Matrices especiales y el método de Gauss-Seidel

Lección 12

Dr. Pablo Alvarado Moya

CE3102 Análisis Numérico para Ingeniería
Área de Ingeniería en Computadores
Tecnológico de Costa Rica

I Semestre 2018

Contenido

Sistemas tridiagonales

- Algunos problemas de ingeniería conducen a sistemas de ecuaciones tridiagonales o a bandas. Ejemplos:
  - Splines
  - Ecuaciones diferenciales
- Métodos convencionales (eliminación de Gauss, descomposición **LU** de Doolittle o Crout) son ineficientes para estos casos.

Algoritmo de Thomas

Supóngase que se tiene el sistema tridiagonal

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

El algoritmo de Thomas es una descomposición LU optimizada:

1. Descomposición

$$\begin{array}{l} \text{for } (k = 2; k \leq n; k++) \ \{ \\ \quad e_k \ /= f_{k-1} \\ \quad f_k \ -= e_k \cdot g_{k-1} \\ \} \end{array}$$

2. Sust. hacia adelante

$$\begin{array}{l} \text{for } (k = 2; k \leq n; k++) \ \{ \\ \quad r_k \ -= e_k \cdot r_{k-1} \\ \} \end{array}$$

3. Sust. hacia atrás

$$\begin{array}{l} x_n = r_n / f_n \\ \text{for } (k = n-1; k > 0; k--) \ \{ \\ \quad x_k = (r_k - g_k \cdot x_{k+1}) / f_k \\ \} \end{array}$$

Cálculo *in-situ*

El algoritmo calcula *in-situ* las matrices **LU**

$$\begin{bmatrix} f_1 & g_1 & & & & \\ e_2 & f_2 & g_2 & & & \\ & e_3 & f_3 & g_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & e_{n-1} & f_{n-1} & g_{n-1} \\ & & & & e_n & f_n \end{bmatrix} =$$

$$\begin{bmatrix} 1 & & & & & \\ e'_2 & 1 & & & & \\ & e'_3 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & e'_{n-1} & 1 & \\ & & & & e'_n & 1 \end{bmatrix} \begin{bmatrix} f'_1 & g_1 & & & & \\ & f'_2 & g_2 & & & \\ & & f'_3 & g_3 & & \\ & & & \ddots & \ddots & \\ & & & & f'_{n-1} & g_{n-1} \\ & & & & & f'_n \end{bmatrix}$$

Ejemplo: Algoritmo de Thomas (1)

Ejemplo

Resolver

$$\begin{bmatrix} 2,04 & -1 & & \\ -1 & 2,04 & -1 & \\ & -1 & 2,04 & -1 \\ & & -1 & 2,04 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 40,08 \\ 0,8 \\ 0,8 \\ 200,8 \end{bmatrix}$$

Ejemplo: Algoritmo de Thomas (2)

Solución:

$$\begin{bmatrix} f_1 & g_1 & & \\ e_2 & f_2 & g_2 & \\ & e_3 & f_3 & g_3 \\ & & e_4 & f_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \end{bmatrix}$$

Aquí, $f_i = 2,04$ y $e_i = g_i = -1$.

La descomposición:

```
for (k = 2; k ≤ n; k++) {
    e_k /= f_{k-1}
    f_k -= e_k · g_{k-1}
}
```

$$e_2 = e_2/f_1 = -0,49$$
$$f_2 = f_2 - e_2 \cdot g_1 = 1,55$$
$$e_3 = e_3/f_2 = -0,645$$
$$f_3 = e_3 \cdot g_2 = 1,395$$
$$e_4 = e_4/f_3 = -0,717$$
$$f_4 = f_4 - e_4 \cdot g_3 = 1,323$$

Ejemplo: Algoritmo de Thomas (3)

Entonces, $\mathbf{LU} = \mathbf{A}$:

$$\mathbf{LU} = \begin{bmatrix} 1 & & & \\ -0,49 & 1 & & \\ & -0,645 & 1 & \\ & & -0,717 & 1 \end{bmatrix} \begin{bmatrix} 2,04 & -1 & & \\ & 1,55 & -1 & \\ & & 1,395 & -1 \\ & & & 1,323 \end{bmatrix}$$

La sustitución hacia adelante:

$$\begin{aligned} &\text{for } (k = 2; k \leq n; k++) \ \{ \\ &\qquad r_k -= e_k \cdot r_{k-1} \\ &\} \end{aligned}$$

$$\begin{aligned} r_2 &= r_2 - e_2 \cdot r_1 = 20,8 \\ r_3 &= r_3 - e_3 \cdot r_2 = 14,221 \\ r_4 &= r_4 - e_4 \cdot r_3 = 210,996 \end{aligned}$$

