Apunte de Métodos Numéricos

Franco Frizzo

Diciembre de 2016 - Abril de 2018*

Índice

1. Resolución directa de sistemas de ecuaciones lineales **2**

1.1. Resolución de sistemas sencillos 2

1.2. Eliminación gaussiana 3

1.2.1. Pivoteo 5

1.3. Factorización LU 6

2. Normas matriciales, condicionamiento y estabilidad numérica **9**

3. Factorización de matrices **11**

3.1. Factorización de Cholesky 11

3.2. Factorización QR 14

3.2.1. Método de Householder (Reflexiones) 14

3.2.2. Método de Givens (Rotaciones) 16

3.3. Descomposición en valores singulares 18

3.3.1. Demostración de la existencia 18

4. Resolución iterativa de sistemas de ecuaciones lineales **21**

4.1. Métodos iterativos 21

4.2. Análisis de convergencia 21

4.3. Método de Jacobi 22

4.4. Método de Gauss-Seidel 23

5. Cálculo de autovalores **24**

5.1. Método de la potencia 24

5.2. Método de la potencia inversa 25

6. Interpolación polinómica **26**

*Sí, eso fue lo que tardé en preparar el final

6.1. Interpolación con polinomio único . . . 26
6.1.1. Polinomio interpolador de Lagrange . . . 26
6.1.2. Diferencias divididas . . . 27
6.1.3. Método de Neville . . . 28
6.2. Interpolación fragmentaria . . . 29
6.2.1. Interpolación fragmentaria lineal . . . 29
6.2.2. Interpolación fragmentaria cuadrática . . . 30
6.2.3. *Splines* cúbicos . . . 30

7. Cuadrados mínimos lineales **32**
7.1. Ecuaciones normales . . . 33
7.2. Resolución con factorización QR . . . 34
7.3. Resolución con descomposición en valores singulares . . . 36

8. Ceros de funciones **38**
8.1. Orden de convergencia . . . 38
8.2. Criterios de parada . . . 38
8.3. Método de la bisección . . . 39
8.4. Algoritmos de punto fijo . . . 40
8.5. Método de Newton . . . 41
8.6. Método de la secante . . . 42
8.7. Método *regula falsi* . . . 43

1. Resolución directa de sistemas de ecuaciones lineales

Un **sistema de ecuaciones lineales** es un conjunto de ecuaciones de la forma

$$\begin{cases} a_{1,1}x_1 & + & a_{1,2}x_2 & + & \cdots & + & a_{1,n}x_n & = & b_1 \\ a_{2,1}x_1 & + & a_{2,2}x_2 & + & \cdots & + & a_{2,n}x_n & = & b_2 \\ \vdots & + & \vdots & + & & + & \vdots & = & \vdots \\ a_{n,1}x_1 & + & a_{n,2}x_2 & + & \cdots & + & a_{n,n}x_n & = & b_n \end{cases}$$

donde los $a_{i,j}$ y los b_i son números reales.

Los sistemas de ecuaciones lineales admiten también la representación matricial

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

donde $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$. Esta representación nos facilitará tanto su comprensión como su tratamiento computacional.

Las variables $x_1, \ldots, x_n$ se denominan las **incógnitas** del sistema. Una **solución** de un sistema de ecuaciones lineales es un conjunto de valores para las incógnitas que satisfacen simultáneamente todas las ecuaciones.

Un sistema de ecuaciones lineales puede no tener solución, tener solución única, o tener infinitas soluciones. Si la matriz asociada al sistema es inversible (o, lo que es lo mismo, sus columnas son linealmente independientes) la solución será única. Si, por el contrario, la matriz es singular, podría pasar que el sistema no tenga solución o que tenga infinitas de ellas.

Un sistema de la forma $\mathbf{A} \cdot \mathbf{x} = \mathbf{0}$ se denomina **homogéneo**. Las soluciones de un sistema homogéneo forman un subespacio vectorial. Además, el conjunto de soluciones de cualquier sistema $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ puede obtenerse sumando una solución particular del mismo a las soluciones del sistema homogéneo asociado, $\mathbf{A} \cdot \mathbf{x} = \mathbf{0}$.

En esta sección, expondremos dos métodos para obtener computacionalmente la solución de un sistema de ecuaciones lineales, en el caso de que esta exista y sea única. La resolución de estos sistemas es un problema importante y frecuente en el análisis numérico, ya que estos son útiles a la hora de modelar matemáticamente el comportamiento de problemas provenientes de diversas disciplinas, como la física y la ingeniería, para ser tratados en forma computacional. En muchos de estos modelos aparecen ecuaciones que, o bien son lineales, o pueden aproximarse bien mediante ecuaciones lineales. Estos sistemas también aparecen en la resolución de ecuaciones diferenciales, que son cruciales para muchas disciplinas.

1.1. Resolución de sistemas sencillos

Decimos que una matriz $\mathbf{A} \in \mathbb{R}^{n \times n}$ es:

- **diagonal**, si $(\forall\ 1 \leq i, j \leq n)\ i \neq j \Rightarrow a_{i,j} = 0$; es decir, todos los elementos fuera de la diagonal son nulos.
- **triangular inferior**, si $(\forall\ 1 \leq i, j \leq n)\ i < j \Rightarrow a_{i,j} = 0$; es decir, todos los elementos por encima de la diagonal son nulos.
- **triangular superior**, si $(\forall\ 1 \leq i, j \leq n)\ i > j \Rightarrow a_{i,j} = 0$; es decir, todos los elementos por debajo de la diagonal son nulos.

Existen ciertos sistemas de ecuaciones que se pueden resolver algorítmicamente de forma sencilla. Por ejemplo, si el sistema tiene asociada una matriz diagonal, entonces sus ecuaciones son de la

forma

$$\begin{cases} a_{1,1}x_1 = b_1 \\ a_{2,2}x_2 = b_2 \\ \vdots \\ a_{n,n}x_n = b_n, \end{cases}$$

de donde pueden despejarse fácilmente valores para cada x_i. Se puede notar que existirá una solución única si y solo si $(\forall\ 1 \leq i \leq n)\ a_{i,i} \neq 0$, condición que equivale a que la matriz sea inversible. En tal caso, el costo de hallarla será lineal en la cantidad de ecuaciones, es decir, $\mathrm{O}(n)$.

Por otra parte, si la matriz del sistema es triangular superior, puede aplicarse un algoritmo llamado **sustitución hacia atrás** (*backward substitution*). El mismo consiste en comenzar a partir de la última ecuación, que tiene la forma

$$a_{n,n}x_n = b_n,$$

de donde es sencillo despejar un valor para x_n. Luego, en la ecuación anterior, que tiene la forma

$$a_{n-1,n-1}x_{n-1} + a_{n-1,n}x_n = b_{n-1},$$

puede reemplazarse el valor hallado para x_n, obteniendo así un valor para x_{n-1}. Este procedimiento se repite con las ecuaciones superiores, hasta haber despejado todas las incógnitas del sistema. Al igual que en el caso anterior, esto será posible si y solo si todos los elementos de la diagonal son no nulos; en caso contrario, el sistema no tiene solución única.

Así, la solución hallada mediante el algoritmo de sustitución hacia atrás puede describirse mediante las siguientes ecuaciones:

$$\begin{cases} x_n = \dfrac{b_n}{a_{n,n}} \\ x_{n-1} = \dfrac{b_{n-1} - a_{n-1,n} \cdot x_n}{a_{n-1,n-1}} \\ \vdots \\ x_1 = \dfrac{b_1 - a_{1,2} \cdot x_2 - \cdots - a_{1,n} \cdot x_n}{a_{1,1}} \end{cases}$$

Si la matriz del sistema es triangular inferior, se puede modificar el algoritmo para comenzar despejando en la primera ecuación. A este procedimiento se lo conoce como **sustitución hacia adelante** (*forward substitution*).

Tanto el algoritmo de sustitución hacia adelante como el algoritmo de sustitución hacia atrás tienen un costo cuadrático en la cantidad de ecuaciones del sistema ($\mathrm{O}(n^2)$).

1.2. Eliminación gaussiana

Decimos que dos sistemas de ecuaciones son **equivalentes** si tienen el mismo conjunto de soluciones. El algoritmo de **eliminación gaussiana** consiste en transformar un sistema de ecuaciones cualquiera en otro equivalente, pero que se encuentre en forma triangular superior. De esta forma, puede aplicarse el procedimiento de sustitución hacia atrás para encontrar las soluciones del sistema original.

Para transformar un sistema en otro equivalente, se aplica una serie de operaciones sobre las ecuaciones del mismo, que no modifican su conjunto de soluciones. Estas operaciones son las siguientes:

(1) Intercambiar el orden de dos ecuaciones.

(2) Multiplicar una ecuación por una constante $\lambda \in \mathbb{R}$ no nula.

(3) Sumar a una ecuación el resultado de multiplicar otra por una constante $\lambda \in \mathbb{R}$.

