Problem 13

A)

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

In [ ]:
```
def create_L_matrix(n):
    L = np.identity(n=n)
    return L

def create_U_matrix(n):
    U = np.zeros((n,n))
    return U

def hilbert(n):
    x = np.arange(1, n+1) + np.arange(0, n)[:, np.newaxis]
    return 1.0/x

def lu_decomp(A):
    n = len(A)

    L = create_L_matrix(n)
    U = create_U_matrix(n)


    for k in range(0,n):
        # Ukk
        temp_sum_ukk = 0
        for s in range(0,k):
            temp_sum_ukk += L[k,s]*U[s,k]

        U[k,k] = A[k,k] - temp_sum_ukk

        #Ukj
        for j in range(k+1,n):
            temp_sum_ukj = 0
            for s in range(0,k):
                temp_sum_ukj += L[k,s]*U[s,j]

            U[k,j] = A[k,j] - temp_sum_ukj

        #Lik
        for i in range(k+1,n):
            temp_sum_lik = 0
            for s in range(0,k):
                temp_sum_lik += L[i,s]*U[s,k]

            L[i,k] = (1/U[k,k])*(A[i,k] - temp_sum_lik)

    return L,U
```

In [ ]:
```
A = hilbert(4)

results = lu_decomp(A)
```

```
print("L: ")
print(results[0])
print()
print()
print('U:')
print(results[1])
np.set_printoptions(precision=55)
```

```
L: 
[[1.                 0.                 0.
  0.                ]
 [0.5                1.                 0.
  0.                ]
 [0.3333333333333333 1.0000000000000004 1.
  0.                ]
 [0.25               0.9000000000000004 1.4999999999999951
  1.                ]]


U:
[[1.000000000000000e+00 5.000000000000000e-01 3.333333333333333e-01
  2.500000000000000e-01]
 [0.000000000000000e+00 8.333333333333331e-02 8.333333333333334e-02
  7.500000000000001e-02]
 [0.000000000000000e+00 0.000000000000000e+00 5.555555555555536e-03
  8.333333333333276e-03]
 [0.000000000000000e+00 0.000000000000000e+00 0.000000000000000e+00
  3.571428571429447e-04]]
```

B)

In [ ]:

```
from audioop import reverse


def forwardSub(L,b):
    L_inv = np.linalg.inv(L)
    y = np.matmul(L_inv,b)
    return y

def backwardsSub(U,y):
    #x = np.zeros((len(U),1))
    #print(x)
    #for i in reversed(range(0,len(U))):
        #temp_sum = 0
        #for j in range(i+1,len(U)):
            #print('i: '+str(i)+'  j: '+str(j))
            #temp_sum += x[j]/U[i,j]
        #x[i][0] = (y[i]-temp_sum)/U[i,i]



    U_inv = np.linalg.inv(U)
    x = np.matmul(U_inv,y)
    return x
```

In [ ]:

```
A = hilbert(4)
b = np.matrix([[1],[2],[3],[4]])

results = lu_decomp(A)
```

```python
y = forwardSub(results[0],b)
print('Forward substitution results   y: ')
print(y)
print()

x = backwardsSub(results[1],y)
print('Backwards substitution results  x: ')
print(x)
```

```
Forward substitution results   y:
[[1.                ]
 [1.5               ]
 [1.1666666666666666 ]
 [0.6500000000000048]]

Backwards substitution results  x:
[[  -63.999999999977085]
 [  899.9999999997318  ]
 [-2519.999999999342   ]
 [ 1819.9999999995673  ]]
```

C)

In [ ]:

```python
A = hilbert(4)

checking = np.matmul(A,x)
print('Ab = x where x equals:')
print(checking)
```

```
Ab = x where x equals:
[[1.0000000000001137]
 [2.                ]
 [3.                ]
 [3.999999999999943 ]]
```

As we can see the outcome of the matrix multiplication of matrix A with the solution vector x gives us with in a reasonable range for the true value of b.