Ejemplo: Algoritmo de Thomas (4)

Finalmente, la sustitución hacia atrás:

$$
\begin{aligned}
&x_n = r_n/f_n \\
&\text{for } (k = n-1; k > 0; k--) \{ \\
&\qquad x_k = (r_k - g_k \cdot x_{k+1})/f_k \\
&\}
\end{aligned}
$$

$$
\begin{aligned}
x_4 &= r_4/f_4 = 159,48 \\
x_3 &= (r_3 - g_3 \cdot x_4)/f_3 = 124,538 \\
x_2 &= (r_2 - g_2 \cdot x_3)/f_2 = 93,778 \\
x_1 &= (r_1 - g_1 \cdot x_2)/f_1 = 65,97
\end{aligned}
$$

Almacenamiento eficiente de matriz tridiagonal

Una estrategia de ahorro de memoria para el almacenamiento de matrices tridiagonales a bandas es utilizar

$$\begin{bmatrix} & f_1 & g_1 \\ e_2 & f_2 & g_2 \\ e_3 & f_3 & g_3 \\ \vdots & \vdots & \vdots \\ e_{n-1} & f_{n-1} & g_{n-1} \\ e_n & f_n & \end{bmatrix}$$

donde las diagonales de la matriz original ocupan columnas de la versión "comprimida".

Descomposición de Cholesky

- La descomposición de Cholesky se aplica a
  - matrices simétricas, esto es $\mathbf{A} = \mathbf{A}^T$
  - matrices definidas positivas, esto es $\underline{\mathbf{x}}^T \mathbf{A} \underline{\mathbf{x}} > 0$ para todo vector no nulo $\underline{\mathbf{x}}$
- La descomposición de Cholesky para la matriz simétrica $\mathbf{A}$ es

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T$$

- La matríz $\mathbf{L}^T$ es triangular superior y equivale entonces a la matriz $\mathbf{U}$ de la descomposición $\mathbf{LU}$
- A $\mathbf{L}$ se le conoce como la *raíz cuadrada* de $\mathbf{A}$

¿Cómo derivar las ecuaciones?

Algoritmo de descomposición de Cholesky

Siguiendo el esquema de descomposición de Crout se obtiene

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

$$l_{ki} = \frac{1}{l_{ii}} \left(a_{ki} - \sum_{j=1}^{i-1} l_{ij} l_{kj} \right) \qquad i = 1, 2, \ldots k-1$$

Con matrices simétricas y definidas positivas no es necesario el pivoteo, pues el método es numéricamente estable.

Descomposición **QR**

- Otra descomposición utilizada es

$$\mathbf{A} = \mathbf{QR}$$

- $\mathbf{R}$ es triangular superior
- $\mathbf{Q}$ es ortogonal (i.e. $\mathbf{Q}^T = \mathbf{Q}^{-1}$)
- Su uso para resolver sistemas de ecuaciones se basa en que

$$\begin{aligned} \mathbf{A}\underline{\mathbf{x}} &= \underline{\mathbf{b}} \\ \mathbf{QR}\underline{\mathbf{x}} &= \underline{\mathbf{b}} \\ \mathbf{R}\underline{\mathbf{x}} &= \mathbf{Q}^T\underline{\mathbf{b}} \end{aligned}$$

1. Calcular $\underline{\mathbf{b}}' = \mathbf{Q}^T\underline{\mathbf{b}}$
2. Sustituir hacia atrás $\mathbf{R}\underline{\mathbf{x}} = \underline{\mathbf{b}}'$

Características de la descomposición **QR**

- La descomposición **QR** requiere cerca del **doble** de operaciones que la descomposición **LU**
- Estabilidad numérica del método es alta, por lo que se elige cuando las matrices son mal condicionadas
- Para encontrar **Q** se utiliza una sucesión de transformaciones de Householder

Transformaciones de Householder (1)

