Task 5 – Particle Swarm Optimization (PSO)

The first thing I've to say is that I decided to chose this task because I was going to leave Kuopio on December so maybe that would make things hard in the fact of communicating with other members from a group. Also for the course Deep Learning I had to make a presentation with other students about PSO algorithm, and that was definitely a good introduction to this task.

I have a lot of experience with C language from my home univerity, therefor I decided to do this task using that tool, but after some work, developing the structure for the particles and some more things, I realized that Python is much easier for this task since it was necessary to represent a visualization of the algorithm, even when I've not used Python that much.

I read about the algorithm on the internet, from different sources, wikipedia is an example of this sources but I also used different pages that expose multiple ways of implementing the algorithm.
All the code is commented, but in a simple way of explaining, the main functioning of this algorithm consists on creating different arrays of data, this arrays will contain the position, velocity, personal best fitness of each particle, etc.

This arrays will be initialized in a first iteration with random values, in this iteration the fitness will be calculated, and the values will be organized using bubble-sort algorithm (It's the simplest one with set-sort, implementing merge-sort or quick-sort was rather unnecessary even when those algorithms are more efficient). After the first iteration, at the beginning of the array we will have the best fitness (Thanks to the sorting), of course the arrays of velocity and positions have to be sorted or we'll lose the order and the "particles" will be mixed. When this is done we let the algorithm run for X generations, where new personal best, group best, and velocity are calculated, (Also with the terminal we can see how the fitness is going over the iterations), in this case 150 iterations with 20 "particles" and we can see reprensented with matplotlib library, how the particles look for the "minimum", also after all the data is gathered a line function is represented also with matplotlib.

This is the generation-g_best function: