ID2090 : Introduction to Scientific Computing
Summer-2021

Assignment – 3

Instructions for submission remain same as for first assignment. Don't forget to provide your name, roll number and assignment number on the first page of the PDF you are uploading as part of the tar file. Both the problems carry equal marks. You can use either python or sagemath for this assignment.

[1] This problem can be attempted in 2D. Consider N spherical particles that obey a Gaussian distribution of sizes. These need to be accommodated in the smallest square possible. Your task is **not** to determine the size of that minimal square but to illustrate the process of filling a square of given size with randomly picked particles. State and make assumptions as necessary. Use numerical values for the size distribution and N as required. You are welcome to device any other algorithm but in case you need help, use the following steps as an example:
- List the particles and their sizes
- Estimate the size of the square approximately from the total area of all the particles
- Pick particles in the decreasing order of their size and place them in the square from one of the corners
- Fit smaller spheres in the voids where possible.
- If there are any particles left, increment the size of the square and repeat the above steps until all the particles are in the square.
- Report this configuration using simple 2D graphics as well as listing of locations of the objects inside the square.

Output required: The code as .py or .sage file; a PDF containing the configuration, the sizes of the particles and the size of the square.

Application:

(a) Filling space using objects to minimize the total space occupied is a logistical problem in transport industry. Efficient filling helps save costs.

(b) Consider one of the sides of the square is to be used as exit for the spheres to be taken out. Imagine each sphere is given a sequence number according to which it has to be taken out. Now the problem is brought closer to a traditional logistics problem of a delivery truck. The sequence depends on the path that the delivery truck takes in the city. Longer paths add to the cost and so do multiple trips or larger trucks. Now you see the challenge of the optimization problem.

[2] Consider a vaccine container with a cryogenic liquid (boiling point 77K) filled upto a certain fraction of the height of the container. The inner temperature of the container is measured by a thermocouple. The inner temperature raises steadily with time as a linear function of time due to phenomena such as the cryogenic fluid keeps vaporising steadily and heat flux from ambient atmosphere keeps entering the container steadily. At random times, the operator replenishes the cryogenic liquid to bring the inner temperature down to 77K again. If the temperature of the container raises above say 200 K, then it has to be flagged as unfit for further use. The temperature data from the thermocouple is sent out over GPRS to a remote server. Simulate and plot an example data of this problem. State and make assumptions as needed. Make sure your assumed values for the rate of temperature rise, durations between refill and total duration for plot can illustrate the

phenomena stated in this problem. Organize your code as two scripts – one to generate the thermocouple data and another to detect and flag the container as unfit as per protocol.

Output required: The two codes, thermocouple data generated, plot of the same, PDF containing the details of the model used, brief description on how you identified the rates and random time instances for replenishing the cryogenic liquid, and the total duration.

Application: When a firm wishes to fabricate a device such as the one mentioned above, it is important to have a digital twin to simulate what if cases. One needs to determine the protocol for periodicity of checking the temperature alerts, actions to be taken and ways to detect any falsification of data enroute. A robust digital twin that reproduces the various situations in real life will help ensure a reliable product.

--oo**oo--