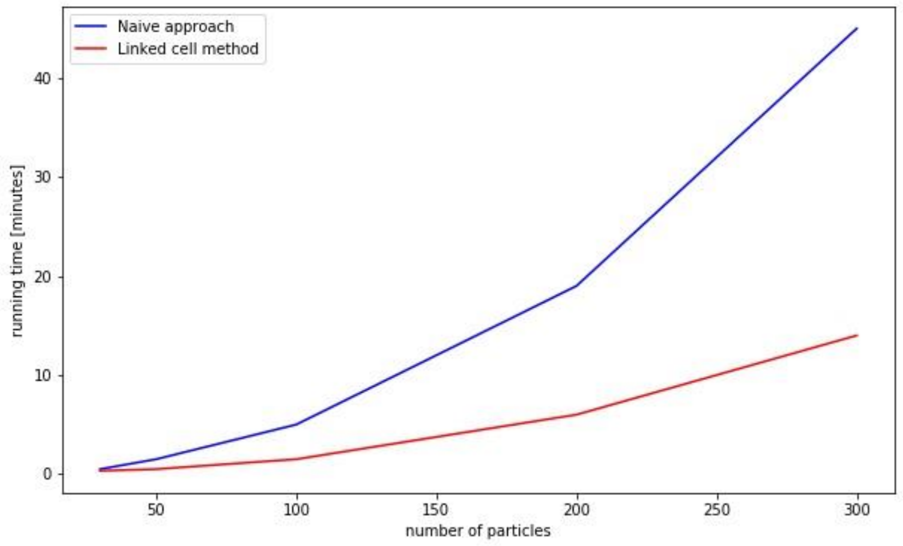
I implemented the Linked cell method considering a 10x10x10 box and cells having edges' length equal to 2. The aim of the method is to perform the simulation in less time than the "naïve" one. I observed that this actually happens, a graph comparing the runtimes for the linked cell code with the old one is displayed below. The linked cell method seems to take one third of the time that the naive method took (at least in the interval of numbers of particles that I explored).

I visualized the results, in ParaView, and they look like the ones produced in the previous exercise. I will summarize the most important steps of my code below, thus that it is (hopefully) easier to spot any conceptual mistakes.

P.s: The new (with respect to exercise 07) part of the code is written in the second cell of the notebook. In the said cell I only define the following three methods.

1) linked_cell: This method returns the "head" and the "linked_list". It implements the encoding of which particle belongs to which cell as explained in the practice lecture. I took the implementation from the book "computer simulations of liquids" at page 198. I slightly modified the indices to fit my previous code.
2) nn: There is a bijective correspondence between the nodes of the grid (created by the cells) and the cells, thus one can identify each cell with the corresponding node. This method takes as input one of the nodes of the grid (corresponding to a specific cell) and returns all the (3**d) nodes identifying the neighbouring cells + the cell itself.
3) Linked_Forces: I create a (rank 3) matrix M=np.zeros((N,N,3)) , N is the number of particles, that will store the interactions between different particles. For each cell (i.e each node in the grid) I consider the neighbouring cells and compute the interactions of the particles in the cell with any particle belonging to the cell itself or to any neighbouring cell. I store the interaction (force) between particle "i" and particle "j" in M[i,j,:], I also use Newton's third law to avoid computing the symmetric element, since I already know that: M[j,i,:] = -M[i,j,:]. In case I rencounter the couple of particles (i,j) I want to avoid computing the interaction again. For this reason for each couple (i,j) I only compute the corresponding interaction if M[i,j,:] is equal to zero (i.e. if it has not been computed yet). When I have visited each cell of the grid I sum over the second axis of M (the axis with the x → M[*,x,*]) to obtain a matrix F containing the total force acting on each particle. The shape of F will be that of a matrix with N rows (N = number of particles) and 3 columns (the dimension of the space were the particles are embedded).


Naive approach
Linked cell method
running time [minutes]
number of particles