inspired by natural memetics and evolution. It is inspired by the interactive behavior and global exchange of information of frogs searching for food laid on discrete stones randomly located in a pond. It combines the advantages of the genetic-based memetic algorithm (MA) and the social behavior-based PSO algorithm with such characteristics as simple concept, fewer parameters adjustment, prompt formation, great capability in global search and easy implementation.

The steps in SFLA include the following:

*__Initial population__: Individual frogs are equivalent to the GA chromosomes, and represent a set of solutions.

*__Sorting and distribution__ : Frogs are sorted in descending order based on their fitness values, then each frog is distributed to a different subset of the whole population called a memeplex, the entire population is divided into *m* memeplexes, each containing *n frogs*

*__Memeplex evolution__: An independent local search is conducted for each frog memeplex, in what is called memeplex evolution.

*__Shuffling:__ After a defined number of memetic evolutionary steps, frogs are shuffled among memeplexes, enabling frogs to interchange messages among different memplexes and ensure that they move to an optimal position, similar to particles in PSO.

*__Terminal condition__ : If a global solution or a fixed iteration number is reached, the algorithm stops

2.3.2. Intelligent Water Drops Algorithm (IWD)

IWD is an innovative population based method proposed by Hamed Shah-hosseini in 2007[12] .It is inspired by the processes in natural river systems constituting the actions and reactions that take place between water drops in the river and the changes that happen in the environment that river is flowing. Based on the observation on the behavior of water drops, an artificial water drop is developed which possesses some of the remarkable properties of the natural water drop.

This Intelligent Water Drop has two important properties:

3... The amount of the soil it carries now, Soil (IWD).

3... The velocity that it is moving now, Velocity (IWD).

The environment in which the water flows depend on the problem under consideration. An IWD moves in discrete finite-length steps. From its current location to its next location, the IWD velocity is increased by the amount nonlinearly proportional to the inverse of the soil between the two locations. Moreover, the IWD‘s soil is increased by removing some soil of the path joining the two locations. The amount of soil added to the IWD is inversely (and nonlinearly) proportional to the time needed for the IWD to pass from its current location to the next location. This duration of time is calculated by the simple laws of physics for linear motion. Thus, the time taken is proportional to the velocity of the IWD and inversely proportional to the distance between the two locations. Another property of an IWD is that it prefers the paths with low soils on its beds to the paths with higher soils on its beds. To implement this behavior of path choosing, a uniform random distribution is used among the soils of the available paths such that the probability of the next path to choose is inversely proportional to the soils of the available paths. The lower the soil of the path, the more chance it has for being selected by the IWD.

2.3.3. Artificial Immune System Algorithm

Proposed by Dasgupta,in 1999 [14].Artificial Immune algorithm is based on clonally selection principle and is a population based algorithm .AIS is inspired by the human immune system which is a highly evolved, parallel and distributed adaptive system that exhibits the following strengths: immune recognition, reinforcement learning, feature extraction, immune memory, diversity and robustness. The artificial immune system (AIS) combines these strengths and has been gaining significant attention due to its powerful adaptive learning and memory capabilities. The main search power in AIS relies on the mutation operator and hence, the efficiency deciding factor of this technique.

The steps in AIS are as follows:

Initialization of antibodies (potential solutions to the problem). Antigens represent the value of the objective function f(x) to be optimized.

Cloning, where the affinity or fitness of each antibody is determined. Based on this fitness the antibodies are cloned; that is the best will be cloned the most. The number of clones generated from the n selected antibodies is given by: Nc =Σ round (β*j/i) © = 1,2…….n , Where Nc is the total number of clones, β is a multiplier factor and j is the population size of the antibodies.

Hypermutation: The clones are then subjected to a hyper mutation process in which the clones are mutated in inverse proportion to their affinity; the best antibody‘s clones are mutated lesser and worst antibody‘s clones are mutated most. The clones are then evaluated along with their original antibodies out of which the best N antibodies are selected for the next iteration. The mutation can be uniform, Gaussian or exponential.

2.4. ECOLOGY

Natural ecosystems provides rich source of mechanisms for designing and solving difficult engineering and computer science problems. It comprises the living organisms along with the abiotic environment with which organisms interact such as air, soil, water etc. There can be numerous and complex types of interactions among the species of ecosystem. Also this can occur as interspecies interaction (between species) or intra species interaction (within species). The nature of these interactions can be cooperative/ competitive. Cooperation includes division of labor and represents the core of sociality. Inter species interaction can be of mainly 3 types based on the outcome of interaction (positive, negative, neutral) termed as mutualism, parasitism, commensalism respectively. Examples of cooperation in nature: within species (i.e., homogeneous cooperation, also called social evolution), as in the social foraging behaviors of animal herds, bird flocks, insect groups and bacterial colonies; between species (i.e., heterogeneous cooperation, also called symbiosis), as in the mutualism between human and honey guide. Moreover Biogeography includes the study of distribution of species over specified time and space.

2.4.1. PS20

Proposed by Hanning Chen and Yunlong Zhu in 2008 [18],inspired by the ideas from the co evolution of symbiotic species in natural ecosystems and heterogeneous interaction between species. , PS2O is a multi-species optimizer which extends the dynamics of the canonical PSO algorithm by adding a significant ingredient that takes into account the symbiotic co evolution between species. The algorithm initially create an ecosystem containing a species set X = {S1, S2,. . .,Sn}, and each species possesses a members set Sn= {xi,x2…xm} i.e., totally n * m (n species and m members within species) individuals co evolve in the ecosystem. The ith member of the kth species is characterized by the vector xik={xi1k,xi2k…xidk} .and the fitness being f(xki) and lower value of the fitness represents the higher ability of survival. Under this presumed external environmental stress, all individuals in this model co evolve to the states of lower and lower fitness by cooperating each other both within species and between species. In each generation t, each individual xki will have social evolution as well as symbiotic evolution. Social evolution resembles the cooperation between individuals of the same species. Due to the socio biological background of the canonical PSO model, xki evolve according to the rules of the canonical PSO algorithm in this process thus accelerating towards the personal best position and the best position found by its neighbors where as symbiotic evolution addresses the cooperation between individuals of distinct species. Xki beneficially interacts with and rewards all its symbiotic partners (individuals of dissimilar species), i.e., each symbiotic partner donates its knowledge to aid other partners. Then xki accelerate towards its symbiotic partner of the best fitness. Finally if all individuals in the ecosystem cannot find a better position after a (here a is a constant) generations, it means that all species suffer a severe external environmental stress. Then randomly choose half species of the ecosystem to go extinct to release this stress for other species to survive. At the same time, randomly initiate equal number of species in the ecosystem for new experimentations and adaptations. In PS2O cooperation occurred in two levels, i.e., species level (interaction between species) and individual level (interaction within species).

2.4.2. Invasive Weed Colony Optimization Invasive

Weed Optimization (IWO) is a numerical stochastic search algorithm proposed by Mehrabian and Lucas in 2006 [19], inspired by the ecological process of weed colonization and distribution. It is capable of solving general multi-dimensional, linear and nonlinear optimization problems with appreciable efficiency .Adapting with their environments, invasive weeds cover spaces of opportunity left behind by improper tillage; followed by enduring occupation of the field. Their behavior changes with time since as the colony become dense there is lesser opportunity of life for the ones with lesser fitness.

The steps of the algorithm are described as below:

Initialization: includes a population of initial solutions being dispread over the D dimensional problem space with random positions.

Fitness Evaluation: Evaluate the individual fitness and rank the population according to their fitness.

Reproduction: Allowed to produce seeds depending on its own and the colony's lowest and highest fitness. This helps to concentrate on the highest fitness values in the search domain and hence increases convergence towards the group best value.

Spatial Dispersal: The generated seeds are being randomly dispersed over the D dimensional search space by normally distributed random numbers with mean equal to zero; but varying variance. The standard deviation (SD), σ, of the random function will be reduced from a previously defined initial value σ initial, to a final value, σ final, in every generation, which is given as follows:

Selection: Select the Pmax or maximum number of plants from the best plants reproduced.

2.4.3. Biogeography- Based Optimization

Biogeography-Based Optimization (BBO) is a global optimization algorithm developed by **Dan Simon** in 2008[20] and is inspired by mathematical models of biogeography by Robert MacArthur and Edward Wilson. Biogeography is the study of distribution of species in nature over time and space; that is the immigration and emigration of species between habitats. The application of this idea to allow information sharing between candidate solutions.Each possible solution is an island and their features that characterize habitability are called suitability index variables (SIV). The fitness of each solution is called its habitat suitability index (©) and depends on many features of the habitat. High-© solutions tend to share their features with low-© solutions by emigrating solution features to other habitats. Low- © solutions accept a lot of new features from high-© solutions by immigration from other habitats. Immigration and emigration tend to improve the solutions and thus evolve a solution to the optimization problem. The value of © is considered as the objective function, and the algorithm is intended to determine the solutions which maximize the © by immigrating and emigrating features of the habitats. In BBO, there are two main operators: migration (which includes both emigration and immigration) and mutation .A habitat H is a vector of N (SIVs) integers initialized randomly. Before optimizing, each individual of population is evaluated and then follows migration and mutation step to reach global minima. In migration the information is shared between habitats that depend on emigration rates μ and immigration rates λ of each solution. Each solution is modified depending on probability *Pmod*that is a user defined parameter. Each individual has its own λ and μ and are functions of the number of species K in the habitat .Poor solutions accept more useful information from good solution, which improve the exploitation ability of algorithm. In BBO, the mutation is used to increase the diversity of the population to get the good solutions.

Features:
* In BBO the original population is not discarded after each generation. It is rather modified by migration.

* Another distinctive feature is that, for each generation, BBO uses the fitness of each solution to determine its immigration and emigration rate.

2.5. Comparative Analysis

This section presents a comparative analysis of the algorithms seen so far in terms of the representation, operators, areas of application and control parameters

NAME OF ALGORITHM	**REPRESENTATION**	**OPERATORS**	**AREAS OF APPLICATION**	**CONTROL PARAMETERS**
GA (Genetic Algorithm)	Binary,Real no.s, Permutation of elements, List of rules, Program elements, Data structure, tree, Matrix	Crossover, Mutation, Selection, Inversion , Gene Silencing	Optimization problems in data mining and rule extraction, dynamic and multiple criteria web-site optimizations , decision thresholds for distributed detection in wireless sensor networks , Computer aided design path planning of mobile robots, fixed charge transportation problem, various scheduling problems ,assignment problems, flight control system design, pattern recognition , reactive power dispatch , sensor-based robot path planning, training of radial basis function, multi-objective vehicle routing problem, minimum energy broadcast problem in wireless ad hoc network, software engineering problems, pollutant emission reduction problem in the manufacturing industry, Power System Optimization problems, port folio Optimization ,optimal learning path in e learning, Web page classification system ,closest sting problem in bioinformatics ,structural optimization, defect identification system, molecular modeling, web service selection, cutting stock problem, drug design, personalized e-learning system, SAT Solvers	Population size, max.generation number, cross over probability, mutation probability, length of chromosome, chromosome encoding

GP	Tree structure (terminals & function set)	Crossover, Reproduction, Mutation, Permutation, Editing, Encapsulation, Decimation	portfolio optimization, Design of image exploring agent, epileptic pattern recognition, automated synthesis of analogue electrical circuits symbolic regression, robotics, data mining(Automatic feature extraction, classification etc.),cancer diagnosis, power transformer fault classification automatic synthesis of analog electrical circuits	Population size, Maximum number of generations, Probability of crossover, Probability of mutation
ES	Real-valued vectors	Mutation, Selection, discrete Recombination	parameter estimation (Hatanaka et al., 1996), image processing (Gonzalez et al., 2001) ,computer vision system (Bergener et al., 2001),Task scheduling and car automation ,structural optimization, Evolution strategy for gas-turbine fault-diagnoses, A multi-parametric evolution strategies algorithm for vehicle routing problems, clustering	Population size, Maximum number of generations, Probability of crossover, Probability of mutation
DE	Real-valued vectors	Crossover, mutation ,selection	unsupervised image classification ,clustering , digital filter design , optimization of non-linear functions , global optimization of non-linear chemical engineering processes and multi-objective optimization .	S Population size ,Nddiamension of problem,F scale factor,Pr probability of crossover
PSO	D dimensional vector for position ,speed,best state	43alutatory, updater and evaluator.	Multimodal biomedical image registration (Wachowiak et al., 2004) and the Iterated Prisoner‘s Dilemma (Franken and Engelbrechet, 2005), classification of instances in multiclass databases, feature selection, web service composition course composition, Power System Optimization problems (economic dispatching), Edge detection in noisy images, finding optimal machining parameter assembly line balancing problem in production and operations management,	number of particles, Dimension of particles, Range of particles, Vmax, Learning factors: c1c2, inertia weight , maximum number of iterations

			various scheduling problems ,vehicle routing problems, prediction of tool life in ANN, multi-objective, dynamic, constrained and combinatorial optimization problems ,QoS in adhoc multicast, Anomaly detection, color image segmentation , sequential ordering problem, constrained portfolio optimization problem, selective particle regeneration for data clustering , Extracting rules from fuzzy neural network, machinery fault detection, Unit commitment computation, Signature verification	
ACO	Undirected graph	Pheromone Update and Measure, trail evaporation	TSP Problem, Quadratic Assignment problem (QAP) Job-Shop Scheduling problem. Dynamic problem of data network routing, a shortest path problem where properties of the system such as node availability vary over time. Continuous optimization and parallel processing implementations . vehicle routing problem ,graph colouring and set covering, agent-based dynamic scheduling, digital image processing, classification problem in data mining, Protein folding problem	number of ants ,iterations , pheromone evaporation rate, amount of reinforcement
PFA	Linear=[x1 ,x2..x]	Dispersal, pollination	Continuous function optimization. Tuning parameters in PID Controllers in higher order systems and RBF Neural Network Parameters Optimization	size of population, the boundary of parameter space, initial value of the maximum number of seeds
AIS	*attribute string*(a real-valued vector),	immune operators(cloning, hyper mutation and	computer security ,anomaly detection, clustering /classification, numeric function optimization, ,learning ,IIR filter	Antibody population size ,. Number of antibodies to be

	integer string , binary string, symbolic string	selection based on elitism)	design, control, robotics, data mining[149],virus detection, pattern recognition , tuning of controllers, multi- modal optimization, job shop scheduling	selected for hyper-mutation ,number of antibodies to be replaced, multiplier factor β
ABC	D-dimensional vector (xi=1,2…D)	Reproduction, replacement of bee, selection	Scheduling problems, image segmentation, capacitated vehicle routing problem , (WSNs),assembly line balancing problem, Solving reliability redundancy allocation problem, training neural networks , XOR, Decoder–Encoder and 3-Bit Parity benchmark problems , pattern classification , reliability redundancy allocation problems, clustering, resource-constrained project scheduling problem, p-center problem	number of food sources which is equal to the number of employed or onlooker bees (SN), the value of limit, the maximum cycle number (MCN)
FSA	Xi = (x1, x2, . . ., xk, . . .,xD),	Swarming ,following, searching	function optimization , Parameter estimation , combinatorial optimization], least squares support vector machine and geo technical engineering problems	Visual distance,max step length ,crowd factor
GSO	Unit vector	Scrounging, ranging, producing	Truss structure design, benchmark functions (He et al., 2006) and applied for optimal power flow problems (Fei et al.,2007), mechanical design optimization problems, multi objective optimization, Optimal placement of FACTS devices, machine condition monitoring, optimal location and capacity of distributed generations.	Population size, percentage of rangers, no: of rangers, Head angle, position, maximum pursuit angle, maximum turning angle, maximum pursuit distance
SFLA	Xi=(xi1, xi 2, ... , xiS)	Replacement, shuffling	Color Image Segmentation , Solving TSP , Automatic recognition of speech emotion water , Unit Commitment Problem , Grid Task scheduling, Optimal viewpoint selection for volume rendering , multi-user detection in DS-CDMA	number of frogs P, number of memeplexes, and number of evolutionary iterations for each memeplex before

			distribution , Fuzzy controller design , Optimal Reactive Power Flow , A Web Document Classification , Mobile robot path planning , classification rule mining , Combined Economic Emission Dispatch , Job-shop scheduling, ground water model calibration problems, Multicast Routing Optimization	shuffling.
BFA	θ*i* (*j*, *k*,*l*) represents *i*-th bacterium at *j*th chemotactic, *k*-th reproductive and l-th elimination dispersal step.	Reproduction, chemotaxis, Dispersion , elimination	inverse airfoil design ,application for harmonic estimation problem in power systems, optimal power system stabilizers design , tuning the PID controller of an AVR, an optimal power flow solution, machine learning, an application of job shop scheduling benchmark problems; the parameters of membership functions and the weights of rules of a fuzzy rule set are estimated, transmission loss reduction ,implemented as the parameter estimation of nonlinear system model (NSM) for heavy oil thermal cracking, evaluation of independent components to work with mixed signals, solve constrained economic load dispatch problems ,application in the null steering of linear antenna arrays by controlling the element amplitudes, applications in multi objective optimization.	Dimension of the search space. ,number of bacteria , number of chemotactic steps , number of elimination and dispersal events , number of reproduction steps , probability of elimination and dispersal, location of each bacterium ,no: of iterations, step size c(i)
IWCO	Vector in D dimensional space	Reproduction, dispersal, selection	Time modulated linear antenna array synthesis, cooperative multiple task assignment of UAV, fractional order PID Controller, Training of Feed-Forward Neural Networks, Nash equilibrium search in electricity markets, blind multi-user detection for MC-CDMA interference suppression over	weed population size, modulation index ,standard deviations

			multipath fading channel, recommender system	
PS2O	D dimensional vector for position , speed, best state	47alutatory, updater ,extinction, evaluator	Cooperative Cognitive Wireless Communication, constructing collaborative service systems (CSSs)	number of particles, Dimension of particles, Range of particles, Vmax, Learning factors: inertia weight , maximum number of iterations
BBO	H=h1,h2..hn as individuals of habitat.	Migration (emigration and immigration) , mutation	general benchmark functions, constrained optimization ,the sensor selection problem for aircraft engine health estimation , power system optimization, groundwater detection and satellite image classification , web-based BBO graphical user interface , global numerical optimization, optimal meter placement for security constrained state estimation	number of habitats (population size), maximum migration rates, mutation rate

2.6. A review on application of algorithms inspired by behavior of bee colonies

Method	Description
Hybrid artificial bee colony algorithm and Bacterial foraging optimization algorithm	To improve the intensification ability of ABC algorithm, it’s used in hybrid with Bacterial Foraging optimization algorithm.

