

IN-CLASS WORK: FRACTAL GROWTH

1. DLA: Definition Read the hand-out and make a sketch of the model. Get me once you are ready.

2. DLA: Flow Chart Sketch the flow chart for the DLA program.

3. DLA: Circle and Random Walk Copy into your working directory

`~kvollmay/classes.dir/capstone_s2012.dir/fractal.dir/DLA3_sample.cc`

The program includes already the initialization of the lattice and the starting point of your random walker, which is on a circle around the midpoint of the lattice. Include in the program (at indicated place) the random walk being done by the particle. You may use the solution to last class' in-class work

`~kvollmay/classes.dir/capstone_s2012.dir/fractal.dir/rand1c.cc`

Update the lattice for each random walk , so that the random walk is visible as a red line (`lattice[x][y]=2;`). Look at your resulting lattice movie with `executable | DynamicLattice -nx 100 -ny 100 -matrix -z 0 2`

4. DLA: Distance

4a. 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 3. the determination of r . (Hint: For a vector (x, y) the length of the vector is $\sqrt{x^2 + y^2}$. In C++ $\sqrt{5}$ is `sqrt(5.0)` and 5^2 is `pow(5.0,2.0)` or `5.0*5.0`. Notice that `sqrt` and `pow` need double or float variables and do not work with `int` variables.) To check your program change the printing such that instead of printing the movie of the lattice it prints for every random walk step two numbers: the step and the distance r . Look at your result with `xmgrace`.

4b. 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 $r \geq 2 * R_{\max}$. Notice that R_{\max} is initialized to $R_{\max} = 3.0$ in the section of initialization, the radius is set to `radius = Rmax + 2`, and `idummy = -6`. Run your program and check that your stopping of the random walk works as intended.

Solutions to Random Walk In-Class Work:

`~kvollmay/classes.dir/capstone_s2012.dir/fractal.dir/rand1b.cc` etc.

5. Stick to Cluster (if time)

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 (use von Neumann neighbors).

5a. Next add to your program of 4b. 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. To be able to check your program, we want to let the random walker to continue walking instead of stopping because of rule 2c. Therefore, just for the sake of checking your program for rule 2d, comment out the walkstop=1 of rule2c. Add to the print command that you print not only r , but also x and y .

5b. Now add to your program of 5a that you also have an integer variable n_{part} which is initialized to be $n_{part}=1$ and gets increased by one whenever a particle sticks to the cluster. Also update `lattice` whenever a particle sticks. To check your program update the lattice such that the random walk is shown as a moving red site (i.e. `lattice[x][y]=0` before random walk step and `lattice[x][y]=2` after walk step). Watch the movie with `DLA5b.out | DynamicLattice -nx 100 -ny 100 -matrix -z 0 2`

5c. Whenever a particle sticks to the cluster, you also need to check if R_{max} has grown and if so, then you need to update R_{max} . Add this to your program.

Upcoming Deadlines:

- **April 12: Results Section of Paper** (analysis, figures, tables, definition of all variables and interpretations in text)
- April 17: Abstract
- April 19: Second Paper 1st Version
- April 26: Second Paper Final Version
- April 24 – May 1: Symposium Talks

IN-CLASS WORK: FRACTAL GROWTH

5. 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 nearest von Neumann neighbor to the cluster.

5a. Either use

```
~kvollmay/classes.dir/capstone_s2012.dir/fractal.dir/DLA4b.cc
```

or use your working program for 4b from last class. Add to the program 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. To be able to check your program, we want to let the random walker to continue walking instead of stopping because of rule 2c. Therefore, just for the sake of checking your program for rule 2d, comment out the walkstop=1 of rule2c. Add to the print command that you print not only r , but also x and y .

5b. Now add to your program of 5a 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. To check your program update the lattice such that the random walk is shown as a moving red site (i.e. $lattice[x][y]=0$ before random walk step and $lattice[x][y]=2$ after walk step). Watch the movie with
DLA5b.out | DynamicLattice -nx 100 -ny 100 -matrix -z 0 2
where $-z 0 2$ sets the colors to be blue for 0, white for 1, and red for 2.

5c. 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.

6. Finish Program: Loop Over Particles

Now you are ready to finish your DLA program! Put back in rule 2c (stopping of random walk when $r \geq 2 Rmax$.) Add to your program the 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 $L=100$, $NPARTMAX=50$. Watch the movie with

```
DLA6.out | DynamicLattice -nx 100 -ny 100 -matrix -z 0 2
```

When you got your program working, please let me know, so that you can show your movie to the class.

Assignment for Tue, April 17: Title & Abstract (I need these for making advertisement posters & pamphlet.)

7. DLA Pattern

Next we try to generate a larger cluster to see its fractal structure (and just because it is fun). Print the lattice no longer for every random walk step, but instead print the lattice only once after the while loop over particles. To be able to print only the middle part of the lattice instead of the complete lattice, have a look at the function `printsmallattice` in

```
~kvollmay/classes.dir/capstone_s2012.dir/fractal.dir/DLA4b.cc
```

Set `L=500,NPARTMAX=3000` and `LPRINT=250`. Have fun with the resulting picture. To make a picture which you can put on your wall:

```
DLA7.out > DLApicture.data
```

```
import DLA.ps
```

```
(or import DLA.png).
```

The resulting `DLA.ps` you can print on a color printer, or to check it on the screen

```
evince DLA.ps (or display DLA.png).
```

8. Fractal Dimension of DLA Cluster (if time)

In the following you will analyze the pattern of the DLA model. You will determine the fractal dimension of one pattern. When you get to this, please get me, because I will give the class an introduction to the fraction dimension.

