Universidad Autónoma de Querétaro

Facultad de Ingeniería

Factorización tipo Cholesky

Análisis numérico

Autor:
David Gómez Torres

15 Septiembre del 2021

ÍNDICE GENERAL

CAPÍTULO 1

INTRODUCCIÓN

Ciertas matrices tienen una estructura particular que puede aprovecharse para desarrollar esquemas de solución eficientes. En esta sección se dedica al estudio de dos de estos sistemas: matrices **a bandas** y **simétricas**. Se describen métodos de eliminación eficiente para ambas.

1.1. Matrices especiales

Una **matriz a bandas** es una matriz cuadrada en la que todos sus elementos son cero, con excepción de una banda centrada sobre la diagonal principal.

Las dimensiones de un sistema a bandas se cuantifica mediante dos parámetros: el ancho de banda (**BW**) y el ancho de media banda **HBW**.
Estos dos valores se relacionan mediante $BW = 2HBW + 1$. En general, un sistema a bandas es aquel para el cual $a_{ij} = 0$ si $|i–j| >$ HBW. Ver figura (1.1).

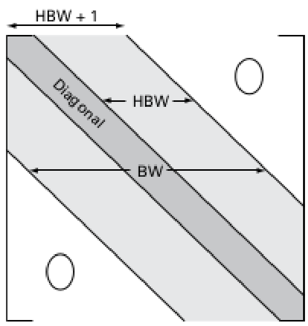

Figura 1.1: Ejemplo de un matriz a bandas

1.1.1. Sistemas tridiagonales

Un sistema tridiagonal (con un ancho de banda 3) se expresa en forma general de la siguiente manera:

$$\begin{bmatrix} f_1 & g_1 & & & & & & \\ e_2 & f_2 & g_2 & & & & & \\ & e_3 & f_3 & g_3 & & & & \\ & & \cdot & \cdot & \cdot & & & \\ & & & \cdot & \cdot & \cdot & & \\ & & & & \cdot & \cdot & \cdot & \\ & & & & & e_{n-1} & f_{n-1} & g_{n-1} \\ & & & & & & e_n & f_n \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ \cdot \\ \cdot \\ \cdot \\ x_{n-1} \\ x_n \end{Bmatrix} = \begin{Bmatrix} r_1 \\ r_2 \\ r_3 \\ \cdot \\ \cdot \\ \cdot \\ r_{n-1} \\ r_n \end{Bmatrix} \tag{1.1}$$

Existe un algoritmo llamado **algoritmo de Thomas**, para resolver la ecuación (1.1). Como una descomposición LU convencional, el algoritmo consiste de tres pasos: *descomposición*, *sustitución hacia adelante* y *sustitución hacia atrás*.

Así, las ventajas de la descomposición LU, como la evaluación de vectores múltiples del lado derecho y el cálculo de la matriz inversa, se obtienen mediante una apropiada aplicación de este algoritmo.

1.1.2. Descomposición de Cholesky

Una **matriz simétrica** es aquella donde $a_{ij} = a_{ji}$ para toda i y j.
En otras palabras, [A] = $[A]^T$

Uno de los métodos más populares usa la **descomposición de Cholesky**. Este algoritmo se basa en el hecho de que una matriz simétrica se descompone así:

$$[A] = [L][L]^T \tag{1.2}$$

Los términos de la ecuación (1.2) se desarrollan al multiplicar e igualar entre sí ambos lados. El resultado se expresa en forma simple mediante relaciones de recurrencia. Para el renglón k-ésimo,

$$l_{ki} = \frac{a_{ki} - \sum_{j=1}^{i-1} l_{ij} l_{kj}}{l_{ii}} \text{para i = 1, 2,.., k − 1} \tag{1.3}$$

y

$$l_{kk} = \sqrt{a_{kk} - \sum_{j=1}^{k-1} l_{kj}^2} \tag{1.4}$$

Debe observar que el algoritmo del método da un error de ejecución si en la evaluación de a_{kk} se obtiene la raíz cuadrada de un número negativo. Sin embargo, cuando la matriz es definida positiva, esto nunca ocurrirá.