2.7. A review on application of algorithms inspired by behavior of ant colonies

Method	Description
Ant Colony System, Genetic Algorithm and Back propagation Network	Ant Colony System, Genetic Algorithm and Back propagation Network
Ant Colony Optimization, Bee colony Optimization And Genetic Algorithm	Some novel method on Bio-inspired adaptive algorithms like ACO, BCO and Genetic algorithm is introduced for Selection of features extracted from mammogram image.
Ant Colony Optimization, Genetic algorithm	A hybrid method for selection and classification of features extracted from mammogram image. Feature selection is performed using GA and ACO, which then fed to a three-layer BPN hybrid with ACO for classification.
Ant Colony Optimization and Genetic Algorithm	A technique is implemented for extraction of suspicious regions using Asymmetric approach. Breast border is detected using GA and Bio-inspired ACO algorithm for nipple identification. Finally suspicious regions are identified by subtracting the images of left and right breast

2.8. A review on application of algorithms inspired by particle swarm

Method	Description
Particle Swarm Optimization	An algorithm to solve course scheduling problem using PSO.
Artificial Bee Colony and Particle Swarm Optimization	To improve the performance of the algorithm using hybrid modified EABCPSO and OABC-PSO
Particle Swarm Optimization, Ant Colony Optimization, Genetic algorithm	A hybrid method to classify themicrocalcifications in mammogram. SGLDM is used for feature extraction. Feature selection is performed using GA, ACO and PSO. The selected features are then fed to a three-layer BPN hybrid with ACO and PSO for classification.
Particle Swarm Optimization, Ant Colony System	A method to automatically detect breast border and nipple position to identify the suspicious Regions in Mammograms based on Asymmetries is proposed. Image is enhanced using median filter. Then the pectoral muscle regions are removed. PSO is used to enhance the detected breast border whereas the ACS to identify the nipple position.
Genetic Algorithm	A new PSO method proposed for feature set selection and classification of micro calcification in mammograms in hybrid with GA and three-layer BPN respectively. Features are extracted using SGLDM.
Particle Swarm Optimization, Ant Colony Optimization, Genetic algorithm(GA)	An improved Computer-aided Decision support system for classifying the tumor and identifying the stages of cancer using neural network in hybrid with PSO and ACO. Multi-objective genetic algorithm has been used for optimal feature extraction.

CHAPTER 3

LITERATURE SURVEY ON BACTERIAL FORAGING OPTIMIZATION ALGORITHM

A new member to the family of nature-inspired optimization algorithms is Bacteria Foraging Optimization Algorithm (BFOA), proposed by Kavin Passino [1] in the year 2002. The key idea of this new algorithm is based on the application of group foraging strategy of a swarm of *E*.coli bacteria in multi-optimal function optimization. Application of group foraging strategy of a swarm of *E*.coli bacteria in multi-optimal function optimization is the key idea of this new algorithm. Bacteria search for nutrients is a manner to maximize energy obtained per unit time. Individual bacterium also communicates with others by sending signals. A bacterium takes foraging decisions after considering two previous factors. The process, in which a bacterium moves by taking small steps while searching for nutrients, is called chemotaxis. The key idea of BFOA is mimicking chemotactic movement of virtual bacteria in the problem search space.

3.1. The Theory of Foraging

Foraging theory is basically the hypothesis that animals seek for food and obtain nutrients in a way that maximizes their intake energy *E* per unit time *T* spent for foraging. Hence, they try to maximize a function like

$$\frac{E}{T}$$

Maximization of such a function provides nutrient sources to survive and additional time for other important activities. Shelter-building and mate-finding activities sometimes bear similarities to foraging. Basically, foraging is very different for different species. Herbivores generally find food easily but must eat a lot of it. Whereas Carnivores generally find difficult to locate food but do not have to eat much since their food is of high energy value. The “environment” establishes the pattern of nutrients that are available and it places constraints on obtaining that. During foraging there can be risks due to predators, the prey may be mobile so it must be chased, and the physiological characteristics of the forager constrain its capabilities and ultimate success. For many animals, nutrients are distributed in “patches”, for example a lake, a bush with berries, group of trees with fruits etc. Foraging involves finding such patches, deciding whether to enter a patch and search for food, and whether to continue searching for food in the current patch or to go find another patch. Patches are generally encountered sequentially, and sometimes great effort and risk are needed to travel from one patch to another. Generally, if an animal finds a nutrient-poor patch, but it expects that there should be a better patch elsewhere, then it will consider risks and efforts to search for another patch. If an animal has been in a patch for some time, it can begin to reduce its resources, so there should be an optimal time to leave the patch and venture out to try to find a richer one.

In fact, some researchers have shown that foraging decision *heuristics* are used very effectively by animals to approximate optimal policies, given the physiological (and other) constraints that are imposed on the animal [1].

3.2. Foraging Search Strategies

Animals searching for food often move in a series of runs, and pauses. This behavior is easily observed, for example, when birds hunt for insects on a lawn. [37] In one approach to the study of foraging search strategies [2], predation is broken into components that are similar for many animals.

- ❖ Predators must search for and locate prey.
- ❖ Next, they pursue and attack the prey.
- ❖ Finally, they “handle” and ingest the prey.

The importance of various components of foraging behavior depends on the relationship between the predator and the prey. If the prey is larger than the predator, then the pursuit, attack and handling can be most important. The prey may be easy to find, but the prey’s size gives it an advantage. If the prey is smaller than the predator, then generally the search component of foraging is most important. Small size can be an advantage for the prey. Since preys are often smaller than predators for many animals, they must be consumed often and in large numbers; this makes the search time limit other components of the predation cycle. In this article, we consider cases where the searching behavior is the dominant factor in foraging. This is the case for many birds, fish, lizards, and insects. Some animals are “cruise”, “ambush” or “Saltatory” searchers.

3.2.1. Cruise search

For the cruise approach to searching, the forager moves continuously through the environment, constantly searching for prey at the boundary of the volume being searched. Tuna fish and hawks are considered as cruise searchers. In cruise search, distance increases at a constant rate dictated by how fast the animal moves in search. To visualize this above mentioned strategies, consider Fig. 9, where distance traveled in searching is plotted against time.

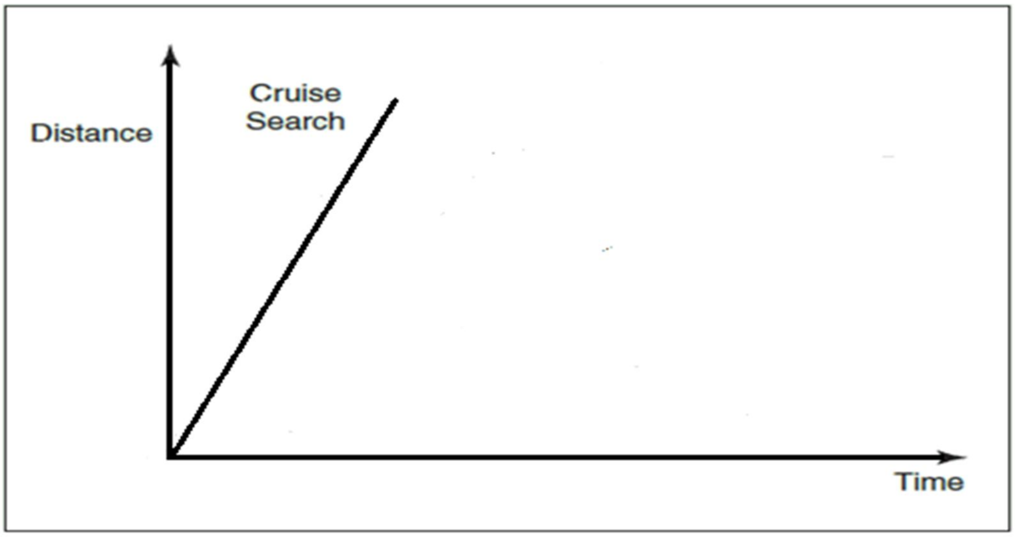


Figure 9
Cruise Search

3.2.2. Ambush search

In ambush search, the forager remains stationary and waits for prey to cross into its strike range. Snakes and Lions are good examples in this regard. Figureb10 depicts the Ambush search strategy.

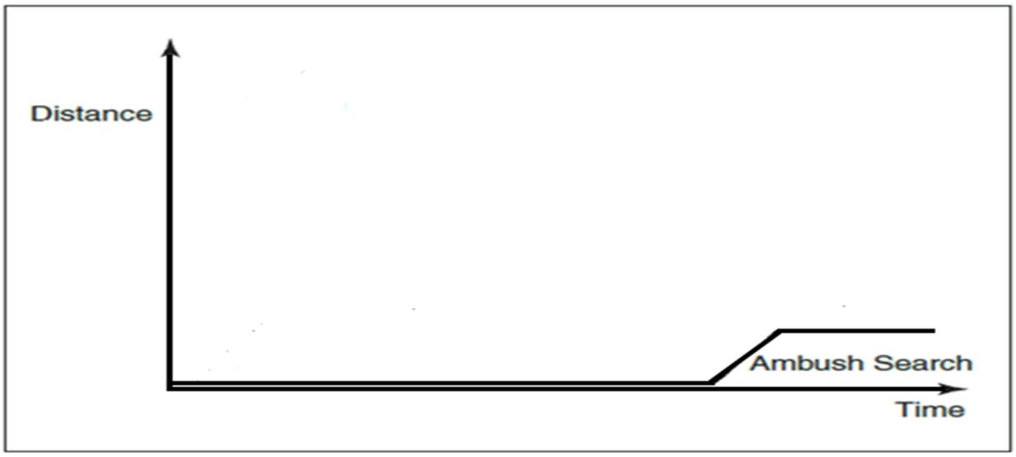


Figure 10
Ambush Search

3.2.3. Saltatory search

The search strategies of many species are actually between the cruise and ambush extremes. Particularly, in “53alutatory search”, an animal will intermittently cruise and sit and wait, possibly changing direction at various times when it stops and possibly while it moves. One of Cody’s (1974) innovations was to record these search movements in terms of time and distance (Figure 1). This pattern of movement has been called “pause-travel” by Andersson (1981). More recently, O’Brien and his colleagues (Evans and O’Brien, 1988; O’Brien et aL, 1989,1990) reviewed the prevalence of this phenomenon and coined the phrase “53alutatory search.” Saltatory search appears as a series of oscillations about a trend line in “Cody” plots. O’Brien et aL (1990) report that 53alutatory search has been observed in ground-feeding birds, planktivorous fish, and some lizards [37]. For this type of search strategies Birds, Fish, Lizards and Insects are best fit examples.

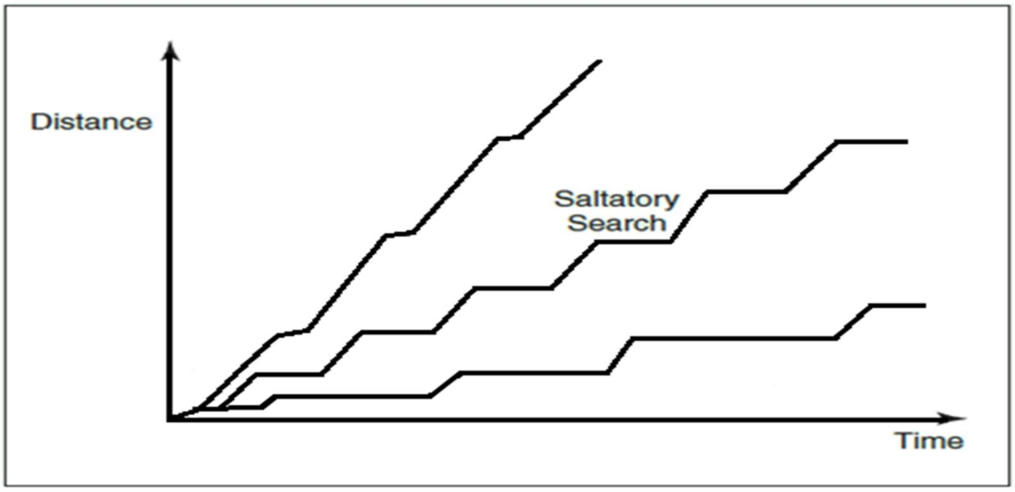


Figure 11
Saltatory Search

In between cruse and ambush search, there are many possible 54alutatory search strategies that are based on an alternating sequence of cruising and waiting. Many animals' foraging strategies seem to lie somewhere on the continuum between ambush and cruise and hence are salutatory search strategies.

3.3. Foraging for Social and Intelligent Living Beings

The above mentioned search strategies for foraging were for single creatures. Clearly, there are many advantages to group (or social) foraging. For group foraging there are some communication methods needed. For human beings, different languages are the means of communication. In other animals, it might be certain movements or chemical secretions or noises or "trail-laying" mechanisms. [38]

Followings are the advantages of group foraging

- More numbers of animals are involved searching for nutrients, so the probability of finding nutrients increases. If any one finds some nutrients, it conveys others in the group about the source of the nutrients.
- Joining a group provides access to an "information center" to assist in survival.[38]
- Increased capability to cope with larger prey. The group can "gang up" on a large prey and kill and ingest it.
- Protection from predators can be provided by members of the group (e.g., in some species the members in the middle of the group are protected by the ones at the edges). Sometimes it is useful to think of a group (swarm) of animals as a single living creature, where via grouping and communication a "collective intelligence" emerges that actually results in more successful foraging for each individual in the group (and the gains can offset effects of food competition within groups; by working together there can be more food than if there is no cooperation). [38]

3.4. The Bacterial Foraging

In this section, we will consider individual and group foraging for bacteria, organisms that are much simpler than ants or humans, but still work together for the benefit of the group. After explaining how bacteria forage, we will model them via a computer simulation.

3.4.1. The E. coli Bacterium

Theodore von Escherich, a German bacteriologist, discovered the bacterium *Escherichia coli* in 1885. The bacterium, commonly known as *E. coli*, can be found in the human intestinal tract and comes in multiple forms, only one of which is deadly. *E. coli* is only two microns in length and one micron wide. It is rod-shaped and covered with small pili for mobility.

Widely known for its lethal capability, *E. coli 0157:H7* is the most common and dangerous strain of *E.coli* and is found in feces and meat. When milk, cider, water, sawdust, and even the air come in contact with cow feces they may become contaminated with *E. coli*. Meat is the primary source of infection in humans.

Cell Parts

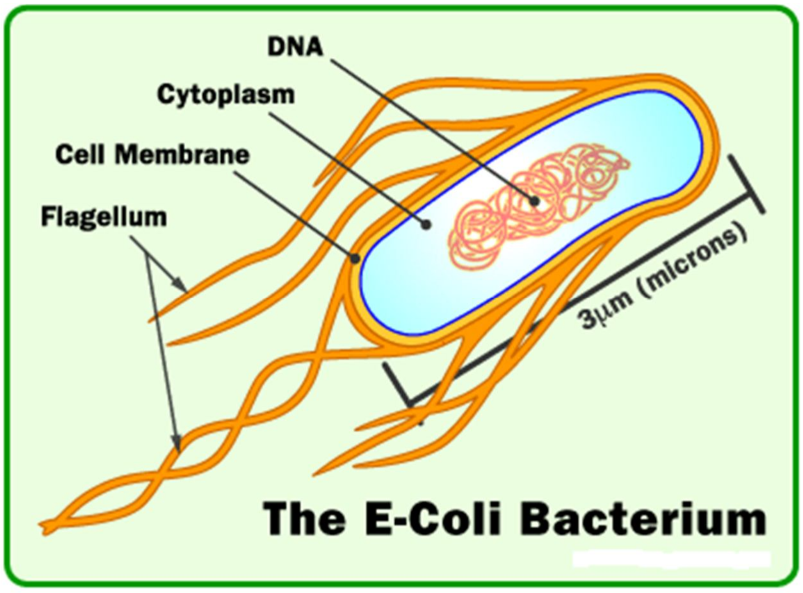


Figure 12

Bacteria are about the simplest cells that exist today. A bacterium is a single, self-contained, living cell. An *Escherichia coli* bacteria (or E. coli bacteria) is typical – it is about one-hundredth the size of a human cell (maybe a micron long and one-tenth of a micron wide), so it is invisible without a microscope. *Escherichia coli* is a Gram-negative, facultative anaerobic, rod-shaped bacterium of the genus *Escherichia*.
A bacterium consists of an outer wrapper called the cell membrane, and inside the membrane is a watery fluid called the cytoplasm. Cytoplasm might be 70-percent water. The other 30 percent is filled with proteins called enzymes that the cell has manufactured, along with smaller molecules like amino acids, glucose molecules and ATP. At the center of the cell is a ball of DNA. If we stretch out this DNA into a single long strand, it would be incredibly long compared to the bacteria – about 1000 times longer.

