Programación: Sistemas unitriangulares inferiores

Objetivos. Programar el método de la *sustitución hacia adelante* para resolver sistemas de ecuaciones lineales con matrices unitriangulares inferiores. Analizar la complejidad computacional del problema.

Requisitos. Matrices triangulares, notación breve para las sumas, experiencia de programación: funciones, ciclos, vectores, matrices.

1. Definición (matrices unitriangulares inferiores). Una matriz cuadrada L se llama *unitriangular inferior* si es triangular inferior y todas sus entradas diagonales son 1. Un sistema de ecuaciones lineales de la forma $Lx = b$ se llama *unitriangular inferior* si la matriz L es unitriangular inferior.

2. Ejemplo. Resolver el siguiente sistema unitriangular inferior de ecuaciones lineales:

$$\begin{cases} x_1 & & & & & & & = & -2; \\ 3x_1 & + & x_2 & & & & & = & -5; \\ -2x_1 & + & 4x_2 & + & x_3 & & & = & 5; \\ -3x_1 & + & x_2 & & & + & x_4 & = & 3. \end{cases}$$

Solución. La primera ecuación nos da el valor de x_1. De la segunda ecuación despejamos x_2 y sustituimos el valor de x_1, etc.:

$$x_1 = \underbrace{\qquad}_{?};$$

$$x_2 = -5 - 3x_1 = -5 - (-6) = \underbrace{\qquad}_{?};$$

$$x_3 = 5 - \big(-2x_1 + 4x_2\big) = 5 - \big(\qquad\qquad\big) = \underbrace{\qquad}_{?};$$

$$x_4 = \underbrace{\qquad}_{?} - \big(\underbrace{\qquad\qquad\qquad}_{?} + 0x_3\,\big) = \underbrace{\qquad}_{?} - \underbrace{\qquad}_{?} = \underbrace{\qquad}_{?}.$$

Comprobación:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 3 & 1 & 0 & 0 \\ -2 & 4 & 1 & 0 \\ -3 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} -2 \\ \hline \\ \hline \\ \hline \\ \end{bmatrix} = \begin{bmatrix} -2+0+0+0 \\ \hline \\ \hline \\ \hline \\ \end{bmatrix} = \begin{bmatrix} -2 \\ \hline \\ \hline \\ \hline \\ \end{bmatrix}. \qquad \checkmark \ \square$$

Notación breve para sumas (repaso)

3. Ejemplos.

$$\sum_{k=4}^{6} b_k = \underset{\text{con } k=4}{b_k} + \underset{\text{con } k=5}{b_k} + \underset{\text{con } k=6}{b_k} = b_4 + b_5 + b_6.$$

$$\sum_{r=2}^{5} a_r = a_2 + a_3 + a_4 + a_5, \qquad \sum_{j=3}^{6} c_j = \underbrace{\qquad\qquad}_{?},$$

$$\sum_{k=2}^{3} c_k d_k = c_2 d_2 + c_3 d_3, \qquad \sum_{p=1}^{4} b_p y_p = \underbrace{\qquad\qquad\qquad\qquad}_{?}.$$

4. Sumas que dependen de parámetros.
La primera suma depende de un parámetro k (esto significa que k no se elimina aún cuando escribimos los sumandos en la forma explícita); la segunda depende de i:

$$\sum_{j=1}^{4} A_{k,j} = A_{k,1} + A_{k,2} + A_{k,3} + A_{k,4}; \qquad \sum_{j=2}^{3} A_{i,j} x_j = \underbrace{\qquad\qquad}_{?}.$$

5. Ejemplos: escribir con $\sum$.

$$A_{5,1} y_1 + A_{5,2} y_2 = \sum_{j=1}^{2} A_{5,j} y_j. \qquad b_2 + b_3 + b_4 + b_5 + b_6 = \sum_{j=} \underbrace{\qquad}_{?}.$$

$$c_3 a_3 + c_4 a_4 + c_5 a_5 = \sum \underbrace{\qquad\qquad}_{?}. \qquad a_2 b_3 + a_3 b_4 + a_4 b_5 = \sum \underbrace{\qquad\qquad}_{?}.$$

6. Convenio: la suma del conjunto vacío de sumandos es cero.

Ejemplo: $\displaystyle\sum_{j=3}^{2} a_j = \sum_{\substack{j\in\mathbb{Z}:\\ 3\le j\le 2}} a_j = \sum_{j\in\emptyset} a_j = 0.$

Fórmulas de la sustitución hacia adelante en el caso de un sistema unitriangular inferior (se recomienda deducir las fórmulas antes de la clase práctica)

7. Fórmulas para $n = 4$. Consideremos un sistema de ecuaciones de la forma $Lx = b$, donde L es una matriz real unitriangular inferior de orden 4.

$$\begin{cases} x_1 & & & & & & & = & b_1; \\ L_{2,1}x_1 & + & x_2 & & & & & = & b_2; \\ L_{3,1}x_1 & + & L_{3,2}x_2 & + & x_3 & & & = & b_3; \\ L_{4,1}x_1 & + & L_{4,2}x_2 & + & L_{4,3}x_3 & + & x_4 & = & b_4. \end{cases}$$

La primera ecuación nos da el valor de la incógnita x_1. De la segunda ecuación despejamos x_2 (expresamos x_2 a través de x_1). De la tercera ecuación despejamos la incógnita x_3 (la expresamos a través de x_1 y x_2). De la cuarta ecuación despejamos x_4.

