Random Walks

1. Uniform Random Numbers

1a. Copy the program
~kvollmay/classes.dir/capstone_s2005.dir/random_walks.dir/float_rand0-1.cc
into your working directory. Compile the program and let it run. Have a look at the program to learn how to get random numbers. Whenever you want to write a program in which you use random numbers you need to do the following step:
1. Include in the header of your program the two lines:

```
double randomd(long *);
long idummy = -7;
```

2. Include at the end of your program the definition of the function randomd, i.e. lines 19–49 of float_rand0-1.cc.
3. whenever you want another random number use
`randomd(&idummy)`

1b. Have a look at which random numbers you get: To do so set up the following unix-command:
`executable | gawk '{print NR,$0}' | xgraph -m -nl`

2. Probability

2a. Write a program that prints 500 numbers. Each number is with probability $p = 0.3$ "1" and otherwise "0". Run the program and check it with xgraph as in 1.

2b. Change your program from 2a. such that it reads in from screen the probability p and the number of random numbers N. The program then produces N numbers, where each number is with probability p "1" and otherwise "0". Use an array to store the N numbers. The program then counts and prints out how many of the drawn numbers are "1" and how many are "0". Run your program a few times for different values of p and N.

3. Random Walk in One Dimension

3a. Modify your program of 2a. to program a random walk in one dimension on a lattice. Set the probability P to hop to the right (length 1).Use a constant for P and start with $P = 0.5$. Use `NSTEPS = 5000` time steps. Start with $x = 0$ and print $x(t)$. Have a look at your result with `xgraph`. Rerun your program for $P = 0.2$;

3b. Get me before you work on this program. Set $P = 0.5$. The next step is a preparation for 3c. Our goal in 3c will be to determine $\langle x\rangle(N)$ and $\langle x^2\rangle(N)$ where N is the number of random steps. Determine for N =`NSTEPS=5000` what the final x and x^2 is. Find for `NORUNS = 1000` simulation runs of your random walk the average value for the final (after `NSTEPS` random walk steps) x and x^2.

3c. Definitely get me before you work on this program. Next store your averages of x and x^2 in an array to obtain $\langle x\rangle(N)$ and $\langle x^2\rangle(N)$. As result of your program print three columns N, $\langle x\rangle(N)$, and $\langle x^2\rangle(N)$. Look at N, $\langle x\rangle(N)$. Run your program with `executable.out > outfile`. Then look at $\langle x\rangle(N)$ with `xgraph -m outfile` and at $\langle x^2\rangle(N)$ with `gawk '{print $1,$3}' outfile | xgraph -m` .Does your result agree with the theory? Repeat the simulation but with $P = 0.2$ instead. Does your result agree with the theory?

3. Random Walk in One Dimension

3d. Copy
`~kvollmay/classes.dir/capstone_s2005.dir/random_walks.dir/rand3b.cc`
into your working directory. You will modify this program in 3e. to determine $\langle x \rangle(N)$ and $\langle \Delta x^2 \rangle(N)$. Explain what the program `rand3b.cc` does. Make a flow chart of the program.

3e. Get me before you work on this program. Modify `rand3b.cc` such that it stores your sums of x and x^2 in an array to obtain $\langle x \rangle(N)$ and $\langle x^2 \rangle(N)$. As result of your program print three columns N, $\langle x \rangle(N)$, and $\langle \Delta x^2 \rangle(N)$. Run your program with `executable.out > outfile`. Then look at $\langle x \rangle(N)$ with `xgraph outfile` and look at $\langle \Delta x^2 \rangle(N)$ with `gawk '{print $1,$3}' outfile | xgraph` .Does your result agree with the theory?

3f. Repeat 3e but with $P = 0.2$ instead. Does your result agree with the theory?

Fractal Growth: DLA

1. Flow Chart Get together in groups of two and draw the flow chart for the DLA-program. Your flow chart should be for a program which grows a cluster of `NPARTMAX` particles. Which loops do you need? Which decision statements do you need?

2. Random Walk in Two Dimensions Copy
`~kvollmay/classes.dir/capstone_s2005.dir/fractal.dir/DLA2_sample.cc`
into your working directory. Add to the program the random walk step (see "`add here the random walk step in two dimensions (x,y)`"). Print the complete lattice after every random walk step. Look at your result with
`executable.out | DynamicLattice -nx 100 -ny 100 -matrix`.