question 1 views

Daily Challenge 27.2

(**Due: Sunday 4/14 at 12:00 noon Eastern**)

In DC 24.3, you investigated the probability distribution for eigenvalues of random *symmetric* matrices, i.e. matrices A with the property that $A^T = A$.

We restricted to this case because the eigenvalues of symmetric matrices are real numbers, while the eigenvalues of general matrices can be complex. In the symmetric case, the answer you found was that the eigenvalue distribution tended toward the *semicircle distribution*, namely

$$f(x) = \frac{2}{\pi R^2}\sqrt{R^2 - x^2}.$$

Today let's relax this assumption and investigate non-symmetric matrices.

(**Part a**) Write a function that runs an experiment to generate a large number of $N \times N$ matrices, where N is an input, then collects the eigenvalues of all of those matrices and returns them in a list.

The entries of the matrix should be random numbers drawn from a Gaussian distribution, like numpy.random.normal ("normal" is another word for "Gaussian"). Don't symmetrize the matrix by adding it to its transpose, as you did in DC 24.3.

Your function signature may look like

```
def run_random_matrix_experiment(n_trials=1000, N=10):
    """
    Generates n_trials matrices, each NxN, and returns a list of all their eigenvalues
    """

    eig_list = []

    for trial in range(n_trials):
        ## Make a random NxN matrix with normally-distributed entries
        ## You can use numpy.random.normal with the size option
        ## Then get the eigenvalues using numpy.linalg.eig and put them in the list

    return eig_list
```

(**Part b**) Run an experiment and take a look at your eigenvalues. They should be complex numbers like `1+2j` or something.

Write a function which takes in a list of complex numbers $z = x + iy$ and makes a matplotlib scatter plot of the points (x, y) in the complex plane.

You probably want something like

```
reals = [z.real for z in eig_list]
imags = [z.imag for z in eig_list]
plt.scatter(reals, imags)
```

(**Part c**) First run an experiment with small matrices, maybe 5×5 or 10×10. Plot the results. You should get something that doesn't look totally uniform. It might be in the rough shape of a circle but there will be some bunching up of the eigenvalues.

Then run an experiment with larger matrices, like 1000×1000. The scatter plot should tend toward a uniform distribution on a circle, i.e. the eigenvalues should be evenly spread out with no bunching. This is called the circular law.

How does the radius of the circle depend on N? For example, if I increase N from 1000×1000 to $10,000 \times 10,000$, how does the size of the circle change?

[If I recall correctly, the answer is that the radius grows like $\sqrt{N}$, but check me on this. Once upon a time I was very interested in such questions.]

Push the code to Github.

daily_challenge

Updated 2 days ago by Christian Ferko

the instructors' answer, *where instructors collectively construct a single answer*

Click to start off the wiki answer

followup discussions *for lingering questions and comments*