Find the Eigen values for the given matrix

Find the Eigen vectors for the given matrix

In [1]:

```
from sympy.interactive import printing
printing.init_printing(use_latex=True)
from sympy import *
from sympy.solvers import solve
from time import time


import sympy as sp

start1=time()
a11,a12,a13,a21,a22,a23,a31,a32,a33=map(int,input("Enter the matrix coefficients : ").split())
r1,r2,r3=[a11,a12,a13],[a21,a22,a23],[a31,a32,a33]
amat=sp.Matrix((r1,r2,r3))
print("Input matrix")
display(amat)


s1=a11+a22+a33
display('S1={0}+{1}+{2}'.format(a11,a22,a33))
display('s1={0}'.format(s1))
s2=(((a22*a33)-(a32*a23))+((a11*a33)-(a31*a13))+((a11*a22)-(a21*a12)))
display('s2=((({0}*{1})-({2}*{3}))+(({4}*{5})-({6}*{7}))+(({8}*{9})-({10}*{11}))'.format(a22,a33,a32,a23,a11,a33,a31,a13,
display('s2=({0}+{1}+{2})'.format(((a22*a33)-(a32*a23)),((a11*a33)-(a31*a13)),((a11*a22)-(a21*a12))))
s3=det(amat)
display('s3=det(amat)')
display("s3={0}".format(s3))
display('s1={0} s2={1} s3={2}'.format(s1,s2,s3))
x=sp.symbols('λ')
eq1=sp.Function("eq1")
eq1=Eq((x**3 - (s1)*x**2 +(s2)*x- s3))
display(eq1)
k=sp.factor(eq1)
display(k)
K=solve(k)
print("Eigen values are : ",K)

mat2=Matrix(([x,0,0],[0,x,0],[0,0,x]))
```

```
a,b,c,d,e,f,mi=sp.symbols('x1 x2 x3 * = -> -')
print(">>>>>>>>>>>>>>>>> To find eigen vector <<<<<<<<<<<<<<")
print("({0}-{1}{2}){3}={4}".format('A',x,'I','X',0))
cmat=sp.Matrix((0,0,0))
display(amat,mi,mat2,e,cmat,f)
display(amat-mat2)
xmat=sp.Matrix((a,b,c))
display(d,xmat,e,cmat)


for i in range(0,len(K)):
    print("\n\nIf λ={0}".format(K[i]))
    print("({0}-{1}{2}){3}={4}".format('A',K[i],'I','X',0))
    lmat1=Matrix(([K[i],0,0],[0,K[i],0],[0,0,K[i]]))
    display(amat,mi,lmat1,e,amat-lmat1)
    print('now')
    display(amat-lmat1,d,xmat,e,cmat,f)

    eq2=sp.Function("eq2")
    eq3=sp.Function("eq3")
    eq4=sp.Function("eq4")

    eq2=Eq((a11-K[i])*a+a12*b+a13*c)
    eq3=Eq(a21*a+(a22-K[i])*b+a23*c)
    eq4=Eq(a31*a+a32*b+(a33-K[i])*c)

    display(eq2,eq3,eq4)


    b1,b2,b3=[(a11-K[i]),a12,a13,0],[a21,(a22-K[i]),a23,0],[a31,a32,(a33-K[i]),0]

    system=Matrix((b1,b2,b3))
    print('\nBy Gaussian Elimination Method.....\n')
    n=sp.solve_linear_system(system,a,b,c)
    display(n)
    if(eq2!=eq3 and eq3!=eq4):
        print("\nBy Using Cofactor Method\n")
        xm=((a12*a23)-((a22-K[i])*a13))
        ym=(((a11-K[i])*a23)-(a21*a13))
        zm=(((a11-K[i])*(a22-K[i]))-(a21*a12))
        if(xm==0 and ym==0 and zm==0):
            subst=input("Which variable you want to assume the value : ")
            if(subst=='x1' or subst=='X1'):
                xm=1
            if(subst=='x2' or subst=='X2'):
```

```
                ym=-1
            if(subst=='x3' or subst=='X3'):
                zm=1
        print('({0}/{1})=(-({2}/{3}))=({4}/{5})'.format(a,xm,b,ym,c,zm))
        if(xm==(-ym)==zm):
            if(xm==(-ym)==zm==0):
                v1=Matrix([xm,-ym,zm])
            else:
                v1=Matrix([xm/xm,-ym/(-ym),zm/zm])
        elif((xm%2==0 and -ym%2==0 and zm%2==0)):
            v1=Matrix([xm/2,-ym/2,zm/2])
            if(xm!=1 and -ym!=1 and zm!=1):
                xm/=2
                ym/=2
                zm/=2
        elif((xm%3==0 and -ym%3==0 and zm%3==0)):
            v1=Matrix([xm/3,-ym/3,zm/3])
            if(xm!=1 and -ym!=1 and zm!=1):
