Factorización LDL^t

Juan Manuel Fornos Andrade
Jorge Álvarez Espiño
UVigo

E-mail: juaforn@outlook.com

E-mail: jorgeaespino@gmail.com
diciembre 2021

Abstract. Aplicaremos la factorización LDL^t mediante nuestro programa desarrollado en Fortran a un ejemplo sencillo de corrientes en circuitos

1. Módulos, programa y ficheros

-**Módulos**

- 'checks.f90' : Contiene las rutinas para comprobar las restricciones para aplicar la descomposición LDL^t, que la matriz sea simétrica y definida positiva. Contiene también una descomposición LU (la realizada en clase) para calcular los determinantes de las submatrices pertinentes para comprobar si una matriz es definida positiva.
- 'decomposition.f90' : Contiene las rutinas para realizar la descomposición LDL^t, imprimiendo las matrices pertinentes por pantalla a la vez y realizando una comprobación de concordancia en los cálculos (A - LDL^t), así como los métodos de resolución de sistemas mediante sustitución hacia adelante, hacia atrás y para matrices diagonales (*si hubiéremos programado más descomposiciones tendría más sentido dedicarle un módulo aparte, pero para este único caso decidimos incluirlo en el mismo*).

-**Programa**

- 'LDLt.f90': Resuelve un sistema lineal, $ax = b$, gracias al método LDL^t, introduciendo la dimensión n, la matriz de coeficientes a, y el vector columna b.

-**Ficheros**

- 'data.dat' : Fichero de datos para introducir dimensión, matriz y vector b
- 'output.dat' : Fichero de salida

- 'errors.dat' : Fichero para guardar los errores lanzados en la ejecución del programa
- 'Makefile' : Guarda los comandos para realizar la compilación y ejecución, se incluye el parámetro show para ver el output.dat por terminal

2. Resolución circuito eléctrico por método de mallas.

Se dice que un circuito teórico está resuelto cuando se ha calculado el voltaje y la corriente que circulará por cada uno de los elementos del circuito. Para poder conocer estas magnitudes vamos a emplear el método de corriente de malla. Para lo que procederemos de la siguiente forma:

(i) Determinar las mallas que tiene el circuito.

(ii) Dibujar corrientes de malla en sentido horario en cada una de las mallas.

(iii) Escribir un sistema de matrices de la siguiente forma:

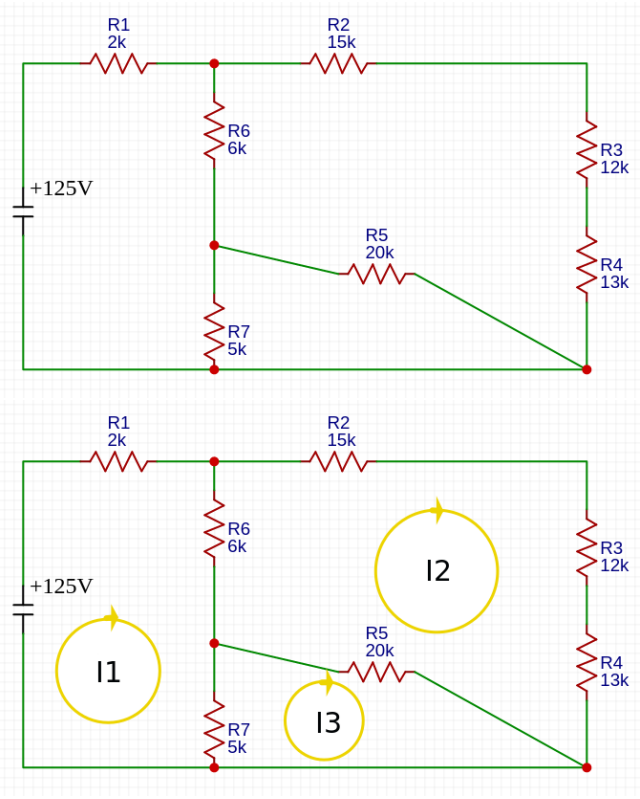
$$\begin{bmatrix} R11 & R12 & R13 \\ R21 & R22 & R23 \\ R31 & R32 & R33 \end{bmatrix} \cdot \begin{bmatrix} I1 \\ I2 \\ I3 \end{bmatrix} = \begin{bmatrix} V1 \\ V2 \\ V3 \end{bmatrix}$$

En el vector de voltajes, debemos escribir la suma de los voltajes de las fuentes que tenemos en cada malla anteponiendo el signo + o – por el que salgamos al movernos en sentido horario.

En la matriz de resistencia, los elementos de la diagonal principal son la suma de las resistencias de cada una de las mallas. Así, R11 será la suma de las resistencias de la malla 1, R22 será la suma de las resistencias de la malla 2, etc. Los demás elementos son la suma de las resistencias frontera entre dos mallas cambiadas de signo. Así R12 será la suma entre las resistencias comunes a las mallas 1 y 2 cambiadas de signo dado que la corriente I2 va en sentido contrario a I1.

