Eigenvalues and Eigenvectors

Few concepts to remember from linear algebra

Let $\boldsymbol{A}$ be an $n \times m$ matrix and the linear transformation $\boldsymbol{y = Ax}$

$$\boldsymbol{x} \in \mathcal{R}^{\boldsymbol{m}} \xrightarrow{\boldsymbol{A}} \boldsymbol{y} \in \mathcal{R}^{\boldsymbol{n}}$$

- Rank: maximum number of linearly independent columns or rows of $\boldsymbol{A}$
- Range$(\boldsymbol{A}) = \{\boldsymbol{y = Ax} \mid \forall \boldsymbol{x}\}$
- Null$(\boldsymbol{A}) = \{\boldsymbol{x} \mid \boldsymbol{Ax = 0}\}$

Eigenvalue problem

Let $\boldsymbol{A}$ be an $n \times n$ matrix:

$\boldsymbol{x} \neq \boldsymbol{0}$ is an **eigenvector** of $\boldsymbol{A}$ if there exists a scalar λ such that

$$\boldsymbol{A}\,\boldsymbol{x} = \lambda\,\boldsymbol{x}$$

where λ is called an **eigenvalue**.

If $\boldsymbol{x}$ is an eigenvector, then $\alpha\boldsymbol{x}$ is also an eigenvector. Therefore, we will usually seek for **normalized eigenvectors**, so that

$$\|\boldsymbol{x}\| = 1$$

Note: When using Python, numpy.linalg.eig will normalize using $p=2$ norm.

How do we find eigenvalues?

Linear algebra approach:

$$\boldsymbol{A}\,\boldsymbol{x} = \lambda\,\boldsymbol{x}$$
$$(\boldsymbol{A} - \lambda\,\boldsymbol{I})\boldsymbol{x} = \boldsymbol{0}$$

Therefore the matrix $(\boldsymbol{A} - \lambda\,\boldsymbol{I})$ is singular $\Longrightarrow det(\boldsymbol{A} - \lambda\,\boldsymbol{I}) = 0$

$p(\lambda) = det(\boldsymbol{A} - \lambda\,\boldsymbol{I})$ is the characteristic polynomial of degree n.

In most cases, there is no analytical formula for the eigenvalues of a matrix (Abel proved in 1824 that there can be no formula for the roots of a polynomial of degree 5 or higher) $\Longrightarrow$ **Approximate the eigenvalues numerically**!

Example

$$\boldsymbol{A} = \begin{pmatrix} 2 & 1 \\ 4 & 2 \end{pmatrix} \qquad det\begin{pmatrix} 2-\lambda & 1 \\ 4 & 2-\lambda \end{pmatrix} = 0$$

Solution of characteristic polynomial gives: $\lambda_1 = 4, \lambda_2 = 0$

To get the eigenvectors, we solve: $\boldsymbol{A}\,\boldsymbol{x} = \lambda\,\boldsymbol{x}$

$$\begin{pmatrix} 2-(4) & 1 \\ 4 & 2-(4) \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad \boldsymbol{x} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

$$\begin{pmatrix} 2-(0) & 1 \\ 4 & 2-(0) \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad \boldsymbol{x} = \begin{pmatrix} -1 \\ 2 \end{pmatrix}$$

Notes:

The matrix $\boldsymbol{A}$ is singular (det(A)=0), and rank($\boldsymbol{A}$)=1

The matrix has two distinct real eigenvalues

The eigenvectors are linearly independent

Diagonalizable Matrices

A $n \times n$ matrix $\boldsymbol{A}$ with n linearly independent eigenvectors $\boldsymbol{u}$ is said to be **diagonalizable**.

$$\boldsymbol{A}\,\boldsymbol{u_1} = \lambda_1\,\boldsymbol{u_1},$$
$$\boldsymbol{A}\,\boldsymbol{u_2} = \lambda_2\,\boldsymbol{u_2},$$
$$\dots$$
$$\boldsymbol{A}\,\boldsymbol{u_n} = \lambda_n\,\boldsymbol{u_n},$$

In matrix form:

$$\boldsymbol{A}\,(\boldsymbol{u_1} \quad \dots \quad \boldsymbol{u_n}) = (\lambda_1\boldsymbol{u_1} \quad \dots \quad \lambda_n\boldsymbol{u_n}) = (\boldsymbol{u_1} \quad \dots \quad \boldsymbol{u_n}) \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \lambda_n \end{pmatrix}$$