An E. coli bacterium has a distinctive, capsule shape. The outer portion of the cell is the cell membrane, shown here in orange. In E. coli, there are actually two closely-spaced membranes protecting the cell. At the center of the cell is its DNA. The DNA is like a wadded-up ball of string. There is no protection for the DNA in a bacterium – the wadded-up ball floats in the cytoplasm roughly in the center of the cell. Attached to the outside of the cell are long strands called **flagella**, which propel the cell.

The pili are used for a type of gene transfer to other *E. coli* bacteria, and flagella (singular-flagellum) are used for locomotion. The cell is about 1 µm in diameter and 2 µm in length. The *E. coli* cell only weighs about 1 picogram and is about 70% water. Its entire genome has been sequenced; it contains 4,639,221 of the A, C, G, and T "letters"—adenosine, cytosine, guanine, and thymine—arranged into a total of 4,288 genes. Mutations in *E. coli* occur at a rate of about 10^{-7} per gene, per generation, and can affect its physiological aspects. *E. coli* bacteria occasionally engage in a type of "sex" called "conjugation" where small gene sequences are unidirectionally transferred from one bacterium to another via an extended pilus. When *E. coli* grows, it gets longer, and then divides in the middle into two "daughters." Given sufficient food and held at the temperature of about 37^0 C, *E. coli* can synthesize and replicate everything it needs to make a copy of itself in about 20 min; hence growth of a population of bacteria is exponential with a relatively short time to double. The *E. coli* bacterium has a control system that enables it to search for food and try to avoid noxious substances. For instance, it swims away from alkaline and acidic environments and toward more neutral ones.

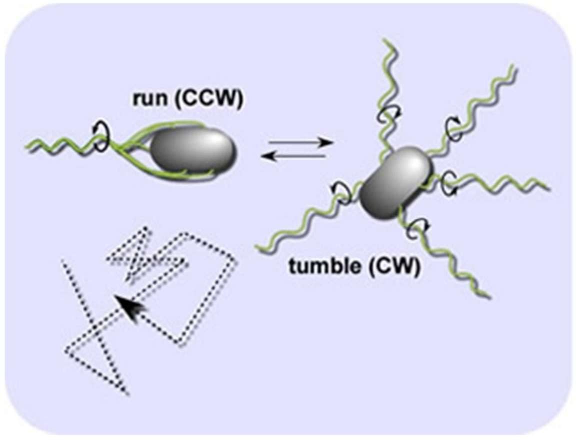
3.4.2 Swimming and Tumbling via Flagella

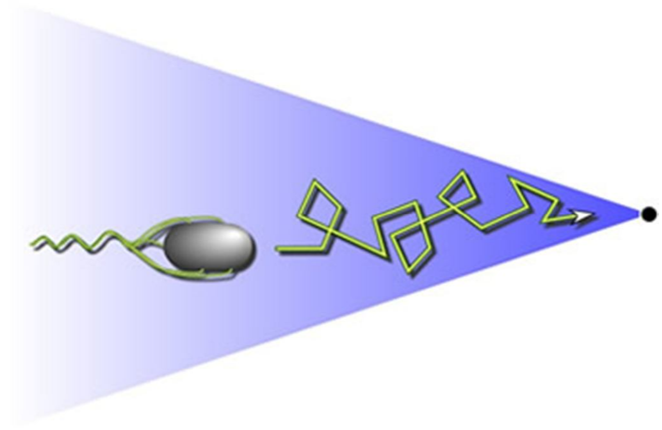
A motile E. coli propels itself from place to place by rotating its long, whip-like structures called flagella. To move forward, motors in the cell's wall move the flagella into bundle together and spin at about 100-200 revolutions per second counterclockwise and the organism "swims". But when the bacteria have to make a turn, one tail separates itself out and rotation abruptly changes to clockwise, the bacterium "tumbles" in place and seems incapable of going anywhere. Once it's oriented the right way, *E. coli* bundles up its tails and spins them all counterclockwise. Swimming is more frequent as the bacterium approaches a chemo attractant (food). Tumbling, change in direction is more frequent as the bacterium moves away from the chemo attractant. It is a complex combination of swimming and tumbling that keeps them in areas of higher food concentrations.

In the above-mentioned algorithm the bacteria undergoes chemotaxis, where they like to move towards a nutrient gradient and avoid noxious environment. Generally the bacteria move for a longer distance in a friendly environment. Figure 13 depicts how clockwise and counter clockwise movement of a bacterium take place in a nutrient solution.[38]

When they get food in sufficient, they are increased in length and in presence of suitable temperature they break in the middle to from an exact replica of itself. This phenomenon inspired Passino to introduce an event of reproduction in BFOA. Due to the occurrence of sudden environmental changes or attack, the chemotactic progress may be destroyed and a group of bacteria may move to some other places or some other may be introduced in the swarm of concern. This constitutes the event of elimination-dispersal in the real bacterial population, where all the bacteria in a region are killed or a group is dispersed into a new part of the environment.

Each flagellum is a left-handed helix configured so that as the base of the flagellum (i.e., where it is connected to the cell) rotates counterclockwise, as viewed from the free end of the flagellum looking toward the cell, it produces a force against the bacterium so it pushes the cell. If a flagellum rotates clockwise, it will pull at the cell. From an engineering perspective, the rotating shaft at the base of the flagellum is quite an interesting contraption that seems to use what biologists call a “universal joint” (so the rigid flagellum can “point” in different directions relative to the cell). In addition, the mechanism that creates the rotational forces to spin the flagellum in either direction is described by biologists as a biological motor [8], [15]. The motor is quite efficient in that it makes a complete revolution using only about 1,000 protons, and thereby *E. coli* spends less than 1% of its energy for motility. An *E. coli* bacterium can move in two different ways; it can run (swim for a period of time) or it can tumble, and it alternates between these two modes of operation its entire lifetime (it is rare that the flagella will stop rotating). To tumble after a run, the cell slows down or stops first; since bacteria are so small, they experience almost no inertia, only viscosity, so when a bacterium stops swimming, it stops within the diameter of a proton [14]. In one type of medium, on a run the bacteria swim at a rate of about 10-20 µm/s, but in a rich medium they can swim even faster [16]. This is a relatively fast rate for a living organism to travel; Call the time interval during which a run occurs the “run interval.” [8], [12]. Runs are not perfectly straight since the cell is subject to Brownian movement that causes it to wander off course by about 30^0 in 1 s in one type of medium, so this is how much it typically can deviate on a run. In a certain medium, after about 10 s it drifts off course more than 90^0 and hence essentially forgets the direction it was moving [8]. Finally, note that in many bacteria and media the motion of the flagella can induce other motions.





(a) (b)

Figure 13

Random and biased walks. *Left:* A random walk in isotropic environments. When the cell’s motors rotate CCW, the flagellar filaments form a trailing bundle that pushes the cell forward. When one or more of the flagellar motors reverses to CW rotation, that filament undergoes a shape change (owing to the torque reversal) that disrupts the bundle. Until all motors once again turn in the CCW direction, the filaments act independently to push and pull the cell in a chaotic tumbling motion. Tumbling episodes enable the cell to try new, randomly-determined swimming directions. *Right:* A biased walk in a chemoeffector gradient. Sensory information suppresses tumbling whenever the cell happens to head in a favorable direction. The cells cannot head directly up-gradient because they are frequently knocked off course by Brownian motion.

3.4.3 Bacterial Motile Behavior : Climbing to Nutrient Gradients

The motion patterns that the bacteria will generate in the presence of chemical attractants and repellants are called chemotaxes. For *E. coli*, encounters with serine or aspartate result in attractant responses, whereas repellant responses result from the metal ions Ni and Co, changes in pH, amino acids like leucine, and organic acids like acetate. Generally, as a group, they will try to find food and avoid harmful phenomena, and when viewed under a microscope, they show a type of intelligent behavior, since they seem to intentionally move as a group. To explain chemotaxis motions in E. Coli, we must simply explain how they decide how long to run.If an *E. coli* is in some substance that it does not have food or noxious substances, and if it is in this medium for a long time, then the flagella will simultaneously alternate between moving clockwise and counterclockwise so that the bacterium will alternately tumble and run. This alternation between the two modes will move the bacterium, but in random directions, and this enables it to "search" for nutrients (see Fig. 14(b)). If the bacteria are placed in a homogeneous concentration of serine, then a variety of changes occurs in the characteristics of their motile behavior. For instance, mean run length and mean speed increase and mean tumble time decreases. They still show, a basic type of searching behavior; even though the bacterium has some food, it persistently searches for more. As an example of tumbles and runs in the isotropic homogeneous medium described above, in one trial motility experiment lasting 29.5 s there were 26 runs, the maximum run length was 3.6 s, and the mean speed was about 21 µm/s [8], [12]. Suppose that the bacterium happens to encounter a nutrient gradient (e.g., serine), as shown in Fig. 14(c). The *change* in the concentration of the nutrient triggers a reaction such that the bacterium will spend more time swimming and less time tumbling. The directions of movement are "biased" toward increasing nutrient gradients. The cell does not change its *direction* on a run

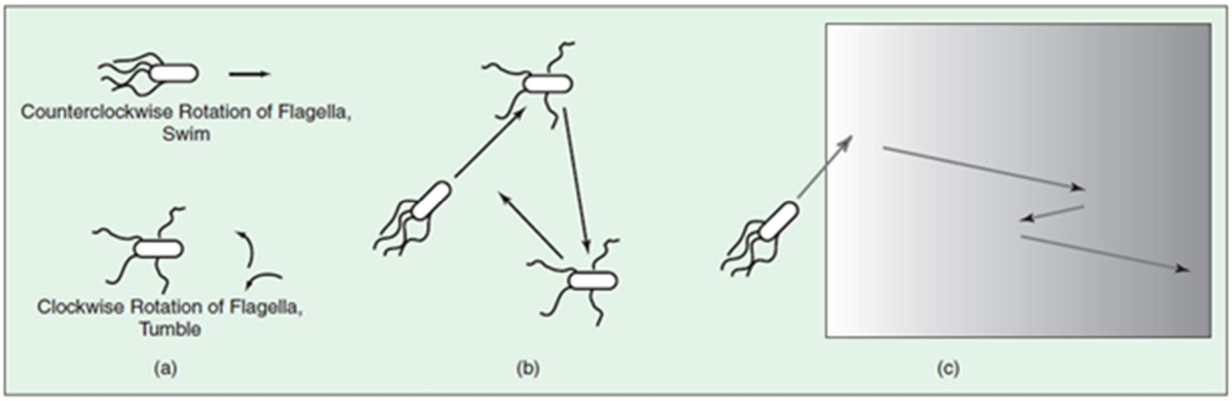

Counterclockwise Rotation of Flagella, Swim
Clockwise Rotation of Flagella, Tumble
(a)
(b)
(c)

Figure 14
Motile Behavior of Bacteria Cell

due to changes in the gradient— the tumbles basically determine the direction of the run, aside from the Brownian influences mentioned above. On the other hand, typically if the bacterium happens to swim down a concentration gradient or into a positive gradient of noxious substances, it will return to its baseline behavior so that essentially it tries to search for a way to climb back up the gradient. For instance, under certain conditions, for a wild-type cell swimming up serine gradients, the mean run length is 2.19 ± 3.43 s, but if it swims down a serine gradient the mean run length is 1.40 ± 1.88 s [12]. Finally, suppose that the bacterium reaches a region with constant nutrient concentration after having been on a positive gradient for some time. In this case, after a period of time the bacterium will return to the same proportion of swimming and tumbling as when it was in the neutral substance, so that it returns to its standard searching behavior. It is never satisfied with the amount of surrounding food; it always seeks higher concentrations. Considering the deviations in direction due to Brownian movement , the bacterium basically uses as much time as it can in making decisions about climbing gradients [14]. Basically, the bacterium is trying to swim from places with low concentrations of nutrients to places with high concentrations. An opposite type of behavior is used when it encounters noxious substances. If the various concentrations move with time, then the bacterium will tend to "chase" after the more favorable environments and run from harmful ones.

3.4.4 Sensing and Decision-Making

In an ever-changing environment, it is essential that organisms are able to sense these changes and to respond appropriately. The sensors are the receptor proteins that are signaled directly by external substances. Possible responses include alterations in genetic material expression and/or active movement towards or away from an environment. Most sensory pathways in eukaryotic organisms rely on serine, threonine or tyrosine protein kinases, whereas the most common sensory pathways in prokaryotes use a HAP (Histidine-Aspartate Phosphorelay) system. Bacteria can sense a vast range of environmental signals, from the concentrations of nutrients and toxins to oxygen levels, pH, osmolarity and the intensity and wavelength of light. Bacteria are able to respond to a changing environment, and one way to respond is to move. The transduction of sensory signals alters the concentration of small phosphorylated response regulators that bind to the rotary flagellar motor and cause switching [39].

The flagellar locomotion in *E. coli* is chiefly controlled by the soluble HPK chemotaxis protein, CheA and two RRs. CheAsenses changes through transmembrane chemoreceptors, which induce the trans-autophosphorylation of dimeric CheAon a histidine residue. The correct interplay between locomotory system and other sensing systems are essential for numerous sensory signals to result in a balanced behavioural response (Ref.5)
The sensor is very sensitive, in some cases requiring less than ten molecules of attractant to trigger a reaction, and attractants can trigger a swimming reaction in less than 200 ms. On the other hand, the corresponding threshold for encountering a homogeneous medium after being in a nutrient-rich one is larger.

Also, a type of time averaging is occurring in the sensing process. The receptor proteins then affect signaling molecules inside the bacterium. Also, there is in effect an "adding machine" and an ability to compare values to arrive at an overall decision about which mode the flagella should operate in; essentially, the different sensors add and subtract their effects, and the more active or numerous have a greater influence on the final decision. The sensory and decision making system in *E. coli* is probably the best understood one in biology; here, we are ignoring the underlying chemistry needed for a full explanation.

The "decision-making" system in the *E. coli* bacterium must have some ability to sense a *derivative*, and hence it has a type of memory! Experiments have shown that it performs a type of sampling, and roughly speaking, it remembers the concentration a moment ago, compares it with a current one, and makes decisions based on the difference. In summary, we see that with memory, a type of addition mechanism, an ability to make comparisons, a few simple internal "control rules," and its chemical sensing and locomotion capabilities, the bacterium is able to achieve a complex type of search and avoidance behavior. Evolution has designed this control system. It is robust and clearly very successful at meeting its goals of survival when viewed from a population perspective.

