Métodos Numéricos

Resolución de Sistemas de Ecuaciones Lineales

Diego Passarella

Universidad Nacional de Quilmes

Sistemas de Ecuaciones Lineales

Se estudia la resolución de problemas del tipo

$$Ax = b$$

con A una matriz cuadrada e inversible ($det(A) \neq 0$, por lo tanto, posee solución única).

En general, la resolución $x = A^{-1}b$ no es computacionalmente viable.

Hay que utilizar otras estrategias.

Sistemas de Ecuaciones Lineales

Métodos para resolver S.E.L:

- Directos: Para sistemas relativamente pequeños y con matrices densamente pobladas. Se basan en operar sobre la matriz A o su expandida $[A|b]$ para llevarla a una forma que sea fácilmente resoluble (matriz diagonal o triangular). Se obtiene la solución (idealmente exacta) en un número finito de operaciones conocido *a priori*.
- Iterativos: Para sistemas grandes y ralos (muchos ceros en la matriz A). Se genera una sucesión que aproxima a la solución por medio de la repetición de operaciones "matriz x vector" en cada iteración. Se llega a una aproximación de la solución con una dada tolerancia en una cantidad *a priori* desconocida de pasos.

Matriz triangular inferior

Una matriz triangular inferior es de la forma:

$$L = \begin{pmatrix} l_{1,1} & 0 & 0 & \cdots & 0 \\ l_{2,1} & l_{2,2} & 0 & \cdots & 0 \\ l_{3,1} & l_{3,2} & l_{3,3} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{n,1} & l_{n,2} & l_{n,3} & \cdots & l_{n,n} \end{pmatrix}$$

Algoritmo de descenso

Resolución de sistemas con matriz triangular inferior:

Algoritmo:

1) $x_1 = b_1 / l_{1,1}$

2) para i=2 hasta n

$$x_i = \frac{b_i - \sum_{j=1}^{i-1} x_j l_{i,j}}{l_{i,i}}$$

Matriz triangular superior

Una matriz triangular superior es de la forma:

$$U = \begin{pmatrix} u_{1,1} & u_{1,2} & u_{1,3} & \cdots & u_{1,n} \\ 0 & u_{2,2} & u_{2,3} & \cdots & u_{2,n} \\ 0 & 0 & u_{3,3} & \cdots & u_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & u_{n,n} \end{pmatrix}$$

Algoritmo de remonte

Resolución de sistemas con matriz triangular superior:

<u>Algoritmo:</u>

1) $x_n = b_n / u_{n,n}$

2) para i=n-1 hasta 1

$$x_i = \frac{b_i - \sum_{j=i+1}^{n} x_j u_{i,j}}{u_{i,i}}$$

Eliminación de Gauss

En cada etapa se hacen cero los elementos debajo de la diagonal principal. Se trabaja con la matriz expandida $\widetilde{A} = [A|b]$.

Algoritmo:

1) para k=1 hasta n-1
(recorro las n-1 primeras columnas)

2) para i=k+1 hasta n
(recorro todas las filas debajo de la diagonal)

$$l_{i,k} = \widetilde{a}_{i,k} / \widetilde{a}_{k,k}$$

...

Eliminación de Gauss

...

3) para j=k hasta n+1

(recorro toda la k-ésima fila)

$$\widetilde{a}_{i,j} = \widetilde{a}_{i,j} - l_{i,k}\widetilde{a}_{k,j}$$

Condición importante: $\widetilde{a}_{k,k} \neq 0$ siempre.
Este método puede generar grandes errores cuando $\widetilde{a}_{k,k}$ es pequeño. En ese caso, hay que sumar una etapa de pivoteo en cada $k-$ésimo paso.
El sistema a resolver termina siendo:

$$PAx = Pb$$

Donde P es la matriz de permutaciones resultante del pivoteo

Factorización LU

Idea: Transformar a la matriz A en el producto de dos matrices, una triangular superior y otra triangular inferior de manera que:

$$A = LU$$

$$Ax = b \Rightarrow LUx = b$$

con lo que se termina resolviendo dos sistemas de matriz diagonal

$$Ly = b$$

$$Ux = y$$

La factorización LU no es única, por lo que se impone que la diagonal principal de L esté formada por 1's.

