Scientific Computing Project 1

Jakob Schauser, pwn274

September 2023

a

(1) Implement centers,radii = gershgorin(A)

Okay,

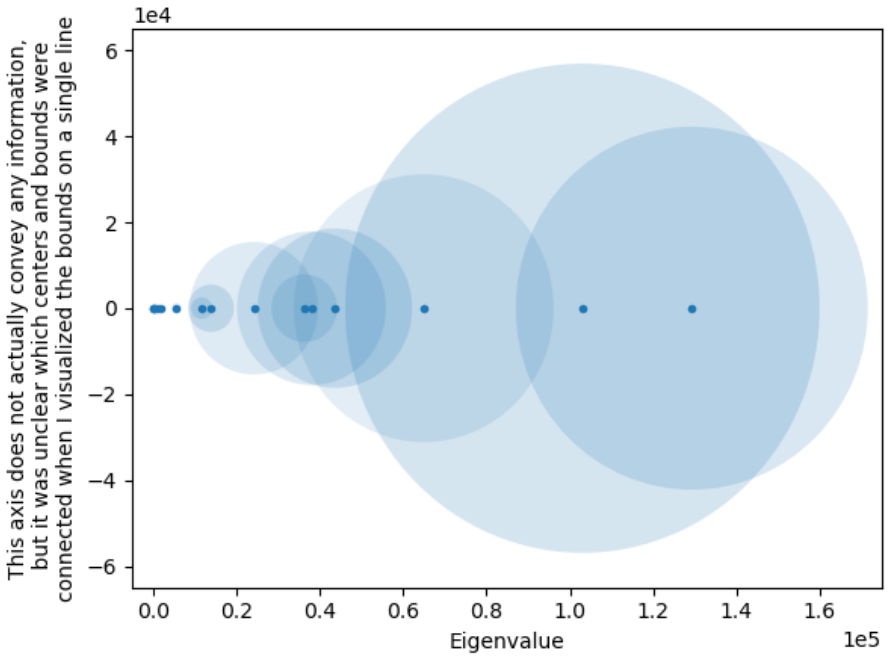
```
def gershgorin(A : ArrayLike) -> (ndarray, ndarray):
    radii = np.empty(A.shape[0])
    for i in range(A.shape[0]):
        # Sum of absolute values of all elements in row i, minus the absolute value of
    the diagonal element
        # this will most likely give a dumb floating point error, but I choose to think
    I will be fine
        radii[i] = np.sum(np.abs(A[i,:])) - np.abs(A[i,i])

    centers = np.diag(A)

    return centers, radii
```

(2) Localize the eigenvalues of K, and report the disk centers and radii

Here is what I found:

```
 The centers are:
 [1.29292219e+05 1.03041439e+05 6.49675787e+04 4.36124119e+04
 3.62737515e+04 3.79900994e+04 2.41669711e+04 1.16511587e+04
 1.38650805e+04 5.60054767e+03 1.17303879e+03 1.76077135e+03
 2.88423061e+02 8.68968906e+01 1.38933458e+01]
The radii are:
[4.22316467e+04 5.69275189e+04 3.11606709e+04 1.85323487e+04
 7.87051614e+03 1.78547026e+04 1.54147167e+04 2.51204139e+03
 5.50296953e+03 1.19528329e+03 3.41990248e+02 4.01214726e+02
 7.14173194e+01 2.54029437e+00 2.59094641e-01]
```

But that is impossible to gather anything from, so I made a graphical visualization:



b

(1) Implement a function `lambda = rayleigh_qt(A,x)`

ok

```
def rayleigh_qt(A : ArrayLike,x : ArrayLike) -> float:
    _lambda = x.T @ A @ x/(x.T @ x)
    return _lambda
```

(2) Implement a function x, k = power_iterate(A,x0) for power iteration. Don't forget to choose and implement a suitable convergence criterion

👍

```
def power_iterate(A : ArrayLike, x0 : ArrayLike) -> (ndarray, int):
    x0 = x0.copy()
    for k in range(1000):
        # copy for comparison
        ls = x0.copy()
        x0 = A @ x0

        x0 = x0/np.linalg.norm(x0)

        if (np.linalg.norm(x0 - ls) < 1e-8):
            break

    return x0, k
```

(3) Test it by finding the largest eigenvalue of the example matrices. Report the eigenvalue found, the Rayleigh residual (the residual of the Raleigh quotient as a least-squares system), and the number k of iterations used.

