Power Method: Eigenvalue & Eigenvector

MPHYCC-05 Unit-IV Semester-II

Power Method for Approximating Eigenvalue & vector:

In the previous section we saw that the eigenvalues of n by n matrix are computed by solving the characteristic equation $|A - \lambda I| = 0$. The equation is a polynomial in λ of degree n. For large value of n, it is difficult and time consuming to solve the polynomial equation. Moreover, numerical techniques for approximating roots of polynomial equations of high degree are sensitive to rounding errors. The method presented here can be used only to find the eigenvalue of matrix A which is largest in absolute value. And the eigenvalue is known as the **dominant eigenvalue** of A. Although this restriction may seem severe, dominant eigenvalues are of primary interest in many physical applications. For instance, Google uses it to calculate the PageRank of documents in their search engine The power iteration method is especially suitable for sparse matrices, such as the web matrix, or as the matrix-free method that does not require storing the coefficient matrix explicitly, but can instead access a function evaluating matrix-vector products.

The power method can be used when:

- The matrix A is n by n matrix and having n linearly independent eigenvectors, v_1, v_2, v_3, …, v_n
- The eigenvalues can be ordered in magnitude as

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \ldots . \geq |\lambda_n|$$

And in this case λ_1 is called as dominant eigenvalues of A.

Now if A satisfies these conditions, then any vector, x^o can be written as a linear combination of v_1, v_2, v_3, …, v_n as:

$$x^o = c_1 v_1 + c_2 v_2 + c_3 v_3 + \cdots + c_n v_n$$

Where c's are the coefficients. Operating A on the above equation gives:

$$Ax^o = \lambda_1 c_1 v_1 + \lambda_2 c_2 v_2 + \lambda_3 c_3 v_3 + \cdots + \lambda_n c_n v_n$$

$$A^2 x^o = \lambda_1^2 c_1 v_1 + \lambda_2^2 c_2 v_2 + \lambda_3^2 c_3 v_3 + \cdots + \lambda_n^2 c_n v_n$$

$$A^3 x^o = \lambda_1^3 c_1 v_1 + \lambda_2^3 c_2 v_2 + \lambda_3^3 c_3 v_3 + \cdots + \lambda_n^3 c_n v_n$$

……..

$$A^k x^o = \lambda_1^k c_1 v_1 + \lambda_2^k c_2 v_2 + \lambda_3^k c_3 v_3 + \cdots + \lambda_n^k c_n v_n$$

If *k* is very large then the first term on the right hand side is dominating over other terms as λ_1 is greater than the other eigenvalues.

For large powers of *k*, and by properly scaling this sequence, we will be able to find the good approximation of the dominant eigenvector of *A* say x. Once we have x, then using **Rayleigh quotient** which states: "If **x** is an eigenvector of a matrix *A*, then its corresponding eigenvalue is given by

$$\frac{Ax.x}{x.x} = \lambda$$

Thus we could able to calculate the eigenvalue. We can also use the alternative way to calculate the eigenvalues which is as follows.

The last equation as:

$$A^k x^o = \lambda_1^k c_1 v_1 + \lambda_2^k c_2 v_2 + \lambda_3^k c_3 v_3 + \cdots + \lambda_n^k c_n v_n$$

As *k* tends to infinity the first term is also tending infinity. Thus to make it finite, we divide the last equation with λ_1^k which transform the last equation as:

$$\frac{A^k x^o}{\lambda_1^k} = c_1 v_1 + \frac{\lambda_2^k}{\lambda_1^k} c_2 v_2 + \frac{\lambda_3^k}{\lambda_1^k} c_3 v_3 + \cdots + \frac{\lambda_n^k}{\lambda_1^k} c_n v_n$$

Because λ_1 is greater than the other eigenvalues thus as *k* goes to infinity, the above equation becomes

$$\frac{A^k x^o}{\lambda_1^k} = c_1 v_1$$

As long as $c_1 \neq 0$, this equation can give us an approximation to λ_1. And $c_1 \neq 0$ guaranteed when x^o is not orthogonal to v_1. Operating the above equation with A one more time lead to:

$$\frac{A^{k+1} x^o}{\lambda_1^{k+1}} = c_1 v_1$$

Now λ_1 is computed from these two equations; by multiplying these equations with a vector y which is not orthogonal to v_1 and dividing them lead us to:

$$\frac{A^{k+1} x^o . y}{A^k x^o . y} = \lambda_1$$

The powers of A give the name as power method. And $\frac{A^k x^o}{|A^k x^o|}$ converges to the (a multiple of) corresponding eigenvalue. The convergence is slowly if the second largest eigenvalue is close in magnitude to the dominant eigenvalue. Following steps are adopted in the power method:

- Assume a simple possible dominating eigenvector with norm equal to 1
- Apply the matrix A on the vector till it converges
- Find the approximated dominating eigenvector
- Calculate the dominating eigenvalues using Rayleigh quotient method or other as mention in this document