                xm/=3
                ym/=3
                zm/=3
        elif((xm%5==0 and -ym%5==0 and zm%5==0)):
            v1=Matrix([xm/5,-ym/5,zm/5])
            if(xm!=1 and -ym!=1 and zm!=1):
                xm/=5
                ym/=5
                zm/=5
        elif((xm%7==0 and -ym%7==0 and zm%7==0)):
            v1=Matrix([xm/7,-ym/7,zm/7])
            if(xm!=1 and -ym!=1 and zm!=1):
                xm/=7
                ym/=7
                zm/=7
        else:
            v1=Matrix([xm,-ym,zm])
        print("\n\nEigen vector when {0}={1} : ".format(x,K[i]))
        display(v1)
    elif(eq2==eq3==eq4):
        print("\nBy Substitution Method.....\n")
        print('\nLet X2=0\n')
        ys=0
        xm=eq2.subs(b,ys)
        display(xm)
        display(solve(xm))
        xs=int(input('X1 substitution : '))
        display(xm.subs(a,xs))
```

```
        display(solve(xm.subs(a,xs)))
        zs=solve(xm.subs(a,xs))
        print('C=',*zs)
        v2=Matrix([xs,ys,*zs])
        print("\nEigen vectors when {0}={1} and when X2=0 \n".format(x,K[0]))
        display(v2)
        print('\n\nLet C=0\n')
        zs=0
        xm=eq2.subs(c,zs)
        display(xm)
        display(solve(xm))
        xs=int(input('X1 substitution : '))
        display(xm.subs(a,xs))
        display(solve(xm.subs(a,xs)))
        ys=solve(xm.subs(a,xs))
        print('B=',*ys)
        v3=Matrix([xs,*ys,zs])
        display(v3)
        print("\nEigen vectors when {0}={1} and when X3=0 \n".format(x,K[0]))
        display(v3)
end1=time()
print("Execution Time")
display(end1-start1)
print("Using Built in function")
start=time()
ev=amat.eigenvals()
j=amat.eigenvects()
print("Eigen value")
display(*ev)
print("Eigen Vector")
display(j)
end=time()
print("Execution time ")
display(end-start)
```

```
Enter the matrix coefficients : 1 2 3 3 -2 1 1 -6 -5
Input matrix
```

```
c:\users\elcot\appdata\local\programs\python\python39\lib\site-packages\IPython\lib\latextools.py:126: MatplotlibDeprecat
ionWarning:
The to_png function was deprecated in Matplotlib 3.4 and will be removed two minor releases later. Use mathtext.math_to_i
mage instead.
  mt.to_png(f, s, fontsize=12, dpi=dpi, color=color)
c:\users\elcot\appdata\local\programs\python\python39\lib\site-packages\IPython\lib\latextools.py:126: MatplotlibDeprecat
ionWarning:
The to_rgba function was deprecated in Matplotlib 3.4 and will be removed two minor releases later. Use mathtext.math_to_
image instead.
```

```
  mt.to_png(f, s, fontsize=12, dpi=dpi, color=color)
c:\users\elcot\appdata\local\programs\python\python39\lib\site-packages\IPython\lib\latextools.py:126: MatplotlibDeprecat
ionWarning:
The to_mask function was deprecated in Matplotlib 3.4 and will be removed two minor releases later. Use mathtext.math_to_
image instead.
  mt.to_png(f, s, fontsize=12, dpi=dpi, color=color)
c:\users\elcot\appdata\local\programs\python\python39\lib\site-packages\IPython\lib\latextools.py:126: MatplotlibDeprecat
ionWarning:
The MathtextBackendBitmap class was deprecated in Matplotlib 3.4 and will be removed two minor releases later. Use mathte
xt.math_to_image instead.
  mt.to_png(f, s, fontsize=12, dpi=dpi, color=color)
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & -2 & 1 \\ 1 & -6 & -5 \end{bmatrix}$$

