February 12, 2015 PHYS 338 Spring 2015

In-Class Work: Fractal Growth

1. Random Walk in Two Dimensions

1a. For the fractal growth DLA model we will need a random walk in two dimensions. Write a python-program for a random walker on a two dimensional lattice, starting at $x = 0$ and $y = 0$ and (print and) look at $x(t)$ and $y(t)$. (You may use the solution program from last class `~kvollmay/classes.dir/phys338.dir/phys338_s2015.dir/randomwalk.dir/classrndwalk2a.py`.

1b. Movie

Next let's make a movie of your random walk. Define a lattice (`lattice`) of size 100x100 and initialize it for all sites equal to zero. Put your initial walker at site $x = 50$ and $y = 50$. We want to make a movie of the random walker where we mark on the lattice the current random walker site with the lattice value 2 and we mark any previously visited site with 1 (This is just for our fun.). To make a movie we first make an image for every random-walk step. **(So please use only `NSTEPS=50` random walk steps!)** To see how to make these pictures see the example
`~/classes.dir/phys338.dir/phys338_s2015.dir/linux_python_intro.dir/classpython11.py`
Once you have all pictures in `frame*` you can run the movie with

```
animate -delay 10 -pause 5 frame*
```

2. Fractal Growth: Background

Read in our textbook §13.2 Kinetic Growth Processes (text only, no problems, no JAVA code) Epidemic model, Eden model, and diffusion limited aggregation. In case you do do not have your textbook with you, there is a draft from a public webpage:
`~kvollmay/classes.dir/phys338.dir/phys338_s2015.dir/bookdraft.dir/csm_ch13.pdf`
Please get me, when you get here. I will give a mini-intro about cellular automata and about fractals.

3. Initialize the lattice with all lattice sites being zero and only in the middle of the lattice is a seed with value one. Use a lattice size `LATSIZE=100`. We want next to implement that a new random walker starts randomly somewhere on a circle with midpoint in the middle of the lattice and with radius `RMAX+2`. For the DLA-program we will use `RMAX=2` but for now use `RMAX=20`. To check that you draw your random walker indeed equally likely on a circle, add to your program that you put 50 initial walkers on their starting point (not yet with random walk steps) and mark for each of these 50 starting points the lattice with the value 2. Make only one image of your lattice after these 50 markings on the lattice. To get a pdf-file of your frame use `imshow` as above and use for example `plt.savefig('frame3.pdf')`

4. DLA: Circle and Random Walk Take out of your program now the loop over 50 starting points, instead start your random walker at one point on the circle and use `RMAX=2`. Add to your program a loop over $NSTEP = 50$ random walk steps. To check your program, assign to each site, which is visited by the random walker, the value 2. Print the lattice after this random walk and look at it.

February 17, 9:30 – 11:00, 2015 PHYS 338 Spring 2015

In-Class Work: Fractal Growth

Today we will program the DLA-model! To guide you through the process of implementing a larger task, as the DLA program, I will give an intro at the beginning of class. We will develop the flow chart and the following steps guide you through the approach of deviding the bigger task into several smaller tasks, which you program and check.

3. Start Random Walker on Circle Copy the program `~kvollmay/classes.dir/phys338.dir/phys338_s2015.dir/fractal.dir/classfractal3.py` into your working directory. Read the program, to convince yourself what it does and run the program. Look at the resulting `frame3.pdf`.

4. DLA: Random Walk Take out of this program now the loop over 50 starting points, instead start the random walker at **only one** point on the circle and use `RMAX=2`. Add to your program a loop over `NSTEP=50` random walk steps. To check your program, assign to each site, which is visited by the random walker, the value 2. Print the lattice after this random walk and look at it. (Hint: You may have a look at the random-walk program `~kvollmay/classes.dir/phys338.dir/phys338_s2015.dir/fractal.dir/classfractal1a.py`)

5. DLA: Distance 5a. We work next on step 2c of the rules, for which you need to keep track of the distance r of your walker from the midpoint of the lattice. Add to your program of 4. the determination of r . (Hint: For $\sqrt{}$ 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 2d, 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).

Homework #7: (see webpage)

At the beginning of next class, Thursday, Feb. 19, hand in:

- hardcopy of your "first version" of your first paper
- hardcopy of your previous model & bibliography with my comments
- hardcopy of your previous background/intro with my comments
- electronic copy of your first paper as pdf-file in your `~/share.dir/` (with read-permission).

In-Class Work: Talk Tools

1. Sample File(s) for Latex Beamer:

Copy

```
~kvollmay/classes.dir/phys338.dir/phys338_s2015.dir/talks.dir/beamer_example.tex
~kvollmay/classes.dir/phys338.dir/phys338_s2015.dir/talks.dir/fig[1-3].eps
```

into your working directory. Have a look at `beamer_example.tex`.

2. Compile:

The commands for compiling this sample file and for looking at the resulting pdf-file are as comments at the beginning of `beamer_example.tex`. Since I teach you graphic tools (`xfig`, and `xmgrace`) with which you can make eps-files, choose option `(A)`. Paste the commands on the command line and hit `Enter`.