This corresponds to a similarity transformation

$$\boldsymbol{AU} = \boldsymbol{UD} \Leftrightarrow \boldsymbol{A} = \boldsymbol{UDU^{-1}}$$

Example

$$\boldsymbol{A} = \begin{pmatrix} 2 & 1 \\ 4 & 2 \end{pmatrix} \quad det\begin{pmatrix} 2-\lambda & 1 \\ 4 & 2-\lambda \end{pmatrix} = 0$$

Solution of characteristic polynomial gives: $\lambda_1 = 4, \lambda_2 = 0$

To get the eigenvectors, we solve: $\boldsymbol{A}\,\boldsymbol{x} = \lambda\,\boldsymbol{x}$

$$\begin{pmatrix} 2-(4) & 1 \\ 4 & 2-(4) \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad \boldsymbol{x} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

or normalized eigenvector ($p = 2$ norm)

$$\boldsymbol{x} = \begin{pmatrix} 0.447 \\ 0.894 \end{pmatrix}$$

$$\begin{pmatrix} 2-(0) & 1 \\ 4 & 2-(0) \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad \boldsymbol{x} = \begin{pmatrix} -1 \\ 2 \end{pmatrix}$$

$$\boldsymbol{x} = \begin{pmatrix} -0.447 \\ 0.894 \end{pmatrix}$$

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^{-1} \qquad \boldsymbol{U} = \begin{pmatrix} 0.447 & -0.447 \\ 0.894 & 0.894 \end{pmatrix} \qquad \boldsymbol{D} = \begin{pmatrix} 4 & 0 \\ 0 & 0 \end{pmatrix}$$

Notes:

The matrix $\boldsymbol{A}$ is singular (det(A)=0), and rank($\boldsymbol{A}$)=1

Since $\boldsymbol{A}$ has two linearly independent eigenvectors, the matrix $\boldsymbol{U}$ is full rank, and hence, the matrix $\boldsymbol{A}$ is diagonalizable.

Some things to remember about eigenvalues:

- Eigenvalues can have zero value
- Eigenvalues can be negative
- Eigenvalues can be real or complex numbers
- A $n \times n$ real matrix can have complex eigenvalues
- The eigenvalues of a $n \times n$ matrix are not necessarily unique. In fact, we can define the multiplicity of an eigenvalue.
- If a $n \times n$ matrix has n linearly independent eigenvectors, then the matrix is diagonalizable

How can we get eigenvalues numerically?

Assume that $\boldsymbol{A}$ is diagonalizable (i.e., it has n linearly independent eigenvectors $\boldsymbol{u}$). We can propose a vector $\boldsymbol{x}$ which is a linear combination of these eigenvectors:

$$\boldsymbol{x} = \alpha_1 \boldsymbol{u}_1 + \alpha_2 \boldsymbol{u}_2 + \cdots + \alpha_n \boldsymbol{u}_n$$

Then we evaluate $\boldsymbol{A}\,\boldsymbol{x}$:

$$\boldsymbol{A}\,\boldsymbol{x} = \alpha_1 \boldsymbol{A}\boldsymbol{u}_1 + \alpha_2 \boldsymbol{A}\boldsymbol{u}_2 + \cdots + \alpha_n \boldsymbol{A}\boldsymbol{u}_n$$

And since $\boldsymbol{A}\boldsymbol{u}_1 = \lambda_1 \boldsymbol{u}_1$ we can also write:

$$\boldsymbol{A}\,\boldsymbol{x} = \alpha_1 \lambda_1 \boldsymbol{u}_1 + \alpha_2 \lambda_2 \boldsymbol{u}_2 + \cdots + \alpha_n \lambda_n \boldsymbol{u}_n$$

where λ_i is the eigenvalue corresponding to eigenvector $\boldsymbol{u}_i$ and we ***assume***

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|$$

Power Iteration

Our goal is to find an eigenvector $\boldsymbol{u}_i$ of $\boldsymbol{A}$. We will use an iterative process, where we start with an initial vector, where here we assume that it can be written as a linear combination of the eigenvectors of $\boldsymbol{A}$.

$$\boldsymbol{x}_0 = \alpha_1 \boldsymbol{u}_1 + \alpha_2 \boldsymbol{u}_2 + \cdots + \alpha_n \boldsymbol{u}_n$$