8a. To avoid having to run the DLA program again and again, let us first prepare one pattern, which you then will analyze in 8b. Use your program of 7. and run it with `L=500, NPARTMAX=10000, LPRINT=350` and

```
DLA7.out > bigDLAcluster.data
```

to print the final pattern only once at the end of the program into the file “`bigDLAcluster.data`”.

8b. Now write a program which reads in the 350x350 matrix from your file of 8a. To get the fractal dimension d_f we use the following relations.

$$\ln(N) = \ln(c) + d_f * \ln(b) \quad (4)$$

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.(5) 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 length b , where you center your lattice of length b around the midpoint of your 350x350 lattice. Loop over the length of your lattice and print out $\ln(N)$ as a function of $\ln(b)$. Let's say you do

```
DLA8b.out > lnNoflnb.data
```

8c. Next we fit a line to our data from 8b stored in file `lnNoflnb.data`. For this we use `gnuplot`. So type in the command line “`gnuplot`”. Then type “`plot "lnNoflnb.data"`”. 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.data" via a,b`”. The resulting `a` is the fractal dimension d_f . You can look at the data and fit with “`plot "lnNoflnb.data",f(x)`”

9. History of DLA Particles (if time)

To understand the shape of the DLA clusters, let us now indicate the history of the particles with their color: Print “5” for the first 500 particles, “6” for the next 500 particles, “7” for the following 500 particles etc.. Look at the final result with `DynamicLattice`.

Hint: Adjust your check if the random walker is next to a particle of the cluster.

You might recognize the resulting picture (-)

IN-CLASS WORK: FRACTAL GROWTH

8. Fractal Dimension of DLA Cluster (if time)

In the following you will analyze the pattern of the DLA model. You will determine the fractal dimension of one pattern.

8a. To avoid having to run the DLA program again and again, let us first prepare one pattern, which you then will analyze in 8b. Either use

```
~kvollmay/classes.dir/capstone_s2012.dir/fractal.dir/DLA7.cc
```

or your finished and working DLA-program of last class. Run the program with $L=500$, $NPARTMAX=10000$, $LPRINT=350$ and

```
DLA7.out > bigDLAcluster.data
```

to print the final pattern only once at the end of the program into the file "bigDLAcluster.data".

You may check (and enjoy) the resulting picture:

```
DynamicLattice -nx 350 -ny 350 -matrix < bigDLAcluster.data
```

8b. Now we want to write a program which reads in the 350x350 matrix from your file of 8a and determines the fractal dimension. You may copy

```
~kvollmay/classes.dir/capstone_s2012.dir/fractal.dir/DLA8b_sample.cc
```

which takes care of reading in the bigDLAcluster.data. To get the fractal dimension d_f we use the following relations.

$$\ln(N) = \ln(c) + d_f * \ln(b) \quad (5)$$

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.(5) 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 length b , where you center your lattice of length b around the midpoint of your 350x350 lattice. Loop over the length of your lattice and print out $\ln(N)$ as a function of $\ln(b)$. Let's say you do

```
DLA8b.out > lnNoflnb.data
```

8c. Next we fit a line to our data from 8b stored in file lnNoflnb.data. For this we use gnuplot. So type in the command line "gnuplot". Then type

```
plot "lnNoflnb.data"
```

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.data" via a,b
```

The resulting a is the fractal dimension d_f . You can look at the data and fit with

```
plot "lnNoflnb.data",f(x)
```

11. Sticking Probability Variation (if time)

11a Next we work on a variation to the model. A random walker next to a cluster particle does not always stick to the cluster. Instead the random walker particle sticks only with probability `PSTICK`. Start with `PSTICK=0.2`. Use your working program or you may start with `~kvollmay/classes.dir/capstone_s2012.dir/fractal.dir/DLA7.cc` (so 500x500 lattice and `NPARTMAX = 10000` and print only once at the end a square lattice of size `LPRINT=350`). To avoid that a non-sticking particle walks on top of already existing cluster particles, do a random walk step only if it would put your random walker onto an empty lattice site. So lines 83-86 are not always done (not for `x,y`) and similarly do lines 62-69 only if your random walk step is allowed.

11b. Now let's compare the **fractal dimension** of the original DLA-model and of the DLA-model with the sticking probability . Repeat the steps of 8. to determine the fractal dimension.

11c. (if time)

Determine the fractal dimension for varying `PSTICK`.

9. History of DLA Particles (if time)

To understand the shape of the DLA clusters, let us now indicate the history of the particles with their color: Print "5" for the first 500 particles, "6" for the next 500 particles, "7" for the following 500 particles etc.. Look at the final result with `DynamicLattice`.

Hint: Adjust your check if the random walker is next to a particle of the cluster.

You might recognize the resulting picture (-)

Upcoming Deadlines:

- April 17: Title & Abstract (today, please as text of email)
- April 20: Second Paper 1st Version
- April 26: Second Paper Final Version
- April 24: Symposium Talks of Dan, Will, Kota and Miles
- April 26: Symposium Talks of Jessica, Evan, Lia and Meghan
- May 1: Symposium Talks of Matt, Kin, Jacob and Sowande