Visualization in Python
PHYS 250 (Autumn 2024) – Lecture 4

David Miller

Department of Physics and the Enrico Fermi Institute
University of Chicago

October 10, 2024

Outline

Random diffusion (I)

Consider a general, uncorrelated random walk where at each time step Δt the particle's position x changes by a step ℓ:

$$x(t + \Delta t) = x(t) + \ell(t). \tag{1}$$

Let the probability distribution for each step be $\chi(\ell)$, which in our case is a discrete probability (e.g. for the 2D random walk, $\chi(\ell) = \delta(|\ell| - L)$ with equal probability in $\pm x, \pm y$). We will assume that χ has mean zero and standard deviation a. The first few moments of χ are therefore:

$$\int \chi(z)dz = 1 \tag{2}$$

$$\int z\chi(z)dz = 0 \tag{3}$$

$$\int z^2\chi(z)dz = a^2 \tag{4}$$

Random diffusion (II)

For the particle to go from x' at time t to x at time $t + \Delta t$, the step $\ell(t)$ must be $x - x'$. This happens with probability $\chi(x - x')$ times the probability density $\rho(x', t)$ that it started at x'. Integrating over original positions x', we have:

$$
\begin{align}
\rho(x, t + \Delta t) &= \int_{-\infty}^{+\infty} \rho(x', t)\chi(x - x')dx' \tag{5} \\
&= \int_{-\infty}^{+\infty} \rho(x - z, t)\chi(z)dz \tag{6}
\end{align}
$$

where we have changed variables $x' \to z = x - x'$. Now, perform a Taylor expansion in z:

$$
\begin{align}
\rho(x, t + \Delta t) &\approx \rho(x, t) + \frac{1}{2}\frac{\partial^2 \rho}{\partial x^2}\int z^2 \chi(z)dz \tag{7} \\
&\approx \rho(x, t) + \frac{a^2}{2}\frac{\partial^2 \rho}{\partial x^2} \tag{8}
\end{align}
$$

Slow random diffusion

If the diffusion is also slow, such that the time derivative of ρ is approximately linear with respect to time and $\rho(x, t + \Delta t) - \rho(x, t) \approx (\frac{\partial \rho}{\partial t})\Delta t$, then

$$\frac{\partial \rho}{\partial t} = \frac{a^2}{2\Delta t} \frac{\partial^2 \rho}{\partial x^2}. \tag{9}$$

This is the diffusion equation Eq. **??** with $D = \frac{a^2}{2\Delta t}$.

The point is that we obtained an analytical description of a random walk via the diffusion equation under minimal assumptions: the probability distribution is broad and slowly varying compared to the size and time of the individual steps.

Visualizing the random walk

Even just in our examples on Tuesday, we took the time to visualize what we were doing:
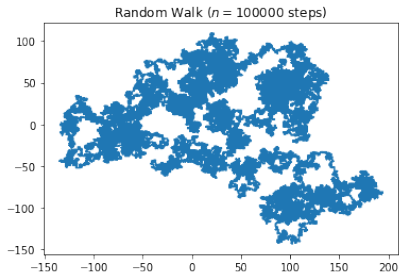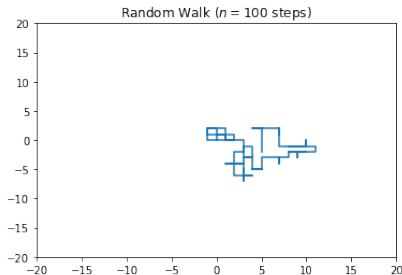
These were fairly simple to setup:

```python
plt.plot(walk[0],walk[1],label= 'Random walk')
plt.title("Random Walk ($n="+str(nsteps)+"$ steps)")
```

Taking a step back on plotting and visualization

All of the visualization tools that we will discuss are powerful enough for professional scientific work and are free or open source.

- Commercial packages such as Matlab, AVS, Amira, and Noesys produce excellent scientific visualization but are less widely available.

- Mathematica and Maple have excellent visualization packages as well, but they are not nearly as useful for with large numerical data sets.

The powerhouse is `matplotlib`

A very powerful library of plotting functions callable from within Python that is capable of producing publication quality figures in a number of output formats. It is, by design, similar to the plotting packages with MATLAB, and is made more powerful by its use of the **`numpy`** package for numerical work. In addition to 2-D plots, Matplotlib can also create interactive, 3-D visualizations of data.

An aside on philosophy

One of the absolute **best** parts about data analysis is making that *final, concise, precise figure* that clearly conveys the point of the results that you've been working on for so long.

But more than that, actually **looking at your data** is critical (as I showed clearly in the second lecture, I hope!).

- **visualizations may provide deep insights into problems by letting us see and *handle* the functions with which we are working**
- **visualization also assists in the debugging process**
- **visualizations are, simply, often just beautiful**

In thinking about ways to present your results, keep in mind that **the point of visualization is to make the science clearer and to communicate your work to others**. It follows then that you should make all figures as **clear, informative, and self-explanatory as possible**, especially if you will be using them in presentations without captions. This means labels for all curves and data points, a title, and labels on the axes.
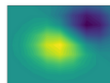
Matplotlib documentation

Matplotlib has fantastic documentation with nearly and infinite number of examples of how to build clear, useful, efficient data visualizations and figures.

"Hands-on"!

Today will be mostly going through examples, and doing hands-on exercises before we get back into the physics in a deep way next week.