Con el sistema en forma matricial, dichas operaciones entre ecuaciones se traducen a operaciones entre filas de la matriz, y pueden representarse como el producto a izquierda por ciertas matrices particulares, llamadas **matrices elementales**. Una matriz elemental es cualquiera de las siguientes:

(1) Una matriz $\mathbf{P}$, obtenida de permutar filas o columnas de la identidad; a esta matriz se la llama **matriz de permutación**. Si $\mathbf{A}$ es una matriz cualquiera, entonces $\mathbf{P} \cdot \mathbf{A}$ es una permutación de las filas de $\mathbf{A}$, y $\mathbf{A} \cdot \mathbf{P}$ es una permutación de las columnas de $\mathbf{A}$.

(2) Una matriz $\mathbf{E_1} = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & \lambda & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}$, obtenida de cambiar el 1 de la i-ésima fila de la identidad por un valor λ cualquiera. Si $\mathbf{A}$ es una matriz cualquiera, multiplicar a izquierda (a derecha) por $\mathbf{E_1}$ multiplica por λ la i-ésima fila (columna) de $\mathbf{A}$.

(3) Una matriz $\mathbf{E_2} = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & 1 & \cdots & 0 \\ \vdots & \lambda & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}$, obtenida de cambiar un 0 de la identidad, ubicado en la posición i, j, por un valor λ cualquiera. Si $\mathbf{A}$ es una matriz cualquiera, multiplicar a izquierda (a derecha) por $\mathbf{E_2}$ la suma λ veces la fila j-ésima (la columna i-ésima) a la fila i-ésima (la columna j-ésima) de $\mathbf{A}$.

Las matrices elementales son inversibles, y el producto por la inversa de una matriz elemental revierte la operación que esta realiza sobre las filas de una matriz cualquiera.

El algoritmo de eliminación gaussiana opera sobre la **matriz extendida** del sistema, que es la matriz

$$\tilde{\mathbf{A}} = \left[\begin{array}{cccc|c} a_{1,1} & a_{1,2} & \dots & a_{1,n} & b_1 \\ a_{2,1} & a_{2,2} & \dots & a_{2,n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n,1} & a_{n,2} & \dots & a_{n,n} & b_n \end{array}\right].$$

Como cada iteración efectúa cambios sobre la matriz $\tilde{\mathbf{A}}$, utilizaremos la notación $\tilde{\mathbf{A}}^{(k)}$ para referirnos al resultado luego de la k-ésima iteración del proceso, mientras que con $a_{i,j}^{(k)}$ y $b_i^{(k)}$ haremos referencia a cada uno de sus elementos.

La idea del algoritmo es aplicar operaciones de filas en forma consecutiva hasta llevar $\tilde{\mathbf{A}}$ a una forma triangular superior. El método itera sobre las columnas de la matriz, buscando en cada paso colocar ceros en los lugares que se encuentran debajo de la diagonal. Es decir, el invariante del algoritmo es que, en la k-ésima iteración, todas las columnas hasta la $k-1$ tienen ceros debajo de la diagonal, asegurando que, tras k iteraciones, la matriz quedará en forma triangular superior.

Más precisamente, en la k-ésima iteración, se resta a todas las filas a partir de la $k+1$ un múltiplo de la fila k-ésima, con un factor $m_i^{(k)}$ elegido convenientemente para cada fila i. Esto significa que, para todo $i = k+1, \dots, n$, los coeficientes de la fila i-ésima quedarán

$$a_{i,j}^{(k)} = a_{i,j}^{(k-1)} - m_i^{(k)} \cdot a_{k,j}^{(k-1)},$$

y como se quiere dejar un 0 en la columna k-ésima, es decir, $a_{i,k}^{(k)} = 0$, debe tomarse, para cada fila i, el multiplicador

$$m_{i,k} = \frac{a_{i,k}^{(k-1)}}{a_{k,k}^{(k-1)}}.$$

Es importante notar que solo es posible efectuar el k-ésimo paso del algoritmo si $a_{k,k}^{(k-1)} = a_{k,k} \neq 0$, es decir, si el k-ésimo elemento de la diagonal no es nulo. Se puede relajar ligeramente esta hipótesis, admitiendo $a_{k,k}^{(k-1)} = 0$, si todos los demás elementos de esa columna debajo de la diagonal también son nulos; en ese caso directamente se puede omitir la k-ésima iteración. En cualquier otro caso, el algoritmo falla.

Como cada paso del algoritmo coloca ceros debajo de la diagonal en la columna k-ésima, y no modifica los ceros que fueron ubicados en otras columnas por los pasos previos, la matriz $\tilde{\mathbf{A}}^{(n-1)}$ que se obtiene tras $n-1$ iteraciones del proceso es triangular superior.

A continuación se presenta el algoritmo en forma de pseudocódigo. Analizando el mismo, se puede ver que su complejidad es de orden cúbico ($\mathrm{O}(n^3)$).

Algoritmo de eliminación gaussiana

Entrada: $\mathbf{A} \in \mathbb{R}^{n \times n}$ y $\mathbf{b} \in \mathbb{R}^n$.
Salida: $\mathbf{x} \in \mathbb{R}^n$ tal que $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$.

```
para k = 1, ..., n − 1 hacer
    si a_{k,k} ≠ 0 entonces
        para i = k + 1, ..., n hacer
            m_{i,k} ← a_{i,k} / a_{k,k}
            F_i ← F_i − m_{i,k} · F_k
    en otro caso
        si a_{i,k} ≠ 0 para algún i ∈ k + 1, ..., n entonces
            fallar
```

La notación F_i representa aquí la i-ésima fila de la matriz ampliada del sistema.

1.2.1. Pivoteo

En cada paso del algoritmo de eliminación gaussiana, llamamos **pivote** al elemento de la diagonal sobre el cual estamos trabajando (en el paso k-ésimo, el pivote es $a_{k,k}^{(k-1)}$). La técnica de **pivoteo** consiste en realizar operaciones sobre la matriz, intercambiando sus filas (o sus columnas) para modificar el pivote sin alterar las soluciones del sistema asociado. Esto puede ser deseable por dos razones:

1. Como se vio anteriormente, si en algún paso del algoritmo el pivote es cero pero hay un elemento no nulo debajo de él, el procedimiento falla. Gracias al pivoteo, puede cambiarse el pivote por un elemento no nulo y proseguir con el algoritmo. Se puede demostrar que toda matriz admite eliminación gaussiana con pivoteo, incluso aquellas para las cuales falla la versión sin pivoteo.
2. Debido a que la computadora trabaja con arimética finita (una aproximación discreta de los números reales), durante las operaciones se suele perder precisión en los resultados. Es deseable que los algoritmos eviten que estos errores se amplifiquen durante iteraciones posteriores, propiedad que se conoce como *estabilidad numérica*. En el caso de la eliminación gaussiana, la

estabilidad numérica mejora cuando el pivote tiene un valor absoluto grande, por lo que se puede utilizar pivoteo para intercambiarlo por uno de mayor valor absoluto y así mejorar la precisión de los resultados obtenidos.

Si se quiere aplicar eliminación gaussiana con pivoteo, es necesario determinar un criterio para elegir el pivote en cada iteración. Existen principalmente dos formas de hacerlo.

- El **pivoteo parcial** consiste en intercambiar el pivote por un elemento de la misma columna, considerando el propio pivote y los elementos que se encuentran por debajo de él, y eligiendo el de mayor valor absoluto. Por lo tanto, se lleva a cabo intercambiando dos filas de la matriz. Esta técnica solo requiere considerar, a lo sumo, n posibles valores para el pivote. Garantiza que se elegirá un pivote no nulo (a menos que el elemento de la diagonal y todos los que estén debajo sean nulos), y permite mejorar la estabilidad numérica.
- El **pivoteo completo** considera toda la submatriz que falta reducir, eligiendo como pivote al elemento de mayor valor absoluto. Se lleva a cabo intercambiando dos filas y dos columnas de la matriz (intercambiar columnas equivale a alterar el orden de las variables del sistema, por lo que los intercambios de columnas deberán ser revertidos en la solución que se obtenga). Esta técnica permite mejorar aún más la estabilidad numérica, pero es poco utilizada por resultar considerablemente menos eficiente, ya que la búsqueda del pivote tiene una complejidad cuadrática.

1.3. Factorización LU

Dada una matriz $\mathbf{A} \in \mathbb{R}^{n \times n}$, una factorización LU para $\mathbf{A}$ es una escritura:

$$\mathbf{A} = \mathbf{L} \cdot \mathbf{U}$$

donde $\mathbf{L} \in \mathbb{R}^{n \times n}$ es triangular inferior con unos en la diagonal y $\mathbf{U} \in \mathbb{R}^{n \times n}$ es triangular superior.

Dada la factorización LU de una matriz $\mathbf{A}$, resolver un sistema de ecuaciones $\mathbf{A} \cdot \mathbf{x} = b$ asociado resulta sencillo. En primer lugar, se puede notar que $\mathbf{L}$ es inversible, ya que es triangular inferior sin ceros en la diagonal. Entonces,

$$\begin{aligned} \mathbf{A} \cdot \mathbf{x} &= \mathbf{b} && \text{sii} \\ \mathbf{L} \cdot \mathbf{U} \cdot \mathbf{x} &= \mathbf{b} && \text{sii} \\ \mathbf{U} \cdot \mathbf{x} &= \mathbf{L}^{-1} \cdot \mathbf{b} && \text{sii} \\ \mathbf{U} \cdot \mathbf{x} &= \mathbf{y} \end{aligned}$$

donde $\mathbf{y} = \mathbf{L}^{-1} \cdot \mathbf{b}$ o, lo que es lo mismo, $\mathbf{y}$ es la única solución del sistema de ecuaciones $\mathbf{L} \cdot \mathbf{y} = \mathbf{b}$. Por lo tanto, basta con resolver consecutivamente dos sistemas de ecuaciones:

- $\mathbf{L} \cdot \mathbf{y} = \mathbf{b}$,
- $\mathbf{U} \cdot \mathbf{x} = \mathbf{y}$.

