pj03-Dec17

December 31, 2018

In [2]:
```
import numpy as np
import timeit
import matplotlib.pyplot as plt
```

In [3]:
```
def Jacobi(A):
    n     = A.shape[0]            # matrix size #columns = #lines
    maxit = 100                   # maximum number of iterations
    eps   = 1.0e-15               # accuracy goal
    pi    = np.pi
    info  = 0                     # return flag
    ev    = np.zeros(n,float)     # initialize eigenvalues
    U     = np.zeros((n,n),float) # initialize eigenvector
    for i in range(0,n): U[i,i] = 1.0

    for t in range(0,maxit):
        s = 0;    # compute sum of off-diagonal elements in A(i,j)
        for i in range(0,n): s = s + np.sum(np.abs(A[i,(i+1):n]))
        if (s < eps): # diagonal form reached
            info = t
            for i in range(0,n):ev[i] = A[i,i]
            break
        else:
            limit = s/(n*(n-1)/2.0)       # average value of off-diagonal elements
            for i in range(0,n-1):       # loop over lines of matrix
                for j in range(i+1,n):  # loop over columns of matrix
                    if (np.abs(A[i,j]) > limit):      # determine (ij) such that |A(i,j)
                                                        # value of off-diagonal element
                        denom = A[i,i] - A[j,j]       # denominator of Eq. (3.61)
                        if (np.abs(denom) < eps): phi = pi/4         # Eq. (3.62)
                        else: phi = 0.5*np.arctan(2.0*A[i,j]/denom)  # Eq. (3.61)
                        si = np.sin(phi)
                        co = np.cos(phi)
                        for k in range(i+1,j):
                            store  = A[i,k]
                            A[i,k] = A[i,k]*co + A[k,j]*si  # Eq. (3.56)
                            A[k,j] = A[k,j]*co - store *si  # Eq. (3.57)
                        for k in range(j+1,n):
```

```
                        store  = A[i,k]
                        A[i,k] = A[i,k]*co + A[j,k]*si  # Eq. (3.56)
                        A[j,k] = A[j,k]*co - store *si  # Eq. (3.57)
                    for k in range(0,i):
                        store  = A[k,i]
                        A[k,i] = A[k,i]*co + A[k,j]*si
                        A[k,j] = A[k,j]*co - store *si
                    store = A[i,i]
                    A[i,i] = A[i,i]*co*co + 2.0*A[i,j]*co*si +A[j,j]*si*si  # Eq. (3
                    A[j,j] = A[j,j]*co*co - 2.0*A[i,j]*co*si +store *si*si  # Eq. (3
                    A[i,j] = 0.0                                            # Eq. (3
                    for k in range(0,n):
                        store  = U[k,j]
                        U[k,j] = U[k,j]*co - U[k,i]*si  # Eq. (3.66)
                        U[k,i] = U[k,i]*co + store *si  # Eq. (3.67)
            info = -t # in case no convergence is reached set info to a negative value "-t"
        return ev,U,t
```

```
In [6]: A = np.array([[5 , -1, 0, 0, 0], [-1, 4.5, 0.2, 0, 0], [0, 0.2, 1, -0.4, 0], [0, 0, -0.4
        print(A.shape)
```

```
(5, 5)
```

```
In [8]: np.set_printoptions(precision=5)
        ev,U = np.linalg.eig(A)
        print ("RESULT FROM numpy.linalg.eig")
        print ("Eigenvalues = ", ev)
        print ("Eigenvectors = ", U)
```

```
RESULT FROM numpy.linalg.eig
Eigenvalues =  [5.784    0.8903   2.07071 3.72756 4.02743]
Eigenvectors =  [[ 0.7867    0.01393  0.00792 -0.6161   -0.03568]
 [-0.61677  0.05725  0.02321 -0.78395 -0.0347 ]
 [-0.02615 -0.9636   -0.24232 -0.0527   -0.09642]
 [ 0.00431 -0.23564  0.66023 -0.03259  0.71238]
 [ 0.00155  0.11169 -0.71047 -0.04479  0.69336]]
```

```
In [9]: ev,U,t = Jacobi(A)
        print ("JACOBI METHOD: Number of rotations = ", t)
        print ("Eigenvalues = ", ev)
        print ("Eigenvectors = ", U)
```

```
JACOBI METHOD: Number of rotations =  12
Eigenvalues =  [5.784    3.72756 0.8903   4.02743 2.07071]
Eigenvectors =  [[ 0.7867    0.6161   -0.01393 -0.03568 -0.00792]
 [-0.61677  0.78395 -0.05725 -0.0347   -0.02321]
 [-0.02615  0.0527    0.9636   -0.09642  0.24232]
```

```
[ 0.00431  0.03259  0.23564  0.71238 -0.66023]
[ 0.00155  0.04479 -0.11169  0.69336  0.71047]]
```