```
'S1=1+-2+-5'
's1=-6'
's2=(((-2*-5)-(-6*1))+((1*-5)-(1*3))+((1*-2)-(3*2))'
's2=(16+-8+-8)'
's3=det(amat)'
's3=0'
's1=-6 s2=0 s3=0'
```

```
c:\users\elcot\appdata\local\programs\python\python39\lib\site-packages\sympy\core\relational.py:495: SymPyDeprecationWar
ning:

Eq(expr) with rhs default to 0 has been deprecated since SymPy 1.5.
Use Eq(expr, 0) instead. See
https://github.com/sympy/sympy/issues/16587 for more info.

  SymPyDeprecationWarning(
```

$$\lambda^3 + 6\lambda^2 = 0$$

$$\lambda^2 (\lambda + 6) = 0$$

```
Eigen values are :  [-6, 0]
>>>>>>>>>>>>>>>>>> To find eigen vector <<<<<<<<<<<<<<<
(A-λI)X=0
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & -2 & 1 \\ 1 & -6 & -5 \end{bmatrix}$$

—

$$\begin{bmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{bmatrix}$$

$$=$$

$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$->$$

$$\begin{bmatrix} 1-\lambda & 2 & 3 \\ 3 & -\lambda-2 & 1 \\ 1 & -6 & -\lambda-5 \end{bmatrix}$$

$$*$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$=$$

$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

```
If λ=-6
(A--6I)X=0
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & -2 & 1 \\ 1 & -6 & -5 \end{bmatrix}$$

$$-$$

$$\begin{bmatrix} -6 & 0 & 0 \\ 0 & -6 & 0 \\ 0 & 0 & -6 \end{bmatrix}$$

$$=$$

$$\begin{bmatrix} 7 & 2 & 3 \\ 3 & 4 & 1 \\ 1 & -6 & 1 \end{bmatrix}$$

now

$$\begin{bmatrix} 7 & 2 & 3 \\ 3 & 4 & 1 \\ 1 & -6 & 1 \end{bmatrix}$$

$$*$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$=$$

$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$->$$

$$7x_1 + 2x_2 + 3x_3 = 0$$

$$3x_1 + 4x_2 + x_3 = 0$$

$$x_1 - 6x_2 + x_3 = 0$$

By Gaussian Elimination Method.....

$$\left\{x_1 : -\frac{5x_3}{11},\ x_2 : \frac{x_3}{11}\right\}$$

By Using Cofactor Method

(x1/-10)=(-(x2/-2))=(x3/22)

Eigen vector when λ=-6 :

$$\begin{bmatrix} -5 \\ 1 \\ 11 \end{bmatrix}$$

```
If λ=0
(A-0I)X=0
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & -2 & 1 \\ 1 & -6 & -5 \end{bmatrix}$$

$$-$$

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$=$$

$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & -2 & 1 \\ 1 & -6 & -5 \end{bmatrix}$$

```
now
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & -2 & 1 \\ 1 & -6 & -5 \end{bmatrix}$$

$$*$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$=$$

$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$->$$

$$x_1 + 2x_2 + 3x_3 = 0$$

$$3x_1 - 2x_2 + x_3 = 0$$

$$x_1 - 6x_2 - 5x_3 = 0$$

```
By Gaussian Elimination Method.....
```

$$\{x_1 : -x_3,\ x_2 : -x_3\}$$

```
By Using Cofactor Method

(x1/8)=(-(x2/-8))=(x3/-8)


Eigen vector when λ=0 :
```

$$\begin{bmatrix} 4 \\ 4 \\ -4 \end{bmatrix}$$

```
Execution Time
```

$$26.4237039089203$$

```
Using Built in function
Eigen value
```

$$-6$$

$$0$$

```
Eigen Vector
```

$$\left[\left(-6,\ 1,\ \left[\begin{bmatrix} -\frac{5}{11} \\ \frac{1}{11} \\ 1 \end{bmatrix} \right] \right),\ \left(0,\ 2,\ \left[\begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix} \right] \right) \right]$$