Como ambas matrices son triangulares, los sistemas se pueden resolver con un costo $\mathrm{O}(n^2)$, usando los algoritmos de sustitución hacia adelante y hacia atrás, respectivamente.

Hallar la factorización LU, por su parte, tiene una complejidad de $\mathrm{O}(n^3)$ (como se verá luego con más detalle). Por lo tanto, si se la quiere utilizar para resolver un sistema desde cero, el costo es el mismo que el de aplicar eliminación gaussiana. La ventaja de la factorización LU aparece si se quieren resolver varios sistemas de ecuaciones que comparten la matriz asociada; es decir, si se tiene una matriz $\mathbf{A}$ y una familia de vectores $\mathbf{b}_1, \ldots, \mathbf{b}_k$, y se quieren hallar soluciones para todos los sistemas $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}_i$ con $i \in \{1, \ldots, k\}$. En este caso, usando eliminación gaussiana, habría que pagar un costo $\mathrm{O}(n^3)$ para resolver cada sistema. Es mucho más eficiente calcular primero la factorización LU de $\mathbf{A}$, pagando el costo $\mathrm{O}(n^3)$ solo una vez, y luego resolver cada sistema con un costo $\mathrm{O}(n^2)$.

La factorización LU está intrínsecamente relacionada al algoritmo de eliminación gaussiana. Esto se debe a que la matriz $\mathbf{U}$ puede interpretarse como el resultado de aplicar eliminación gaussiana (sin pivoteo) sobre $\mathbf{A}$, mientras que $\mathbf{L}$ es el producto de las matrices elementales que representan a las operaciones involucradas en el proceso.

Recordemos lo que sucede en el paso k-ésimo del algoritmo de eliminación gaussiana: para cada $i \in \{k+1, \dots, n\}$, se calcula un multiplicador $m_{i,k}$ y se realiza la operación de filas $\mathrm{F}_i \leftarrow \mathrm{F}_i - m_{i,k} \cdot \mathrm{F}_k$.[1] Esta operación se puede representar como el producto a izquierda por una matriz elemental $\mathbf{M}_{i,k}$, que es similar a la identidad, pero tiene el valor $-m_{i,k}$ en la posición i, k. Por lo tanto, considerando todas las operaciones de filas, podemos sintetizar el k-ésimo paso de la siguiente manera (por simplicidad trabajaremos con la matriz $\mathbf{A}$, en lugar de usar la matriz extendida $\tilde{\mathbf{A}}$):

$$\begin{aligned}\mathbf{A}^{(k)} &= \mathbf{M}_{n,k} \cdot \ldots \cdot \mathbf{M}_{k+1,k} \cdot \mathbf{A}^{(k-1)} \\ &= \mathbf{M}_k \cdot \mathbf{A}^{(k-1)}.\end{aligned}$$

La matriz $\mathbf{M}_k$, que es el producto de las matrices elementales de cada operación de filas, es similar a la identidad pero en la k-ésima columna tiene, debajo de la diagonal, los valores $-m_{k+1,k}, \dots, -m_{n,k}$. Además, $\mathbf{M}_k$ es inversible; su inversa es muy parecida, pero los $m_{i,k}$ no están cambiados de signo.

De forma similar, podemos sintetizar todo el algoritmo de eliminación gaussiana como

$$\begin{aligned}\mathbf{A}^{(n-1)} &= \mathbf{M}_{n-1} \cdot \ldots \cdot \mathbf{M}_1 \cdot \mathbf{A}^{(0)} \\ &= \mathbf{M} \cdot \mathbf{A}^{(0)} \\ &= \mathbf{M} \cdot \mathbf{A}.\end{aligned}$$

Llamaremos $\mathbf{U}$ a la matriz $\mathbf{A}^{(n-1)}$, que es triangular superior por ser el resultado del algoritmo de eliminación gaussiana. Por su parte, la matriz $\mathbf{M}$ es triangular inferior, con unos en la diagonal, y cada posición i, k debajo de la diagonal contiene el valor $-m_{i,k}$. Se trata de una matriz inversible; su inversa es similar, pero los $m_{i,k}$ no aparecen cambiados de signo. Como

$$\mathbf{U} = \mathbf{M} \cdot \mathbf{A},$$

si llamamos $\mathbf{L} = \mathbf{M}^{-1}$, tenemos que

$$\mathbf{L} \cdot \mathbf{U} = \mathbf{A},$$

con $\mathbf{L}$ triangular inferior con unos en la diagonal. Se trata, por lo tanto, de una factorización LU para $\mathbf{A}$.

De lo anterior se puede concluir que si una matriz admite el algoritmo de eliminación gaussiana sin pivoteo, entonces tiene factorización LU: si se puede ejecutar el algoritmo, la factorización se obtiene considerando como $\mathbf{U}$ a la matriz triangulada y construyendo $\mathbf{L}$ a partir de los multiplicadores calculados en el proceso. De allí que la complejidad de obtener la factorización también sea de orden cúbico. Por otro lado, si una matriz posee factorización LU, entonces admite el algoritmo de eliminación gaussiana sin pivoteo; a partir de la matriz $\mathbf{L}$ pueden recuperarse los multiplicadores de cada paso del algoritmo, mientras que $\mathbf{U}$ es la matriz triangulada que se obtiene como resultado final.

No todas las matrices admiten factorización LU. A modo de ejemplo, consideremos una matriz $\mathbf{A} \in \mathbb{R}^{n \times n}$. Si tiene factorización LU, entonces existen $a, b, c, d \in \mathbb{R}$ tales que

$$\mathbf{A} = \mathbf{L} \cdot \mathbf{U} = \begin{bmatrix} 1 & 0 \\ a & 1 \end{bmatrix} \cdot \begin{bmatrix} b & c \\ 0 & d \end{bmatrix} = \begin{bmatrix} b & c \\ ab & ac + d \end{bmatrix}.$$

[1] En el caso en que para cierta columna k, el elemento de la diagonal y todos los que están debajo de él sean nulos, el algoritmo no hace nada; en ese caso puede considerarse que todos los $m_{i,k}$ valen 0.

La matriz $\begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$ no cumple esta propiedad, por lo que no puede tener factorización LU.

Sin embargo, teniendo en cuenta que toda matriz admite eliminación gaussiana con pivoteo, puede considerarse la matriz inversible $\mathbf{P}$ que resulta de multiplicar todas las matrices elementales que representan las operaciones del pivoteo. Así $\mathbf{P} \cdot \mathbf{A} = \mathbf{A'}$, donde $\mathbf{A'}$ admite eliminación gaussiana sin pivoteo, y por lo tanto, tiene factorización LU. Luego, podemos escribir

$$\begin{aligned} \mathbf{A} &= \mathbf{P}^{-1} \cdot \mathbf{A'} \\ &= \mathbf{P}^{-1} \cdot \mathbf{L} \cdot \mathbf{U}. \end{aligned}$$

Esta escritura, que siempre existe, se conoce como factorización PLU de $\mathbf{A}$.

Con respecto a la unicidad de la factorización LU, no siempre es posible garantizarla (por ejemplo, la matriz nula tiene infinitas factorizaciones LU distintas). Sin embargo, si $\mathbf{A}$ es inversible y tiene factorización LU, entonces se puede demostrar que esta es única.

Los siguientes dos resultados, que se enuncian sin demostración, hacen referencia a la existencia de factorización LU para ciertos tipos particulares de matrices:

- Las matrices estrictamente diagonal-dominantes por filas, es decir, aquellas que cumplen que $|a_{i,i}| > \sum_{j=1, j\neq i} |a_{i,j}|$ para todo $i \in \{1, \ldots, n\}$, siempre admiten factorización LU.
- Las matrices inversibles admiten factorización LU si y solo si todos sus menores principales son no nulos. Los menores principales de una matriz son los determinantes de sus submatrices principales, es decir, aquellas submatrices cuadradas que contienen su esquina superior izquierda.

2. Normas matriciales, condicionamiento y estabilidad numérica

A la hora de resolver problemas que involucran números reales utilizando la computadora, siempre debe tenerse en cuenta que el abordaje de los mismos es **numérico**, es decir, opera tan solo con aproximaciones de los números reales, dentro de lo permitido por la aritmética finita de la computadora. Esto produce, inevitablemente, errores de redondeo que pueden ocasionar pérdida de exactitud en los resultados; por lo tanto, es importante tener cuidado en evitar que dichos errores se propaguen de formas no deseadas. En esta breve sección presentaremos algunos conceptos que serán útiles más adelante para estudiar la forma en que distintos métodos pueden verse afectados por errores de tipo numérico.