And multiply by $\boldsymbol{A}$ to get:

$$\begin{gathered}
\boldsymbol{x}_1 = \boldsymbol{A}\,\boldsymbol{x}_0 = \alpha_1 \lambda_1 \boldsymbol{u}_1 + \alpha_2 \lambda_2 \boldsymbol{u}_2 + \cdots + \alpha_n \lambda_n \boldsymbol{u}_n \\
\boldsymbol{x}_2 = \boldsymbol{A}\,\boldsymbol{x}_1 = \alpha_1 (\lambda_1)^2 \boldsymbol{u}_1 + \alpha_2 (\lambda_2)^2 \boldsymbol{u}_2 + \cdots + \alpha_n (\lambda_n)^2 \boldsymbol{u}_n \\
\vdots \\
\boldsymbol{x}_k = \boldsymbol{A}\,\boldsymbol{x}_{k-1} = \alpha_1 (\lambda_1)^k \boldsymbol{u}_1 + \alpha_2 (\lambda_2)^k \boldsymbol{u}_2 + \cdots + \alpha_n (\lambda_n)^k \boldsymbol{u}_n
\end{gathered}$$

Or rearranging…

$$\boldsymbol{x}_k = (\lambda_1)^k \left[\alpha_1 \boldsymbol{u}_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \boldsymbol{u}_2 + \cdots + \alpha_n \left(\frac{\lambda_n}{\lambda_1} \right)^k \boldsymbol{u}_n \right]$$

Power Iteration

$$\boldsymbol{x}_k = (\lambda_1)^k \left[\alpha_1 \boldsymbol{u}_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k \boldsymbol{u}_2 + \cdots + \alpha_n \left(\frac{\lambda_n}{\lambda_1}\right)^k \boldsymbol{u}_n \right]$$

Assume that $\alpha_1 \neq 0$, the term $\alpha_1 \boldsymbol{u}_1$ dominates the others when k is very large.

Since $|\lambda_1| > |\lambda_2|$, we have $\left(\frac{\lambda_2}{\lambda_1}\right)^k \ll 1$ when k is large

Hence, as k increases, $\boldsymbol{x}_k$ converges to a multiple of the first eigenvector $\boldsymbol{u}_1$, i.e.,

$$\lim_{k\to\infty} \frac{\boldsymbol{x}_k}{(\lambda_1)^k} = \alpha_1 \boldsymbol{u}_1 \quad \text{or} \quad \boldsymbol{x}_k \to \alpha_1 (\lambda_1)^k \boldsymbol{u}_1$$

How can we now get the eigenvalues?

If $\boldsymbol{x}$ is an eigenvector of $\boldsymbol{A}$ such that

$$\boldsymbol{A}\,\boldsymbol{x} = \lambda\,\boldsymbol{x}$$

then how can we evaluate the corresponding eigenvalue λ?

$$\lambda = \frac{\boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}}$$

Rayleigh coefficient

Normalized Power Iteration

$$\boldsymbol{x}_k = (\lambda_1)^k \left[\alpha_1 \boldsymbol{u}_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \boldsymbol{u}_2 + \cdots + \alpha_n \left(\frac{\lambda_n}{\lambda_1} \right)^k \boldsymbol{u}_n \right]$$

$\boldsymbol{x_0}$ = arbitrary nonzero vector

$$\boldsymbol{x_0} = \frac{\boldsymbol{x_0}}{\|\boldsymbol{x_0}\|}$$

for $k = 1,2, \ldots$

$$\boldsymbol{y}_k = \boldsymbol{A}\, \boldsymbol{x}_{k-1}$$

$$\boldsymbol{x}_k = \frac{\boldsymbol{y}_k}{\|\boldsymbol{y}_k\|}$$

Normalized Power Iteration

$$\boldsymbol{x}_k = (\lambda_1)^k \left[\alpha_1 \boldsymbol{u}_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \boldsymbol{u}_2 + \cdots + \alpha_n \left(\frac{\lambda_n}{\lambda_1} \right)^k \boldsymbol{u}_n \right]$$

What if the starting vector $\boldsymbol{x_0}$ have no component in the dominant eigenvector $\boldsymbol{u}_1$ ($\alpha_1 = 0$)?