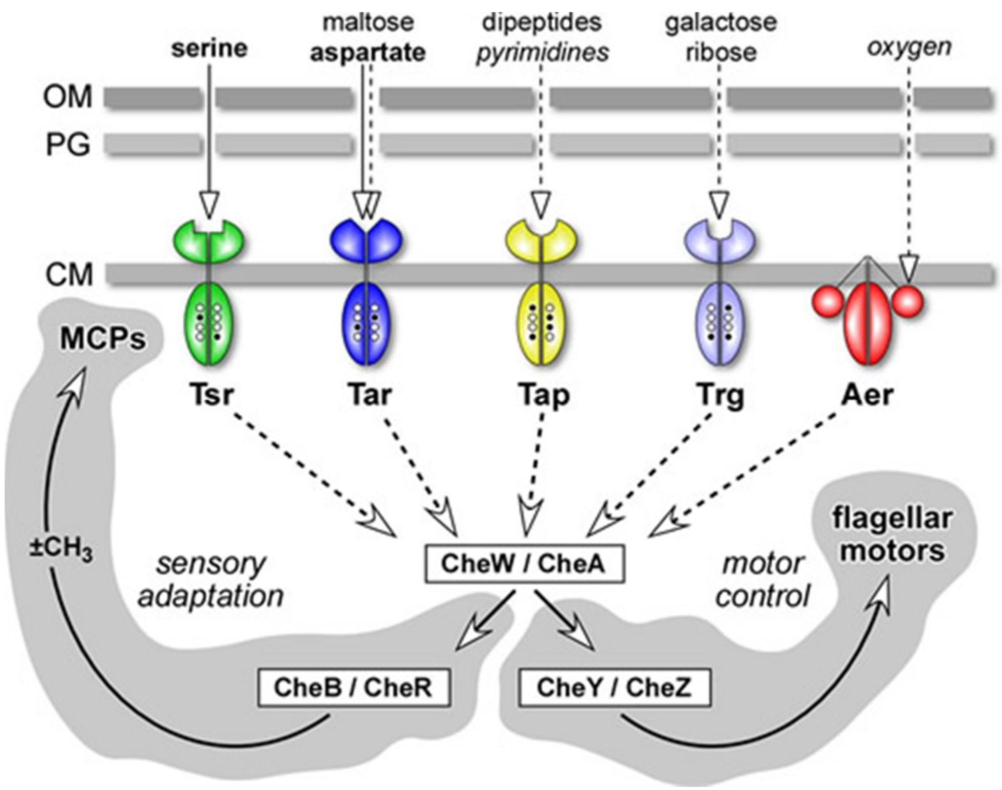


Figure 15

Signaling components and circuit logic. ***E. coli*** **receptors employ a common set of cytoplasmic signaling proteins: CheW and CheA interact with receptor molecules to form stable ternary complexes that generate stimulus signals; CheY transmits those signals to the flagellar motors, CheZ controls their lifetime; CheR (methyltransferase) and CheB (methylesterase) regulate MCP methylation state. Abbreviations: OM (outer membrane); PG (peptidoglycan layer of the cell wall); CM (cytoplasmic membrane).**

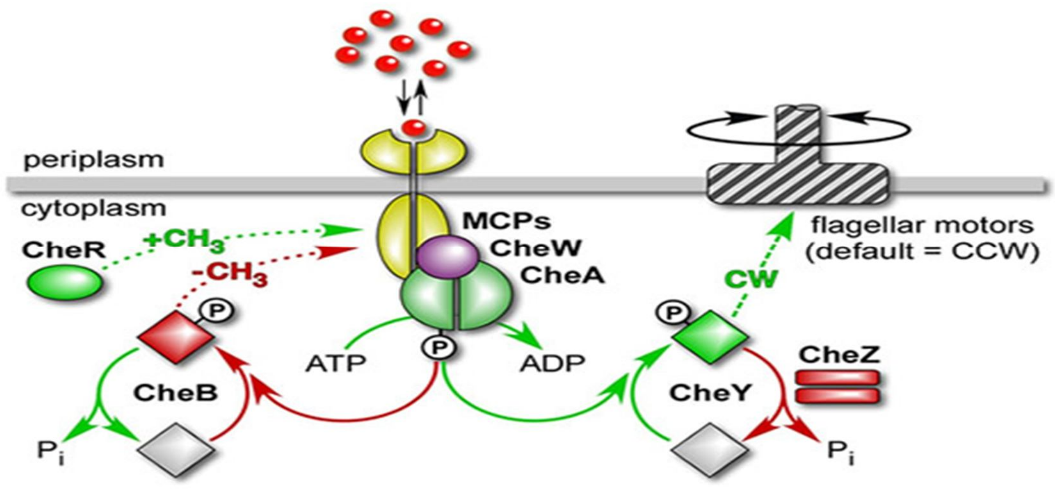


Figure 16

Phosphorelay signaling. The flagellar motors of *E. coli* spin CCW by default; the signaling pathway modulates the level of phospho-CheY, the signal for CW rotation. Reactions and components that augment CW rotation are depicted in green; those that augment CCW rotation are depicted in red.

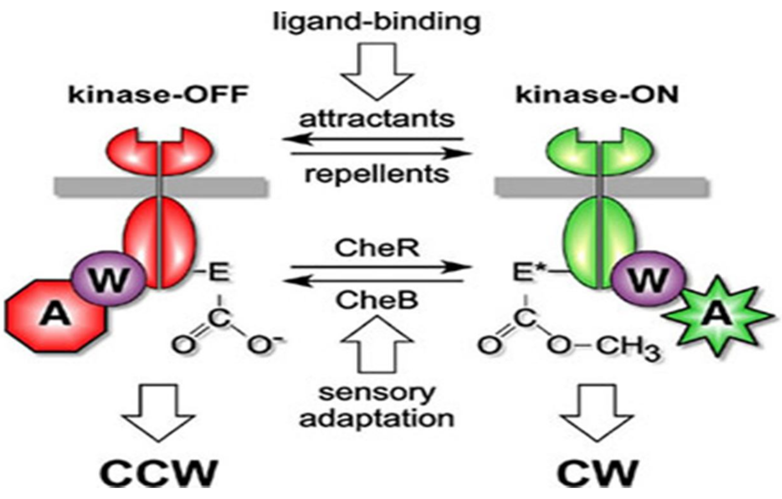


Figure 17

Two-state model of receptor signaling. The cartoons depict receptor ternary complexes in kinase-active (on) and kinase-inactive (off) signaling states. Changes in chemoeffector occupancy drive the complexes toward one state or the other. During sensory adaptation, changes in receptor methylation level shift signaling complexes toward the opposing state to restore a balance between CCW and CW signal outputs. The actual stoichiometry and structure of receptor signaling complexes are not known. Note that receptor methyl groups are attached to specific glutamic acid (E) residues in the MCP cytoplasmic domain, forming glutamyl methyl esters and neutralizing the negative charge on the glutamyl carboxyl group. Charge neutralization probably plays an important role in controlling receptor output state.

3.4.5 Elimination and Dispersal Events

It is possible that the local environment where a population of bacteria live changes either gradually (e.g., via consumption of nutrients) or suddenly due to some other influence. Events can occur such that all the bacteria in a region are killed or a group is dispersed into a new part of the environment.

For example, local significant increases in heat can kill a population of bacteria that are currently in a region with a high concentration of nutrients we can consider heat as a type of noxious influence. Or it may be that water or some animal will move populations of bacteria from one place to another in the environment. Over long periods of time, such events have spread various types of bacteria into virtually every part of our environment—from our intestines to hot springs and underground environments.

The effect of elimination and dispersal events on chemotaxis : they have both the effect of possibly destroying chemotactic progress, but they also have the effect of assisting in chemotaxis, since dispersal may place bacteria near good food sources. From a broader perspective, elimination and dispersal are parts of the population-level long-distance motile behavior.

3.4.6 Bacterial Motility and Swarming

Most bacteria are motile, and many types have analogous taxes capabilities to *E. coli* bacteria. The specific sensing, actuation, and decision-making mechanisms are different [10], [18]. Some bacteria can search for oxygen, and hence their motility behavior is based on aerotaxis, whereas others search for desirable temperatures resulting in thermotaxis. Actually, the *E. coli* is capable of thermotaxis in that it seeks warmer environments with a temperature range of 20 to 37 ^{0}C.

Other bacteria search for or avoid light of certain wavelengths, and this is called phototaxis. Actually, the *E. coli* tries to avoid intense blue light, so it is also capable of phototaxis. Some bacteria swim along magnetic lines of force that enter the earth, so that in the northern hemisphere they swim toward the north magnetic pole and in the southern hemisphere they swim toward the south magnetic pole.

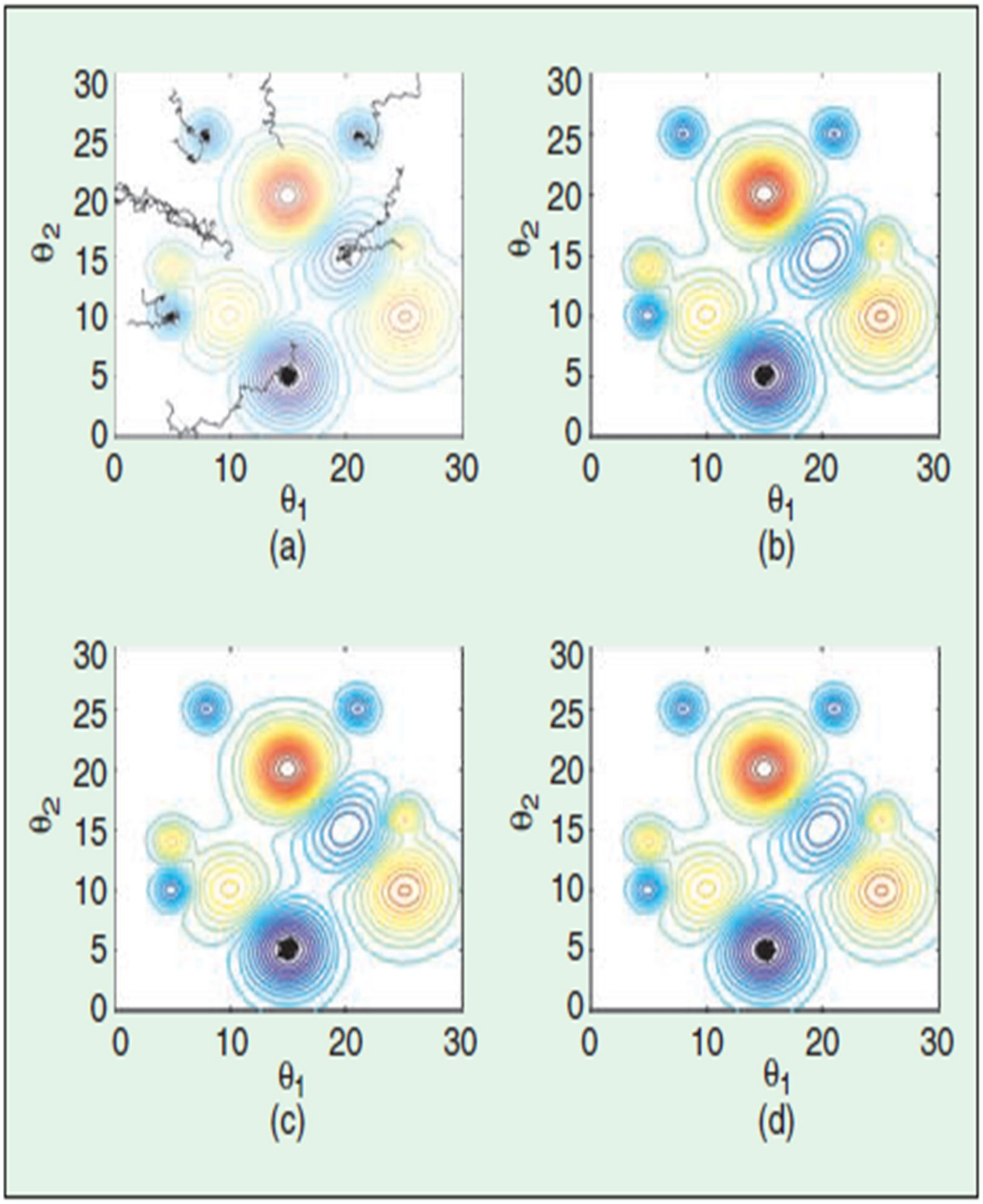


Figure 18
Bacterial Motion Trajectories Contour Plot

A particularly interesting group behavior has been demonstrated for several motile species of bacteria, including *E.coli* and *S. typhimurium*, where intricate stable spatiotemporal patterns (swarms) are formed in semisolid nutrient media [18]-[22]. (Microbiologists reserve the term "swarming" for other characteristics of groups of bacteria.

Here, we abuse the terminology and favor using the terminology that is used for higher forms of animals such as bees.) When a group of *E. coli* cells is placed in the center of a semisolid agar with a single nutrient chemo-effector (sensor), they move out from the center in a traveling ring of cells by moving up the nutrient gradient created by consumption of the nutrient by the group. Moreover, if high levels of succinate are used as the nutrient, then the cells release the attractant aspartate so that they congregate into groups and hence move as concentric patterns of groups with high bacterial density. The spatial order results from outward movement of the ring and the local releases of the attractant; the cells provide an attraction signal to each other so they swarm together. Pattern formation can be suppressed by a background of aspartate (since it seems that this will in essence scramble the chemical signal by eliminating its directionality). The pattern seems to form based on the dominance of the two stimuli (cell-to-cell signaling and foraging).

The role of these patterns in natural environments is not fully understood; however, there is evidence that stress to the bacteria results in them releasing chemical signals toward which other bacteria are chemotactic. If enough stress is present, then a whole group can secrete the chemical signal, strengthening it, and hence an aggregate of the bacteria forms. It seems that this aggregate forms to protect the group from the stress (e.g., by effectively hiding many cells in the middle of the group). It also seems that the aggregates of the bacteria are not necessarily stationary; under certain conditions they can migrate, split, and fuse. This has led researchers to hypothesize that other communication methods are being employed that are not yet understood.
As another example, biofilms exist that can be composed of multiple types of bacteria (e.g., *E. coli*) that can coat various objects (e.g., roots of plants or medical implants). It seems that both motility and "quorum sensing" [18], [23] are involved in biofilm formation. A biofilm is a mechanism for keeping a bacterial species in a fixed location, avoiding overcrowding and avoiding nutrient limitation and toxin production by packing them in a low density in a polysaccharide matrix [23]. Secreted chemicals provide a mechanism for the cells to sense population density, but motility seems to assist in the early stages of biofilm formation. Researchers also think that chemotactic responses are used to drive cells to the outer edges of the biofilm where nutrient concentrations may be higher.

Finally, it should be noted that other types of bacteria exhibit swarm behaviors [23]. For instance, the luminous bacteria *Vibrio fischeri* will emit light when its population density reaches a certain threshold. *Streptomycete* colonies can grow a branching network of long fiberlike cells that can penetrate and degrade vegetation and then feed on the resulting decaying matter (in terms of combinatorial optimization, you may think of finding optimal trees or graphs). *Myxococcus xanthus*, one type of *myxobacterium* (slime bacteria), exhibits relatively exotic foraging and survival behaviors [18], [23]-[27]. For instance, while moving across solid surfaces (via gliding), they secrete slime trails and tend to follow slime trails of each other. Some *myxobacteria* prey on other bacteria (in what has been likened to the behavior of wolves). Mutants of *Myxococcus xanthus* can exhibit "social" and "adventurous" motility (essentially different group foraging behaviors). Under starvation conditions, they form aggregates called fruiting bodies where some cells die and others form spores. These fruiting bodies can then be transported via insects or wind into more favorable environments where the spores germinate and form a new colony. Simulations of *myxobacteria* based on a stochastic cellular automata approach are described in [28] and [29].

There is a great diversity of strategies for foraging andsurvival, even at the bacterial level! Clearly, we cannot outline them all here. Our objective is simply to explain how motile behaviors in both individual and groups of bacteria implement foraging and hence optimization.

3.4.7 *E. coli* Bacterial Swarm Foraging for Optimization

Suppose that we want to find the minimum of $J(\theta)$, $\theta \in R^p$, where we do not have measurements or an analytical description of the gradient$\nabla J(\theta)$. Here, we use ideas frombacterial foraging to solve this nongradient optimization problem. First, suppose that θ is the position of a bacterium and $J(\theta)$ represents the combined effects of attractants and repellants from the environment, with, for example, $J(\theta) < 0$, $J(\theta) = 0$, and $J(\theta) > 0$ representing that the bacterium at location θ is in nutrient-rich, neutral, and noxious environments, respectively. Basically, chemotaxis is a foraging behavior that implements a type of optimization where bacteria try to climb up the nutrient concentration (find lower and lower values of $J(\theta)$), avoid noxious substances, and search for ways out of neutral media (avoid being at positions θ where $J(\theta) \geq 0$). It implements a type of biased random walk.

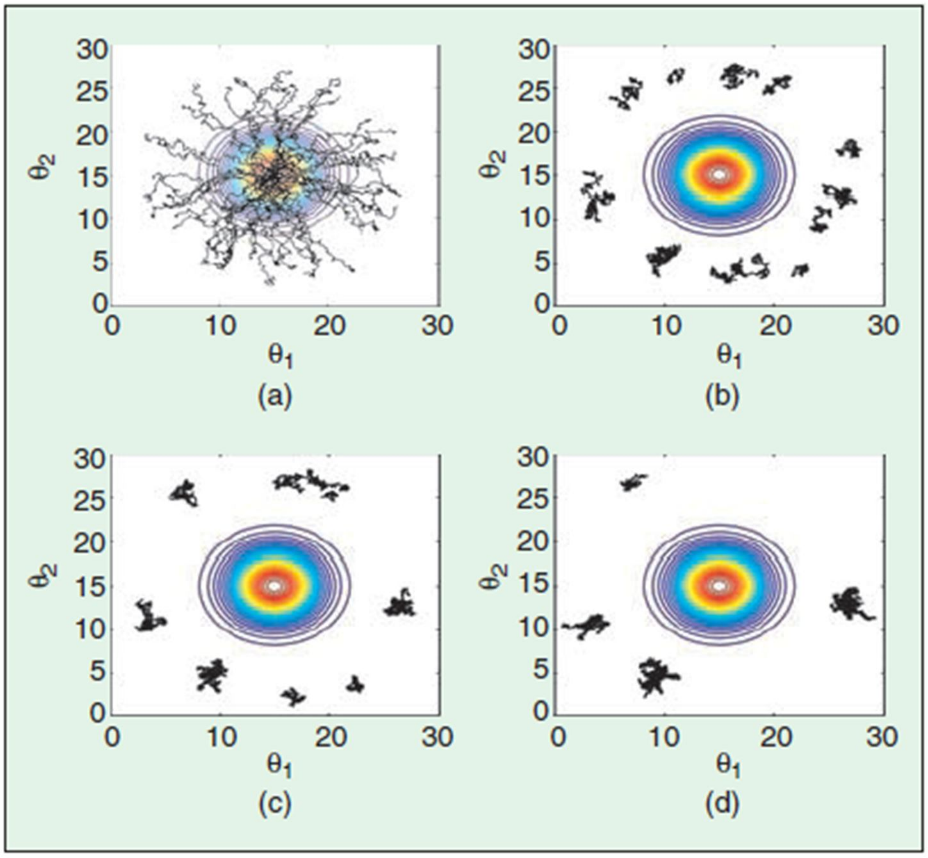


Figure 19

Swarm Behavior of E.Coli

3.4.8 Chemotaxis, Swarming, Reproduction, Elimination, and Dispersal

Define a chemotactic step to be a tumble followed by a tumble or a tumble followed by a run. Let *j* be the index for the chemotactic step. Let *k* be the index for the reproduction step. Let*l* be the index of the elimination-dispersal event.

