Imagry- home assignment-Noy Shulman

The assignment:

Using Python, write a function taking the following inputs: `N_points` and `N_dim`, which are integers, and `min_d` which is a floating point number. The function should output N_points random points in such a way that no two points should be closer than a distance `min_d` as measured on the surface of the sphere and each point sampled randomly from uniform distribution on the sphere surface Please attach the code with a short explanation of its workings and an example plot in 3D**.**

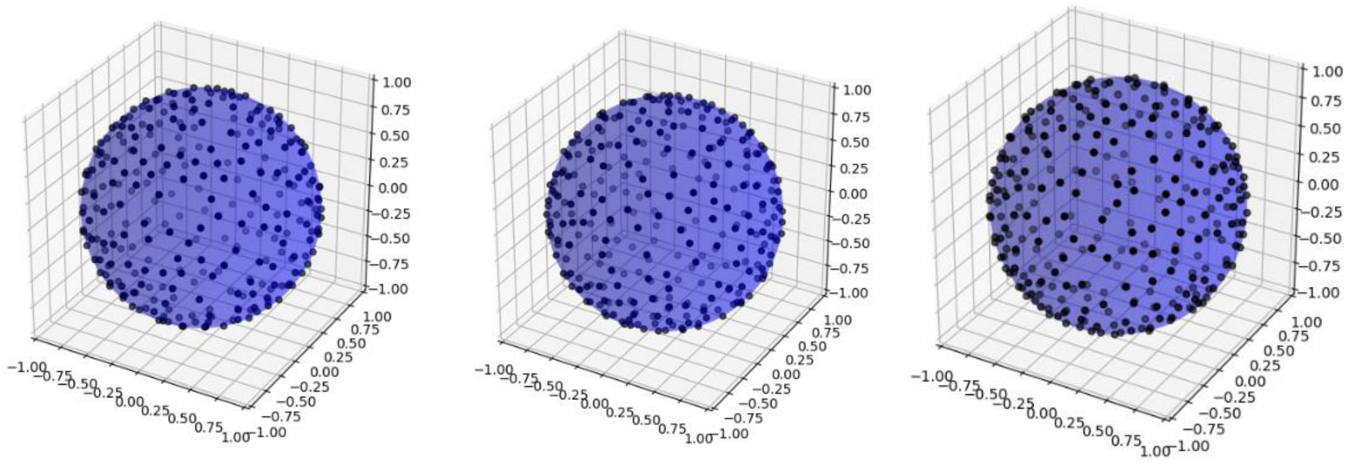
My Algorithm:

1. Sample a point from uniform distribution on a N_dim dimensional sphere.
2. Check if the point is at least min_d from all other points.
3. If it is, add it to the list of points to be returned in the future.
   If not, return to step one without adding the point to the list.
4. Stop when you successfully sampled N_points or predefined time out.

The algorithm does not always converge even when the specified arguments are solvable. To solve this, we can use a relaxation method to move points that are too close together until convergence. Doing so will satisfy the distance condition but will force us to alter the location of the points after being generated, but then the points aren't sampled from truly uniform distribution.

Another solution that approximately solves this problem is using Poisson-disc sampling on the sphere. This will generate the approximate max amount of points that satisfy the distance condition on the sphere. The next step is randomly sampling N_points points out of those points to get the required amount. Like the relaxation method, this solution gives a "good" solution but the points aren't sampled from truly uniform distribution.

Examples in 3D with N_points = 300, min_d = 0.13:



The code:

https://github.com/noyshu/HW_imagry