Una **norma matricial** es una extensión del concepto de norma vectorial a matrices. Se trata de una función Φ que asigna a cada matriz un escalar (para nosotros, siempre será un real) no negativo, y que se anula solamente en la matriz $\mathbf{0}$; además, preserva el producto por escalares ($\Phi(\lambda \cdot \mathbf{A}) = |\lambda| \cdot \Phi(\mathbf{A})$) y cumple la desigualdad triangular ($\Phi(\mathbf{A} + \mathbf{B}) \leq \Phi(\mathbf{A}) + \Phi(\mathbf{B})$).

Existen distintos tipos de normas matriciales. Nos interesaremos especialmente en las **normas inducidas** por normas vectoriales. Si φ es una norma vectorial, definimos la norma matricial inducida por φ como

$$\Phi(\mathbf{A}) = \text{máx}\{\varphi(\mathbf{A} \cdot \mathbf{v}) : \varphi(\mathbf{v}) = 1\}.$$

Las siguientes dos propiedades valen para cualquier norma matricial inducida Φ:

(i) Φ es *submultiplicativa*: $\Phi(\mathbf{A} \cdot \mathbf{B}) \leq \Phi(\mathbf{A}) \cdot \Phi(\mathbf{B})$.

(ii) Φ es *consistente*: $\Phi(\mathbf{A} \cdot \mathbf{v}) \leq \Phi(\mathbf{A}) \cdot \varphi(\mathbf{v})$.

A continuación se enumeran las tres normas vectoriales más comunes y sus correspondientes normas matriciales.

- **Norma 1** o norma del taxista:

$$\|(v_1, \ldots, v_n)\|_1 = |v_1| + \cdots + |v_n|.$$

  La norma matricial inducida cumple que $\|\mathbf{A}\|_1 = \text{máx}_j \|\text{col}_j(\mathbf{A})\|_1$.

- **Norma 2** o norma euclídea:[2]

$$\|(v_1, \ldots, v_n)\|_2 = \sqrt{(v_1)^2 + \cdots + (v_n)^2}.$$

- **Norma infinito** o norma del máximo:

$$\|(v_1, \ldots, v_n)\|_\infty = \max_{1 \leq i \leq n} |v_i|.$$

  La norma matricial inducida cumple que $\|\mathbf{A}\|_\infty = \text{máx}_i \|\text{fil}_i(\mathbf{A})\|_1$.

Las normas matriciales nos brindan herramientas para caracterizar sistemas de ecuaciones problemáticos, donde pequeños errores numéricos pueden magnificarse y producir soluciones considerablemente inexactas. Los sistemas que presentan este tipo de inconvenientes se dice que están **mal condicionados**.

A modo de ejemplo, se puede pensar en un sistema de dos ecuaciones con dos variables. Cada ecuación del sistema puede pensarse interpretarse como una recta en el plano; una solución del sistema es un punto de intersección entre ambas rectas. Consideremos un sistema como el de la figura 1, donde las rectas son casi paralelas.

[2] En general, para $p \in \mathbb{N}$, puede definirse la *norma p* como $\|(v_1, \ldots, v_n)\|_p = \sqrt[p]{(v_1)^p + \cdots + (v_n)^p}$

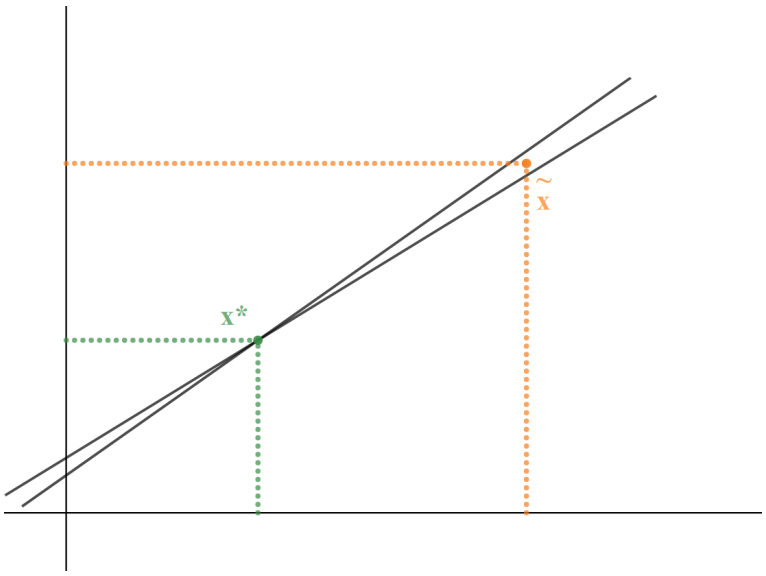

Figura 1: Sistema de ecuaciones mal condicionado.

Si bien el sistema tiene solución única $\mathbf{x}^*$, un pequeño error numérico al resolver el sistema puede producir un resultado como $\tilde{\mathbf{x}}$, donde las rectas también están muy próximas, pero que se encuentra a una distancia considerable de $\mathbf{x}^*$. Si consideramos la formulación matricial del sistema $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$, entonces $\tilde{\mathbf{x}}$ es solución de un sistema $\mathbf{A} \cdot \mathbf{x} = \tilde{\mathbf{b}}$ con $\tilde{\mathbf{b}}$ muy próximo a $\mathbf{b}$. Es decir, un pequeño error en $\mathbf{b}$ se tradujo en un error considerablemente más grande al obtener la solución $\mathbf{x}^*$.

Supongamos que tenemos una matriz $\mathbf{A}$ inversible. Considerando una norma vectorial $\|\bullet\|$ cualquiera y su norma matricial inducida, podemos demostrar las siguientes dos propiedades, que establecen una cota para el error que puede generarse en la solución $\mathbf{x}^*$ (o error hacia adelante) de un sistema $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ a partir de un error en la entrada $\mathbf{b}$ (o error hacia atrás):

(i) $\|\mathbf{x}^* - \tilde{\mathbf{x}}\| \leq \|\mathbf{b} - \tilde{\mathbf{b}}\| \cdot \|\mathbf{A}^{-1}\|$.

(ii) $\dfrac{\|\mathbf{x}^* - \tilde{\mathbf{x}}\|}{\|\mathbf{x}^*\|} \leq \dfrac{\|\mathbf{b} - \tilde{\mathbf{b}}\|}{\|\mathbf{b}\|} \cdot \|\mathbf{A}\| \cdot \|\mathbf{A}^{-1}\|$.

En la segunda propiedad se observa que el factor según el cual un error relativo en las entradas puede magnificarse al producir un error relativo en las soluciones es

$$\kappa(\mathbf{A}) = \|\mathbf{A}\| \cdot \|\mathbf{A}^{-1}\|.$$

A este valor se lo conoce como el **número de condición** de $\mathbf{A}$. Para cualquier matriz, se cumple que $\kappa(\mathbf{A}) \geq \kappa(\mathbf{I}) = 1$. Si el número de condición de una matriz es alto, entonces la matriz (y, en consecuencia, cualquier sistema de ecuaciones asociado a ella) está mal condicionada.

Cabe destacar que estar bien o mal condicionado es una propiedad inherente del sistema de ecuaciones que se busca resolver. Para un mismo sistema, distintos algoritmos pueden ofrecer resultados de mayor o menos precisión, según el tipo de operaciones que realizan y la magnitud de los errores que estas introducen en los datos. A la propiedad de un algoritmo de manipular los datos sin producir errores numéricos considerables se la denomina **estabilidad numérica**.

3. Factorización de matrices

En esta sección se presentarán tres maneras distintas de factorizar matrices. La primera de ellas, la **factorización de Cholesky**, solo puede aplicarse a matrices que cumplen con determinadas características; las otras dos, la **factorización QR** y la **descomposición en valores singulares** o **SVD**, son aplicables a matrices arbitrarias.

Una de las principales razones por las que estas factorizaciones resultan útiles es porque proporcionan formas de resolver sistemas de ecuaciones lineales que tienen propiedades deseables, como eficiencia, reutilizabilidad y estabilidad numérica. No obstante, las factorizaciones QR y SVD también tienen aplicaciones a la hora de resolver otros tipos de problemas numéricos, como el problema de cuadrados mínimos lineales, que es abordado en la sección 7.

3.1. Factorización de Cholesky

Decimos que una matriz $\mathbf{A} \in \mathbb{R}^{n\times n}$ es **simétrica** si $\mathbf{A} = \mathbf{A}^{\mathrm{T}}$. Si una matriz simétrica $\mathbf{A}$ es inversible y admite factorización LU, entonces también admite un tipo de factorización que llamaremos LDL. La misma consiste en una escritura

$$\mathbf{A} = \mathbf{L} \cdot \mathbf{D} \cdot \mathbf{L}^{\mathrm{T}}$$

donde $\mathbf{L}$ es triangular inferior con unos en la diagonal y $\mathbf{D}$ es una matriz diagonal.