When to stop in the power method: In principle we would like to stop when the calculated value of dominating eigenvalue, λ_1^{cal} match with the actual eigenvalueλ_1^{actu}. However, in the most realistic problem we do not know theλ_1^{actu}. Thus we estimate $|\lambda_1^{cal} - \lambda_1^{actu}|$ and stop when this value is quite small. The $|\lambda_1^{cal} - \lambda_1^{actu}|$ can be calculated by using the relation:

$$|\lambda_1^{cal} - \lambda_1^{actu}| \leq \sqrt{\frac{A^{k+1}x^o.A^{k+1}x^o}{A^kx^o.A^kx^o} - (\lambda_1^{cal})^2}$$

$$= \sqrt{\frac{A^{k+1}x^o.A^{k+1}x^o}{A^kx^o.A^kx^o} - \left(\frac{A^{k+1}x^o.A^kx^o}{A^kx^o.A^kx^o}\right)^2}$$

Example:

The following example demonstrates this method very well.

Example 1: Use the power method to estimate the largest eigenvalue of the following matrix.

$$A = \begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix}$$

Solution: Let the initial guess eigenvector is

$$x^o = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

We operate the Matrix A to this eigenvector till having the approximate convergence. Thus

$$Ax^o = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \quad A^2x^o = \begin{bmatrix} 7 \\ 6 \end{bmatrix} \quad A^3x^o = \begin{bmatrix} 25 \\ 26 \end{bmatrix} \quad A^4x^o = \begin{bmatrix} 103 \\ 102 \end{bmatrix}$$

$$A^5x^o = \begin{bmatrix} 409 \\ 410 \end{bmatrix} \quad A^6x^o = \begin{bmatrix} 1639 \\ 1638 \end{bmatrix} \quad A^7x^o = \begin{bmatrix} 6553 \\ 6554 \end{bmatrix} \quad A^8x^o = \begin{bmatrix} 26215 \\ 26214 \end{bmatrix}$$

Now the resultant matrix converges to: $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$

Thus the dominating eigenvector is: $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$

In-order to get the corresponding dominating eigenvalue; we assume y:

$y = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ Because this y is not orthogonal to the dominating eigenvector: Thus the required eigenvalue is:

$$\lambda_1 = \frac{A^6x^o.y}{A^5x^o.y} = \frac{1639}{409} = 4.0073\ldots$$

Therefore, the dominating eigenvector and corresponding dominating Eigenvalue is:

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \text{ \& } 4$$

Python scripts and outputs:

Python script is given below to find the dominating eigenvalue and corresponding eigenvector of a given matrix.

Script:

```
……
# importing the numerical libraries of python
import numpy as np

# Entering the matrix, initial guess, number of iteration
A = eval(input('Enter the matrix A:'))# as np.array([[a11,a12],[a21,a22]])
x1 = eval(input('Enter initial guess of x:'))  # as [x01,x02]
x2 = eval(input('Enter initial guess of x:'))  # as [x01,x02] exactly same as x1
n = eval(input('Enter the number of Iterations:'))

# using the for loop and calculating the multiplication
#of power matrix and initial guess and looking for convergence
for i in range (n):
    x1=np.dot(A, x1)
    norm=np.linalg.norm(x1)
    #normalized form
    B=x1/norm
    print(B)
# using the for loop and calculating the multiplication of
#power matrix and initial guess just one power less than the previous one
for i in range (n-1):
    x2=np.dot(A, x2)

#calculation of approximate dominating eigenvector
C=np.min(abs(B))
eigvect=B/C

#calculating the approximate dominating corresponding eigenvalue
y = eval(input('Enter a non-orthogonal matrix:'))# as [y01,y02]
eigvalue=np.dot(x1,y)/np.dot(x2,y)

# printing dominating eigenvalue and corresponding eigenvalue
print('Dominating eigenvalue is', eigvalue)
print('and corresponding eigenvector is', eigvect)
……
```

Outputs:

After compiling the above python script and using the same matrix as described in the example with different initial guesses; we have following outputs:

……
With initial guess [1, 0] and for iteration 10

```
Enter the matrix A:np.array([[1,3],[2,2]])
Enter initial guess of x:[1,0]
Enter initial guess of x:[1,0]
Enter the number of Iterations:10
[0.4472136  0.89442719]
[0.7592566  0.65079137]
[0.69310872 0.72083306]
[0.71054763 0.70364911]
[0.70624288 0.70796963]
[0.70732253 0.70689097]
[0.70705283 0.70716073]
[0.70712027 0.70709329]
[0.70710341 0.70711015]
[0.70710762 0.70710594]
Enter a non-orthogonal matrix:[1,0]
Dominating eigenvalue is 4.000028610393202
and corresponding eigenvalue is [1.00000238 1.        ]
```
……...

Thus eigenvalue and eigenvector is close to expected value that is 4 & [1, 1]. However, for the initial guess [1, 1] and iteration 10; we have the exact required result as shown below.