- La forma de la matriz de transformación de Householder es

$$\mathbf{Q}_1 = \mathbf{I} - 2\frac{\underline{\mathbf{u}}\underline{\mathbf{u}}^T}{\underline{\mathbf{u}}^T\underline{\mathbf{u}}} = \mathbf{I} - 2\frac{\underline{\mathbf{u}}\underline{\mathbf{u}}^T}{\|\underline{\mathbf{u}}\|^2} = \mathbf{I} - 2\underline{\bar{\mathbf{u}}}\underline{\bar{\mathbf{u}}}^T$$

  para un vector dado $\underline{\mathbf{u}}$, y su normalización $\underline{\bar{\mathbf{u}}} = \underline{\mathbf{u}}/\|\underline{\mathbf{u}}\|$
- Nótese que $\mathbf{Q}_1$ es simétrica: $\mathbf{Q}_1^T = \mathbf{Q}_1$
- $\mathbf{Q}_1$ es ortogonal, pues

$$\begin{aligned}\mathbf{Q}_1\mathbf{Q}_1^T &= \mathbf{Q}_1\mathbf{Q}_1 \\ &= (\mathbf{I} - 2\underline{\bar{\mathbf{u}}}\underline{\bar{\mathbf{u}}}^T)(\mathbf{I} - 2\underline{\bar{\mathbf{u}}}\underline{\bar{\mathbf{u}}}^T) \\ &= \mathbf{I} - 2\underline{\bar{\mathbf{u}}}\underline{\bar{\mathbf{u}}}^T - 2\underline{\bar{\mathbf{u}}}\underline{\bar{\mathbf{u}}}^T + 4\underline{\bar{\mathbf{u}}}\underline{\bar{\mathbf{u}}}^T = \mathbf{I}\end{aligned}$$

Transformaciones de Householder (2)

- Sea $\mathbf{P}_1 = \mathbf{Q}_1\mathbf{A}$
- La primera columna de $\mathbf{P}_1$ es $\underline{\mathbf{p}}_{:1} = \mathbf{Q}_1\underline{\mathbf{a}}_{:1}$
- Elíjase $\underline{\mathbf{u}} = \underline{\mathbf{a}}_{:1} \mp \|\underline{\mathbf{a}}_{:1}\|\underline{\mathbf{e}}_0$ para generar $\mathbf{Q}_1 = \mathbf{I} - 2\underline{\mathbf{u}}\underline{\mathbf{u}}^T/\|\underline{\mathbf{u}}\|^2$ con el vector unitario $\underline{\mathbf{e}}_0 = [1, 0 \ldots 0]^T$
- Para la magnitud $\|\underline{\mathbf{u}}\|^2$ se cumple:

$$\begin{aligned}
\|\underline{\mathbf{u}}\|^2 &= \| \underline{\mathbf{a}}_{:1} \mp \|\underline{\mathbf{a}}_{:1}\| \underline{\mathbf{e}}_0 \|^2 \\
&= (a_{11} \mp \|\underline{\mathbf{a}}_{:1}\|)^2 + a_{21}^2 + a_{31}^2 + \ldots + a_{n1}^2 \\
&= a_{11}^2 \mp 2a_{11}\|\underline{\mathbf{a}}_{:1}\| + \|\underline{\mathbf{a}}_{:1}\|^2 + a_{21}^2 + a_{31}^2 + \ldots + a_{n1}^2 \\
&= 2\|\underline{\mathbf{a}}_{:1}\|^2 \mp 2a_{11}\|\underline{\mathbf{a}}_{:1}\|
\end{aligned}$$

Transformaciones de Householder (3)

- Entonces:

$$\begin{aligned}
\underline{\mathbf{p}}_{:1} = \mathbf{Q}\underline{\mathbf{a}}_{:1} &= \left(\mathbf{I} - 2\frac{\mathbf{u}\mathbf{u}^T}{\|\underline{\mathbf{u}}\|^2}\right)\underline{\mathbf{a}}_{:1} = \underline{\mathbf{a}}_{:1} - 2\frac{\mathbf{u}\mathbf{u}^T}{\|\underline{\mathbf{u}}\|^2}\underline{\mathbf{a}}_{:1} \\
&= \underline{\mathbf{a}}_{:1} - \frac{2\underline{\mathbf{u}}(\underline{\mathbf{a}}_{:1} \mp \|\underline{\mathbf{a}}_{:1}\|\,\underline{\mathbf{e}}_0)^T\underline{\mathbf{a}}_{:1}}{\|\underline{\mathbf{u}}\|^2} \\
&= \underline{\mathbf{a}}_{:1} - \frac{2\underline{\mathbf{u}}(\|\underline{\mathbf{a}}_{:1}\|^2 \mp \|\underline{\mathbf{a}}_{:1}\|\,a_{11})}{2\|\underline{\mathbf{a}}_{:1}\|^2 \mp 2a_{11}\|\underline{\mathbf{a}}_{:1}\|} \\
&= \underline{\mathbf{a}}_{:1} - \frac{\underline{\mathbf{u}}(\|\underline{\mathbf{a}}_{:1}\|^2 \mp a_{11}\,\|\underline{\mathbf{a}}_{:1}\|)}{\|\underline{\mathbf{a}}_{:1}\|^2 \mp a_{11}\|\underline{\mathbf{a}}_{:1}\|} \\
&= \underline{\mathbf{a}}_{:1} - \underline{\mathbf{u}} = \underline{\mathbf{a}}_{:1} - (\underline{\mathbf{a}}_{:1} \mp \|\underline{\mathbf{a}}_{:1}\|\,\underline{\mathbf{e}}_0) \\
&= \pm\|\underline{\mathbf{a}}_{:1}\|\,\underline{\mathbf{e}}_0
\end{aligned}$$