CAPÍTULO 2

METODOLOGÍA

2.1. Sistema tridiagonal

Problema 11.1

a)Realice los mismos cálculos para el sistema tridiagonal.

$$\begin{bmatrix} 0.8 & -0.4 & \\ -0.4 & 0.8 & -0.4 \\ & -0.4 & 0.8 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 41 \\ 25 \\ 105 \end{Bmatrix}$$

1. Solución por método de Thomas

```python
import numpy as np

a = np.array([-0.4,-0.4])
b = np.array([0.8,0.8,0.8])
c = np.array([-0.4,-0.4])
d = np.array([41,25,105])

def thomas(a, b, c, d)
    n = len(b)
    x = np.zeros(n)
#descomposicion
    for k in range(1,n):
        q = a[k]/b[k-1]
        b[k] = b[k] - c[k-1]*q
        d[k] = d[k] - d[k-1]*q
#sustitucion hacia atras
    q = d[n-1]/b[n-1]
    x[n-1] = q
    for k in range(n-2,-1,-1):
        q = (d[k]-c[k]*q)/b[k]
        x[k] = q
    return x

thomas(a,b,c,d)
```

2.2. Algoritmo Crank-Nikolson

Problema 11.3

El sistema tridiagonal que sigue debe resolverse como parte de un algoritmo mayor (Crank-Nicolson) para solucionar ecuaciones diferenciales parciales:

$$\begin{bmatrix} 2.01475 & -0.020875 & & \\ -0.020875 & 2.01475 & -0.020875 & \\ & -0.020875 & 2.01475 & -0.020875 \\ & & -0.020875 & 2.01475 \end{bmatrix} \times \begin{Bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{Bmatrix} = \begin{Bmatrix} 4.175 \\ 0 \\ 0 \\ 2.0875 \end{Bmatrix}$$

```python
import numpy as np

a = np.array([-0.020875,-0.020875,-0.020875])
b = np.array([2.01475,2.01475,2.01475,2.01475])
c = np.array([-0.020875,-0.020875,-0.020875])
d = np.array([41,25,105])

def thomas(a, b, c, d)
    n = len(b)
    x = np.zeros(n)

#descomposicion
    for k in range(1,n):
        q = a[k]/b[k-1]
        b[k] = b[k] - c[k-1]*q
        d[k] = d[k] - d[k-1]*q

#sustitucion hacia atras
    q = d[n-1]/b[n-1]
    x[n-1] = q
    for k in range(n-2,-1,-1):
        q = (d[k]-c[k]*q)/b[k]
        x[k] = q
    return x

thomas(a,b,c,d)
```

2.3. Sistema simétrico

Problema 11.5

Haga los mismos cálculos para el sistema simétrico que sigue:

$$\begin{bmatrix} 6 & 15 & 55 \\ 15 & 55 & 225 \\ 55 & 25 & 225 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \end{Bmatrix} = \begin{Bmatrix} 152.6 \\ 585.6 \\ 2488.8 \end{Bmatrix}$$

Además de resolver para la descomposición de Cholesky, empléela para solucionar cuál es el valor de las a.

```python
import math
import numpy as np

a = [[6,15,55],[15,55,225],[55,25,225]]
b = [[152.6],[585.6],[2488.8]]

def cholesky(a,b):
    n = len(a)
    b = np.zeros([n]) #vector aumentado
    x = np.zeros([n]) #soluciones
    L = [[0,0,0],[0,0,0],[0,0,0]] #matriz L de ceros

    for i in range(n):
        for k in range(i+1):
            tmp_sum = sum(L[i][j] * L[k][j] for j in range(k))

            if (i==k):  #elementos de la diagonal
                L[i][k] = math.sqrt(a[i][i] - tmp_sum)
            else:
                L[i][k] = (1 / L[k][k] * (a[i][k] - tmp_sum))
    #sustitucion hacia atras
    x[n-1] = b[n-1]/a[n-1,n-1]
    for i in range(n-2,-1,-1):
        sum = 0
        for j in range(0,n):
            sum = sum + a[i,j] * x[j]
        x[i] = (b[i] - sum)/a[i,i]

        #imprimir matriz L resultante
    print("La matriz L: \n")
    for line in L:
        print ('  '.join(map(str, line)))

    print("Soluciones al sistema: ", x)  #soluciones al sistema

cholesky(a,b)
```