Los elementos del vector de corrientes son las corrientes de malla.

Como caso concreto escogemos el circuito de la siguiente figura:



Del cual obtenemos el siguiente sistema y la matriz de coeficientes a factorizar:

$$\begin{bmatrix} 13 & -6 & -5 \\ -6 & 66 & -20 \\ -5 & -20 & 25 \end{bmatrix} \cdot \begin{bmatrix} I1 \\ I2 \\ I3 \end{bmatrix} = \begin{bmatrix} 125 \\ 0 \\ 0 \end{bmatrix}$$

El valor al que debemos llegar para cada corriente:

- $I1 = 12,5A$
- $I2 = 2,5A$
- $I3 = 4,5A$

3. 3. Funcionamiento del programa y outputs

Lo que sigue se ha ejecutado en la terminal de Ubuntu 20.04 . Para compilar y ejecutar simplemente escribimos *make* (para hacer uso de nuestro Makefile), que sería equivalente a usar los comandos:

```
~$ gfortran checks.f90 decomposition.f90 LDLt.f90
~$ ./a.out < data.dat > output.dat 2> errors.dat
```

con el primero compilariamos y con el segundo ejecutaríamos. En concreto, empleamos *make show*, que añadiría el tercer comando *cat output.dat*:

```
~$ make show
gfortran checks.f90 decomposition.f90 LDLt.f90
./a.out < data.dat > output.dat 2> errors.dat
cat output.dat
 ------------------------------
 The matrix dimension n is:
           3
 The coefficient matrix a is:
   13.000000000000000        -6.0000000000000000        -5.0000000000000000
  -6.0000000000000000         66.000000000000000        -20.000000000000000
  -5.0000000000000000        -20.000000000000000         25.000000000000000
 Vector b:
   125.00000000000000
   0.0000000000000000
   0.0000000000000000
 ------------------------------
 Matrix is symmetric
 Matrix is positive definite
 ------------------------------
 Matrix L:

   1.0000000000000000         0.0000000000000000         0.0000000000000000
 -0.46153846153846156         1.0000000000000000         0.0000000000000000
 -0.38461538461538464        -0.35279805352798049         1.0000000000000000
 Matrix D:

   13.000000000000000         0.0000000000000000         0.0000000000000000
   0.0000000000000000         63.230769230769234         0.0000000000000000
   0.0000000000000000         0.0000000000000000         15.206812652068129
 Matrix Lt:

   1.0000000000000000        -0.46153846153846156        -0.38461538461538464
   0.0000000000000000         1.0000000000000000        -0.35279805352798049
   0.0000000000000000         0.0000000000000000         1.0000000000000000
 ERROR: a - L*D*Lt =
   0.0000000000000000         0.0000000000000000        -3.5527136788005009E-015
   0.0000000000000000         0.0000000000000000        -3.5527136788005009E-015
   0.0000000000000000         0.0000000000000000        -3.5527136788005009E-015

 LDLt decomposition done successfully
 ------------------------------
 Solution vector x:
```

```
x1 =    12.500000000000000
x2 =    2.4999999999999996
x3 =    4.4999999999999991
```

Como vemos obtenemos la misma solución, aunque no de manera exacta, solo por ver como saldría con otro ejemplo, escogemos $R11 = 10$ y $R22 = 60$ (obviamente la solución de las corrientes cambiará), y comprobamos como no arroja error númerico a la hora de realizar la descomposición y el cálculo $L \cdot D \cdot L^t$:

```
~$ make show
gfortran checks.f90 decomposition.f90 LDLt.f90
./a.out < data.dat > output.dat 2> errors.dat
cat output.dat
```

```
The matrix dimension n is:
           3
The coefficient matrix a is:
   10.000000000000000       -6.0000000000000000       -5.0000000000000000
  -6.0000000000000000        60.000000000000000       -20.000000000000000
  -5.0000000000000000       -20.000000000000000        25.000000000000000
Vector b:
   125.00000000000000
   0.0000000000000000
   0.0000000000000000
```

```
Matrix is symmetric
Matrix is positive definite
```

```
Matrix L:

   1.0000000000000000        0.0000000000000000        0.0000000000000000
 -0.59999999999999998        1.0000000000000000        0.0000000000000000
 -0.50000000000000000       -0.40780141843971635        1.0000000000000000
Matrix D:

   10.000000000000000        0.0000000000000000        0.0000000000000000
   0.0000000000000000        56.399999999999999        0.0000000000000000
   0.0000000000000000        0.0000000000000000        13.120567375886523
Matrix Lt:

   1.0000000000000000       -0.59999999999999998       -0.50000000000000000
   0.0000000000000000        1.0000000000000000       -0.40780141843971635
```

```
0.0000000000000000   0.0000000000000000   1.0000000000000000

LDLt decomposition done successfully
```

```
Solution vector x:
x1 =    18.581081081081081
x2 =    4.2229729729729737
x3 =    7.0945945945945956
```