La demostración proviene de considerar la factorización LU de $\mathbf{A}$, $\mathbf{A} = \mathbf{L} \cdot \mathbf{U}$. Como $\mathbf{L}$ tiene unos en la diagonal debe ser inversible y, como $\mathbf{A}$ es inversible, $\mathbf{U}$ tiene que serlo también; lo mismo vale para sus respectivas matrices traspuestas. Es claro que

$$\mathbf{A} \;=\; \mathbf{L} \cdot \mathbf{U} \;=\; \mathbf{L} \cdot \mathbf{U} \cdot (\mathbf{L}^{\mathrm{T}})^{-1} \cdot \mathbf{L}^{\mathrm{T}}.$$

Definimos $\mathbf{D} = \mathbf{U} \cdot (\mathbf{L}^{\mathrm{T}})^{-1}$, con lo que $\mathbf{A} = \mathbf{L} \cdot \mathbf{D} \cdot \mathbf{L}^{\mathrm{T}}$. Resta ver que $\mathbf{D}$ es diagonal. Como $\mathbf{A}$ es simétrica, tenemos que

$$\mathbf{L} \cdot \mathbf{U} = (\mathbf{L} \cdot \mathbf{U})^{\mathrm{T}} = \mathbf{U}^{\mathrm{T}} \cdot \mathbf{L}^{\mathrm{T}}.$$

Si multiplicamos a izquierda por $\mathbf{L}^{-1}$ y a derecha por $(\mathbf{L}^{\mathrm{T}})^{-1}$, obtenemos que

$$\mathbf{U} \cdot (\mathbf{L}^{\mathrm{T}})^{-1} = \mathbf{L}^{-1} \cdot \mathbf{U}^{\mathrm{T}}.$$

Ambos lados de la ecuación son iguales a $\mathbf{D}$. Del lado izquierdo de la igualdad aparece un producto entre dos matrices triangulares superiores, que es otra matriz triangular superior. Análogamente, del lado derecho se tiene una matriz triangular inferior. Es decir, $\mathbf{D}$ es simultáneamente una matriz triangular superior y triangular inferior, lo cual significa que es una matriz diagonal.

Por otro lado, decimos que una matriz $\mathbf{A} \in \mathbb{R}^{n\times n}$ es (simétrica y) **definida positiva** (s.d.p.) si es simétrica[3] y para todo $\mathbf{x} \in \mathbb{R}^n$ no nulo se cumple que $\mathbf{x}^{\mathrm{T}} \cdot \mathbf{A} \cdot \mathbf{x} > 0$. Las matrices definidas positivas siempre son inversibles, y tienen elementos positivos en su diagonal.

Alternativamente, las matrices definidas positivas pueden caracterizarse como aquellas matrices simétricas cuyos **menores principales** son todos positivos. A partir de esta caracterización, se puede demostrar que las matrices definidas positivas siempre admiten factorización LU, y por lo tanto, al ser simétricas, poseen una factorización LDL.

[3] Algunos autores definen la noción de matriz definida positiva independientemente de la matriz simétrica, mientras que otros requieren la simetría como una condición para que una matriz sea definida positiva; aquí se tomó el primer criterio, que es el adoptado por *Numerical Analysis* (9th Edition) de Richard L. Burden y J. Douglas Faires.

Dada una matriz $\mathbf{A}$, llamamos **factorización de Cholesky** a su escritura en la forma

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

donde $\mathbf{L}$ es una matriz triangular inferior con elementos positivos en la diagonal. A continuación demostraremos que una matriz es definida positiva si y solo si tiene factorización de Cholesky.

($\Rightarrow$) Consideremos una matriz definida positiva $\mathbf{A} \in \mathbb{R}^{n \times n}$. Como enunciamos anteriormente, dicha matriz debe admitir una factorización LDL, $\mathbf{A} = \mathbf{L} \cdot \mathbf{D} \cdot \mathbf{L}^{\mathrm{T}}$.

En primer lugar, se puede verificar que los elementos de la diagonal de $\mathbf{D}$ deben ser positivos. Basta considerar $i \in \{1, \dots, n\}$ cualquiera; como $\mathbf{L}^{\mathrm{T}}$ es inversible, se puede tomar $\mathbf{x} \in \mathbb{R}^n$ tal que $\mathbf{L}^{\mathrm{T}} \cdot \mathbf{x} = \mathbf{e}_i$, donde $\mathbf{e}_i$ es un vector que tiene un uno en la posición i-ésima y ceros en las posiciones restantes. Luego,

$$\begin{aligned} 0 &< \mathbf{x}^{\mathrm{T}} \cdot \mathbf{A} \cdot \mathbf{x} \\ &= \mathbf{x}^{\mathrm{T}} \cdot \mathbf{L} \cdot \mathbf{D} \cdot \mathbf{L}^{\mathrm{T}} \cdot \mathbf{x} \\ &= (\mathbf{L}^{\mathrm{T}} \cdot \mathbf{x})^{\mathrm{T}} \cdot \mathbf{D} \cdot (\mathbf{L}^{\mathrm{T}} \cdot \mathbf{x}) \\ &= \mathbf{e}_i^{\mathrm{T}} \cdot \mathbf{D} \cdot \mathbf{e}_i \\ &= d_{i,i}. \end{aligned}$$

Esto permite definir

$$\sqrt{\mathbf{D}} = \begin{bmatrix} \sqrt{d_{1,1}} & 0 & \cdots & 0 \\ 0 & \sqrt{d_{2,2}} & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & \sqrt{d_{n,n}} \end{bmatrix}$$

de modo que $\mathbf{D} = \sqrt{\mathbf{D}} \cdot \sqrt{\mathbf{D}} = \sqrt{\mathbf{D}} \cdot \sqrt{\mathbf{D}}^{\mathrm{T}}$. Así, reemplazando en la escritura LDL, obtenemos

$$\begin{aligned} \mathbf{A} &= \mathbf{L} \cdot \mathbf{D} \cdot \mathbf{L}^{\mathrm{T}} \\ &= \mathbf{L} \cdot \sqrt{\mathbf{D}} \cdot \sqrt{\mathbf{D}}^{\mathrm{T}} \cdot \mathbf{L}^{\mathrm{T}} \\ &= (\mathbf{L} \cdot \sqrt{\mathbf{D}}) \cdot (\mathbf{L} \cdot \sqrt{\mathbf{D}})^{\mathrm{T}}. \end{aligned}$$

En esta última escritura, por las características de $\mathbf{L}$ y $\sqrt{\mathbf{D}}$, la matriz $\mathbf{L} \cdot \sqrt{\mathbf{D}}$ resulta triangular inferior con elementos positivos en la diagonal; por lo tanto, se trata de una factorización de Cholesky para $\mathbf{A}$.

($\Leftarrow$) Si $\mathbf{A} = \mathbf{L} \cdot \mathbf{L}^{\mathrm{T}}$, entonces necesariamente $\mathbf{A}$ es simétrica. Basta ver que para todo $\mathbf{x} \neq \mathbf{0}$, $\mathbf{x}^{\mathrm{T}} \cdot \mathbf{A} \cdot \mathbf{x} > 0$.

$$\mathbf{x}^{\mathrm{T}} \cdot \mathbf{A} \cdot \mathbf{x} \;=\; \mathbf{x}^{\mathrm{T}} \cdot \mathbf{L} \cdot \mathbf{L}^{\mathrm{T}} \cdot \mathbf{x} \;=\; \left(\mathbf{L}^{\mathrm{T}} \cdot \mathbf{x}\right)^{\mathrm{T}} \cdot \left(\mathbf{L}^{\mathrm{T}} \cdot \mathbf{x}\right) \;=\; \|\mathbf{L}^{\mathrm{T}} \cdot \mathbf{x}\|_2^2,$$

y al ser $\mathbf{L}^{\mathrm{T}}$ inversible, $\mathbf{L}^{\mathrm{T}} \cdot \mathbf{x} \neq \mathbf{0}$, por lo que la norma necesariamente es positiva.

Una forma de computar la factorización de Cholesky de una matriz $\mathbf{A}$ surge de observar cómo se podría recuperar un elemento de $\mathbf{A}$ si ya se contara con dicha factorización. Sea $\mathbf{A} = \mathbf{L} \cdot \mathbf{L}^{\mathrm{T}}$ y consideremos, sin pérdida de generalidad, $i \geq j$; se puede notar que entonces

$$a_{i,j} = \mathrm{fil}_i(\mathbf{L}) \cdot \mathrm{col}_j(\mathbf{L}^{\mathrm{T}}) = \sum_{k=1}^{j} l_{i,k} \cdot l_{j,k}.$$

La idea es recorrer la matriz $\mathbf{L}$ de forma ordenada, por columnas, e ir utilizando la ecuación anterior para obtener, a partir de $\mathbf{A}$, los valores que corresponden en cada posición.