Here is what I've found:

Example #	Largest eigenvalue	iterations	residual
1	4.0	1	0.0
2	4.0	1	0.0
3	12.2989583961	17	2.398e-08
4	16.11684396	7	1.456e-09
5	68.6420807405	6	9.186e-09
6	2.000	23	5.754e-09

(4) What is the largest eigenvalue of K? Visualize your eigenfunction using show waves(x,basis set) to see the wave, and show nodes(x,basis set) to see where the sand will gather. The eigenfunction for the largest eigenvalue should have nodes along an 8 x 8 grid.

The eigenvalue is: 151362.6665
The number of iterations was: 39

I visualized the eigenfuntion, and the nodes did indeed lie on a grid

c

(1) Write a Rayleigh quotient iteration function x, k = rayleigh iterate(A,x0, shift0)

I implemented the Rayleigh quotient iteration as it is described in Algorithm 4.4 in the book. The most important parts are shown below:

```
def rayleigh_iterate(A : ArrayLike, x0 : ArrayLike, shift0 : float) -> (ndarray, int,
    float):
    x = x0.copy()
    shift = shift0

    for k in range(100):
        x = lu_solve(A - shift * np.eye(A.shape[0]), x)

        x = x/np.linalg.norm(x)

        shift = rayleigh_qt(A, x)

        if (rayleigh_residual(A,x,shift) < 1e-8):
            break

    return x, k, shift
```

(2) Test it with the example matrices, and report the eigenvalues found, Rayleigh residual, and the number k of iterations used

The highest eigenvalues are shown here:

Example #	Highest eigenvalue found	actual	residual	iterations used	All eigenvalues found
1	4	4	6.28031e-16	1	yes
2	4.0	4	0.0	1	yes
3	12.29895	12.2990	5.71866e-09	7	yes
4	16.11684	16.1168	3.95239e-14	4	yes
5	68.64208	68.6420	3.74047e-12	7	no
6	2.000000	2	8.02106e-11	1	no

As can be seen, my "find-all-eigenvalues"-function was not perfect. For example #5 and example #6 my algorithm does not find all the correct eigenvalues. The found eigenvalues can be seen here:

Found eigenvalues for #5				
68.64208074	-3.64208074	-3.64208074	-3.64208074	-3.64208074
Actual eigenvalues for #5				
68.64208074	-3.64208074	0.	0.	0.
Found eigenvalues for #6				
2	2	1	1	1
Actual eigenvalues for #6				
2	2	2	1	0

d

(1) Why can you not get all the eigenvalues and -vectors with pure power iteration?

Power iteration finds the eigenvector with the highest eigenvalue only. I am not sure if this is the answer you are looking for, though.

(2) Use your Rayleigh quotient iteration function together with the Gershgorin centers to calculate as many eigenvectors and eigenvalues of K as you can. Check using show nodes(x,basis set) that the eigenfunction with lowest eigenvalue looks like a cross

I find all eigenvalues of the K-matrix and numpy agrees with me. The lowest eigenstate plottet using the given tool looks as follows:

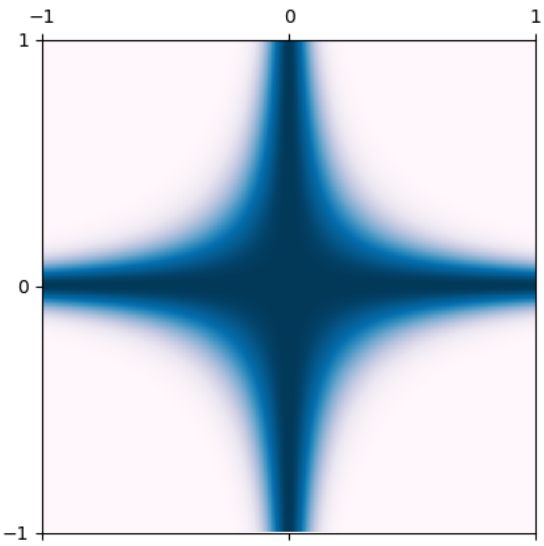


Figure 1: You must have seen this image a lot by now.

