Sistemas de Ecuaciones Lineales I

- **Preliminares:** Expresión matricial. Dificultades numéricas.

Expresión matricial

- Todo sistema de ecuaciones lineales puede escribirse matricialmente:

$$\begin{cases} a_{11}x_1 + \cdots + a_{1n}x_n &= b_1 \\ \vdots & \vdots \\ a_{n1}x_1 + \cdots + a_{nn}x_n &= b_n \end{cases} \iff \boldsymbol{Ax} = \boldsymbol{b},$$

donde

$$\boldsymbol{A} := \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix} \in \mathbb{R}^{n\times n} \quad \text{y} \quad \boldsymbol{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} \in \mathbb{R}^n$$

son los datos y $\boldsymbol{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \in \mathbb{R}^n$ es el vector de incógnitas.

Matriz inversa

- El sistema de ecuaciones lineales $\boldsymbol{Ax} = \boldsymbol{b}$ tiene solución única si y sólo si $\boldsymbol{A}$ es una matriz no singular.

  Recordemos que una matriz $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es no singular si y sólo si se cumple cualquiera de estas condiciones:

  1. $\boldsymbol{A}$ es invertible: $\exists \boldsymbol{A}^{-1} \in \mathbb{R}^{n\times n} : \boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{I}$;
  2. $\det(\boldsymbol{A}) \neq 0$;
  3. todas las filas (y columnas) de $\boldsymbol{A}$ son l.i.: $\text{rango}(\boldsymbol{A}) = n$.
  4. 0 no es valor propio de $\boldsymbol{A}$: $0 \notin \sigma(\boldsymbol{A})$.

- Si $\boldsymbol{A}$ es no singular, entonces

$$\boldsymbol{Ax} = \boldsymbol{b} \qquad \Longleftrightarrow \qquad \boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}.$$

  Sin embargo, en general, no es conveniente calcular la matriz inversa $\boldsymbol{A}^{-1}$ para resolver un sistema de ecuaciones, pues hacerlo así resulta mucho más costoso computacionalmente.

Matriz inversa (cont.)

- Por el contrario, una manera natural de calcular la inversa de una matriz $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ consiste en resolver n sistemas de ecuaciones lineales. Si llamamos $\boldsymbol{c}^1, \ldots, \boldsymbol{c}^n$ a las columnas de $\boldsymbol{A}^{-1}$:

$$\boldsymbol{A}^{-1} = \left[\begin{array}{c|c|c} & & \\ \boldsymbol{c}^1 & \cdots & \boldsymbol{c}^n \\ & & \end{array}\right], \qquad \boldsymbol{c}^1, \ldots, \boldsymbol{c}^n \in \mathbb{R}^n,$$

entonces

$$\left[\begin{array}{c|c|c} & & \\ \boldsymbol{A}\boldsymbol{c}^1 & \cdots & \boldsymbol{A}\boldsymbol{c}^n \\ & & \end{array}\right] = \boldsymbol{A}\left[\begin{array}{c|c|c} & & \\ \boldsymbol{c}^1 & \cdots & \boldsymbol{c}^n \\ & & \end{array}\right] = \boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{I} = \left[\begin{array}{c|c|c} & & \\ \boldsymbol{e}^1 & \cdots & \boldsymbol{e}^n \\ & & \end{array}\right],$$

donde las columnas $\boldsymbol{e}^1, \ldots, \boldsymbol{e}^n$ de $\boldsymbol{I}$ son los vectores de la base canónica de $\mathbb{R}^n$. Por lo tanto, $\boldsymbol{A}^{-1}$ puede calcularse columna por columna resolviendo:

$$\boxed{\boldsymbol{A}\boldsymbol{c}^i = \boldsymbol{e}^i, \qquad i = 1, \ldots, n.}$$

Dificultades numéricas

Los siguientes aspectos deben tenerse en cuenta al diseñar un algoritmo para resolver un sistema de ecuaciones lineales:

- Costo operacional. El tiempo de cálculo del computador necesario para resolver el sistema debe ser lo menor posible.