2.4. Descomposición Cholesky

Problema 11.7

Calcule la descomposición de Cholesky de:

$$[A] = \begin{bmatrix} 9 & 0 & 0 \\ 0 & 25 & 0 \\ 0 & 0 & 4 \end{bmatrix}$$

```python
import math
import numpy as np

a=[[9,0,0],
   [0,25,0],
   [0,0,4]] #matriz 3x3 del problema

L = np.array([[0,0,0],[0,0,0],[0,0,0]]) #matriz L de ceros


def cholesky(a):
    n = len(a)

    for i in range(n):
        for k in range(i+1):
            sumatoria = sum(L[i][j] * L[k][j] for j in range(k))

            if (i==k):  #elementos de la diagonal
                L[i][k] = math.sqrt(a[i][i] - sumatoria)
            else:
                L[i][k] = (1 / L[k][k] * (a[i][k] - sumatoria))

    print("La matriz L: \n")
    for line in L:
        print (' '.join(map(str, line)))

cholesky(a)
```

CAPÍTULO 3

ANEXOS

3.1. Anexo A: Evidencia del funcionamiento de los código reportados

3.1.1. Ejercicio 11.1

```
Matriz resultante
 [[10.          2.          -1.        ]
 [ 0.         -5.4         1.7        ]
 [ 0.          0.          5.35185185]]

Vector resultante
 [ 27.          -53.4         -32.11111111]

Soluciones al sistema:
 [ 0.5  8.  -6. ]
```

Figura 3.1: Método de Thomas

3.1.2. Ejercicio 11.3

```
Matriz U
 [[ 8.    4.   -1.  ]
 [ 0.    6.    0.75]
 [ 0.    0.    6.5 ]]
Matriz L
 [[ 1.          0.          0.        ]
 [-0.25        1.          0.        ]
 [ 0.25       -0.33333333  1.        ]]
Inversa
 [[ 0.09935897 -0.07371795  0.02884615]
 [ 0.04487179  0.16025641 -0.01923077]
 [-0.02564103  0.05128205  0.15384615]]
```

Figura 3.2: Descomposición de Thomas

3.1.3. Ejercicio 11.5

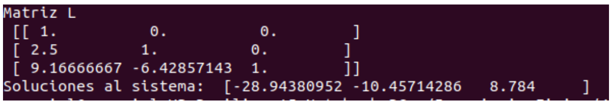

Figura 3.3: Descomposición Cholesky con solución al sistema

3.1.4. Ejercicio 11.7

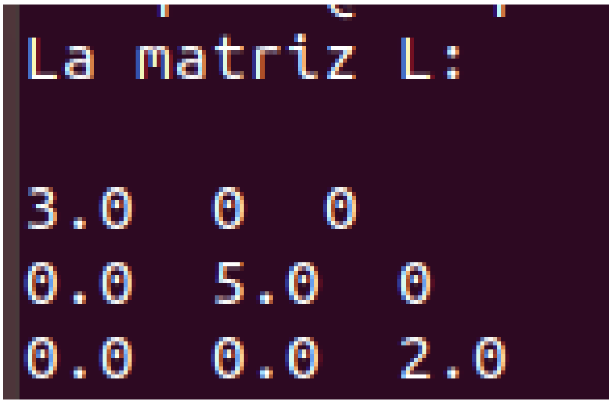

Figura 3.4: Descomposición de Cholesky

CAPÍTULO 4

BIBLIOGRAFÍA

1. Chapra, S. C., Canale, R. P. (2011). *Numerical methods for engineers* (Vol. 1221). New York: Mcgraw-hill.