Pregunta 7

April 25, 2019

Autor:
Lazaro Camasca Edson
Analisis Numerico
Universidad Nacional de Ingenieria

0.0.1 Programe el método de eliminación gaussiana

Sin intercambio de filas y resuelva el sistema $ Hx = b $, donde $H(i;j) = 1/(i+j1)$ y $b(j) = 1$

```
In [1]: from toolNick import *
        import numpy as np

toolNick se ha importado correctamente.
```

```
In [2]: def elimGauss(a, b, p = False, v = False):
            #v nos muestra el procedimiento detallado de la eliminacion
            #p se utiliza si requiere pivotacion total al inicio

            A_b = np.c_[a, b] # Matriz aumentada
            (fil, col) = A_b.shape #Guarda el #filas y #columnas
            line = "============================================="
            print("Matriz aumentada [H|b] al inicio: \n".format(A_b,line))

            if p: A_b = pivoteo_Total(A_b, v)

            for i in range(fil): # se condiera dim(a)
                #El pivoteo parcio no el solo cuando se tiene 0 en la diagonal
                #Para realizar el pivoteo parcial eliminar la condicion A_b[i,i]==0

                #Si se quiere un pivoteo parcial quitar esta condicion
                if A_b[i, i] == 0 :
                    P = pivoteo(A_b, i)
                    if v : print("P_{}:\n{}".format(i+1,P))
                    A_b = np.matmul(P, A_b)

                #Obtenemos el L_i
                L_i = get_L(A_b, i, v)
```

```
            A_b = np.matmul(L_i, A_b)

            #mostrar L(i) * (A|b)
            if v : print("L_{} * [H|b]:\n{}\n{}".format(i+1,A_b,line))

        #U sera todo menos la ultima columna
        U = A_b[:, :col - 1]
        #b_sol sera solo la ultima columna
        b_sol = A_b[:, col - 1]
        return resolverMTriangularSup(U, b_sol)
```

0.0.2 Ejemplo

```
In [3]: n = 3
        H = np.fromfunction(lambda i,j: 1/((i+1)+(j+1)-1),(n,n))
        print("H_",H)
        b = np.array([1, 1 ,1]); b.reshape(3,1)
        print("b_: \n", b)
```

```
H_ [[1.         0.5        0.33333333]
 [0.5        0.33333333 0.25      ]
 [0.33333333 0.25       0.2       ]]
b_:
 [1 1 1]
```

Como los indices de python empiezan en 0 cambiamos la funcion $H(i;j) = 1/(i+j1)$ por $H(i;j) = 1/[(i+1)+(j+1)1]$

```
In [4]: elimGauss(H,b,v=True)
```

```
Matriz aumentada [H|b] al inicio:

alpha_1:
[[0.        ]
 [0.5       ]
 [0.33333333]]
alpha_1 * e_1:
[[0.         0.         0.        ]
 [0.5        0.         0.        ]
 [0.33333333 0.         0.        ]]
L_1:
[[ 1.          0.          0.        ]
 [-0.5         1.          0.        ]
 [-0.33333333  0.          1.        ]]
L_1 * [H|b]:
[[1.         0.5        0.33333333 1.        ]
 [0.         0.08333333 0.08333333 0.5       ]
 [0.         0.08333333 0.08888889 0.66666667]]
```

```
==========================================
alpha_2:
[[0.]
 [0.]
 [1.]]
alpha_2 * e_2:
[[0. 0. 0.]
 [0. 0. 0.]
 [0. 1. 0.]]
L_2:
[[ 1.  0.  0.]
 [ 0.  1.  0.]
 [ 0. -1.  1.]]
L_2 * [H|b]:
[[ 1.00000000e+00  5.00000000e-01  3.33333333e-01  1.00000000e+00]
 [ 0.00000000e+00  8.33333333e-02  8.33333333e-02  5.00000000e-01]
 [ 0.00000000e+00 -1.38777878e-17  5.55555556e-03  1.66666667e-01]]
==========================================
alpha_3:
[[0.]
 [0.]
 [0.]]
alpha_3 * e_3:
[[0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]]
L_3:
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
L_3 * [H|b]:
[[ 1.00000000e+00  5.00000000e-01  3.33333333e-01  1.00000000e+00]
 [ 0.00000000e+00  8.33333333e-02  8.33333333e-02  5.00000000e-01]
 [ 0.00000000e+00 -1.38777878e-17  5.55555556e-03  1.66666667e-01]]
==========================================
```

Out[4]: array([3., -24., 30.])