Let $P(j,k,l) = [\theta^i(j,k,l) \mid i = 1,2,\ldots\ldots,S]$
represent the position of each member in the population of the *S* bacteria at the *j*th chemotactic step, *k*th reproduction step, and *l*th elimination-dispersal event. Here, let $J(i\ ,\ j,k,l)$ denote the cost at the location of the *i*th bacterium $\theta^i(j,k,l)$ □ R*p* (sometimes we drop the indices and refer to the *i*th bacterium position as θ^i). Note that we will interchangeably refer to *J* as being a "cost" (using terminology from optimization theory) and as being a nutrient surface (in reference to the biological connections). For actual bacterial populations, *S* can be very large (e.g., $S = 10^9$), but $p = 3$.

In our computer simulations, we will use much smaller population sizes and will keep the population size fixed. We can allow $p > 3$ so we can apply the method to higher dimensional optimization problems.

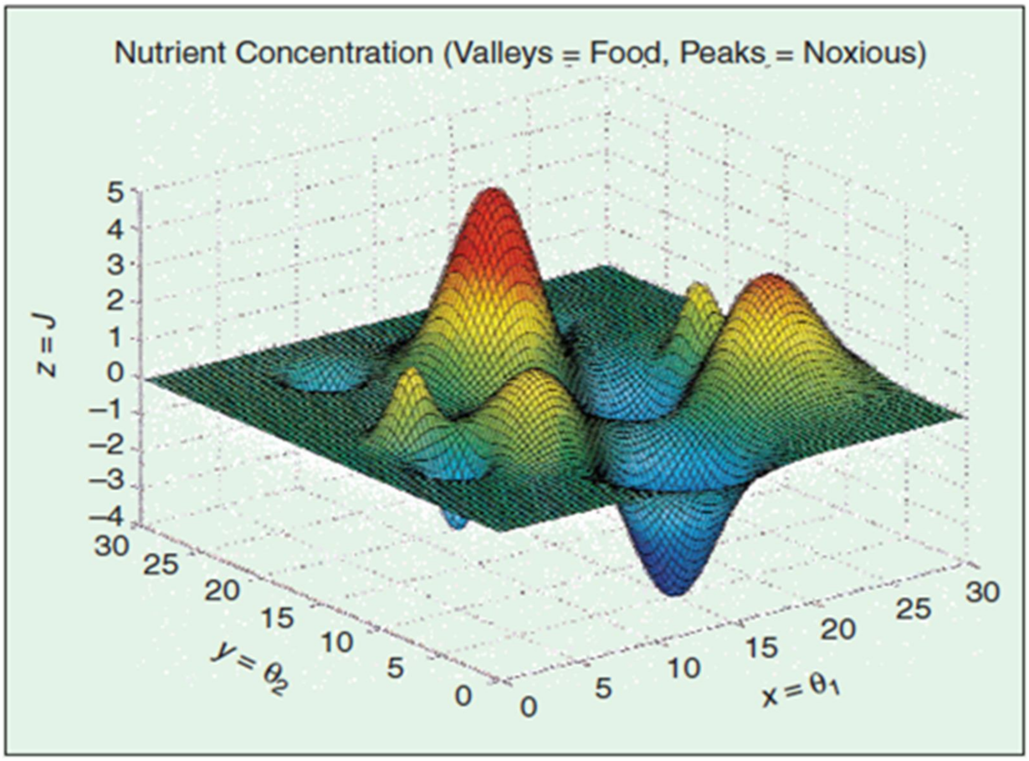


Figure 20
Nutrient Landscape

Let *Nc* be the length of the lifetime of the bacteria as measured by the number of chemotactic steps they take during their life. Let $C(i) > 0, i = 1,2,\ldots\ldots,S,$ denote a basic chemotactic step size that we will use to define the lengths of steps during runs. To represent a tumble, a unit length random direction, say □(*j*), is generated; this will be used to define the direction of movement after a tumble. In particular, we let

$$\theta^i(j+1,k,l) = \theta^i(j,k,l)+C(i)\,\square(j)$$

so that $C(i)$ is the size of the step taken in the random direction specified by the tumble.

If at $\theta^i(j+1,k,l)$ the cost $J(i, j+1,k,l)$ is better (lower) than at $\theta^i(j,k,l)$, then another step of size $C(i)$ in this same direction will be taken, and again, if that step resulted in a position with a better cost value than at the previous step, another step is taken. This swim is continued as long as it continues to reduce the cost, but only up to a maximum number of steps, *Ns*. This represents that the cell will tend to keep moving if it is headed in the direction of increasingly favorable environments.

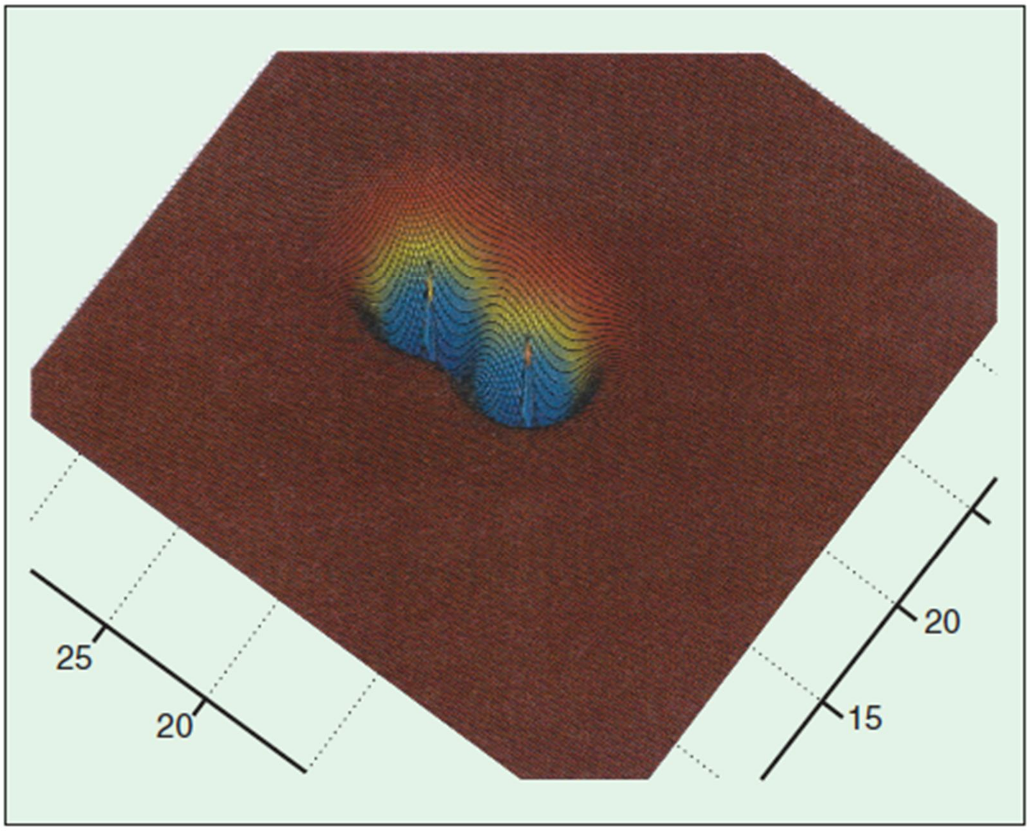


Figure 21
Cell to Cell Chemical Attraction

The above discussion was for the case where no cell-released attractants are used to signal other cells that they should swarm together. Here, we will also have cell-to-cell signaling via an attractant and will represent that with J^i_{cc} $(\theta,\theta^i(j,k,l))$, $i = 1,2,\ldots\ldots\ldots,S$, for the *i*th bacterium. Let,

$$d\text{attract} = 0.1$$

be the depth of the attractant released by the cell (a quantification of how much attractant is released) and

$$w_{\text{attract}} = 02.$$

be a measure of the width of the attractant signal (a quantification of the diffusion rate of the chemical). The cell also repels a nearby cell in the sense that it consumes nearby nutrients and it is not physically possible to have two cells at the same location. To model this, we let

$$h_{\text{repellant}} = d_{\text{attract}}$$

be the height of the repellant effect (magnitude of its effect) and

$$w_{\text{repellant}} = 10$$

be a measure of the width of the repellant. The values for these parameters are simply chosen to illustrate general bacterial behaviors, not to represent a particular bacterial chemical signaling scheme. The particular values of the parameters were chosen with the nutrient profile that we will use later in Fig. 3 in mind. For instance, the depth and width of the attractant is small relative to the nutrient concentrations represented in Fig. 9. Let

$$\begin{aligned} J_{cc}(\theta,P(j,k,l)) &= \sum_{i=1}^{S} J_{cc}^{i}(\theta,\theta^{i}(j,k,l)) \\ &= \sum_{i=1}^{S}[-d_{attract}\exp(-w_{attract}\sum_{m=1}^{p}(\theta_m-\theta_m^i)^2] \\ &\quad +[-h_{repellant}\exp(-w_{repellant}\sum_{m=1}^{p}(\theta_m-\theta_m^i)^2] \end{aligned}$$

denote the combined cell-to-cell attraction and repelling effects,
where $\theta = [\theta_1,\theta_2,\ldots\ldots,\theta_p]^T$ is a point on the optimization domain and θ_m^i is the *m*th component of the *i*th bacterium position θS^i (for convenience we omit some of the indices).

An example for the case of $S = 2$ and the above parameter values is shown in Fig. 10. Here, note that the two sharp peaks represent the cell locations, and as you move radially away from the cell, the function decreases and then increases (to model the fact that cells far away will tend not to be attracted, whereas cells close by will tend to try to climb down the cell-to-cell nutrient gradient toward each other and hence try to swarm). Note that as each cell moves, so does its J^i_{cc} $(\theta,\theta^i(j,k,l))$ function, and this represents that it will release chemicals as it moves. Due to the movements of all the cells, the function $J_{cc}(\theta,P(j,k,l))$ is *time varying* in that if many cells come close together there will be a high amount of attractant and hence an increasing likelihood that other cells will move toward the group. This produces the swarming effect. When we want to study swarming, the *i*th bacterium, $i = 1,2,\ldots.,S$, will hill-climb on

$$J(i,j,k,l)) + J_{cc}(\theta,P)$$

(rather than the $J(i, j,k,l)$ defined above) so that the cells will try to find nutrients, avoid noxious substances, and at the same time try to move toward other cells, but not too close to them. The $J_{cc}(\theta,P)$ function dynamically deforms the search landscape as the cells move to represent the desire to swarm (i.e., we model mechanisms of swarming as a minimization process).
After *Nc* chemotactic steps, a reproduction step is taken.
Let *Nre* be the number of reproduction steps to be taken. For convenience, we assume that *S* is a positive even integer. Let

$$S_r = \frac{S}{2}$$

be the number of population members who have had sufficient nutrients so that they will reproduce (split in two) with no mutations. For reproduction, the population is sorted in order of ascending accumulated cost (higher accumulated cost represents that a bacterium did not get as many nutrients during its lifetime of foraging and hence is not as "healthy" and thus unlikely to reproduce); then the *Sr* least healthy bacteria die and the other *Sr* healthiest bacteria each split into two bacteria, which are placed at the same location. Other fractions or approaches could be used in place of (1); this method rewards bacteria that have encountered a lot of nutrients and allows us to keep a constant population size, which is convenient in coding the algorithm. Let N_{ed} be the number of elimination-dispersal events, and for each elimination-dispersal event each bacterium in the population is subjected to elimination-dispersal with probability p_{ed} . We assume that the frequency of chemotactic steps is greater than the frequency of reproduction steps, which is in turn greater in frequency than elimination-dispersal events (e.g., a bacterium will take many chemotactic steps before reproduction, and several generations may take place before an elimination-dispersal event).

Clearly, we are ignoring many characteristics of the actual biological optimization process in favor of simplicity and capturing the gross characteristics of chemotactic hill-climbing and swarming. For instance, we ignore many characteristics of the chemical medium and we assume that consumption does not affect the nutrient surface (e.g., while a bacterium is in a nutrient-rich environment, we do not increase the value of J near where it has consumed nutrients), where clearly in nature bacteria modify the nutrient concentrations via consumption.

A tumble does not result in a perfectly random new direction for movement; however, here we assume that it does. Brownian effects buffet the cell so that after moving a small distance, it is within a pie-shaped region with its start point at the tip of the piece of pie. Basically, we assume that swims are straight, whereas in nature they are not. Tumble and run lengths are exponentially distributed random variables, not constant, as we assume. Run-length decisions are actually based on the past 4 s of concentrations, whereas here we assume that at each tumble, older information about nutrient concentrations is lost. Although naturally asynchronous, we force synchronicity by requiring, for instance, chemotactic steps of different bacteria to occur at the same time, all bacteria to reproduce at the same time instant, and all bacteria that are subjected to elimination and dispersal to do so at the same time. We assume a constant population size, even if there are many nutrients and generations. We assume that the cells respond to nutrients in the environment in the same way that they respond to ones released by other cells for the purpose of signaling the desire to swarm (a more biologically accurate model of the swarming behavior of certain bacteria is given in [22]). Clearly, other choices for the criterion of which bacteria should split could be used (e.g., based only on the concentration at the end of a cell's lifetime, or on the quantity of noxious substances that were encountered). We are also ignoring conjugation and other evolutionary characteristics. For instance, we assume that $C(i)$, *Ns* , and *Nc* remain the same for each generation. In nature it seems likely that these parameters could evolve for different environments to maximize population growth rates. The intent here was simply to come up with a simple model that only represents certain aspects of the foraging behavior of bacteria.

3.5 Bacterial Foraging Optimization Algorithm

For initialization, we must choose *p, S, Nc , Ns , Nre , Ned , ped* , and the *C*(*i*), *i* = 1,2,…., *S*. If we use swarming, we also have to pick the parameters of the cell-to-cell attractant functions; here we will use the parameters given above.

Also, initial values for the θ^i, *i* = 1,2,….., *S*, must be chosen. Choosing these to be in areas where an optimum value is likely to exist is a good choice. The algorithm that models bacterial population chemotaxis, swarming, reproduction, elimination, and dispersal is given here (initially, $j = k = l = 0$).

For the algorithm, updates to the θ^i automatically result in updates to *P*. Clearly, we could have added a more sophisticated termination test than simply specifying a maximum number of iterations.

1) Elimination-dispersal loop: $l = l + 1$

2) Reproduction loop: $k = k + 1$

3) Chemotaxis loop: $j = j + 1$

a) For $i = 1,2,\ldots\ldots, S$, take a chemotactic step for bacterium *i* as follows.

b) Compute $J(i, j,k,l)$.
Let $J(i, j,k,l) = J(i, j,k,l) + J_{cc}(\theta^i(j,k,l),P(j,k,l))$
(i.e., add on the cell-to-cell attractant effect to the nutrient concentration).

c) Let$J_{last} = J(I,j,k,l)$ to save this value since we may find a better cost via a run.

d) Tumble: Generate a random vector $\Delta(i) \in Rp$ with each element $\Delta_m(i)$, $m = 1,2,\ldots\ldots\ldots\ldots,p$, a random number on $[-1,1]$.

e) Move: Let
$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\frac{\Delta(i)}{\sqrt{\Delta T(i)\Delta(i)}}$
This results in a step of size *C*(*i*) in the direction of the tumble for bacterium *i*.

e) Compute $J(i, j+1,k,l)$, and then let $J(i, j+1,k,l) = J(i, j+1,k,l) + J_{cc}(\theta^i(j+1,k,l),P(j+1,k,l))$

g) Swim (note that we use an approximation since we decide swimming behavior of each cell as if the bacteria numbered $\{1,2,\ldots\ldots\ldots..,i\}$ have moved and $\{i + 1,i + 2,\ldots\ldots, S\}$ have not; this is much simpler to simulate than simultaneous decisions about swimming and tumbling by all bacteria at the same time):
i) Let $m = 0$ (counter for swim length).
ii) While $m < N_s$(if have not climbed down too long)
• Le t$m = m+1$.
• I f $J(i, j+1,k,l) < J_{last}$ (if doing better), let
$J_{last} = J(i, j+1,k,l)$ and let $\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\frac{\Delta(i)}{\sqrt{\Delta T(i)\Delta(i)}}$

and use this $\theta^i(j+1,k,l)$ to compute the *new* $J(i, j+1,k,l)$ as we did in f.
• Else, let $m = Ns$. This is the end of the while statement.

h) Go to next bacterium ($i+1$) if $i \neq S$ (i.e., go to b) to process the next bacterium.

4) If $j < Nc$, go to step 3. In this case, continue chemotaxis, since the life of the bacteria is not over.

5) Reproduction:
a) For the given k and l, and for each $i = 1,2,\ldots, S$, let

$$J^i_{health} = \sum_{j=1}^{N_c+1} J(i,j,k,l)$$

be the health of bacterium i (a measure of how many nutrients it got over its lifetime and how successful it was at avoiding noxious substances). Sort bacteria and chemotactic parameters$C(i)$ in order of ascending cost J_{health} (higher cost means lower health).

b) The S_r bacteria with the highest J_{health} values die and the other S_r bacteria with the best values split (and the copies that are made are placed at the same location as their parent).

6) If $k < Nre$, go to step 2. In this case, we have not reached the number of specified reproduction steps, so we start the next generation in the chemotactic loop.