Demo "Power Iteration

Normalized Power Iteration

$$\boldsymbol{x}_k = (\lambda_1)^k \left[\alpha_1 \boldsymbol{u}_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \boldsymbol{u}_2 + \cdots + \alpha_n \left(\frac{\lambda_n}{\lambda_1} \right)^k \boldsymbol{u}_n \right]$$

What if the first two largest eigenvalues (in magnitude) are the same, $|\lambda_1| = |\lambda_2|$?

$$\boldsymbol{x}_k = (\lambda_1)^k \alpha_1 \boldsymbol{u}_1 + (\lambda_1)^k \left(\frac{\lambda_2}{\lambda_1} \right)^k \alpha_2 \boldsymbol{u}_2 + (\lambda_1)^k \left[\ldots + \alpha_n \left(\frac{\lambda_n}{\lambda_1} \right)^k \boldsymbol{u}_n \right]$$

Demo “Power Iteration

Potential pitfalls

1. Starting vector $\boldsymbol{x_0}$ may have no component in the dominant eigenvector $\boldsymbol{u}_1$ ($\alpha_1 = 0$). This is usually unlikely to happen if $\boldsymbol{x_0}$ is chosen randomly, and in practice not a problem because rounding will usually introduce such component.

2. Risk of eventual overflow (or underflow): in practice the approximated eigenvector is normalized at each iteration (Normalized Power Iteration)

3. First two largest eigenvalues (in magnitude) may be the same: $|\lambda_1| = |\lambda_2|$. In this case, power iteration will give a vector that is a linear combination of the corresponding eigenvectors:
   - If signs are the same, the method will converge to correct magnitude of the eigenvalue. If the signs are different, the method will not converge.
   - This is a "real" problem that cannot be discounted in practice.

Error

$$\boldsymbol{x}_k = (\lambda_1)^k \left[\alpha_1 \boldsymbol{u}_1 + \underbrace{\alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k \boldsymbol{u}_2 + \cdots + \alpha_n \left(\frac{\lambda_n}{\lambda_1}\right)^k \boldsymbol{u}_n}_{Error} \right]$$

We can see from the above that the rate of convergence depends on the ratio $\left|\frac{\lambda_2}{\lambda_1}\right|$, that is:

$$\left\| (\lambda_1)^{-k} \boldsymbol{x}_k - \alpha_1 \boldsymbol{u}_1 \right\| = O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k \right)$$

Convergence and error

$$\boldsymbol{x}_k = \boldsymbol{u}_1 + \underbrace{\left(\frac{\alpha_2}{\alpha_1}\right)\left(\frac{\lambda_2}{\lambda_1}\right)^k \boldsymbol{u}_2}_{\boldsymbol{e}_k} + \cdots$$

$$\frac{\boldsymbol{e}_{k+1}}{\boldsymbol{e}_k} \approx \left|\frac{\lambda_2}{\lambda_1}\right|$$

Power method has linear convergence, which is quite slow.

Iclicker question

Suppose you are given a matrix with eigenvalues 3, 4, and 5.
You use (normalized) power iteration to approximate one of the eigenvectors $\|\mathbf{x}\|$. For simplicity, assume $\|\mathbf{x}\| = 1$. Your initial guess $\mathbf{x}_0$ has a norm of the error $\|\mathbf{x} - \mathbf{x}_0\| = 0.3$.

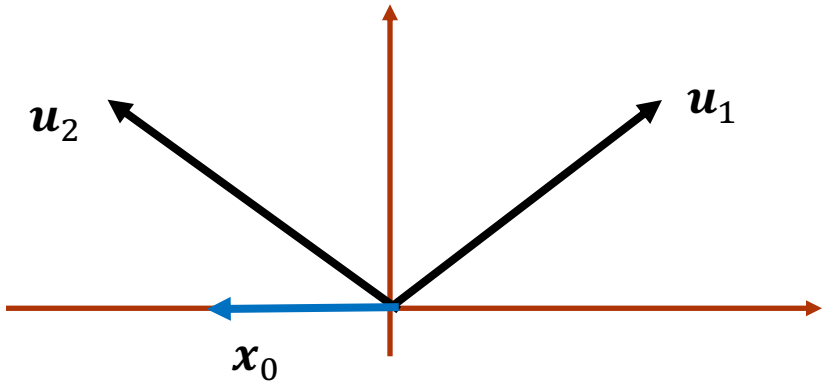
How big will the error be after three rounds of normalized power iteration?

(Note that for normalized power iteration, all vectors under consideration have norm 1, so the absolute and the relative error are the same.)