- Para computar la esquina superior izquierda de $\mathbf{L}$, tenemos en cuenta que $a_{1,1} = (l_{1,1})^2$. Por lo tanto,
$$l_{1,1} = \sqrt{a_{1,1}}.$$
- Para completar la primer columna, observamos que si $i > 1$, entonces $a_{i,1} = l_{i,1} \cdot l_{1,1}$. Luego,
$$l_{i,1} = \frac{a_{i,1}}{l_{1,1}}.$$
- Consideremos ahora una columna $j > 1$. Como $\mathbf{L}$ es triangular inferior, el primer elemento a computar es el de la diagonal. Podemos observar que $a_{j,j} = \sum_{k=1}^{j}(l_{j,k})^2 = \sum_{k=1}^{j-1}(l_{j,k})^2 + (l_{j,j})^2$. Entonces,
$$l_{j,j} = \sqrt{a_{j,j} - \sum_{k=1}^{j-1}(l_{j,k})^2}.$$
- Para calcular los restantes elementos, si $1 < j < i$, tenemos $a_{i,j} = \sum_{k=1}^{j} l_{i,k} \cdot l_{j,k} = \sum_{k=1}^{j-1} l_{i,k} \cdot l_{j,k} + l_{i,j} \cdot l_{j,j}$. Así, obtenemos que
$$l_{i,j} = \frac{1}{l_{j,j}} \cdot \left(a_{i,j} - \sum_{k=1}^{j-1} l_{i,k} \cdot l_{j,k} \right).$$

Si los pasos se siguen en orden, en cada uno de ellos solo se utilizan elementos de $\mathbf{L}$ que ya fueron calculados previamente. Por lo tanto, tenemos un algoritmo bien definido para obtener la factorización de Cholesky de cualquier matriz definida positiva. Además, el algoritmo determina $\mathbf{L}$ de forma unívoca partiendo de una ecuación que debe ser satisfecha por toda factorización de Cholesky de $\mathbf{A}$; esto demuestra que la factorización de Cholesky es única.

A continuación, se presenta el algoritmo escrito en forma de pseudocódigo.

Factorización de Cholesky

Entrada: $\mathbf{A} \in \mathbb{R}^{n \times n}$ definida positiva.
Salida: $\mathbf{L}$ triangular superior, con elementos positivos en la diagonal, tal que $\mathbf{A} = \mathbf{L} \cdot \mathbf{L}^{\mathrm{T}}$.

1 $l_{1,1} \leftarrow \sqrt{a_{1,1}}$
2 **para** $i = 2, \ldots, n$ **hacer**
3 $\quad l_{i,1} \leftarrow \dfrac{a_{i,1}}{l_{1,1}}$
4 **para** $j = 2, \ldots, n$ **hacer**
5 $\quad l_{j,j} \leftarrow \sqrt{a_{j,j} - \sum_{k=1}^{j-1}(l_{j,k})^2}$
6 $\quad$ **para** $i = j+1, \ldots, n$ **hacer**
7 $\quad\quad l_{i,j} \leftarrow \dfrac{1}{l_{j,j}} \cdot \left(a_{i,j} - \sum_{k=1}^{j-1} l_{i,k} \cdot l_{j,k} \right)$

Puede observarse que la complejidad del algoritmo es $\mathrm{O}(n^3)$. Si bien se trata de la misma complejidad asintótica que la de obtener una factorización LU, las constantes son mejores; en la práctica, computar una factorización de Cholesky es aproximadamente el doble de rápido que obtener una factorización LU.

3.2. Factorización QR

Dos vectores $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ se dicen **ortogonales** ($\mathbf{x} \perp \mathbf{y}$) si su producto interno $\langle \mathbf{x}, \mathbf{y} \rangle$ es 0. Un conjunto de vectores es **ortogonal** si sus elementos son ortogonales dos a dos. Un conjunto de vectores es **ortonormal** si es ortogonal y la norma de todos sus elementos es 1. Los elementos de un conjunto ortonormal siempre son linealmente independientes.

Decimos que una matriz $\mathbf{Q} \in \mathbb{R}^{n \times n}$ es **ortogonal** si sus columnas forman un conjunto ortonormal. Se puede demostrar, además, que una matriz es ortogonal si y solo si es inversible y su transpuesta es igual a su inversa, es decir, $\mathbf{Q}^{\mathrm{T}} = \mathbf{Q}^{-1}$. El producto de matrices ortogonales es también ortogonal, es decir, si $\mathbf{Q_1}$ y $\mathbf{Q_2}$ son ortogonales, $\mathbf{Q_1} \cdot \mathbf{Q_2}$ es ortogonal.

Las matrices ortogonales poseen la importante propiedad de estar asociadas a las transformaciones lineales que *preservan norma 2*. En otras palabras, una matriz $\mathbf{Q} \in \mathbb{R}^{n \times n}$ es ortogonal si y solo si, para todo $x \in \mathbb{R}^n$, $\|\mathbf{Q} \cdot \mathbf{x}\|_2 = \|\mathbf{x}\|_2$. De esto se desprende también que el número de condición de una matriz ortogonal (con respecto a la norma 2) es $\kappa(\mathbf{Q}) = 1$. Todo esto nos indica que las matrices ortogonales son muy estables numéricamente, lo cual las vuelve muy atractivas a la hora de resolver problemas en forma computacional.

Dada una matriz $\mathbf{A} \in \mathbb{R}^{m \times n}$, una **factorización QR** de $\mathbf{A}$ es una escritura

$$\mathbf{A} = \mathbf{Q} \cdot \mathbf{R}$$

donde $\mathbf{Q} \in \mathbb{R}^{m \times m}$ es ortogonal y $\mathbf{R} \in \mathbb{R}^{m \times n}$ es triangular superior. Toda matriz admite una factorización QR, incluso aquellas que no son cuadradas, como se verá más adelante al analizar los algoritmos que permiten obtenerla.

Con respecto a la unicidad de la factorización QR, puede asegurarse para matrices inversibles si se agrega la restricción de que los elementos en la diagonal de $\mathbf{R}$ sean positivos.

Una de las aplicaciones clave de esta factorización es la resolución de sistemas de ecuaciones lineales. Si se tiene el sistema $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$, utilizando la factorización QR, se lo puede reescribir como

$$\mathbf{Q} \cdot \mathbf{R} \cdot \mathbf{x} = \mathbf{b}$$

Al ser $\mathbf{Q}$ ortogonal, se puede multiplicar a ambos lados por $\mathbf{Q}^{\mathrm{T}}$ y se tiene

$$\mathbf{Q} \cdot^{\mathrm{T}} \mathbf{Q} \cdot \mathbf{R} \cdot \mathbf{x} = \mathbf{Q}^{\mathrm{T}} \mathbf{b}$$

o lo que es lo mismo,

$$\mathbf{R} \cdot \mathbf{x} = \mathbf{Q}^{\mathrm{T}} \cdot \mathbf{b}$$

Como $\mathbf{R}$ es triangular superior, este sistema se puede resolver de forma eficiente con sustitución hacia atrás, con la importante ventaja de que $\mathbf{Q}^{\mathrm{T}}$ es sumamente estable, evitando la introducción de errores numéricos al calcular el producto con $\mathbf{b}$.

3.2.1. Método de Householder (Reflexiones)

Las dos formas que estudiaremos para computar la factorización QR de una matriz se basan en el mismo principio: ir aplicando sucesivas transformaciones a la matriz, todas ellas definidas por matrices ortogonales, hasta llevarla a una forma triangular superior. La diferencia está en el tipo de estas transformaciones. En el caso del **método de Householder**, se trata de **matrices de reflexión**.

Una matriz de reflexión en $\mathbb{R}^{n \times n}$ envía a cualquier vector a su opuesto por un determinado hiperplano de $\mathbb{R}^{n \times n}$, es decir, un subespacio de dimensión $n - 1$. Intuitivamente, se trata de una

transformación que no altera la norma 2 de los vectores y, por lo tanto, la matriz asociada a ella debe ser ortogonal.

Cualquier hiperplano se puede determinar unívocamente mediante un vector no nulo $\mathbf{u} \in \mathbb{R}^n$ ortogonal a él, para el cual la matriz de reflexión $\mathbf{W}$ cumplirá $\mathbf{W} \cdot \mathbf{u} = -\mathbf{u}$. Por otro lado, para cualquier $\mathbf{v}$ que sea ortogonal a $\mathbf{u}$ (o, lo que es lo mismo, que sea parte del hiperplano), debe suceder que $\mathbf{W} \cdot \mathbf{v} = \mathbf{v}$.

Partiendo de las dos condiciones anteriores puede demostrarse que, si se toma $\mathbf{u}$ unitario (es decir, $\|\mathbf{u}\|_2 = 1$), la matriz de reflexión correspondiente al hiperplano ortogonal a $\mathbf{u}$ es

$$\mathbf{W} = \mathbf{I} - 2 \cdot \mathbf{u} \cdot \mathbf{u}^{\mathrm{T}}.$$

Además, la matriz $\mathbf{W}$ resulta ortogonal.

Un resultado que nos será útil es que, si se tienen dos vectores $\mathbf{v}$ y $\mathbf{w} \in \mathbb{R}^n$ de igual norma 2, siempre es posible encontrar un hiperplano tal que la reflexión de $\mathbf{v}$ en el mismo sea $\mathbf{w}$. Dicho hiperplano será perpendicular a la recta que pasa por los extremos de $\mathbf{v}$ y $\mathbf{w}$, por lo que se lo puede describir mediante el versor

$$\mathbf{u} = \frac{\mathbf{v} - \mathbf{w}}{\|\mathbf{v} - \mathbf{w}\|_2}.$$

Partiendo de esta idea, buscaremos tomar una matriz $\mathbf{A} \in \mathbb{R}^{m \times n}$ y usar reflexiones sucesivas para triangularla, enviando cada una de sus columnas a un vector de igual norma, pero que tenga ceros en los lugares donde sea necesario.

