Pregunta 15

April 25, 2019

Autor:
Lazaro Camasca Edson
Analisis Numerico
Universidad Nacional de Ingenieria #### Programe un procedimiento que realice la factorización Cholesky de una matriz A. Aplíquelo a la matriz

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

```
toolNick se ha importado correctamente.
```

0.0.1 Factorización Cholesky

```
In [9]: def cholesky(a, b, p = False, v = True):

            fil, col = a.shape
            assert fil == col, "La matriz de coeficientes no es cuadrada."
            assert np.all(np.linalg.eigvals(a) > 0), "Matriz de coeficientes no definida \
            positiva"
            assert esSimetrica(A) == True, "La matriz no es simetrica, los resultados no seran

            line = "================================="
            G = np.zeros((fil, col))

            for k in range(fil):
                if a[k, k] > 0 :
                    G[k, k] = np.sqrt(a[k, k] - G[k, :k].dot(G[k, :k]))
                for  i in range(k + 1, fil):
                    G[i, k] = (a[i, k] - G[i, :k].dot(G[k, :k])) / G[k, k]
                if v == True:
                   print("{}\nG_{}\n{}\n".format(line,k+1,G))

            # Ax = b  &  A = G * G_t
            print("{}\nG_\n{}\n".format(line,G))
            # Ax = b -> G*G_t x = b
            # Gy = b  hallamos y  ->  G_t * x = y
            # G_t x = y  hallamos x
```

```
            y = resolverMTriangularInf(G, b)
            x = resolverMTriangularSup(G.T, y)

            return x
```

Definimos A y b

```
In [10]: A = np.array([[4, -2, 2],
                       [-2, 2, -4],
                       [2, -4, 11]])

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

```
Matriz de coeficientes A:
 [[ 4 -2  2]
 [-2  2 -4]
 [ 2 -4 11]]
Matriz b:
 [1 1 1]
```

Resolvemos el sistema

```
In [11]: cholesky(A,b,v=True)
```

```
==================================
G_1
[[ 2.  0.  0.]
 [-1.  0.  0.]
 [ 1.  0.  0.]]

==================================
G_2
[[ 2.  0.  0.]
 [-1.  1.  0.]
 [ 1. -3.  0.]]

==================================
G_3
[[ 2.  0.  0.]
 [-1.  1.  0.]
 [ 1. -3.  1.]]

==================================
G_
```

```
[[ 2.  0.  0.]
 [-1.  1.  0.]
 [ 1. -3.  1.]]
```

Out[11]: array([6. , 16.5, 5.])