Transformaciones de Householder (4)

- Así, la transformación de Householder $\mathbf{Q}_1$ pone en **cero** todos los elementos en la columna bajo el primer elemento: $p_{i1} = 0$ para $i > 1$
- La transformación $\mathbf{Q}_2$ opera sobre la matriz que resulta de eliminar la primera fila y la primera columna, y pone en cero todos los elementos en la columna bajo el segundo elemento:

$$\mathbf{Q}_2(\mathbf{Q}_1\mathbf{A}) = \left[\begin{array}{c|c} 1 & \underline{\mathbf{0}}^T \\ \hline \underline{\mathbf{0}} & {}^{(n-1)}\underline{\mathbf{Q}}_2 \end{array}\right] \left[\begin{array}{c|c} p_{11} & [p_{12}, p_{13}, \ldots p_{1n}] \\ \hline \underline{\mathbf{0}} & {}^{(n-1)}\underline{\mathbf{P}}_1 \end{array}\right]$$

$$= \left[\begin{array}{cc|c} p_{11} & p_{12} & [p_{13}, p_{14} \ldots p_{1n}] \\ 0 & p'_{22} & [p'_{23}, p'_{24}, \ldots p'_{2n}] \\ \hline \underline{\mathbf{0}} & \underline{\mathbf{0}} & {}^{(n-2)}\underline{\mathbf{P}}_2 \end{array}\right]$$

Transformaciones de Householder (5)

- De forma equivalente para $\mathbf{Q}_3$

$$\mathbf{Q}_3(\mathbf{Q}_2\mathbf{Q}_1\mathbf{A}) = \left[\begin{array}{cc|c} 1 & 0 & \underline{\mathbf{0}}^T \\ 0 & 1 & \underline{\mathbf{0}}^T \\ \hline \underline{\mathbf{0}} & \underline{\mathbf{0}} & {}^{(n-2)}\underline{\mathbf{Q}}_3 \end{array}\right] \left[\begin{array}{cc|c} p_{11} & p_{12} & [p_{13} \cdots p_{1n}] \\ 0 & p'_{22} & [p'_{23} \cdots p'_{2n}] \\ \hline \underline{\mathbf{0}} & \underline{\mathbf{0}} & {}^{(n-2)}\underline{\mathbf{P}}_2 \end{array}\right]$$

$$= \left[\begin{array}{ccc|c} p_{11} & p_{12} & p_{13} & [p_{14}, \cdots p_{1n}] \\ 0 & p'_{22} & p'_{23} & [p'_{24}, \cdots p'_{2n}] \\ 0 & 0 & p''_{33} & [p''_{34} \cdots p''_{3n}] \\ \hline \underline{\mathbf{0}} & \underline{\mathbf{0}} & \underline{\mathbf{0}} & {}^{(n-3)}\underline{\mathbf{P}}_3 \end{array}\right]$$

- Se repiten transformaciones hasta llegar a $\mathbf{Q}_{n-1}$, por lo que

$$\mathbf{R} = \mathbf{Q}_{n-1}\mathbf{Q}_{n-2}\cdots\mathbf{Q}_1\mathbf{A}$$

Transformaciones de Householder (6)

- Puesto que las transformaciones de Householder son ortogonales entonces

$$\mathbf{Q} = (\mathbf{Q}_{n-1}\mathbf{Q}_{n-2}\cdots\mathbf{Q}_1)^{-1} = \mathbf{Q}_1^T\mathbf{Q}_2^T\cdots\mathbf{Q}_{n-1}^T$$

- Pivoteo es necesario solo para matrices casi singulares

Métodos iterativos

Gauss-Seidel (1)