Para describir el proceso, fijamos $\mathbf{A}^{(0)} = \mathbf{A}$; luego, para cada $k \in \{1, \ldots, n-1\}$, consideremos la matriz $\mathbf{A}^{(k-1)}$ obtenida en el paso anterior. El objetivo del paso k-ésimo del algoritmo será poner ceros debajo de la diagonal en la k-ésima columna de la matriz. Podemos notar que

$$\mathbf{A}^{(k-1)} = \left[\begin{array}{ccc|ccc} * & \cdots & * & & & \\ & \ddots & \vdots & & \mathbf{*} & \\ & & * & & & \\ \hline & & & w_{k,k} & \cdots & w_{k,n} \\ & \mathbf{0} & & \vdots & \ddots & \vdots \\ & & & w_{m,k} & \cdots & w_{m,n} \end{array}\right]$$

Teniendo en cuenta solamente la submatriz que todavía no fue triangulada, podemos lograr nuestro objetivo buscando una reflexión $\tilde{\mathbf{W}}_k \in \mathbb{R}^{(m-k+1)\times(m-k+1)}$ tal que

$$\tilde{\mathbf{W}}_k \cdot \begin{bmatrix} w_{k,k} \\ w_{k+1,k} \\ \vdots \\ w_{m,k} \end{bmatrix} = \begin{bmatrix} \|(w_{k,k}, \ldots, w_{m,k})\| \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Llamando $\mathbf{v}$ y $\mathbf{w}$ a los dos vectores que aparecen en la ecuación anterior, respectivamente, podemos definir el vector unitario $\tilde{\mathbf{u}}_k = \frac{\mathbf{v}-\mathbf{w}}{\|\mathbf{v}-\mathbf{w}\|_2}$ y así la matriz de reflexión asociada $\tilde{\mathbf{W}}_k = \mathbf{I} - 2 \cdot \tilde{\mathbf{u}} \cdot \tilde{\mathbf{u}}^{\mathrm{T}}$ cumple con lo que necesitamos. Si ahora completamos con ceros las primeras $k-1$ posiciones de $\tilde{\mathbf{u}}_k$, definiendo $\mathbf{u}_k = \left(\begin{array}{c|c} \mathbf{0} & \tilde{\mathbf{u}}_k \end{array}\right)$, la matriz de reflexión asociada $\mathbf{W}_k = \mathbf{I} - 2 \cdot \mathbf{u} \cdot \mathbf{u}^{\mathrm{T}} \in \mathbb{R}^{m \times m}$ tiene la forma

$$\mathbf{W}_k = \left[\begin{array}{c|c} \mathbf{I}_{k-1} & \mathbf{0} \\ \hline \mathbf{0} & \tilde{\mathbf{W}}_k \end{array}\right],$$

por lo que $\mathbf{A}^{(k)} = \mathbf{W}_k \cdot \mathbf{A}^{(k-1)}$ mantendrá sin cambios sus primeras $k-1$ columnas, pero también tendrá ceros debajo de la diagonal en la columna k-ésima.

Podemos concluir, entonces, que la matriz $\mathbf{A}^{(n-1)}$ será triangular superior. Resumiendo lo hecho en todos los pasos, y llamando $\mathbf{R} = \mathbf{A}^{(n-1)}$, tenemos que:

$$\begin{aligned}\mathbf{R} = \mathbf{A}^{(n-1)} &= \mathbf{W}_{n-1} \cdot \ldots \cdot \mathbf{W}_1 \cdot \mathbf{A} \\ &= \mathbf{W} \cdot \mathbf{A}\end{aligned}$$

donde $\mathbf{W}$, el producto de todas las $\mathbf{W}_k$, es también ortogonal por ser el producto de matrices ortogonales. Finalmente, si definimos $\mathbf{Q} = \mathbf{W}^{-1} = \mathbf{W}^{\mathrm{T}}$, que es también ortogonal, resulta que

$$\mathbf{A} = \mathbf{Q} \cdot \mathbf{R},$$

es decir, hemos obtenido una factorización QR para $\mathbf{A}$.

La complejidad del método de Householder es cúbica en el tamaño de la matriz ($\mathrm{O}(n^3)$); si se realiza un análisis más fino, se puede verificar que la cantidad de operaciones de punto flotante necesarias es de alrededor de $\frac{2}{3} \cdot n^3$.

3.2.2. Método de Givens (Rotaciones)

El **método de Givens**, por su parte, realiza las transformaciones a través de un tipo de matrices ortogonales conocidas como **matrices de rotación**.

Las matrices de rotación reciben este nombre porque la transformación lineal inducida por ellas hace rotar a cualquier vector, en torno al origen, un ángulo fijo θ. Es evidente que las transformaciones de este tipo preservan la longitud de los vectores, por lo que las matrices correspondientes son ortogonales.

Por ejemplo, si se consideran vectores en $\mathbb{R}^2$ y un ángulo θ, se puede definir la matriz de rotación $\mathbf{W}$ correspondiente a dicho ángulo[4] a partir de los valores que deberá tomar la transformación en una base de $\mathbb{R}^2$. Se quiere que

$$\mathbf{W} \cdot \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}\cos(\theta)\\-\operatorname{sen}(\theta)\end{bmatrix} \qquad \text{y} \qquad \mathbf{W} \cdot \begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}\operatorname{sen}(\theta)\\\cos(\theta)\end{bmatrix},$$

por lo que

$$\mathbf{W} = \begin{bmatrix}\cos(\theta) & \operatorname{sen}(\theta)\\-\operatorname{sen}(\theta) & \cos(\theta)\end{bmatrix}.$$

Es sencillo verificar que, para cualquier valor de θ, esta matriz es ortogonal.

Nuestro objetivo es usar sucesivas rotaciones para ir eliminando componentes de las columnas de una matriz, hasta lograr triangularla. En el caso de una matriz $\mathbf{A} \in \mathbb{R}^{2\times 2}$,

$$\mathbf{A} = \begin{bmatrix}a_1 & b_1\\a_2 & b_2\end{bmatrix},$$

queremos rotar la primera columna de modo tal que su segunda coordenada pase a tener un 0; en otras palabras, la columna debe pasar a ser un vector paralelo al eje de abscisas. Adoptaremos la convención de que apuntará en el sentido positivo. Luego, como las rotaciones preservan la norma 2, buscamos una matriz de rotación que cumpla

$$\mathbf{W} \cdot \begin{bmatrix}a_1\\a_2\end{bmatrix} = \begin{bmatrix}\|(a_1, a_2)\|_2\\0\end{bmatrix}.$$

[4] Para construir las matrices de rotación, interpretaremos que los ángulos expresan giros en el sentido de las agujas del reloj.

Se puede ver que el ángulo θ de la rotación deberá ser tal que

$$\cos(\theta) = \frac{a_1}{\|(a_1, a_2)\|_2} \qquad \text{y} \qquad \text{sen}(\theta) = \frac{a_2}{\|(a_1, a_2)\|_2}.$$

Por lo tanto,

$$\mathbf{W} = \begin{bmatrix} \dfrac{a_1}{\|(a_1, a_2)\|_2} & \dfrac{a_2}{\|(a_1, a_2)\|_2} \\ -\dfrac{a_2}{\|(a_1, a_2)\|_2} & \dfrac{a_1}{\|(a_1, a_2)\|_2} \end{bmatrix}.$$

Aplicando esta rotación sobre $\mathbf{A}$, se obtiene una matriz $\mathbf{R}$ triangular inferior. Como $\mathbf{W} \cdot \mathbf{A} = \mathbf{R}$, entonces $\mathbf{A} = \mathbf{W}^{-1} \cdot \mathbf{R} = \mathbf{W}^{\mathrm{T}} \cdot \mathbf{R}$. Dado que $\mathbf{W}^{\mathrm{T}}$ es ortogonal, llamando $\mathbf{Q} = \mathbf{W}^{\mathrm{T}}$, tenemos una factorización QR para $\mathbf{A}$.

Esto se puede generalizar a matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$. La idea es iterar sobre las columnas de la matriz, e ir rotándolas hasta que contengan ceros en las posiciones deseadas. Para cada columna, se realizarán varias rotaciones sucesivas, tantas como ceros sea necesario colocar en ella.

Más en detalle, si se está trabajando con la columna j-ésima, para cada $i > j$ se definirá $\mathbf{W}_{i,j} \in \mathbb{R}^{m \times m}$ de la siguiente forma:

$$\begin{aligned} w_{i,i} &= \frac{a_i}{\|(a_i, a_j)\|_2}, & w_{i,j} &= \frac{a_j}{\|(a_i, a_j)\|_2}, \\ w_{j,i} &= -\frac{a_j}{\|(a_i, a_j)\|_2}, & w_{j,j} &= \frac{a_i}{\|(a_i, a_j)\|_2}, \end{aligned}$$

y el resto de las posiciones al igual que la matriz identidad. Los valores de las posiciones de $\mathbf{A}$ se van modificando a lo largo del proceso, por lo que siempre se consideran los obtenidos en la iteración anterior del algoritmo. Es sencillo verificar que cada $\mathbf{W}_{i,j}$ es una matriz ortogonal.