A) 0.1536
B) 0.192
C) 0.09
D) 0.027

Iclicker question

The matrix $\boldsymbol{A} = \begin{pmatrix} 3 & 1 \\ 1 & 3 \end{pmatrix}$ has eigenvalues $(4,2)$ and corresponding eigenvectors $\boldsymbol{u_1} = (1,1)$ and $\boldsymbol{u_2} = (-1,1)$.



Suppose we want to use the normalized power iteration, starting from $x_0 = (-0.5,0)$. Select the correct statement

A) Normalized power iteration will not converge

B) Normalized power iteration will converge to the eigenvector corresponding to the eigenvalue 2.

C) Normalized power iteration will converge to the eigenvector corresponding to the eigenvalue 4.

Iclicker question

Suppose $\boldsymbol{x}$ is an eigenvector of $\boldsymbol{A}$ such that

$$\boldsymbol{A}\,\boldsymbol{x} = \lambda\,\boldsymbol{x}$$

What is an eigenvalue of $\boldsymbol{A}^{-1}$?

A) λ
B) $-\lambda$
C) $1/\lambda$
D) $-\frac{1}{\lambda}$
E) Can't tell without knowing λ

Inverse Power Method

Previously we learned that we can use the Power Method to obtain the largest eigenvalue and corresponding eigenvector, by using the update

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\,\boldsymbol{x}_k$$

Suppose there is a single smallest eigenvalue of $\boldsymbol{A}$. With the previous ordering

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots > |\lambda_n|$$

the smallest eigenvalue is λ_n. When computing the eigenvalues of the inverse matrix $\boldsymbol{A}^{-1}$, we get the following ordering

$$\left|\frac{1}{\lambda_n}\right| > \left|\frac{1}{\lambda_{n-1}}\right| \geq \cdots \geq \left|\frac{1}{\lambda_1}\right|$$

And hence we can use the Power Method update on the matrix $\boldsymbol{A}^{-1}$ to compute the dominant eigenvalue $\frac{1}{\lambda_n}$, i.e.,

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}^{-1}\,\boldsymbol{x}_k$$

Iclicker question

Which code snippet is the best option to compute the smallest eigenvalue of the matrix $\boldsymbol{A}$?

A)
```
x = x0/la.norm(x0)
for k in range(30):
    x = la.solve(A, x)
    x = x/la.norm(x)
```

B)
```
x = x0/la.norm(x0)
for k in range(30):
    x = la.inv(A)@x
    x = x/la.norm(x)
```

C)
```
x = x0/la.norm(x0)
for k in range(30):
    P, L, U = sla.lu(A)
    y = sla.solve_triangular(L, np.dot(P.T, x), lower=True)
    x = sla.solve_triangular(U, y)
    x = x/la.norm(x)
```

D)
```
x = x0/la.norm(x0)
P, L, U = sla.lu(A)
for k in range(30):
    y = sla.solve_triangular(L, np.dot(P.T, x), lower=True)
    x = sla.solve_triangular(U, y)
    x = x/la.norm(x)
```

E) I have no idea!

Inverse Power Method

Note that the update

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}^{-1}\,\boldsymbol{x}_k$$

can be instead written as

$$\boldsymbol{A}\,\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$$

Where $\boldsymbol{x}_k$ is know and we need to solve for $\boldsymbol{x}_{k+1}$ (we are just solving a linear system of equations!). Since the matrix $\boldsymbol{A}$ does not change from iteration to the next, we can factorize the matrix once and then perform a series of backward and forward substitutions.

Recall $\boldsymbol{PA} = \boldsymbol{LU}$ and $\boldsymbol{A}\,\boldsymbol{x} = \boldsymbol{b}$ resulting in $\boldsymbol{LU}\,\boldsymbol{x} = \boldsymbol{Pb}$

Hence we can efficiently solve

$$\boldsymbol{L}\,\boldsymbol{y} = \boldsymbol{P}\,\boldsymbol{x}_k$$
$$\boldsymbol{U}\,\boldsymbol{x}_{k+1} = \boldsymbol{y}$$

Cost of computing eigenvalues using inverse power iteration

```
x = x0/la.norm(x0)
for k in range(30):
    x = la.solve(A, x)
    x = x/la.norm(x)
```

```
x = x0/la.norm(x0)
for k in range(30):
    x = la.inv(A)@x
    x = x/la.norm(x)
```

```
x = x0/la.norm(x0)
for k in range(30):
    P, L, U = sla.lu(A)
    y = sla.solve_triangular(L, np.dot(P.T, x), lower=True)
    x = sla.solve_triangular(U, y)
    x = x/la.norm(x)
```

```
x = x0/la.norm(x0)
P, L, U = sla.lu(A)
for k in range(30):
    y = sla.solve_triangular(L, np.dot(P.T, x), lower=True)
    x = sla.solve_triangular(U, y)
    x = x/la.norm(x)
```

Iclicker question

What is the approximated cost of computing the largest eigenvalue using Power Method?

