MSE-213 Autumn 2024 – Python Practice Week 3

Run in the conda mse213 environment, use a Jupyter notebook. If you have trouble with that, you can use the online environment noto.epfl.ch

If you don't know how to proceed, stackoverflow.org, google.com, chatgpt.com can be very helpful. For a specific Numpy function, numpy.org has the documentation, or just type "np.function?" into Jupyter (where function is the name of your function of course).

Task 1: The random walk.

The random walk is a simple model of great importance for understanding phenomena such as thermal transport, charge diffusion, the stock market, bacteria and many others.

Write a script that implements a random walk. That is, starting from 0, at every step decide randomly, with a 50:50 probability to either increase the number by one (move right) or reduce it by one (move left), and find the value after n-steps.

- As a first stage, implement this with a for-loop. Start with a list containing only [0] and append randomly a number that is either one larger or one smaller than the previous number. To make a random choice, you can use np.random.randint(2), which gives you a random integer that is either 0 or 1 (the "endpoint" 2 is not included in randint()and the standard start is 0).
- Make a line-plot showing the position of one trajectory after 100 steps.

To be more efficient (and later scale to larger dimensions), instead of running a loop, you can directly use the np.random.randint() function to give you an array of random numbers. For documentation and more examples, see https://numpy.org/doc/stable/reference/random/generated/numpy.random.randint.html For example, if you use

```
np.random.randint(2,size=100)*2-1
```

You will get an array of 100 numbers which are either +1 or -1 (we have scaled the 0,1 that randint gives us).

To then get the position after i steps, you can use the np.cumsum() function (but be careful to not miss the starting point – you can use np.concatenate() to add it).

- Make a line-plot to confirm that this worked correctly. Then, using size=(10,100) in randint you can generate 10 trajectories at once – but make sure you sum over the right axis in cumsum. Plot the result to check.
- Hence, create 1000 trajectories, and plot a histogram of their final positions after (a) 10 steps, (b) 100 steps. (c)1000 steps
- Compute the mean and standard deviation (SD) of those distributions of final positions after 10 / 100 / 1000 steps.
- Plot the SD as a function of step number (optional: show mean +-SD as shaded areas)

- Now change the probability to 60:40 and 70:30 and 99:1, again plot the SD and the mean as a function of step size (optional: with SD shaded). What type of function do you recognize, and how does the slope depend on probability?
- Optional: Plot the SD and the mean position after 1000 steps as a function of probability for moving left.

Task 2: Be a random number generator

Generating "true" random number is hard, in fact there are often unwanted correlations that appear. Let's illustrate that with "hand-made" random numbers. For this you will need a keyboard, if you don't have one, work with your neighbour.

- Keeping 8 fingers on the 1-8 numbers on your keyboard, hit the keys as randomly as possible until you have created a string containing at least 500 digits (this will take about 1 minute. Using
  `a = np.array(list(str(3974582739452...871263)),dtype=int)`
  you can convert this into an array.
- Use np.size(a) to check how long your array is, then create another array b of the same size containing np.random.randint-generated random numbers.
- Compute the mean and standard deviation of a and b. Are they what you expect?
- Plot a histogram of a and b (use plt.hist([a,b],bins=np.arange(1-0.5,8+1+0.5)) to get the right bin centers). Does it fit with what you expect?

Even if a distribution has the "expected" probability for each number, there may still be hidden correlations. To find some, let's look at the *distance* between consecutive numbers. To compute it, we can use

```
aDistance = a[1:]-a[0:-1]
```

- For Laplacian random numbers from 1-8, the probability one number to be the same as the previous number is 1/8. How often does this occur in a? How often in b? (a quick was to find this is plotting the histogram of aDistance). What do you make of this?
- Optional: The probability of the number 6 coming after 5 or 7 would be 2/7 for random numbers. What is the frequency of this event in a and in b?
- Optional: Go further and compute the frequency of a number being the same as the one before AND the one before that (probability 1/36).