……...

```
Enter the matrix A:np.array([[1,3],[2,2]])
Enter initial guess of x:[1,1]
Enter initial guess of x:[1,1]
Enter the number of Iterations:10
[0.70710678 0.70710678]
[0.70710678 0.70710678]
[0.70710678 0.70710678]
```

[0.70710678 0.70710678]
[0.70710678 0.70710678]
[0.70710678 0.70710678]
[0.70710678 0.70710678]
[0.70710678 0.70710678]
[0.70710678 0.70710678]
[0.70710678 0.70710678]
Enter a non orthogonal matrix:[1,0]
Dominating eigenvalue is 4.0
and corresponding eigenvalue is [1. 1.]
……...

For a different matrix with initial guess [1,1] and iteration 10, we have:

Enter the matrix A:np.array([[0.5,0.5],[0.2,0.8]])
Enter initial guess of x:[1,1]
Enter initial guess of x:[1,1]
Enter the number of Iterations:10
[0.70710678 0.70710678]
[0.70710678 0.70710678]
[0.70710678 0.70710678]
[0.70710678 0.70710678]
[0.70710678 0.70710678]
[0.70710678 0.70710678]
[0.70710678 0.70710678]
[0.70710678 0.70710678]
[0.70710678 0.70710678]
[0.70710678 0.70710678]
Enter a non-orthogonal matrix:[1,0]
Dominating eigenvalue is 1.0
and corresponding eigenvector is [1. 1.]
……...

For a 3 by 3 matrix with initial guess [1,1,1] and iteration 10 we have:

Enter the matrix A:np.array([[1,2,0],[-2,1,2],[1,3,1]])
Enter initial guess of x:[1,1,1]
Enter initial guess of x:[1,1,1]
Enter the number of Iterations:10

[0.50709255 0.16903085 0.84515425]
[0.38235956 0.38235956 0.84119102]
[0.39056673 0.4426423 0.80717125]
[0.41103969 0.41103969 0.81369082]
[0.41010447 0.40452482 0.81741911]
[0.40793707 0.40793707 0.81680763]
[0.40804075 0.40866324 0.81639274]
[0.40828286 0.40828286 0.81646201]
[0.40827133 0.4082022 0.8165081]
[0.40824445 0.40824445 0.81650042]
Enter a non-orthogonal matrix:[1,0,0]
Dominating eigenvalue is 2.9996613326559984
and corresponding eigenvector is [1. 1. 2.00002823]

Therefore, the exact dominating eigenvector and eigenvector is 3 & [1, 1, 2] which is very close to the approximated value resulting from this iterative method. However, we can have the exact value if we take the initial guess as [1, 1, 2] as we can see below.

Enter the matrix A:np.array([[1,2,0],[-2,1,2],[1,3,1]])
Enter initial guess of x:[1,1,2]
Enter initial guess of x:[1,1,2]
Enter the number of Iterations:10
[0.40824829 0.40824829 0.81649658]
[0.40824829 0.40824829 0.81649658]
[0.40824829 0.40824829 0.81649658]
[0.40824829 0.40824829 0.81649658]
[0.40824829 0.40824829 0.81649658]
[0.40824829 0.40824829 0.81649658]
[0.40824829 0.40824829 0.81649658]
[0.40824829 0.40824829 0.81649658]
[0.40824829 0.40824829 0.81649658]
[0.40824829 0.40824829 0.81649658]
Enter a non-orthogonal matrix:[1,0,0]
Dominating eigenvalue is 3.0
and corresponding eigenvector is [1. 1. 2.]
……..

Inverse Power Method:

It is method to find the smallest eigenvalue and corresponding eigenvector of a real non-singular matrix A if method is convergent.

Let A be an n by n matrix with eigenvalues $\lambda_1, \lambda_2, \lambda_3,...\lambda_n$ and associated eigenvectors $v_1, v_2, v_3,...v_n$. Let q be a number for which $\det(A-qI) \neq 0$ so q is not an eigenvalue of A.

Let $B = (A-qI)^{-1}$ so the eigenvalues of B are given by $\mu_1 = \frac{1}{\lambda_1 - q}$, $\mu_2 = \frac{1}{\lambda_2 - q}$, $\mu_3 = \frac{1}{\lambda_3 - q}$, …, $\mu_n = \frac{1}{\lambda_n - q}$. And corresponding eigenvectors are same as A that is $v_1, v_2, v_3,...v_n$. If λ_k is the eigenvalue that is closest to the number q, then μ_k is the dominant eigenvalue for B and so it can be determined using the power method. Moreover, to find the eigenvalue of A that is smallest in magnitude is equivalent to find the dominant eigenvalue of the matrix $B = A^{-1}$. And this is the inverse power method with q = 0. The steps adopted in the inverse power method are follows.

- First compute the matrix (A-qI)
- Then calculate its inverse that is $B = (A-qI)^{-1}$
- Apply power method and compute the dominating eigenvalue and corresponding eigenvector for matrix B

……