7) Elimination-dispersal: For $i = 1,2,\ldots, S$, with probability p_{ed}, eliminate and disperse each bacterium (this keeps the number of bacteria in the population constant).

To do this, if you eliminate a bacterium, simply disperse one to a random location on the optimization domain.

8) If$l < Ned$, then go to step 1; otherwise end.

3.6 The Flow Chart for Bacterial Foraging Optimization Algorithm

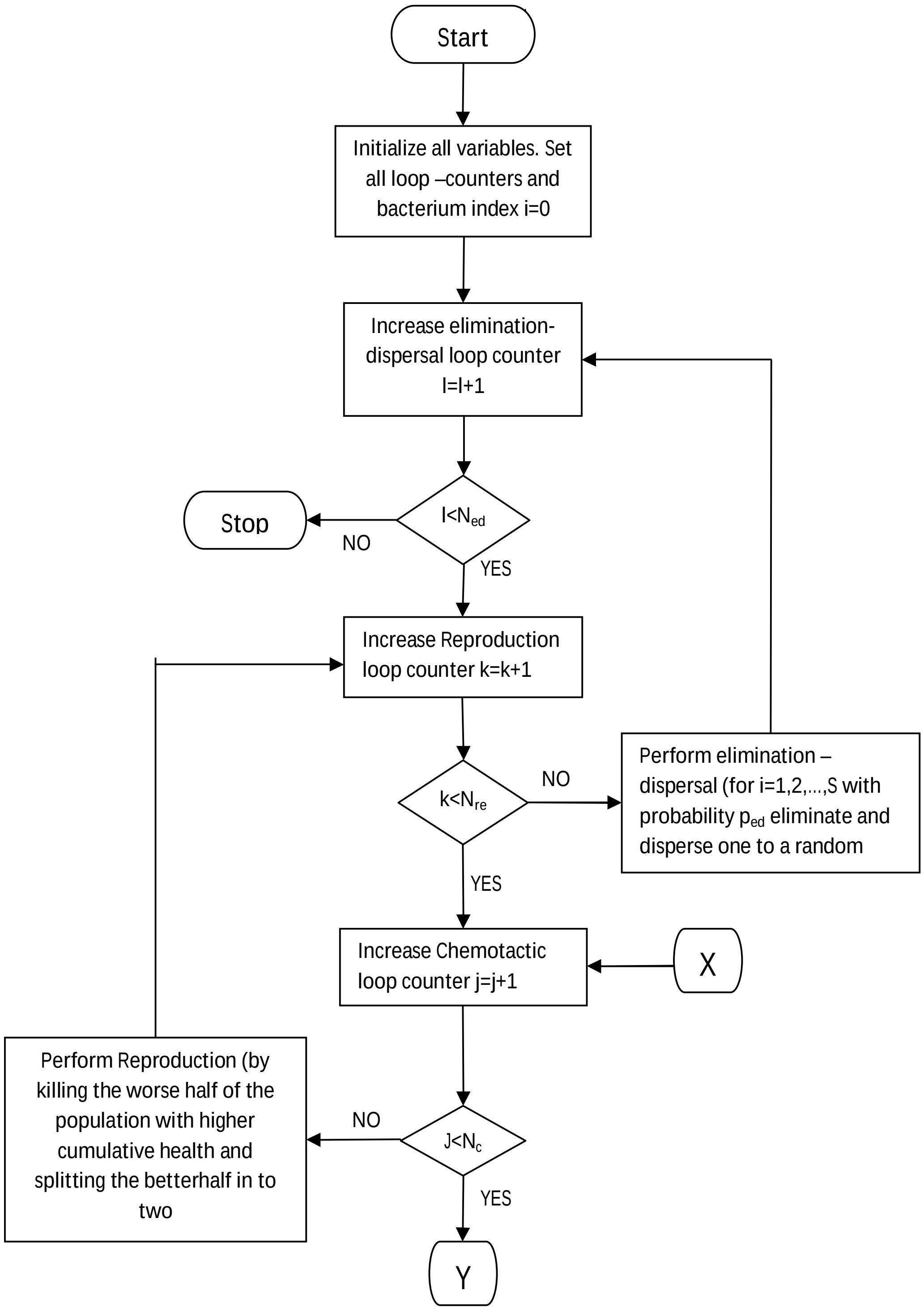

Start
Initialize all variables. Set all loop –counters and bacterium index i=0
Increase elimination-dispersal loop counter l=l+1
l<N_ed
NO
Stop
YES
Increase Reproduction loop counter k=k+1
k<N_re
NO
Perform elimination – dispersal (for i=1,2,...,S with probability p_ed eliminate and disperse one to a random
YES
Increase Chemotactic loop counter j=j+1
X
J<N_c
NO
Perform Reproduction (by killing the worse half of the population with higher cumulative health and splitting the betterhalf in to two
YES
Y

Figure 22
Bacterial Foraging Optimization Algorithm Part I

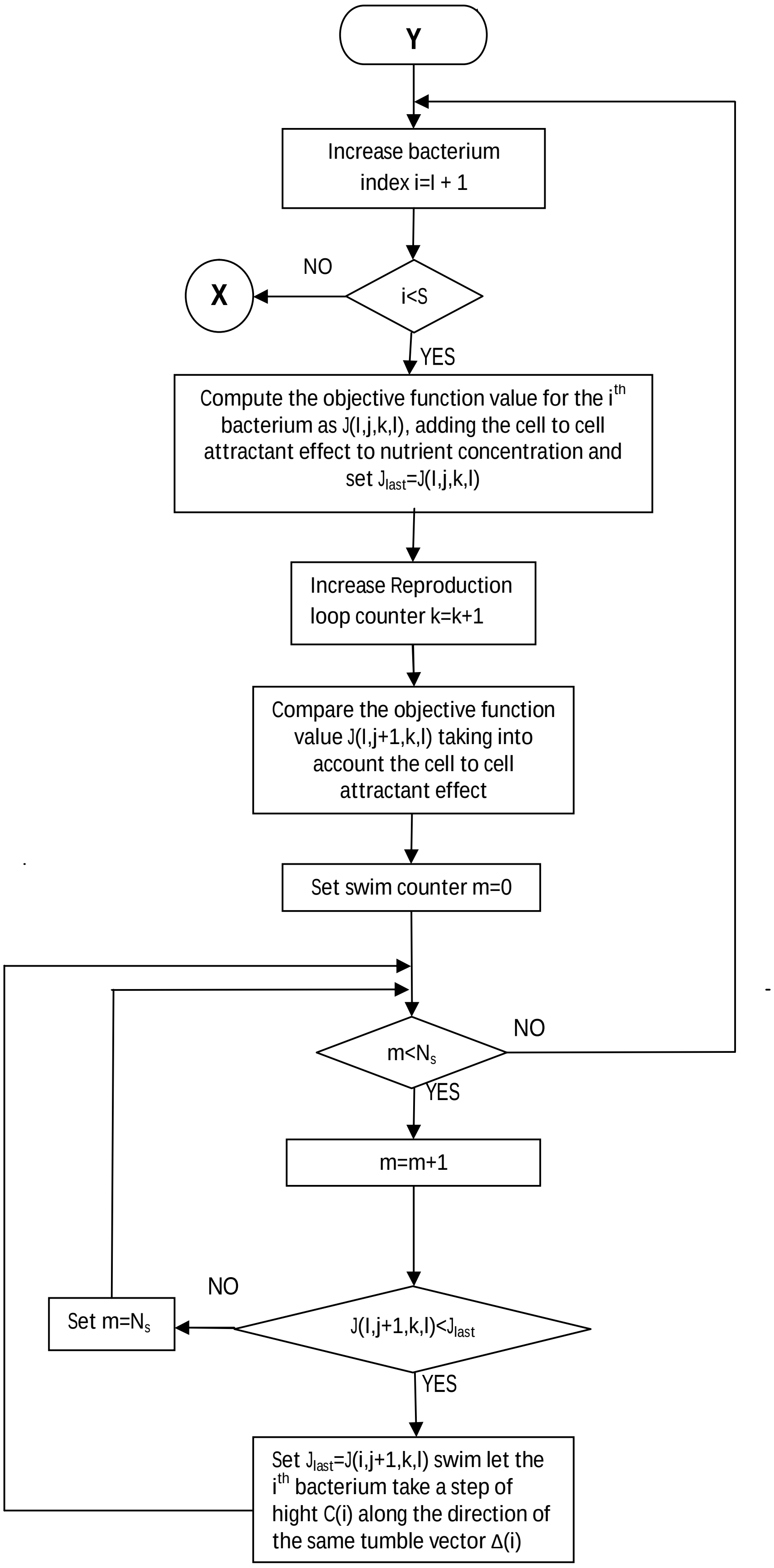

Y
Increase bacterium index i=I + 1
NO
X
i<S
YES
Compute the objective function value for the ith bacterium as J(I,j,k,l), adding the cell to cell attractant effect to nutrient concentration and set J_{last}=J(I,j,k,l)
Increase Reproduction loop counter k=k+1
Compare the objective function value J(I,j+1,k,l) taking into account the cell to cell attractant effect
Set swim counter m=0
NO
m<N_s
YES
m=m+1
NO
Set m=N_s
J(I,j+1,k,l)<J_{last}
YES
Set J_{last}=J(i,j+1,k,l) swim let the ith bacterium take a step of hight C(i) along the direction of the same tumble vector Δ(i)

Figure 23
Bacterial Foraging Optimization Algorithm Part II

3.7 Guidelines for Algorithm Parameter Choices

The bacterial foraging optimization algorithm requires specification of a variety of parameters. First, you can pick the size of the population, *S*. Clearly, increasing the size of *S* can significantly increase the computational complexity of the algorithm. However, for larger values of *S*, is more likely will start at least some bacteria near an optimum point, and over time, it is then more likely that many bacterium will be in that region, due to either chemotaxis or reproduction. If the *C*(*i*)values are too large, then if the optimum value lies in a valley with steep edges, the search will tend to jump out of the valley, or it may simply miss possible local minima by swimming through them without stopping. On the other hand, if the *C*(*i*) values are too small, convergence can be slow, but if the search finds a local minimum it will typically not deviate too far from it. We consider the *C*(*i*) as a type of “step size” for the optimization algorithm. If the attractant width is high and very deep, the cells will have a strong tendency to swarm (they may even avoid going after nutrients and favor swarming). On the other hand, if the attractant width is small and the depth shallow, there will be little tendency to swarm and each cell will search on its own. Social versus independent foraging is then dictated by the balance between the strengths of the cell-to-cell attractant signals and nutrient concentrations. Next, large values for *Nc* result in many chemotactic steps, and hopefully more optimization progress, but of course more computational complexity. If the size of *Nc* is chosen to be too short, the algorithm will generally rely more on luck and reproduction, and in some cases, it could more easily get trapped in a local minimum.

If *Nc* is large enough, the value of *Nre* affects how the algorithm ignores bad regions and focuses on good ones, since bacteria in relatively nutrient-poor regions die. If *Nre* is too small, the algorithm may converge prematurely; however, larger values of *Nre* clearly increase computational complexity. A low value for *Ned* dictates that the algorithm will not rely on random eliination-dispersal events to try to find favorable regions. A high value increases computational complexity but allows the bacteria to look in more regions to find good nutrient concentrations. Clearly, if *ped* is large, the algorithm can degrade to random exhaustive search. If, however, it is chosen appropriately, it can help the algorithm jump out of local optima and into a global optimum.

3.8 References for Chapter 3

[1] D. Stephens and J. Krebs, Foraging Theory. Princeton, NJ: Princeton Univ. Press, 1986.

[2] W. O'Brien, H. Browman, and B. Evans, "Search strategies of foraging animals," Amer. Scientist, vol. 78, pp. 152-160, Mar./Apr. 1990.

[3] E. Bonabeau, M. Dorigo, and G. Theraulaz, Swarm Intelligence: From Natural to Artificial Systems. New York: Oxford Univ. Press, 1999.

[4] C. Adami, Introduction to Artificial Life. New York: Springer-Verlag, 1998.

[5] M. Resnick, Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds. Cambridge, MA: MIT Press, 1994.

[6] S. Levy, Artificial Life: A Report from the Frontier where Computers Meet Biology. New York: Vintage Books, 1992.

[7] T. Audesirk and G. Audesirk, Biology: Life on Earth, 5th ed. Englewood Cliffs, NJ: Prentice Hall, 1999.

[8] H. Berg, "Motile behavior of bacteria," Phys. Today, pp. 24-29, Jan. 2000.

[9] M. Madigan, J. Martinko, and J. Parker, Biology of Microorganisms, 8th ed. Englewood Cliffs, NJ: Prentice Hall, 1997.

[10] F. Neidhardt, J. Ingraham, and M. Schaechter, Physiology of the Bacterial Cell: A Molecular Approach. Sunderland, MA: Sinauer, 1990.

[11] B. Alberts, D. Bray, J. Lewis, M. Raff, K. Roberts, and J.Watson, Molecular Biology of the Cell, 2nd ed. New York: Garland Publishing, 1989.

[12] H. Berg and D. Brown, "Chemotaxis in escherichia coli analysed by three-dimensional tracking," Nature, vol. 239, pp. 500-504, Oct. 1972.

[13] J. Segall, S. Block, and H. Berg, "Temporal comparisons in bacterial chemotaxis," Proc. Nat. Acad. Sci., vol. 83, pp. 8987-8991, Dec. 1986.

[14] H. Berg, Random Walks in Biology. Princeton, NJ: Princeton Univ. Press, 1993.

[15] D. DeRosier, "The turn of the screw: The bacterial flagellar motor," Cell, vol. 93, pp. 17-20, 1998.

[16] G. Lowe, M. Meister, and H. Berg, "Rapid rotation of flagellar bundles in swimming bacteria," Nature, vol. 325, pp. 637-640, Oct. 1987.

[17] T.-M. Yi, Y. Huang, M. Simon, and J. Doyle, "Robust perfect adaptation in bacterial chemotaxis through integral feedback control," PNAS, vol. 97, pp. 4649-4653, April 25, 2000.

[18] J. Armitage, "Bacterial tactic responses," Adv. Microbial Phys., vol. 41, pp. 229-290, 1999.

[19] E. Budrene and H. Berg, "Dynamics of formation of symmetrical patterns by chemotactic bacteria," Nature, vol. 376, pp. 49-53, 1995.

[20] Y. Blat and M. Eisenbach, "Tar-dependent and -independent pattern formation by salmonella typhimurium," J. Bacteriology, vol. 177, pp. 1683-1691, Apr. 1995.

[21] E. Budrene and H. Berg, "Complex patterns formed by motile cells of escherichia coli," Nature, vol. 349, pp. 630-633, Feb. 1991.

[22] D. Woodward, R. Tyson, M. Myerscough, J. Murray, E. Budrene, and H. Berg, “Spatio-temporal patterns generated by salmonella typhimurium,” Biophysi. J., vol. 68, pp. 2181-2189, 1995.

[23] R. Losick and D. Kaiser, “Why and how bacteria communicate,” Sci. Amer., vol. 276, no. 2, pp. 68-73, 1997.

[24] M. McBride, P. Hartzell, and D. Zusman, “Motility and tactic behavior of myxococcus xanthus,” in Myxobacteria II, M. Dworkin and D. Kaiser, Eds. Washington, DC: American Society for Microbiology, 1993, pp. 285-305.

[25] L. Shimkets and M. Dworkin, “Myxobacterial multicellularity,” in Bacteria as Multicellular Organisms, J. Shapiro and M. Dworkin, Eds., New York: Oxford Univ. Press, 1997, pp. 220-244.

[26] H. Reichenbach, “Biology of the myxobacteria: Ecology and taxonomy,” in Myxobacteria II, M. Dworkin and D. Kaiser, Eds. Washington, DC: American Society for Microbiology, 1993, pp. 13-62.

[27] J. Shapiro, “Bacteria as multicellular organisms,” Sci. Amer., vol. 258, pp. 62-69, 1988.

[28] A. Stevens, “Simulations of the gliding behavior and aggregation of myxobacteria,” in Biological Motion (Lecture Notes in Biomathematics, vol. (89), W. Alt and G. Hoffmann, Eds. Berlin: Springer-Verlag, 1990, pp. 548-555.

[29] A. Stevens, “A stochastic cellular automaton, modeling gliding and aggregation of myxobacteria,” SIAM J. Appli. Math., vol. 61, no. 1, pp. 172-182, 2000.

[30] J. Spall, S. Hill, and D. Stark, “Some theoretical comparisons of stochastic optimization approaches,” in Proc. American Control Conf., Chicago, IL, June 2000, pp. 1904-1908.

[31] R. Murray-Smith and T.A. Johansen, Eds., Multiple Model Approaches to Nonlinear Modeling and Control. London, UK: Taylor and Francis, 1996.

[32] K. Kristinsson and G. Dumont, “System identification and control using genetic algorithms,” IEEE Trans. Syst., Man, Cybernet., vol. 22, no. 5, pp.1033-1046, 1992.

[33] L.L. Porter and K. Passino, “Genetic adaptive and supervisory control,” Int. J. Intell. Contr. Syst., vol. 12, no. 1, pp. 1-41, 1998.

[34]W. Lennon and K. Passino, “Genetic adaptive identification and control,” Eng. Applicat. Artif. Intell., vol. 12, pp. 185-200, Apr. 1999.