(I promise this is not just a screen-shot from the assignment text, even though that would have been very funny)

(3) Construct the transformation matrix T whose columns are the eigenvectors in order of ascending eigenvalues, and check that $K = T\Lambda T^{-1}$ with diagonal Λ

I am unsure what to tell you here, but I have checked and (to a precision of 10^{-5}) the T and Λ are correctly constructed :

```
np.isclose(Kmat - T @ np.diag(k_eigs[1]) @ inv(T), np.zeros_like(Kmat)).all()

> True
```

(4) Visualize your solution using the provided function show all wavefunction nodes(T,lambdas,basis set).

So pretty:

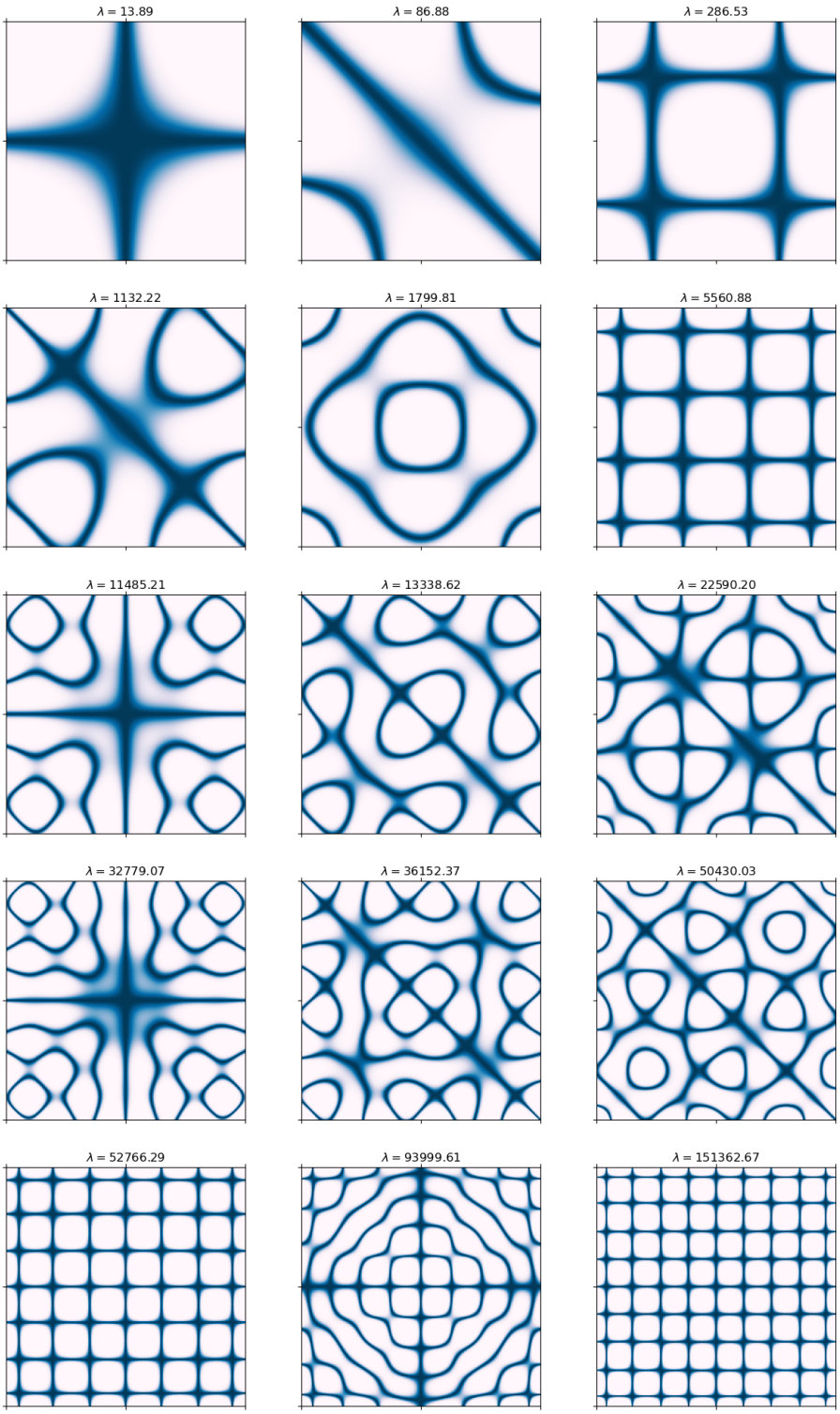


Figure 2: You have most likely also seen this one a couple of times