- Gauss-Seidel es un método **iterativo** que contrasta con los métodos de **eliminación** descritos hasta ahora.
- Similar a los métodos de búsqueda de raíces.
- Supóngase que se tiene un sistema

$$\mathbf{A}\underline{\mathbf{x}} = \underline{\mathbf{b}}$$

- Se pueden plantear con las n ecuaciones del sistema n igualdades de la forma:

Gauss-Seidel (2)

$$x_1 = \frac{1}{a_{11}}\left(b_1 - \sum_{j=2}^{n} a_{1j}x_j\right)$$

$$x_2 = \frac{1}{a_{22}}\left(b_2 - \sum_{\substack{j=1 \\ j\neq 2}}^{n} a_{2j}x_j\right)$$

$$x_3 = \frac{1}{a_{33}}\left(b_3 - \sum_{\substack{j=1 \\ j\neq 3}}^{n} a_{3j}x_j\right)$$

$$\vdots$$

$$x_i = \frac{1}{a_{ii}}\left(b_i - \sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij}x_j\right)$$

$$\vdots$$

$$x_n = \frac{1}{a_{nn}}\left(b_n - \sum_{j=1}^{n-1} a_{ij}x_j\right)$$

Gauss-Seidel y Jacobi

- A partir de un vector inicial $\underline{\mathbf{x}}^{(k-1)}$ se calcula una siguiente aproximación $\underline{\mathbf{x}}^{(k)}$ con las ecuaciones anteriores.
- En el método de **Gauss-Seidel** las ecuaciones se evalúan **en-línea**, (secuencialmente), es decir: para calcular $x_i^{(k)}$ se usan $x_j^{(k)}$ para $j < i$ (los valores ya calculados en la k-ésima iteración) y $x_l^{(k-1)}$ para $l > i$ (los valores calculados en la iteración anterior).
- En el método de **Jacobi** las ecuaciones se evalúan **por lotes** (en paralelo), es decir $\underline{\mathbf{x}}^{(k)}$ se evalúa empleando todo el conjunto de valores anterior $\underline{\mathbf{x}}^{(k-1)}$

Criterio de convergencia

- Criterio relativo

$$|\epsilon_{a,i}| = \left| \frac{x_i^j - x_i^{j-1}}{x_i^j} \right| 100\,\% < \epsilon_s$$

- Criterio absoluto

$$|E_{a,i}| = \left| x_i^j - x_i^{j-1} \right| < E_s$$

Problemas de convergencia de método de Gauss-Seidel

- Método de Gauss-Seidel similar a método de punto fijo.
- $\Rightarrow$ es posible que diverja
- $\Rightarrow$ es posible que si converge lo haga de forma muy lenta.
- Se puede demostrar que sistema converge si es **diagonalmente dominante**, es decir si

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|$$

Mejora del método por medio de relajación (1)

- La relajación es una modificación para mejorar convergencia
- Si el método de Gauss-Seidel produce $\tilde{x}_i^k$, entonces el verdadero valor actualizado se calcula con

$$x_i^{(k)} = \lambda \tilde{x}_i^{(k)} + (1 - \lambda) x_i^{(k-1)}$$

- El método de Gauss-Seidel "puro" se obtiene con $\lambda = 1$
- Con $0 < \lambda < 1$ se obtiene *subrelajación* (se amortiguan oscilaciones)
- Con $1 < \lambda < 2$ se obtiene *sobrerelajación* que asume que el nuevo valor se mueve en la dirección correcta, y método acelera el movimiento
- Elección de λ lo determina el problema concreto y se determina de forma empírica.

Mejora del método por medio de relajación (2)

- Implementación específica para problemas de matrices dispersas (la mayoría de elementos son cero) es eficiente en espacio y cómputo.

Funciones de GNU/Octave

	Matrices
cond	Número de condición de una matriz
norm	Norma vectorial o matricial
rank	Rango de la matriz (número de ecuaciones linealmente independientes)
det	Determinante
trace	Traza de la matriz
	Ecuaciones lineales
chol	Factorización de Cholesky
lu	Descomposición LU
inv	Matriz inversa
qr	Descomposición QR
$x = A \backslash b$	Resuelve $\mathbf{A}\underline{\mathbf{x}} = \underline{\mathbf{b}}$ eficientemente

Resumen

Este documento ha sido elaborado con software libre incluyendo LaTeX, Beamer, GNUPlot, GNU/Octave, XFig, Inkscape, LTI-Lib-2, GNU-Make y Subversion en GNU/Linux