$x_1 =$

$x_2 =$

$x_3 = \underbrace{\quad}_{?} - \Big(\underbrace{\quad}_{?} + \underbrace{\quad}_{?} \Big) = \quad - \sum$

$x_4 =$

8. Fórmulas de la sustitución hacia adelante en el caso de un sistema unitriangular inferior. Generalice las fórmulas del ejercicio anterior. Sea A una matriz unitriangular inferior $n \times n$ y sea $b \in \mathbb{R}^n$. Escriba una fórmula para calcular x_i:

$$x_i = \qquad\qquad . \tag{1}$$

9. Suma sobre el conjunto vacío. Por definición, cualquier suma sobre un conjunto vacío es cero. Por ejemplo,

$$\sum_{i=4}^{3} a_i = \sum_{i \in \mathbb{Z}:\ 4 \le i \le 3} a_i = \sum_{i \in \emptyset} a_i = 0.$$

Escriba la fórmula (1) para $i = 1$ y determine si esta es correcta.

Programación de la sustitución hacia adelante en el caso de sistemas unitriangulares inferiores

10. Escriba una función `solveunilt` que resuelva sistemas de ecuaciones lineales con matrices unitriangulares inferiores.

Entrada: una matriz L y un vector b.

Condiciones que satisface la entrada: L es cuadrada y unitriangular inferior, la longitud de b coincide con el orden de L.

Salida: un vector x tal que $Lx = b$.

Por ejemplo, en el lenguaje MATLAB la función `solveunilt` se debe guardar en el archivo `solveunilt.m` y se puede empezar de la siguiente manera:

```
function [x] = solveunilt(L, b),
   n = length(b);
   x = zeros(n, 1);
   ...
end
```

11. Comprobación con datos pequeños. Escriba una función que haga una comprobación con una matriz unitriangular inferior 3×3.

```
function [] = test1solveunilt(),
   L = [1, 0, 0; 5, 1, 0; -4, -2, 1];
   y = [2; -3; 4];
   b = L * u;
   x = solveunilt(L, b);
   display(x);
   display(y);
end
```

12. Comprobación con matrices pequeñas pseudoaleatorias. Haga una comprobación con $n = 7$ y con datos pseudoaleatorios. Se recomienda el siguiente esquema:

```
function [] = test2solveunitlt(),
   n = 7;
   L = 2 * rand(n, n) - ones(n, n);
   L = tril(L, -1)+ eye(n);
   b = 2 * rand(n, 1) - ones(n, 1);
   x = solveunilt(L, b);
   display(norm(L * x - b));
```

13. Comprobación con datos grandes pseudoaleatorios. Haga comprobaciones con datos pseudoaleatorios y con n grandes, mediendo el tiempo de ejecución.

Análisis de la complejidad computacional

14. Fórmula para la suma de una progresión aritmética (repaso). Recuerde la fórmula para la siguiente suma:

$$\sum_{p=1}^{q} p = \underbrace{\qquad\qquad}_{?}.$$

15. Primera solución (formal). Calcule el número de las operaciones de multiplicación que se necesitan para calcular la expresión

$$\sum_{k=1}^{j-1} L_{j,k} x_k.$$

Calcule el número de las operaciones de multiplicación que se necesitan realizar el algoritmo programado en las páginas anteriores.

16. Número de las entradas de cada tipo en una matriz cuadrada. Consideremos una matriz cuadrada:

$A_{1,1}$	$A_{1,2}$	$A_{1,3}$	$A_{1,4}$
$A_{2,1}$	$A_{2,2}$	$A_{2,3}$	$A_{2,4}$
$A_{3,1}$	$A_{3,2}$	$A_{3,3}$	$A_{3,4}$
$A_{4,1}$	$A_{4,2}$	$A_{4,3}$	$A_{4,4}$

Calcule el número de las entradas (para una matriz cuadrada $n \times n$):

- el número total de las entradas;
- en la diagonal principal;
- fuera de la diagonal principal;
- por debajo de la diagonal principal.

17. Segunda solución (conceptual). Analice la correspondencia entre las entradas de la matriz L (ubicadas por abajo de la diagonal principal) y las operaciones de multiplicación que se usan en algoritmo que hemos programado.

18. La complejidad del algoritmo es la óptima. Explique por qué el problema no se puede resolver aplicando menos operaciones que $\dfrac{n(n-1)}{2}$.