Physics 307: Science and Computers I
Fall 2009

Random Walks

Until now, we have been studying deterministic equations of motion, which work well for two-object, three-object, etc. systems. However, when studying systems with 10^{23} objects, such as a particle moving through a liquid, instead of writing down 10^{23} coupled equations of motion and trying to solve them, we use a simplified description. The simplified description is to treat the molecules of the liquid as exerting random forces on the particle. Random forces are used to describe other complex, many degrees of freedom systems such as the stock market as well as the motion of, say, insects such as ants. Since we are made up of 70 percent water, transport of "material" through our bodies can also be described by random forces exerted on the material.

To simulate a nondeterministic system, a random system, we begin with the canonical random walk in one dimension. As discussed in class last Thursday, the rules of the game of a one-dimensional random walk are equal likelihood for the walker (particle) to move to the left or to the right at each time step. In other words, if you flip a coin and land heads, then the walker hops to the right and if you land tails, the walker hops to the left. The process is random. Also, the walker can either move by an amount unity at each time step, or by some random amount.

Write a code to generate a one-dimensional random walk. To do this you will have to use MATLAB's built-in random number generator, rand.

Once you have your code working, the quantitative signature of a random walk is that the square of the position averaged over many runs scales linearly with time. Modify your code to make this measurement and indeed verify this relation. What is the slope and y-intercept of the line?

Now change the rules of the game to incorporate a bias into your one-dimensional random walk. In other words, make it a little more likely to travel to the left as opposed to the right. Compare plots of position versus time as well as your averaged measurement that you just completed in the unbias case. Does the average of the square of the displacement still scale linearly with time?

Now move on to construct a random walk in two dimensions. How do the rules of the one-dimensional random walk extend to this case?

Now construct a code to simulate the two dimensional random walk and plot y versus x. Use unit step sizes in this case. What do you observe?

What is the equivalent measurement for the average of the square of the position measurement as was done in the one-dimensional random walk case? Do you observe a linear scaling with time as before?

If you wanted to construct a lattice where there is a maximum extent in both the x and y direction, how would you do this in your code? Let's say you implement periodic boundary conditions, which means when the particle "exits" the lattice, it enters back onto the lattice from the opposite side. We will discuss this more in class to clarify.

If you have completed all of the above, think about how to generate a self-avoiding random walk as defined in Sec. 7.2 of the course textbook.