  Una medida standard del costo operacional es la cantidad de operaciones aritméticas $(+,\ -,\ *,\ /)$ que requiere un algoritmo. Éste usualmente se expresa en flop *(floating point operations)*.

- Costo de almacenamiento. La cantidad de posiciones de memoria que requiere el computador para ejecutar un algoritmo (representación de los datos, variables auxiliares, etc.) también debe ser la menor posible.

- Precisión de los resultados. Los algoritmos deben ser estables, en el sentido de amplificar lo menos posible los errores de los datos y los de redondeo.

Costo operacional

- Los sistemas que aparecen en muchas aplicaciones son de gran tamaño. Un sistema de 1000×1000 hoy se considera de tamaño moderado y en algunas aplicaciones deben resolverse sistemas de ecuaciones con cientos de miles de incógnitas.

- Hay métodos que en teoría permiten resolver cualquier sistema de ecuaciones lineales, pero que en la práctica requieren tiempos de cálculo prohibitivos.

- **Mal ejemplo: Regla de Cramer.** Este procedimiento permite calcular explícitamente la solución de un sistema $\boldsymbol{Ax} = \boldsymbol{b}$ mediante:

$$x_i = \frac{\det(\boldsymbol{A}_i)}{\det(\boldsymbol{A})}, \qquad i = 1, \ldots, n,$$

  donde $\boldsymbol{A}_i$ es la matriz que se obtiene a partir de $\boldsymbol{A}$ reemplazándo en ésta su columna i-ésima por el segundo miembro $\boldsymbol{b}$.

  Si los determinantes se calculan mediante la fórmula recursiva usual de desarrollo por fila (o por columna), el costo operacional de la Regla de Cramer es de aproximadamente $(n+1)!$ flop.

Costo operacional (cont.)

- **Buen ejemplo: Método de Eliminación Gaussiana.** Este procedimiento se basa en el método algebraico de **transformaciones elementales**. Su costo operacional veremos que es de aproximadamente $\frac{2}{3}n^3$ flop.
- **Comparación:**

  En un computador de 1 Gflop (10^9 flop) por segundo:

n	10	15	20	100	1000	2000
Regla de Cramer						
flop	4×10^7	2×10^{13}	5×10^{19}	10^{160}	"∞"	"∞"
tiempo	0.04 s	5.5 horas	1500 años	"∞"	"∞"	"∞"
Eliminación Gaussiana						
flop	666	2250	5333	7×10^5	7×10^8	5×10^9
tiempo	0. s	0. s	0. s	0 s	0.73 s	4.88 s

Costo de almacenamiento

- En muchas aplicaciones los sistemas de ecuaciones lineales que deben resolverse involucran matrices de gran tamaño, pero tales que la mayor parte de sus entradas son nulas.

  Estas matrices se denominan **dispersas** o **ralas** (en inglés y en MATLAB, **sparse**) y existen técnicas para almacenarlas que sólo requieren una cantidad de posiciones de memoria aproximadamente igual al número de entradas no nulas de la matriz.

- Los métodos algebraicos usuales (por ejemplo el de **transformaciones elementales**) requieren modificar la matriz original del sistema y, muchas veces, destruyen el carácter disperso de la misma.

- Para evitar esto, estudiaremos también otros procedimientos (**métodos iterativos**) que no modifican la matriz del sistema, por lo que resultarán más convenientes desde el punto de vista del costo de almacenamiento.

  Estos métodos no se basan en calcular la solución exacta del sistema, sino en construir iterativamente aproximaciones cada vez mejores de la misma.

Precisión de los resultados

- La resolución de un sistema de ecuaciones lineales en el computador involucra la propagación de **errores en los datos** y **errores de redondeo**. Por ello:
  1. Hay que disponer de alguna técnica que permita predecir cuando la resolución de un sistema de ecuaciones puede propagar drásticamente estos errores.
  2. Hay que diseñar métodos numéricos estables, que reduzcan la propagación de los errores de redondeo tanto como sea posible.
  3. Hay que diseñar técnicas computacionales que nos permitan, después de calcular la solución de un sistema de ecuaciones, estimar a posteriori la precisión de la solución calculada. Es decir, testear si el error con el que se la calculó está por debajo de una tolerancia aceptable.