[35] S. Brueckner and H. Parunak, “Multiple pheromones for improved guidance,” in Proceedings AEC, 2000 [Online]. Available: www.erim.org/cec/projects/adaptiv/

[36] M. Krieger, J.-B. Billeter, and L. Keller, “Ant-like task allocation and recruitment in cooperative robots,” Nature, vol. 406, pp. 992-995, Aug. 2000.

[37] J. P. Anderson,* D. W. Stephens,* and S. R. Dunbar” Saltatory search: a theoretical analysis” Behavioral Ecology VoL 8 No. 3: 307-517

[38] Kevin M. Passino “Biomimicry of bacterial foraging for distributed optimization and control”.

CHAPTER 4

BACTERIAL FORAGING OPTIMIZATION ALGORITHM AS DE-NOISING FILTER

4. BFOA as De-noising Filter

During image acquisition process due to different processing such as A/D conversion, transmission etc. the digital images gets added with noise. Different types of noises corrupt digital images at different stages of image processing.[1][3] While dealing with medical images, the image gets corrupted by variety of noise mainly from bone, soft tissue, body movement etc. Therefore precautions are taken to reduce the noise to a larger possible extent in the medical images for better diagnosis. However, the demands for noise free image are increasing day by day so is the case for better de-noising filters. Soft Computing techniques in the recent past have been used by researchers to de-noise images. In this thesis, I have presented a technique to de-noise medical images using Bacterial Foraging Optimization technique.
Bacterial Foraging Optimisation developed by Passino in 2002 [8] Bio-inspired Optimisation technique that is derived from the food searching process of E. Coli bacteria. As the bacteria travel in slow speed, it gives the capability to search the pixel without jumping or slipping out pixels thus, improving the quality of image. To test the capability of proposed algorithm, medical images like CT-Scan images of pancreas and brain are considered. Other images are also considered to observe the performance of algorithm as a de-noising filter.

Along with several digital filtering techniques, Soft computing techniques are gaining importance in de-noising process. Several digital filtering techniques used by researchers to remove Salt & Pepper noise have been published. Progressively determining noisy pixel and removal of noisy pixel by switched median filter used in[10]. Modified Median filter used in [2]. Artificial neural network, Fuzzy logic, Genetic Algorithm, Optimisation techniques such as Particle Swarm Optimisation [4] are some important Soft Computing techniques. PSO is applied to de-noise images[11][12] by optimising cost function as structure of similarity, Bacterial Foraging Optimisation [8], Swine Influenza Model Based Optimisation [9] are some important Soft computing techniques. Researchers used various soft computing techniques and hybrid techniques to de-noise images [1]. In this thesis, BFO is used to optimise the output of Adaptive Median filter when images are corrupted with Salt and Pepper noise and output of Average filter is optimized when images are corrupted with Gaussian noise. Experiment is performed on –CT-Scan of pancreas and brain and two other images.

4.1. The BFOA

Bacterial Foraging Optimisation is based on foraging strategy of E.Coli bacteria. Bacteria move in random direction to search favourable direction of increasing nutrients. Thus, this Optimisation technique is useful when gradient of cost function is not known. Bacterial Optimisation is good because of its less mathematical complexity, convergence, accuracy and wide application. This Optimisation is accomplished in four steps-

- Chemo-taxis
- Swarming
- Reproduction and
- Elimination and Dispersal.

(i) **Chemo taxis**: Single Chemo-tactic step completes in tumble, run and tumble if nutrient does not increase in the direction of swim and otherwise it is tumble, run(swim) and followed by run (as per defined limit of swim) if concentration of nutrient increases in the direction of swim. A unit walk with random direction represents a Tumble and unit walk with same direction in the last step indicates Run. Mainly foraging completes in Chemo-tactic step.

(ii) **Swarming**: The cells when stimulated, release an attractant aspirate, which helps them to aggregate into groups and move as concentric patterns of swarms. This helps to achieve global optimum value for cost function. Swarming helps in fast convergence in case of multidimensional cost function.
(iii) **Reproduction**: After calculating fitness value for each bacteria, reproduction allows the half healthy bacteria (with least cost value) to survive and reproduce. The remaining half unhealthy bacteria die. This step helps in the generation of values of variables which are closer to actual value. Fifty percentage of the population is removed in each state and rest fifty percentage reproduce.
(iv) **Elimination and Dispersal**: The chemo taxis provides a basis for local search and reproduction speeds the convergence. But to avoid the trap of bacteria in local minimum Elimination –Dispersal is done.

4.2. Parameter choice of BFOA as de-noising Filter

To test the proposed concept, initially ideal images are corrupted with noise. Now the difference in terms of Mean Square Error between this noisy image and the original image is minimized using BFO.

BFO is used to minimize the Mean Square Error between noisy image and target image to restored image which is closer to actual image. Parameters selected to de noise image are-

Number of bacteria in population used for searching (S) = M×N size of image, in this case it is 100×100 (downsized)
Dimension of search space (p) = 1
Number of Chemo tactic steps (Nc) = 2
Number of swimming steps (Ns) = 1
Number of reproduction steps (Nre) = 2
Number of elimination and dispersal (Ned)=2
Probability of elimination and dispersal (ped) = 0.25
Cost function used to minimize using BFO is MSE.

$$\mathrm{MSE} = \frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N}\left(f'(x,y) - f(x,y)\right)^2$$

Where f'(x, y) = noisy image
F(x, y) = target image
Due to pixel –by- pixel operation, f'(x, y) = P(i, j , k ,l) and f(x, y) = R(x, j, k, l)
J(i, j, k, l) =| P(i, j , k ,l) - R(x, j, k, l) |2
Where
P(i, j , k ,l) = location of ith pixel (bacteria) at jth chemo tactic step, kth reproduction step and lth elimination step.
R(x, j, k, l) = location of xth target pixel at jth chemo tactic step, kth reproduction step and lth elimination step.
J(i, j, k, l) = Cost of ith pixel (bacteria) at jth chemo tactic step, kth reproduction step and lth elimination step.

4.3 Algorithm used for de noising images

For xth target pixel optimization takes in following steps:
Initialize parameters p, S, NC, NS, Nre, Ned, ped, and C(i), i= 1,2,3…………S.
C(i) = Step size in the random direction
Step 1: Elimination –dispersal loop :l=l+1
Step 2: Reproduction loop : k=k+1
Step 3: Chemo taxis loop : j=j+1
a) For i=1,2,…….,S, take a chemo tactic step for bacterium i as follows.
b) Let Jlast=J(I,j,k,l)= | P(i, j , k ,l) - R(x, j, k, l) |2
where P(i, j , k ,l) = location of ith pixel (bacteria) at jth chemotactic step, kth reproduction step and lth elimination step.
R(x, j, k, l) = location of xth target pixel at jth chemo tactic step, kth reproduction step and lth elimination step.
J(i, j, k, l) = Cost of ith pixel (bacteria) at jth chemo tactic step, kth reproduction step and lth elimination step..
c) Tumble: A random vector Δm(i), m= 1,2,…p, a random number in [-1,1].
d) Move: Let (j+1,k,l)= (j,k,l) + C(i) . results in a step C(i) in the direction of the tumble for bacterium i.
e) Swim:
i. Let m= 0 (counter for swim length).
ii. While m< NS
• Let m= m+1
• If J(i,j+1,k,l) < Jlast ,
Let Jlast = J(i,j+1,k,l) and let Pi(j+1,k,l) + C(i)
And use this Pi(j+1,k,l) to compute the new J(j+1,k,l).
• Else, let m = NS.
f) Go to next bacteria (i+1) if $i \neq S$
Step 4: If j < NC, go to step 3.
Step 5: Reproduction:
i) For the given k and l, and for each i= 1,2,…………S, let J(i,j,k,l)
ii) The Sr = S/2 bacteria with the highest value of cost function die and other Sr = S/2 bacteria with the best value (least value of cost function) split.
Step 6: If k < Nre, go to step 2.
Step 7: Elimination- dispersal: Eliminate and dispersal each bacterium.
Step 8: If l < Ned , go to the step 1.
Otherwise end.

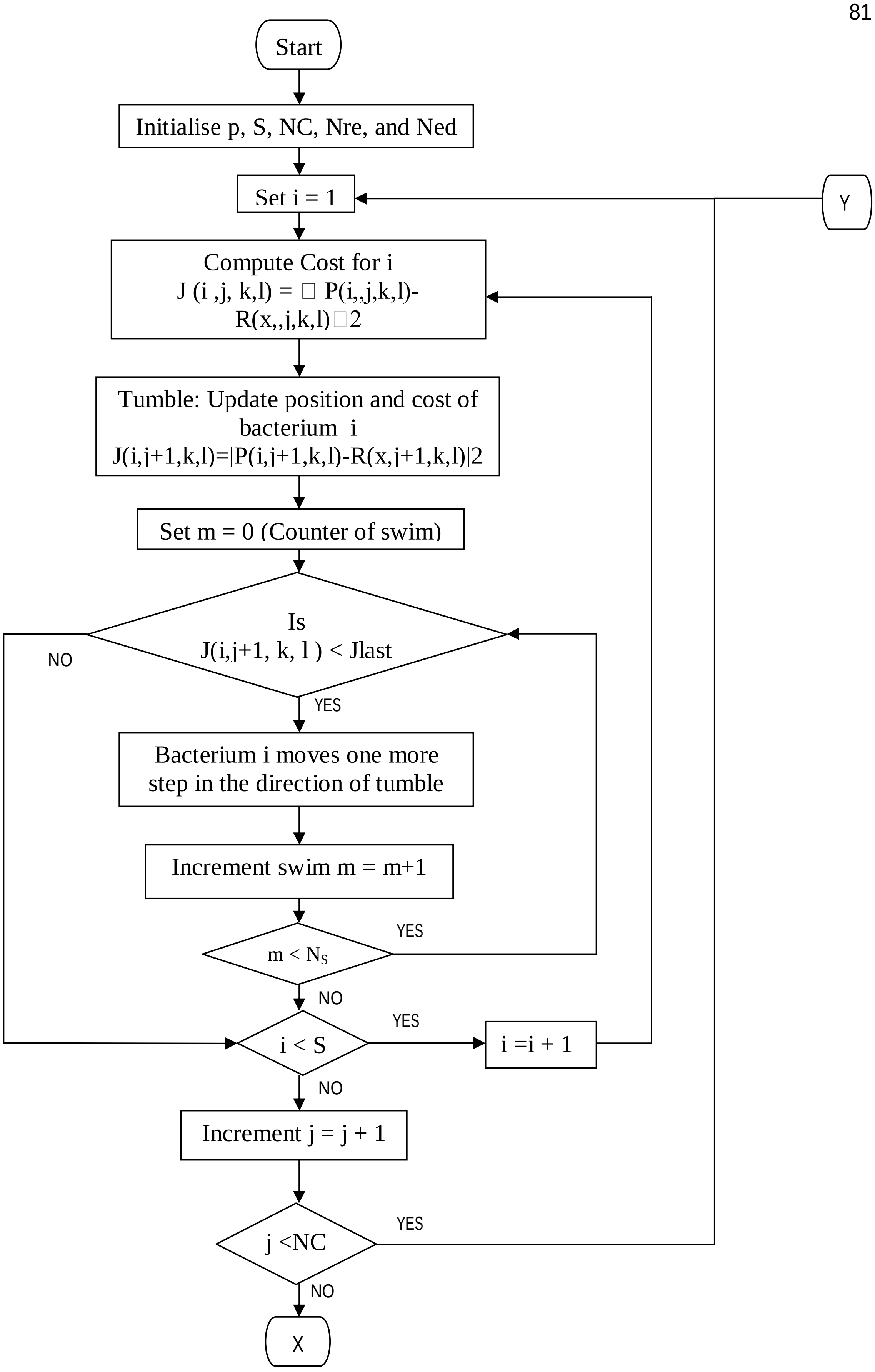

Start
Initialise p, S, NC, Nre, and Ned
Set i = 1
Y
Compute Cost for i
J (i ,j, k,l) = □ P(i,,j,k,l)-R(x,,j,k,l)□2
Tumble: Update position and cost of bacterium i
J(i,j+1,k,l)=|P(i,j+1,k,l)-R(x,j+1,k,l)|2
Set m = 0 (Counter of swim)
Is J(i,j+1, k, l) < Jlast
NO
YES
Bacterium i moves one more step in the direction of tumble
Increment swim m = m+1
YES
m < N_S
NO
YES
i < S
i =i + 1
NO
Increment j = j + 1
YES
j <NC
NO
X

Figure 24 BFOA as Denoising filter Flow Chart Part I

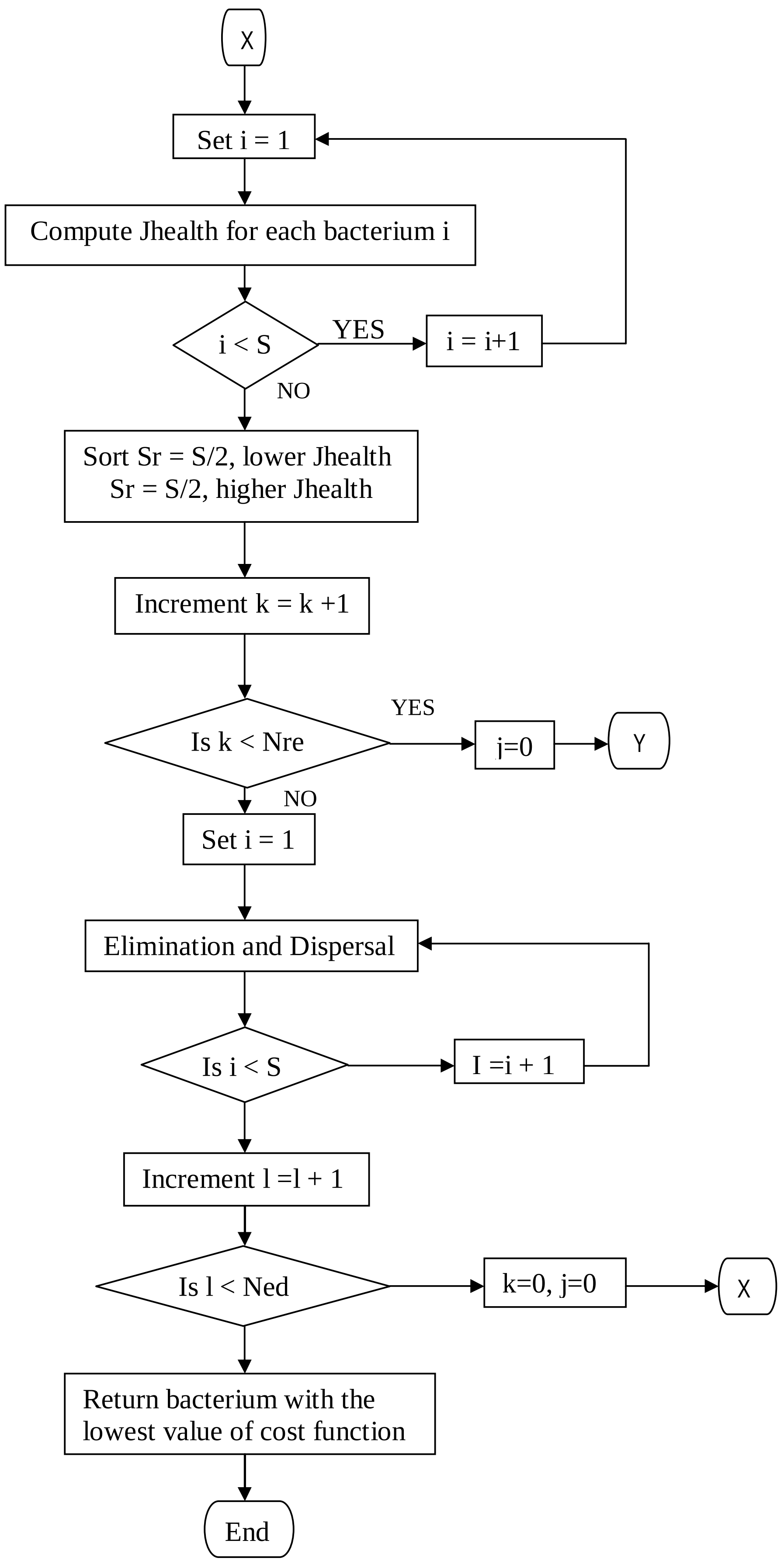
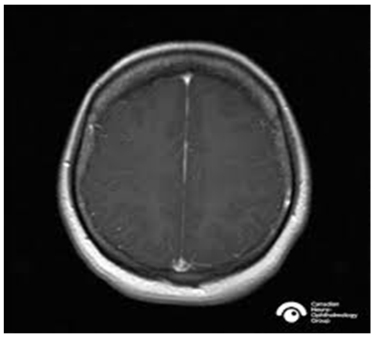
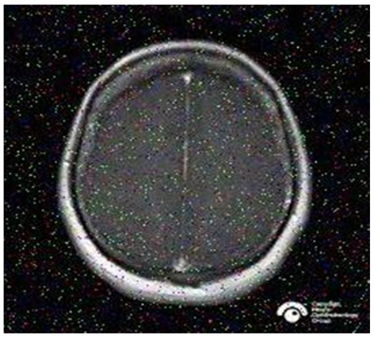


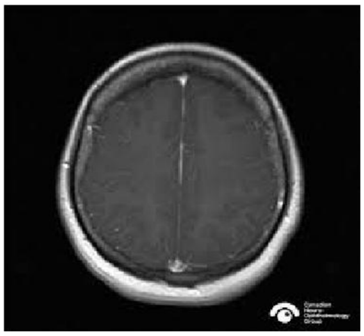
Figure 24 BFOA as Denoising filter Flow Chart Part II

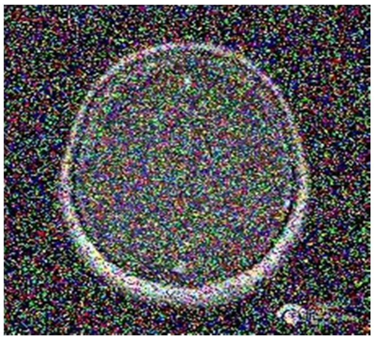
4.4. Experimental Results

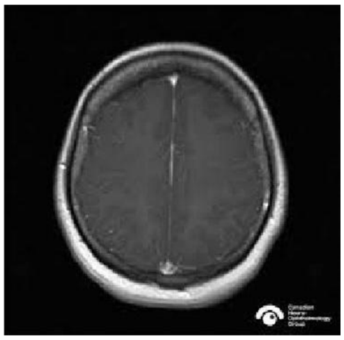
CT scan images of brain and pancreas are initially considered. To reduce the processing time, the images are downsized to (100×100). Salt and pepper noise with varied noise density 0.04 to 0.4 are used. In case of Gaussian noise variance is changed from 0.002 to 0.07. Then medical images like CT-Scan of pancreas and brain are considered. Pancreas being one of the innermost organs, hence is prone to different types of noise. The original medical image is corrupted with the varied noise density or variance.

