**Relevant background material [5]**

The Artificial Bee Colony (ABC) algorithm is an optimisation algorithm based on the foraging behaviour of the honeybee swarm. It belongs to a class of heuristics known as swarm optimisation

Three groups of bees: employed bee, onlookers, and scouts.

Assumption that there is only one artificial employed bee for each for food source (that is, the number of employed bees is equal to the number of food sources available to the hive).

Employed bees go to their food source, come back to the hive, and dance on this area. The bee whose food source has been abandoned becomes a scout and searches for a new food source. Onlookers observe the dances of the employed bees and choose food sources depending on dances.

The position of a food source represents a possible solution to the optimisation problem (presumably it is located in $k$-dimensional space)

The nectar amount of a food sources corresponds to the quality (fitness) of the associated solution.

Each solution is a D-dimensional vector, where D is the number of optimisation parameters.

ABC generates a randomly distributed initial population of $S\_n$ food sources, where $S\_n$ is the size of the swarm.

Selection is stochastic resampling (roulette-wheel selection).

**A detailed pseudocode description of the ABC algorithm [15]**

Fortunately, I have good pseudo-code.

I think I should do this now. Setting up LaTeX is… always a ball-ache.

There are three control parameters used in the basic ABC: the number of food sources / the number of employed or onlooker bees (S\_N), the value of *limit*, and the maximum cycle number (C\_max)

Onlookers and employees carry out the exploitation process in the search space. Scouts control the exploration process.

**A natural language description of the ABC algorithm [10]**

In ABC, a ‘food source’ is a solution.

The *position* of a food source represents a possible solution to the optimisation problem. The nectar amount of a food source corresponds to the fitness of the corresponding solution.

The number of employees is equal to the number of solutions in the population.

At the first step, a randomly distributed initial population (food source position) is generated.

The population is then subject to repeated cycles of the search processes of the employees, onlookers, and scouts.

**Employee exploitation**

The employee produces a modification of the source position in her memory and so identifies a new food source position. If the new food source has more nectar, the bee memorises the new one. This is *exploitation*.

**Onlooker selection**

After all employed bees complete the search process, they share the position information of the sources with the onlookers on the dance area. Each onlooker evaluates the nectar information of the employees and selects a food source using roulette-wheel selection. Each onlooker generates a new source position and checks its nectar amount. If the nectar amount is higher, the bee memorises the new position and forgets the old one.

**Scout exploration**

The number of discarded sources is determined. New sources are randomly produced to replace the abandoned ones.

The ‘nectar’ of a ‘food source’ is the fitness of the solution that ‘food source’ represents.

In the ABC algorithm, a colony consists of three groups of bees: employees, onlookers, and scouts.

Each employee goes to a remembered food source. It evaluates the nectar content of this source, returns to the hive, and dances.

This dance conveys information about the food source: its distance

An employee whose food source has been abandoned becomes a scout.

Each onlooker watches the dances of the employees.

Not very good at all.

Onlookers watch the dances of employees and choose food sources based on these dances.

Exploration: searching a wide area of a search space for promising solution

Exploitation: searching a confined area of a search space based on previous results.

**Details of experiments [15]**

Five classical benchmark algorithms (Srinivasan and Seow, 2003) were used to test the performance of ABC against PSO, PS-EA, and GA:

* Griewank (lol)
* Rastrigin function. Value is 0 at its global minimum at the original. Produces many local, regularly distributed minima, so an optimisation algorithm can easily be trapped. Non-convex. The typical example of a non-linear multi-modal function.
* Rosenbrock function.
* Ackley function. Tests how efficiently an algorithm both explores and exploits.

A function is *multimodal* if it has two or more local optima. An optimisation algorithm must avoid over-exploiting the regions around local minima.

**An overview of results [5]**

Define all concepts used and give a very brief overview of the comparator algorithms and the benchmark functions used in the experimental section.

# <https://en.wikipedia.org/wiki/Artificial_bee_colony_algorithm>

Srinivasan, D., Seow, T.H.: Evolutionary Computation, CEC ’03, 8–12 Dec. 2003, 4(2003), Canberra, Australia, pp. 2292–2297.