```
Execution time
```

$$0.406102895736694$$

In [4]:

```
y=int(input("How many sums going to check "))
for i in range(0,y):
    dim=int(input("Enter the dimension : "))
    if(dim==2):
        a11,a12,a21,a22=map(int,input("Enter the matrix coefficients : ").split())
        r1,r2=[a11,a12],[a21,a22]
```

```
        amat=sp.Matrix((r1,r2))
        print("Input matrix")
        display(amat)
        s1=a11+a22
        display('S1={0}+{1}'.format(a11,a22))
        display('s1={0}'.format(s1))
        display('s2=({0}*{1})-({2}*{3})'.format(a11,a22,a12,a21))
        s2=(a11*a22)-(a12*a21)
        display('s2={0}'.format(s2))
        x=sp.symbols('λ')
        eq1=sp.Function("eq1")
        eq1=Eq((x**2 +(s1)*x- s2))
        display(eq1)
        k=sp.factor(eq1)
        display(k)
        K=solve(k)
        print("Eigen values are : ",*K)
    if(dim==3):
        a11,a12,a13,a21,a22,a23,a31,a32,a33=map(int,input("Enter the matrix coefficients : ").split())
        r1,r2,r3=[a11,a12,a13],[a21,a22,a23],[a31,a32,a33]
        amat=sp.Matrix((r1,r2,r3))
        print("Input matrix")
        display(amat)


        s1=a11+a22+a33
        display('S1={0}+{1}+{2}'.format(a11,a22,a33))
        display('s1={0}'.format(s1))
        s2=(((a22*a33)-(a32*a23))+((a11*a33)-(a31*a13))+((a11*a22)-(a21*a12)))
        display('s2=((({0}*{1})-({2}*{3}))+(({4}*{5})-({6}*{7}))+(({8}*{9})-({10}*{11}))'.format(a22,a33,a32,a23,a11,a33,
        display('s2=({0}+{1}+{2})'.format(((a22*a33)-(a32*a23)),((a11*a33)-(a31*a13)),((a11*a22)-(a21*a12))))
        s3=det(amat)
        display('s3=det(amat)')
        display("s3={0}".format(s3))
        display('s1={0} s2={1} s3={2}'.format(s1,s2,s3))
        x=sp.symbols('λ')
        eq1=sp.Function("eq1")
        eq1=Eq((x**3 - (s1)*x**2 +(s2)*x- s3))
        display(eq1)
        k=sp.factor(eq1)
        display(k)
        K=solve(k)
        print("Eigen values are : ",K)
    P,D=amat.diagonalize()
    print("_____Diagonalization_____")
    display(P,D,P**-1)
```

```
How many sums going to check 2
Enter the dimension : 2
Enter the matrix coefficients : 2 1 0 3
Input matrix
```

$$\begin{bmatrix} 2 & 1 \\ 0 & 3 \end{bmatrix}$$

```
'S1=2+3'
's1=5'
's2=(2*3)-(1*0)'
's2=6'
```

$$\lambda^2 + 5\lambda - 6 = 0$$

$$(\lambda - 1)(\lambda + 6) = 0$$

```
Eigen values are :  -6 1
_____Diagonalization_____
```

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}$$

```
Enter the dimension : 3
Enter the matrix coefficients : 2 2 1 1 3 1 1 2 2
Input matrix
```

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

```
'S1=2+3+2'
's1=7'
's2=(((3*2)-(2*1))+((2*2)-(1*1))+((2*3)-(1*2))'
's2=(4+3+4)'
's3=det(amat)'
's3=5'
's1=7 s2=11 s3=5'
```

$$\lambda^3 - 7\lambda^2 + 11\lambda - 5 = 0$$

$$(\lambda - 5)(\lambda - 1)^2 = 0$$

```
Eigen values are :  [1, 5]
```

_____Diagonalization_____

$$\begin{bmatrix} -2 & -1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 5 \end{bmatrix}$$

$$\begin{bmatrix} -\frac{1}{4} & \frac{1}{2} & -\frac{1}{4} \\ -\frac{1}{4} & -\frac{1}{2} & \frac{3}{4} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{bmatrix}$$

In [ ]: