Fractal Growth

4. Random Walk

Copy the solution program to step 4 of our last inclass work (and solution to HW 9) into your working directory with

```
cp ~kvollmay/share.dir/inclass.dir/classfractal4.py .
```

Look at the program with gedit or any other editor. Run the program and look at the created picture frame4.pdf with

```
   ./classfractal4.py
   atril frame4.pdf
```

5. DLA: Distance

5a. We work next on step IV of the DLA rules, for which you need to keep track of the distance r of your walker from the midpoint of the lattice. Copy `classfractal4.py` into for example `classfractal5a.py` then modify `classfractal5a.py` such that you comment out the red trail marker of setting lattice values to 2 and also comment out the plotting commands. Then add to the program the determination of r . (Hint: For $\sqrt{\quad}$ in python use in the header, as already in the sample file, `import scipy as sp` and then for example $\sqrt{5}$ would be `sp.sqrt(5)`). To check your program print for every random walk step `x,y,LMID,r` and check by hand that you get the right distance from the midpoint of the lattice.

5b. Now replace the loop of 100 random steps with a while loop `while walkstop == 0:`. Set `walkstop = 0` before this while loop and set `walkstop = 1`, as soon as $r \geq 2 * R_{\text{max}}$. Initialize R_{max} to $R_{\text{max}} = 3.0$ and `sp.random.seed(11)`. Run your program and check that your stopping of the random walk works as intended.

6. Stick to Cluster

Next we will work on rule V, the sticking of a random walker particle to the cluster, if the random walker is next to a cluster cell. We use "von Neumann neighbors", which means a neighbor cell up,down,left or right.

6a. Next add to your program of 5b. that whenever the random walker is next (left, right, top, bottom) to a particle of the cluster then the random walk stops (`walkstop` update). Use the flow chart to decide where to add the necessary lines. If you have kept the `print(x,y,LATMID,r)` from 5a (and the same seed) then you can check that your program is working right.

6b. Now add to your program of 6a that you also have an integer variable `npart` which is initialized to be `npart=1` and gets increased by one whenever a particle sticks to the cluster.

Also update `lattice` whenever a particle sticks. Whenever a particle sticks to the cluster, you also need to check if `RMAX` has grown and if so, then you need to update `RMAX`. Add this to your program.

7. Finish Program: Loop Over Particles

Now you are ready to finish your DLA program! Add to your program a while loop over particles. Condition for this while loop are both that the wanted number of cluster particles `NPARTMAX` has not yet been reached and that the radius for the start of the random walk fits into the lattice. Use the flow chart of class to decide where to add this `while`-loop. Use the constants `LATSIZE=100`, `NPARTMAX=50`. Comment out the printing of `(x,y,LATMID,r)`, but print the resulting lattice at the end (so after the particle-loop). In case you would like not to print the complete lattice, you may use the following commands

```
plt.imshow(lattice[int(LATMID-RMAX-2):int(LATMID+RMAX+2),\
  int(LATMID-RMAX-2):int(LATMID+RMAX+2)],interpolation='nearest')
plt.savefig('frame7.pdf')
```