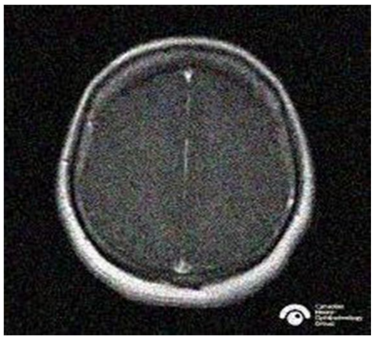

(a)

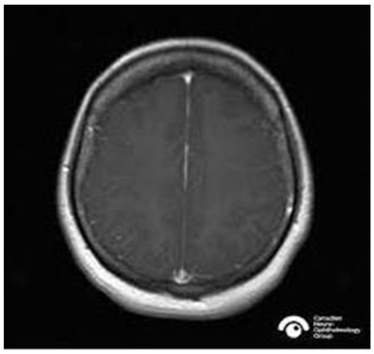

(b)

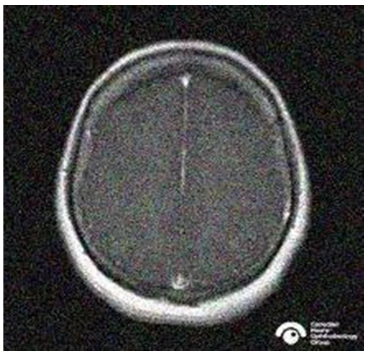

(c)


(d)


(e)


(f)


(g)


(h)

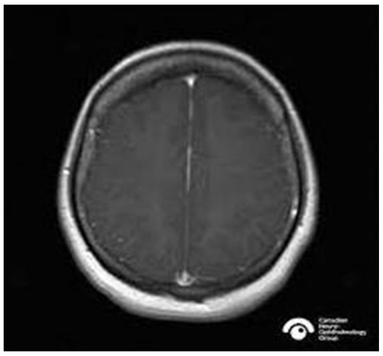

(i)

Figure 26 (a) CT Scan image of brain, (b) Image corrupted with 0.04 Salt & Pepper Noise, (c) Image restored after BFO, (d) corrupted with 0.4 Salt & Pepper Noise, (e) Image restored after BFO, (f) Image corrupted with Gaussian Noise at S.D.= 0.004, (g) Image restored after BFO, (h) Image corrupted with Gaussian Noise at S.D.=0.07, (i) Image restored after BFO.

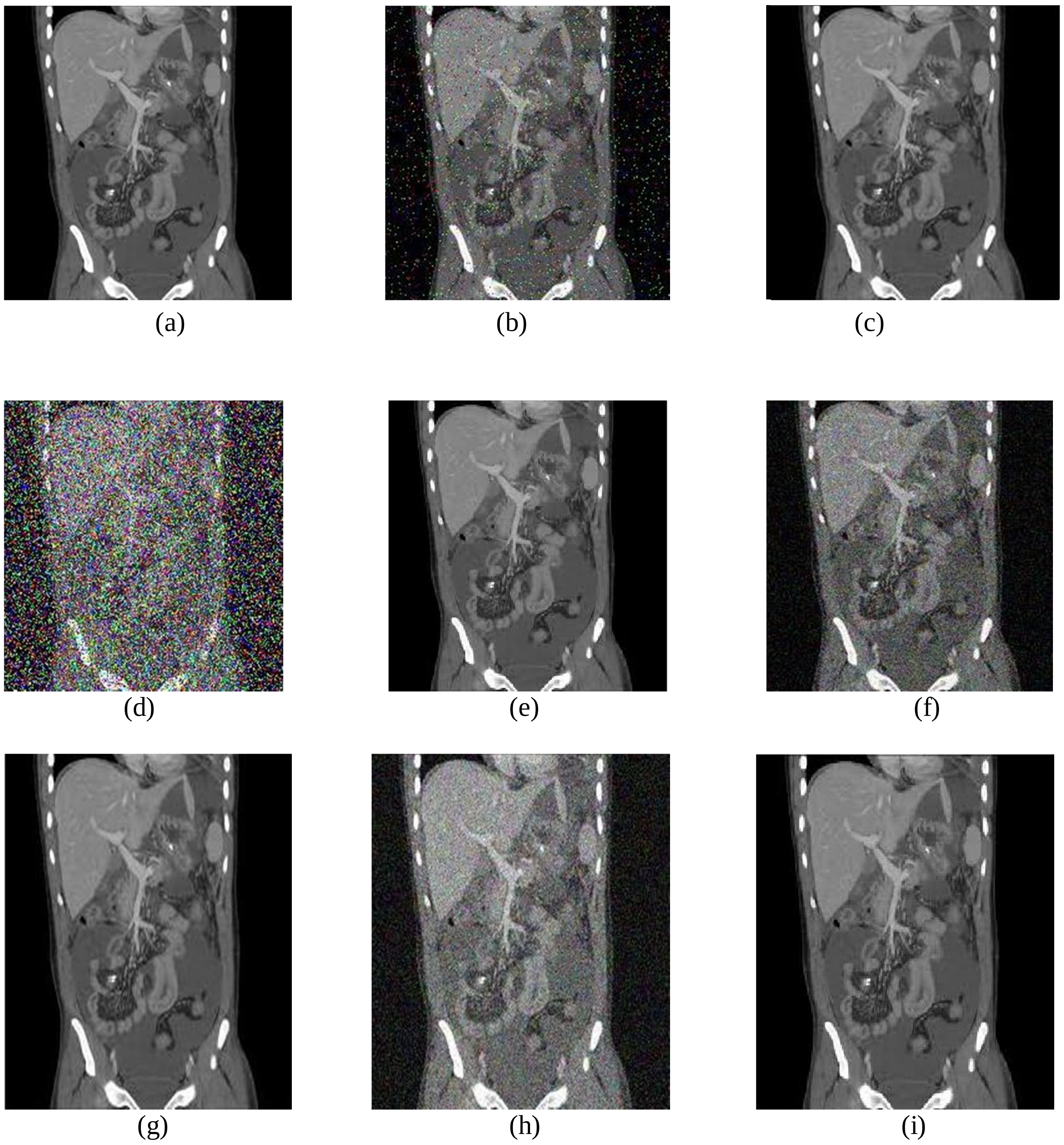
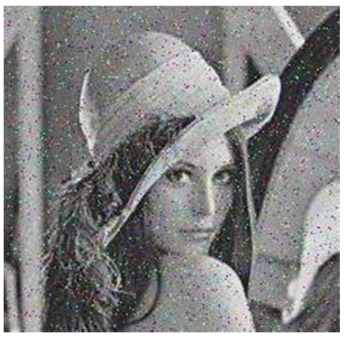
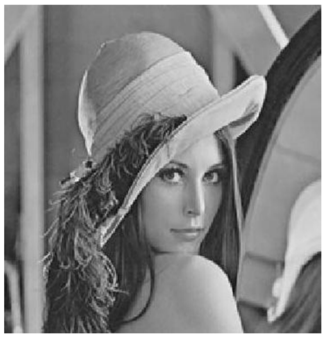
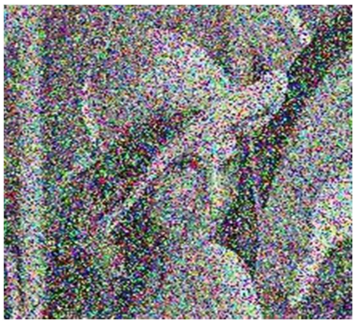
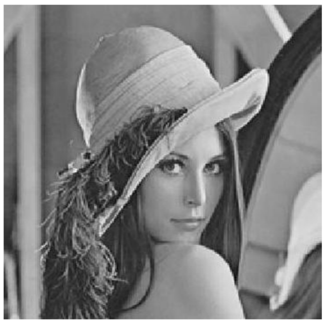
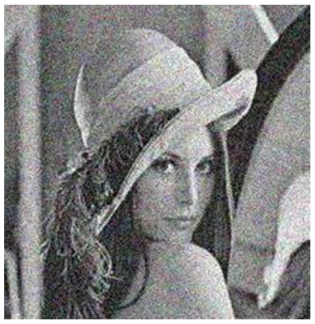


Figure 27 (a) CT Scan image of Pancreas, (b) Image corrupted with 0.04 Salt & Pepper Noise, (c) Image restored after BFO, (d) corrupted with 0.4 Salt & Pepper Noise, (e) Image restored after BFO, (f) Image corrupted with Gaussian Noise at S.D.= 0.004, (g) Image restored after BFO, (h) Image corrupted with Gaussian Noise at S.D.=0.07, (i) Image restored after BFO.

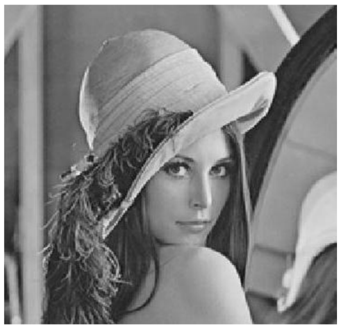
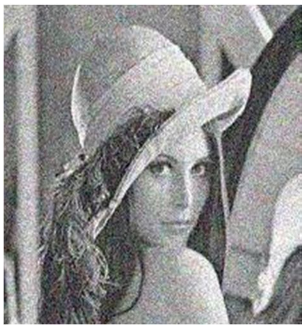
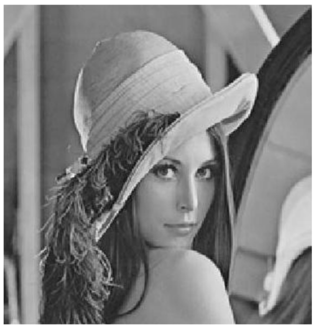
Fig.26 and Fig.27 (a) show the original images of CT scan of brain and Pancreas. Fig.26 and Fig.27 (b),(d),(f),(h) show considered images corrupted with Salt & Pepper noise at 0.04 and 0.4, and Gaussian noise at S.D. 0.004 and 0.07 respectively, Fig.26 and Fig.27 (c),(e),(g),(i) show images restored after BFO filtering.

(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 27 (a) A benchmark image of Lenna, (b) Image corrupted with 0.04 Salt & Pepper Noise, (c) Image restored after BFO, (d) corrupted with 0.4 Salt & Pepper Noise, (e) Image restored after BFO, (f) Image corrupted with Gaussian Noise at S.D.= 0.004, (g) Image restored after BFO, (h) Image corrupted with Gaussian Noise at S.D.=0.07, (i) Image restored after BFO.

The thesis presents an application of BFO as a digital filter to de-noise medical image i.e., CT-Scan and MRI of pancreas. The experimentation in terms of quality matrices like MSE and PSNR show considerable improvement in the quality of restored images. The computational overloading is also low. Thus, this approach of using Soft Computing in conjunction with digital filter will definitely enhance the de-noising capability of digital filters which can find application in sensitive images like medical images.

4.5. REFERENCES

[1] Choubey, Abha, Sinha, G. R., Choubey, Siddharth; "A Hybrid filtering Technique in Medical Image denoising Blending of Neural Network and Fuzzy Inference." ICECT Vol.1, pp. 170-177, April 2011

[2] Daiyan, G.M., Mottalib, M.A.; "Removal of high density Salt and Pepper noise through modified median filter." International Conference on Informatics, Electronoics and vision, pp. 565-570, 2012

[3] Gonzalez,Rafel C. and Woods, Richard E.; "Digital Image Processing." Second edition, Publising House of Electronics Industry, Beijing, 2003

[4] Kennedy,J and Eberhart, R.C; "Particle Swarm Optimisation." Proceedings of IEEE International Conference on Neural Network, , 1995 Piscataway, NJ, pp.1942-1948

[5] Kneey Kal Vin Toh and Nor Ashidi Mat Isa; "Noise Adaptive Fuzzy Switching Median filter for Salt & Pepper Noise Reduction." IEEE Signal Processing Letters, Vol. 17, No. 3, pp. 281-284 , March 2010

[6] Lu Zhang, Jiaming Chen, Yuein Zhu, Zianhua Luo; "Comparisions of several New Denoising Methods for Medical Images." International Conference on Bioinfomatics and Biomedical Engineering, June 2009, pp. 1-4,

[7] Newton, T.H. and Potts, D.G.; "Technical Aspects of Computed Tomography in Radiology of skull and Brain." Vol.5 ISBN-0-8016-3662-0, pp. 3941-3956, 1981

[8] Passino, K.M.; "Biomimicry of Bacterial Foraging for Distributed Optimisation and Control." IEEE Control and System Magzene, pp. 52-67, June 2002

[9] Pattnaik, S.S; Backwad, K.M.; Sohi, B.S.; Ratho, R.K.; S.Devi; "Swine Influenza Model Based Optimisation (SIMBO)." Applied Soft Computing, Vol.1, pp.18-30, Sept. 2012

[10] Zhou Wang David Zhang; "Progressive Switching Median filter for the removal of Impulse Noise from Highly corrupted Images." IEEE Transaction on Circuits and System –II: Analog and Digital Processing, Vol.46,No.1 pp. 78-80 Jan 1999

[11] Zhu Youlin; Huang Cheng; "Image denoising algorithm based on PSO Optimising Structuring element." Control and Decision Conference, , 2012, pp.2404-2408

[12] Saeid Fazli, H. Bouzari and H.M. Pour ―Complex PDE Image
Denoising Based on Particle Swarm Optimization□,International
Congress on Ultra Modern Telecommunications and Control Systems

CHAPTER 5

CONCLUSION

5. Conclusion

Search and optimization problems are ubiquitous through the various realms of science and engineering. This thesis has provided a comprehensive overview of one promising real-parameter optimization algorithm called the Bacterial Foraging Optimization Algorithm (BFOA). BFOA is currently gaining popularity due to its efficiency over other swarm and evolutionary computing algorithms in solving engineering optimization problems. It mimics the individual as well as grouped foraging behavior of *E.coli* bacteria that live in our intestine.
The thesis first outlines the classical BFOA in sufficient details. It then develops a simple mathematical model of the simulated chemotaxis operation of BFOA.

With the help of this model it analyses the chemotactic dynamics of a single bacterium moving over a one-dimensional fitness landscape. The analysis indicates that the chemotactic dynamics has some striking similarity with the classical gradient descent search although the former never uses an analytic expression of the derivative of the objective function. A problem of oscillations near the optimum is identified from the presented analysis and two adaptation rules for the chemotactic step-height have been proposed to promote the quick convergence of the algorithm near the global optimum of the search space.

In recent times, a symbiosis of swarm intelligence with other computational intelligence algorithms has opened up new avenues for the next generation computing systems. The chapter presents an account of the research efforts aiming at hybridizing BFOA with other popular optimization techniques like PSO, DE, and GA for improved global search and optimization. It also discusses the significant applications of BFOA in diverse domains of science and engineering. The content of the chapter reveals that engineering search and optimization problems including those from the fields of pattern recognition, bioinformatics, and machine intelligence will find new dimensions in the light of swarm intelligence techniques like BFOA.

The Bacterial Foraging Optimization technique has gained popularity in solving optimization problems. The algorithm and flowchart for the proposed work is explained and control parameters used are tabulated. Bacterial Foraging Algorithm is based on a computational intelligence technique that is not largely affected by the size and non-linearity of the problem and has converged to the optimal solution to many problems where the most analytical methods fail to converge and also has its advantages such as less computational burden, global convergence, less computational time requirement and can handle more number of objective function.

There are a wide variety of fruitful research directions. There are ways to improve the models (e.g., modeling more dynamics of cell motion). Other species of bacteria could be studied, and indeed it would be interesting to see if Grunbaum’s principle works for others species of social foraging animals.

Moreover, it remains to be seen how practically useful the optimization algorithms are for engineering optimization problems. Claims of practical utility of any optimization algorithm are difficult to make; they depend on the theoretical properties of the algorithm, theoretical and empirical comparisons to other methods, and extensive evaluation on many benchmark problems and real-world problems.