3. Start Your Mini-Project Talk: Copy the `beamer_example.tex` to a second tex-file which will be for your mini-project talk. Change the title to the title of your talk and similarly change author, date and sections.

4. xmgrace

To have some data and an example xmgr-file copy

```
~kvollmay/classes.dir/phys338.dir/phys338_s2015.dir/talks.dir/Noft_Moore.data
~kvollmay/classes.dir/phys338.dir/phys338_s2015.dir/talks.dir/Noft_vonNeumann.data
~kvollmay/classes.dir/phys338.dir/phys338_s2015.dir/talks.dir/fsqt.xmgr
```

To get started with xmgrace type on the command line `xmgrace &`. To pull in a dataset use `Data` → `Import` → `ASCII` and under Selection add `Noft_Moore.data` then click `OK`. Similarly pull in the dataset `Noft_vonNeumann.data` . I will show you next: data: labels, symbols, line symbols, axis changes: line width, label incl. size and tick marks, and legend positioning. To save an xmgrace session use `File` → `SaveAs` (use a filename which ends with `.xmgr`). It is important to use `SaveAs` the first time because default is to overwrite your data-file! For the second time saving you may use `Save`. To continue an xmgrace session use `File` → `Open`. To make an eps-file use `File` → `Print setup` and choose as device `EPS`. This only sets up the printing, to get the eps-file printed use `File` → `Print`.

You may also want to play some with the example `fsqt.xmgr`. Make a figure of $N(t)$ with the `Noft_Moore.data` which would satisfy the expectations on figures for scientific publications and talks. Make an eps-file and include it in your latex beamer file. If time is left you may also want to play some with `fsqt.xmgr`.

In-Class Work: Fractal Growth

8. Finished DLA program

Last class you all worked on the DLA ([T. A. Witten Jr, L. M. Sander, Phys. Rev. Lett. 47, 1400 (1981)]) program. In previous years it took several classes to finish all steps of the DLA program, so even if you did not finish last class the program, you were all showing great progress! Today we will work on how the resulting DLA-cluster can be analyzed, namely you will measure the so called fractal dimension. To ensure that everybody will work on this analysis (instead of finishing their own version of the program), copy the following program into your working directory
`~kvollmay/classes.dir/phys338.dir/phys338_s2015.dir/fractal.dir/classfractal8.py`
I will guide you through this program.

9. Fractal Dimension of DLA Cluster

9a. Read in our textbook pages 491 – 493 to problem 13.1b (incl.). in
`~kvollmay/classes.dir/phys338.dir/phys338_s2015.dir/bookdraft.dir/csm_ch13.pdf`
this corresponds to pages 1 – 3 to problem 13.1b (incl.).

9b. Now lets get ready to analyze the pattern of the DLA model. You will determine the fractal dimension of one pattern using the method of checking squares of length `b`, as just described in class and in the textbook in problem 13.1b.

To avoid having to run the DLA program again and again, let us first prepare one pattern, which you then will analyze in 9c. Run the program of 8., so
`~kvollmay/classes.dir/phys338.dir/phys338_s2015.dir/fractal.dir/classfractal8.py`
This program makes the file `bigDLAcluster.dat` (and a nice pdf-file `frame8.pdf` just for fun). Ensure that you run the program for `LATSIZE=500` and for `NPARTMAX=3000`. This will take a while, but we have to do this only once, because for the analysis we use `bigDLAcluster.dat`.

9c. Now you need a program which reads in the 224x224 matrix from your file `bigDLAcluster.dat`. You may use for this task
`~kvollmay/classes.dir/phys338.dir/phys338_s2015.dir/fractal.dir/classfractal9start.py`
To get the fractal dimension d_f we use the following relation.

$$\ln(N) = \ln(c) + d_\mathsf{f} * \ln(b) \qquad (1)$$

where N is the number of occupied sites, c is some constant and b is the length of your square for which you count the number of occupied sites. You see that Eq.(1) defines d_f and it tells us that if we plot $\ln(N)$ as a function of $\ln(b)$ then we should get a line with slope d_f. So our task is to get N and b. Add to your program that you count the number of occupied sites N for a lattice of lenght b, where you center your lattice of lenght b around the midpoint of your 224 x 224 lattice. Loop over the length of your lattice and print out $ln(N)$ as a function of

ln(b). Let's say you do
`classfractal9c.py` $>$ `lnNoflnb.dat`
Hint: ln(N) is in python `sp.log(N)`

9d. Next we fit a line to our data from 9c stored in file `lnNoflnb.dat`. For this we use gnuplot. So type in the command line "`gnuplot`". Then type "`plot "lnNoflnb.dat"`". Define a function $f(x)$ by typing "`f(x) = a*x + b`". Now fit your data within the xrange [2.0,4.5] to a line by typing "`fit [2.0:4.5] f(x) "lnNoflnb.dat" via a,b`". The resulting a is the fractal dimension d_{f}. You can look at the data and fit with "`plot "lnNoflnb.dat",f(x)`" Compare your fractal dimension with the expected value of 1.71