Factorización LU

Algoritmo:

Utilizar el mismo algoritmo que para la eliminación de Gauss, pero solamente a la matriz A (no a la expandida)

La matriz A resultante va a ser la matriz U y los elementos de pivote $l_{i,j}$ van a ser los del triángulo inferior de L (agregar la diagonal de unos)

La factorización LU es conveniente cuando hay que resolver varios sistemas con la misma matriz A, pero con distintos vectores b

Factorización LU - Cálculo de Determinantes

Recordar que:

$$det(A) = det(A^t) \quad \text{y que} \quad det(A \cdot B) = det(A) \cdot det(B)$$

Por lo tanto, se puede verificar fácilmente que:

$$det(A) = det(L \cdot U) = det(L) \cdot det(U) = det(U) = \sum_{i=1}^{n} u_{i,i}$$

Factorización de Cholesky

Cuando la matriz A es simétrica ($A = A^t$) y definida positiva ($x^t A x > 0 \quad \forall x \neq \overline{0}$), se puede realizar la siguiente factorización:

$$A = LL^t$$

Se descompone a A en el producto de dos matrices triangulares, donde una es la traspuesta de la otra.

Esta factorización requiere menos operaciones que la factorización LU y permite ahorrar memoria, dado que solamente se almacenan los coeficientes de L.

Factorización de Cholesky

Algoritmo:

1) $l_{1,1} = \sqrt{a_{1,1}}$

2) para i=2 hasta n (recorro toda la primera columna)
$l_{i,1} = a_{i,1}/l_{1,1}$

3) para j=2 hasta n-1 (recorro las columnas interiores)
$l_{j,j} = \sqrt{a_{j,j} - \sum_{k=1}^{j-1} l_{j,k}^2}$
para i=j+1 hasta n (recorro los elem. inferiores de la col.)
$l_{i,j} = \left(a_{i,j} - \sum_{k=1}^{j-1} l_{i,k} l_{j,k}\right) / l_{j,j}$

4) $l_{n,n} = \sqrt{a_{n,n} - \sum_{k=1}^{n-1} l_{n,k}^2}$

Importante: Se debe verificar que los argumentos de las raíces sean positivos.

Condicionamiento

Sensibilidad a perturbaciones:

Es interesante analizar la sensibilidad del resultado de x ante perturbaciones en la matriz de coeficientes A o el vector de términos independientes b.

$$Ax = b \rightarrow A(x + \delta x) = (b + \delta b) \quad \text{ó} \; (A + \delta A)(x + \delta x) = b$$

Los errores resultantes están acotados por:

$$\frac{||\delta x||}{||x||} \leq cond(A) \frac{||\delta b||}{||b||}$$

y

$$\frac{||\delta x||}{||x + \delta x||} \leq cond(A) \frac{||\delta A||}{||A||}$$

Número de condición de una matriz

El número de condición se define como:

$$cond_p(A) := ||A||_p \cdot ||A^{-1}||_p$$

Esta forma de calcularlo no suele ser práctica, por lo que hay que usar definiciones más simples que son válidas para cierto tipo de matrices.

- El número de condición de una matriz A mide la sensibilidad de la solución de un S.E.L. respecto a variaciones en A o b.
- El sistema se dice bien condicionado si $cond(A)$ es pequeño ($cond(A) \geq 1$ siempre).
- Se puede mejorar el condicionamiento de un sistema multiplicando filas por coeficientes adecuados (precondicionamiento).

Ejemplo

Analizar el caso siguiente:

$$\begin{pmatrix} 10 & 7 & 8 & 7 \\ 7 & 5 & 6 & 5 \\ 8 & 6 & 10 & 9 \\ 7 & 5 & 9 & 10 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 32 \\ 23 \\ 33 \\ 31 \end{pmatrix}$$