Overview of Memetic Artificial Bee Colony Algorithm: aka “MemeBees”

Purpose: To optimize a multi-variable, single output non-linear, non-continuous black box cost function using a computationally feasible and sufficiently accurate algorithm.

# What is Meme Bees?

* It is a meta-heuristic, evolutionary/swarm optimization global optimization algorithm hybridized with direct search algorithms for local optimization. This hybridization is what makes the algorithm memetic. There are two direct search algorithms available for local optimization, and one of them is chosen based on the current diversity of the candidate solutions. Mostly derived from the paper

## Meta-Heuristic:

* High level procedure that can tackle a variety of functions, without making many assumptions about the function
* Provides a sufficiently good solution, especially with incomplete information. (I.E, do not need to know anything about the derivatives of the function, only its current value)
* Does not guarantee a global optimum solution.
* Parameters can be tuned to improve convergence to global optimum, especially with deeper knowledge of the function.

## Evolutionary Algorithm:

* Improves not just one, but a *population* of candidate solutions through algorithms inspired by Darwinian Natural Selection, *Survival of the Fittest*
* Mutation: Solutions are changed with a degree of stochastic randomness that is adjustable
* Crossover: Solutions are influenced by other solutions to create new candidate solutions(analogous to reproduction)
* Selection: New Candidate solutions which are better survive onto the next generation. This is where the Survival of the Fittest kicks in and is key to understanding the nature of this algorithm.
* Examples include Genetic Algorithm
* The core of the evolutionary component is a variant of the Artificial Bee Colony mixed with genetic algorithm, with two key processes for each generation

## Direct Search:

* Optimizes a *single* solution at a time by generating new solutions and comparing them to the old solutions
* Generates new solutions stochastically, either by exploring in a random direction(Random Walk) or by flipping different variables(Annealing)
* If new solution is better through some criteria, replace the current solution with the new solution, or replace it anyway with a low chance based on a probability function(Annealing). This makes them hill-climbing.
* Two direct search methods are used, and they are selected based on the diversity of the solution so far

### Simulated Annealing:

* Exploratory in nature, can accept worse solutions with a probability function C:\Users\asifm2\Desktop\New folder\MemeBees\MemeBeesMoFunctions\html\anneal1.PNG
* Temperature is a state variable that decays as the algorithm. There’s a higher chance of accepting worse solutions with a higher temperature. Temperature gradually decays as the algorithm runs, making the algorithm more open to worse solutions early on
* Accepting worse solutions earlier on gives the algorithm a chance of not getting trapped in a local optimum.
* Algorithm stops when the temperature drops below a certain threshold

### Optimization problem solution space 3Random Walk with Direction Exploitation (RWDE)

* Exploitative algorithm, does not accept worse solutions.
* Generates new solution by randomly choosing vector direction, moving the solution along that direction by a step size.
* If that direction yields a better solution, that direction is further exploited for a better solution. Exploitation is essentially optimizing the length of the step to yield best the result.
* Step size decreases throughout the running of the algorithm.
* Algorithm halts when step size reaches a certain threshold.

## Hybridization Scheme

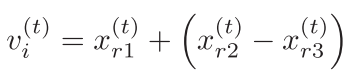
* Two Evolutionary/Swarm Processes (Global Optimization) are combined with a single local search algorithm (Local Optimization), with the single local algorithm being chosen based on the diversity of the solution set so far (more on diversity later). These processes can be described in terms of the ranges of exploration.
* There is also an additional process which replaces a candidate solution if it has not changed for 200 times. The replacement is entirely random within the solution space constrained by upper and lower bounds.
* Evolutionary Processes act on the entire population of candidate solutions, local search algorithm acts on the current best solution so far.
* Local search algorithms run at a fraction of the times the evolutionary process runs, this ratio is adjustable

## Global Optimization:

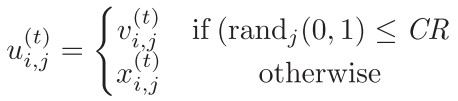
* Population-based global optimization which uses three processes to successively improve solution
* Long Range Stochastic Exploration – sendEmployedBees() – all of the populations undergoes mutation, crossover, and selection
* Medium Range Stochastic Exploration – sendOnlookerBees() – A fraction of the population undergoes mutation, crossover, and selection based on a fitness criteria and some randomness
* Long Range Exploration – sendScouts() – “Expired Solutions” are replaced entirely randomly within the solution space constrained by upperbound and lowerbound.

### sendEmployedBees() :

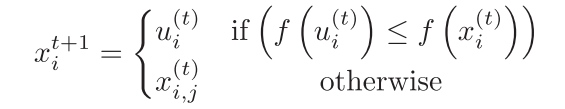
* Long-range stochastic exploration stage
* Mutation: Three candidate solutions are randomly chosen and their addition and subtraction results in a new trial solution. This happens for each candidate solution



* Cross-over: The new trial solutions and old solutions are “crossed” with each other to yield a child solution, such that some elements of the child solution are from new trial, but most are from the old solution. Controlled by Parameter CR, which is the rate at which elements are inherited from the old solution.

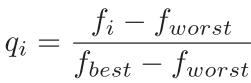


* Selection: Compares fitness of new child solutions and old solutions. If child solutions have better fitness, they become the old solutions.

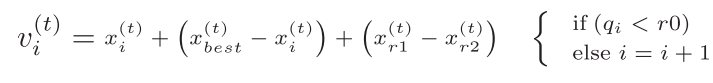


### sendOnlookerBees() :

* Moderate-range stochastic exploration stage since the search effects a subsection of the candidate solutions, based on a fitness criteria.



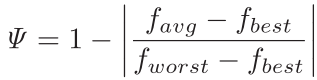
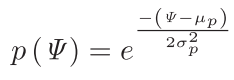
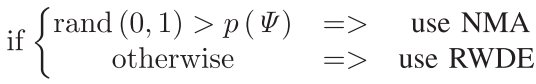
* Different mutation operator based on position of global best at current generation, and fitness criteria is evaluated, which determines the probability a candidate solution is chosen for mutation. More similar to particle swarm here



### sendScouts() :

* If a certain candidate solution is not improved a certain number of times, it is replaced by an entirely random solution within the constraints of the upper and lower bound on variables.

## Local Optimization:

* Periodically improve a single best solution to guide the rest of the population and converge towards a global optimum.
* Choose between two different local search heuristics depending on the current diversity of the candidate solutions.
* Diversity of the current generation of solutions is evaluated, and then is placed in an array which keeps growing. The mean and variance of that array are calculated periodically through the probability function
* Probability function is used to decide between Simulated Annealing or Random Walk with Direction Exploitation

# Testing Procedure

1. Debug and test program to ensure there are no syntactical or run-time errors
2. Validate the algorithm’s ability to optimize by testing it with sample functions commonly used to test optimization algorithm
3. Test the algorithm’s robustness by increasing the number of variables in the functions ot be tested.
4. Observe how different parameters effect the convergence of the algorithm, and how to fine-tune the algorithm for different types of functions(uni-modal, multi-modal, valley shaped, bowl shaped, plate, discontinuous)