Pregunta 10

April 25, 2019

Autor:
Lazaro Camasca Edson
Analisis Numerico
Universidad Nacional de Ingenieria

0.0.1 Programe la factorización LU de Dolittle(L,U=Doolittle(A)) y aplíquelo con

```
In [2]: from toolNick import *
        import numpy as np
```

Algoritmo de factorizacion L1U de Dolittle

```
In [3]: def fac_L1U(A, b, p = False, v = False):
            """recibe una matriz  cuadrada a de coeficientes y la matriz b
            retorna x vector resultado
            para realizar un pivoteo total pasar p = 'total' como parametro
            para una solución detallada pasar v = True como parametro."""

            A_b = np.c_[A, b] # Matriz aumentada
            fil, col = A_b.shape #Guarda el #filas y #columnas
            #assert fil == col, "La matriz de coeficienes no es cuadrada."
            line = "==================================="
            print("Matriz aumentada [A|b] al inicio: \n", A_b)

            if p:
                A_b = pivoteo_Total(A_b, v)

            # Separamos A y b de [A|b]
            A = A_b[:, :col - 1] #A sera todo menos la ultima columna
            b = A_b[:, col - 1] #b sera solo la ultima columna

            #Creamos la matrices L Y U1_ identidad
            U = np.zeros((fil, col))
            L = np.identity(fil)

            for k in range(fil):
                # Hallando U
```

```
            for j in range(k, fil):
                U[k, j] = A[k,j] - L[k, :k]@U[:k, j]

            #Hallando L
            for i in range(k + 1, fil):
                 L[i, k] = (A[i, k] - L[i, :k]@U[:k, k]) /U[k, k]

        if v : print("{}\nL: \n{}\n{}\nU:\n{}\n".format(line,L,line,U))

        y = resolverMTriangularInf(L, b)
        x = resolverMTriangularSup(U, y)

        return x
```

0.0.2 Definiendo A y b

```
In [4]: 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, 8, 30, 41]); 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  8 30 41]
```

Resolviendo el sistema

```
In [5]: fac_L1U(A,b,v=True)
```

```
Matriz aumentada [A|b] al inicio:
 [[ 2  1  1  0  1]
 [ 4  3  3  1  8]
 [ 8  7  9  5 30]
 [ 6  7  9  8 41]]
====================================
L:
[[1. 0. 0. 0.]
 [2. 1. 0. 0.]
 [4. 3. 1. 0.]
```

```
 [3. 4. 1. 1.]]
==================================
U:
[[2. 1. 1. 0. 0.]
 [0. 1. 1. 1. 0.]
 [0. 0. 2. 2. 0.]
 [0. 0. 0. 2. 0.]]
```

Out[5]: array([-1., 2., 1., 3., 0.])