A) $k\,n$
B) $n^2 + k\,n$
C) $k\,n^2$
D) $n^3 + k\,n^2$
E) $NOTA$

Iclicker question

Suppose $\boldsymbol{x}$ is an eigenvector of $\boldsymbol{A}$ such that $\boldsymbol{A}\,\boldsymbol{x} = \lambda_1\,\boldsymbol{x}$ and also $\boldsymbol{x}$ is an eigenvector of $\boldsymbol{B}$ such that $\boldsymbol{B}\,\boldsymbol{x} = \lambda_2\,\boldsymbol{x}$. What is an eigenvalue of

What is an eigenvalue of $(\boldsymbol{A} + \frac{\mathbf{1}}{\mathbf{2}}\boldsymbol{B})^{-1}$?

A) $\frac{\lambda_1}{2\lambda_1+\lambda_2}$

B) $\frac{\lambda_2}{2\lambda_1+\lambda_2}$

C) $\frac{2}{2\lambda_1+\lambda_2}$

D) $\frac{\lambda_1}{2\lambda_2+\lambda_1}$

E) $\frac{\lambda_2}{2\lambda_2+\lambda_1}$

Iclicker question

Suppose $\boldsymbol{x}$ is an eigenvector of $\boldsymbol{A}$ such that $\boldsymbol{A}\,\boldsymbol{x} = \lambda\,\boldsymbol{x}$, but λ is not the largest or smallest eigenvalue. We want to compute the eigenvalue λ that is close to a given number σ. Which of the following modified matrices will give such eigenvalue?

A) $(\boldsymbol{A} - \sigma\boldsymbol{I})$

B) $(\boldsymbol{A} - \sigma\boldsymbol{I})^{-1}$

C) $(1 - \sigma)\,\boldsymbol{A}$

D) $\frac{1}{\sigma}\,\boldsymbol{A}$

E) I still have no clue how to answer to these iclicker questions…

Eigenvalues of a Shifted Inverse Matrix

Suppose the eigenpairs $(\boldsymbol{x}, \lambda)$ satisfy $\boldsymbol{A}\boldsymbol{x} = \lambda\, \boldsymbol{x}$.

We can describe the eigenvalues for the shifted inverse matrix as

$$(\boldsymbol{A} - \sigma \boldsymbol{I})^{-1}\boldsymbol{x} = \bar{\lambda}\, \boldsymbol{x}$$
$$\boldsymbol{I}\boldsymbol{x} = \bar{\lambda}(\lambda \boldsymbol{I} - \sigma \boldsymbol{I})\boldsymbol{x}$$
$$\bar{\lambda} = \frac{1}{\lambda - \sigma}$$

Hence the eigensystem problem is

$$(\boldsymbol{A} - \sigma \boldsymbol{I})^{-1}\boldsymbol{x} = \frac{1}{\lambda - \sigma}\boldsymbol{x}$$

Eigenvalues of a Shifted Inverse Matrix

We use the update

$$(\boldsymbol{A} - \sigma \boldsymbol{I})\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$$

To obtain the eigenpair $(\boldsymbol{x}, \lambda)$ that satisfy $\boldsymbol{A}\boldsymbol{x} = \lambda\, \boldsymbol{x}$ such that λ is an eigenvalue close to the number σ

We can factorize the matrix $\boldsymbol{B} = (\boldsymbol{A} - \sigma \boldsymbol{I})$ such that $\boldsymbol{P}\boldsymbol{B} = \boldsymbol{L}\boldsymbol{U}$ and then efficiently solve

$$\boldsymbol{L}\, \boldsymbol{y} = \boldsymbol{P}\, \boldsymbol{x}_k$$
$$\boldsymbol{U}\, \boldsymbol{x}_{k+1} = \boldsymbol{y}$$