Pregunta 8

April 25, 2019

Autor:
Lazaro Camasca Edson
Analisis Numerico
Universidad Nacional de Ingenieria
Programe la eliminación de Gauss Jordan Muestre una base para el espacio columna de cualquier matriz A, Por ejemplo la matriz del problema 10.
A

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

toolNick se ha importado correctamente.
```

```
In [2]: def gauss_Jordan(a, b, p = False, v = False, i = False):
            # v nos muestra el procedimiento detallado de la eliminacion
            # p se utiliza si requiere pivotacion total al inicio
            # i si se quiere hallar la inversa de la matriz "a"
            A_b = np.c_[a, b] # Matriz aumentada
            (fil, col) = A_b.shape #Guarda el #filas y #columnas
            line = "============================================="

            # Creamos una lista para almacenar las matrices T
            T_list = []
            print("Matriz aumentada [A|b] al inicio:\n{}\n{}".format(A_b,line))

            if p: A_b = pivoteo_Total(A_b, v)

            for i in range(fil): # se condiera dim(a)
                if A_b[i, i] == 0 :
                    P = pivoteo(A_b, i)
                    if v : print("P_\n{}:".format(P))
                    A_b = P @ A_b

                # Obtenemos el T_i
                T_i = get_T(A_b, i, v)
                T_list.append(T_i) # Agregamos para hallar la inversa
```

```
                A_b = T_i @ A_b

                # Mostrar T(i) * (A|b)
                if v : print("T_{} * [A|b]:\n{}\n{}".format(i+1,A_b,line))

            if not(i):
                A_inv = np.identity(fil)
                for i in range(fil):
                    A_inv = T_list[i] @ A_inv
                print("La matriz inversa de A es:\n{}\n{}".format(A_inv,line))

            print("Matriz aumentada [A|b] al final: \n", A_b)
            x = A_b[:, col - 1]
            return x
```

Eliminacion Gauss Jordan

```
In [3]: A = np.array([[2, 1, 1, 0],
                      [4, 3, 3, 1],
                      [8, 7, 9, 5],
                      [6, 7, 9, 8]])

        print("Matriz de coeficientes A:\n", A); (fil,col) = A.shape
        b = np.array([1, 1, 1, 1]); b.reshape(fil,1)
        print("Matriz b: \n", b)
```

```
Matriz de coeficientes A:
 [[2 1 1 0]
 [4 3 3 1]
 [8 7 9 5]
 [6 7 9 8]]
Matriz b:
 [1 1 1 1]
```

```
In [4]: gauss_Jordan(A, b, v=True, i = False)
```

```
Matriz aumentada [A|b] al inicio:
[[2 1 1 0 1]
 [4 3 3 1 1]
 [8 7 9 5 1]
 [6 7 9 8 1]]
==========================================
alpha_1:
[[0.5]
 [2. ]
 [4. ]
 [3. ]]
alpha_1 * e_1:
```

```
[[0.5 0.  0.  0. ]
 [2.  0.  0.  0. ]
 [4.  0.  0.  0. ]
 [3.  0.  0.  0. ]]
T_1:
[[ 0.5  0.   0.   0. ]
 [-2.   1.   0.   0. ]
 [-4.   0.   1.   0. ]
 [-3.   0.   0.   1. ]]
T_1 * [A|b]:
[[ 1.   0.5  0.5  0.   0.5]
 [ 0.   1.   1.   1.  -1. ]
 [ 0.   3.   5.   5.  -3. ]
 [ 0.   4.   6.   8.  -2. ]]
============================================
alpha_2:
[[0.5]
 [0. ]
 [3. ]
 [4. ]]
alpha_2 * e_2:
[[0.  0.5 0.  0. ]
 [0.  0.  0.  0. ]
 [0.  3.  0.  0. ]
 [0.  4.  0.  0. ]]
T_2:
[[ 1.  -0.5  0.   0. ]
 [ 0.   1.   0.   0. ]
 [ 0.  -3.   1.   0. ]
 [ 0.  -4.   0.   1. ]]
T_2 * [A|b]:
[[ 1.   0.   0.  -0.5  1. ]
 [ 0.   1.   1.   1.  -1. ]
 [ 0.   0.   2.   2.   0. ]
 [ 0.   0.   2.   4.   2. ]]
============================================
alpha_3:
[[0. ]
 [0.5]
 [0.5]
 [1. ]]
alpha_3 * e_3:
[[0.  0.  0.  0. ]
 [0.  0.  0.5 0. ]
 [0.  0.  0.5 0. ]
 [0.  0.  1.  0. ]]
T_3:
[[ 1.   0.   0.   0. ]
```

```
 [ 0.   1.  -0.5  0. ]
 [ 0.   0.   0.5  0. ]
 [ 0.   0.  -1.   1. ]]
T_3 * [A|b]:
[[ 1.   0.   0.  -0.5  1. ]
 [ 0.   1.   0.   0.  -1. ]
 [ 0.   0.   1.   1.   0. ]
 [ 0.   0.   0.   2.   2. ]]
============================================
alpha_4:
[[-0.25]
 [ 0.  ]
 [ 0.5 ]
 [ 0.5 ]]
alpha_4 * e_4:
[[-0.   -0.   -0.   -0.25]
 [ 0.    0.    0.    0.  ]
 [ 0.    0.    0.    0.5 ]
 [ 0.    0.    0.    0.5 ]]
T_4:
[[ 1.    0.    0.    0.25]
 [ 0.    1.    0.    0.  ]
 [ 0.    0.    1.   -0.5 ]
 [ 0.    0.    0.    0.5 ]]
T_4 * [A|b]:
[[ 1.   0.   0.   0.   1.5]
 [ 0.   1.   0.   0.  -1. ]
 [ 0.   0.   1.   0.  -1. ]
 [ 0.   0.   0.   1.   1. ]]
============================================
Matriz aumentada [A|b] al final:
 [[ 1.   0.   0.   0.   1.5]
 [ 0.   1.   0.   0.  -1. ]
 [ 0.   0.   1.   0.  -1. ]
 [ 0.   0.   0.   1.   1. ]]


Out[4]: array([ 1.5, -1. , -1. ,  1. ])
```