De esta forma, construyendo iterativamente las matrices de rotación y multiplicando $\mathbf{A}$ a izquierda, se llega a una forma triangular superior

$$\begin{aligned} \mathbf{R} &= (\mathbf{W}_{n,n-1}) \cdot (\mathbf{W}_{n,n-2} \cdot \mathbf{W}_{n-1,n-2}) \cdot \ldots \cdot (\mathbf{W}_{n,1} \cdot \ldots \cdot \mathbf{W}_{2,1}) \cdot \mathbf{A}. \\ &= \mathbf{W} \cdot \mathbf{A} \end{aligned}$$

donde $\mathbf{W}$ es el producto de todas las $\mathbf{W}_{i,j}$, y es por lo tanto una matriz ortogonal. Tomando $\mathbf{Q} = \mathbf{W}^{\mathrm{T}}$, se obtiene

$$\mathbf{A} = \mathbf{Q} \cdot \mathbf{R},$$

una factorización QR para $\mathbf{A}$.

En comparación con el método de Householder, el método de Givens, si bien tiene la misma complejidad asintótica ($\mathrm{O}(n^3)$), debe realizar aproximadamente el doble de operaciones de punto flotante: alrededor de $\frac{4}{3} \cdot n^3$. No obstante, presenta una gran ventaja si se está trabajando con matrices ralas (donde una gran cantidad de las posiciones está ocupada por ceros), ya que cada paso del algoritmo pone un cero en una posición de la matriz, por lo que se puede realizar una optimización colocando ceros solo en las posiciones donde es necesario. En contraste, al usar el método de Householder, cada iteración pone ceros en una columna completa.

3.3. Descomposición en valores singulares

A partir de esta sección, se hace necesario presentar dos nociones importantes que continuarán apareciendo más adelante. Dada una matriz $\mathbf{A} \in \mathbb{R}^{n\times n}$, decimos que $\lambda \in \mathbb{R}$ es un **autovalor** de $\mathbf{A}$ si existe algún vector no nulo $\mathbf{v} \in \mathbb{R}^n$ tal que $\mathbf{A} \cdot \mathbf{v} = \lambda \cdot \mathbf{v}$. En ese caso, a $\mathbf{v}$ se lo conoce como un **autovector** de $\mathbf{A}$ asociado al autovalor λ. Cabe destacar que, si λ es un autovalor de una matriz, entonces existen infinitos autovectores asociados al mismo; los autovectores asociados a un autovalor λ dado conforman un subespacio vectorial de $\mathbb{R}^{n\times n}$.

Sea $\mathbf{A} \in \mathbb{R}^{m\times n}$ una matriz cualquiera, con $\operatorname{rg}(A) = r$. Una **descomposición en valores singulares** de $\mathbf{A}$ es una escritura

$$\mathbf{A} = \mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{V}^{\mathrm{T}},$$

donde

- $\mathbf{U} = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \dots & \mathbf{u}_m \end{bmatrix} \in \mathbb{R}^{m\times m}$ y $\mathbf{V} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \dots & \mathbf{v}_n \end{bmatrix} \in \mathbb{R}^{n\times n}$ son ortogonales, y
- $\mathbf{\Sigma} = \begin{bmatrix} \sigma_1 & 0 & \cdots & & & \\ 0 & \sigma_2 & & & & \\ \vdots & & \ddots & & & \\ & & & \sigma_r & & \\ & & & & 0 & \\ & & & & & \ddots \\ & & & & & & 0 \end{bmatrix} \in \mathbb{R}^{m\times n}$ es diagonal, con $\sigma_i > 0$ para todo $i \in \{1, \dots, r\}$.

Los valores σ_i se denominan los **valores singulares** de $\mathbf{A}$. Por su parte, $\{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_m\}$ y $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n\}$ son bases ortonormales de $\mathbb{R}^m$ y $\mathbb{R}^n$, respectivamente.

Como se verá en la demostración de existencia que se presenta a continuación, se cumple que:

- $\sigma_1^2, \dots, \sigma_r^2$ son los autovalores no nulos tanto de $\mathbf{A} \cdot \mathbf{A}^{\mathrm{T}}$ como de $\mathbf{A}^{\mathrm{T}} \cdot \mathbf{A}$.
- Para cada $i \in \{1, \dots, r\}$, $\mathbf{u}_i$ es un autovector unitario de $\mathbf{A} \cdot \mathbf{A}^{\mathrm{T}}$ asociado al autovalor σ_i^2, mientras que $\mathbf{v}_i$ es un autovector unitario de $\mathbf{A}^{\mathrm{T}} \cdot \mathbf{A}$ asociado al mismo autovalor.
- Los $\mathbf{u}_i$ y $\mathbf{v}_i$ con $i > r$ forman parte del núcleo de $\mathbf{A} \cdot \mathbf{A}^{\mathrm{T}}$ y de $\mathbf{A}^{\mathrm{T}} \cdot \mathbf{A}$, respectivamente.

La descomposición en valores singulares tiene un interés muy particular en diversas aplicaciones específicas, ya que logra condensar en pocos valores (los valores singulares) información muy relevante acerca de la matriz con la que se está trabajando. Un campo en el que suele tener aplicación es el de la estadística. Por otro lado, es una factorización útil a la hora de resolver el problema de cuadrados mínimos lineales, que será abordado en la sección 7.

3.3.1. Demostración de la existencia

Es posible demostrar que toda matriz admite una descomposición en valores singulares. Para demostrarlo, comenzamos por observar que la igualdad enunciada,

$$\mathbf{A} = \mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{V}^{\mathrm{T}},$$

puede expresarse de forma equivalente mediante las siguientes ecuaciones, usando la ortogonalidad de $\mathbf{U}$ y $\mathbf{V}$:

$$\mathbf{A}\cdot\mathbf{V}=\mathbf{U}\cdot\mathbf{\Sigma} \qquad\qquad \mathbf{A}^{\mathrm{T}}\cdot\mathbf{U}=\mathbf{V}\cdot\mathbf{\Sigma}$$

$$\Updownarrow \qquad\qquad \Updownarrow$$

$$\mathbf{A}\cdot\begin{bmatrix}\mathbf{v}_1 & \dots & \mathbf{v}_n\end{bmatrix}=\begin{bmatrix}\mathbf{u}_1 & \dots & \mathbf{u}_m\end{bmatrix}\cdot\mathbf{\Sigma} \qquad \mathbf{A}^{\mathrm{T}}\cdot\begin{bmatrix}\mathbf{u}_1 & \dots & \mathbf{u}_m\end{bmatrix}=\begin{bmatrix}\mathbf{v}_1 & \dots & \mathbf{v}_n\end{bmatrix}\cdot\mathbf{\Sigma}$$

$$\Updownarrow \qquad\qquad \Updownarrow$$

$$\begin{cases}\mathbf{A}\cdot\mathbf{v}_i=\sigma_i\cdot\mathbf{u}_i & \text{para } i=1,\dots,r \quad (1)\\ \mathbf{A}\cdot\mathbf{v}_i=\mathbf{0} & \text{para } i=r+1,\dots,n \quad (2)\end{cases} \qquad \begin{cases}\mathbf{A}^{\mathrm{T}}\cdot\mathbf{u}_i=\sigma_i\cdot\mathbf{v}_i & \text{para } i=1,\dots,r \quad (3)\\ \mathbf{A}^{\mathrm{T}}\cdot\mathbf{u}_i=\mathbf{0} & \text{para } i=r+1,\dots,m \quad (4)\end{cases}$$

En otras palabras, queremos probar la existencia de

- $\mathbf{v}_1,\dots,\mathbf{v}_n\in\mathbb{R}^n$ ortonormales,
- $\mathbf{u}_1,\dots,\mathbf{u}_m\in\mathbb{R}^m$ ortonormales, y
- $\sigma_1,\dots,\sigma_r>0$

que cumplan las ecuaciones (1), (2), (3) y (4).

En primer lugar, vemos que:

(i) Para $i=1,\dots,r$,

$$\begin{aligned}\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}\cdot\mathbf{v}_i&=\sigma_i\cdot\mathbf{A}^{\mathrm{T}}\cdot\mathbf{u}_i\\&=\sigma_i^2\cdot\mathbf{v}_i\end{aligned}$$

es decir, los $\mathbf{v}_1,\dots,\mathbf{v}_r$ deberán ser autovectores de $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}$, con autovalor σ_i^2.

(ii) Para $i=r+1,\dots,n$,

$$\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}\cdot\mathbf{v}_i=\mathbf{A}^{\mathrm{T}}\cdot 0=0$$

es decir, los $\mathbf{v}_{r+1},\dots,\mathbf{v}_n$ deberán ser autovectores de $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}$, con autovalor 0.

Como $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}$ es simétrica, existe una base ortonormal de $\mathbb{R}^n$ compuesta por autovectores de $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}$, propiedad que utilizaremos sin demostración. Esto implica que todos los autovalores de $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}$ son reales.

Además, como $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}$ es semidefinida positiva, todos sus autovalores deben ser no negativos: si λ es un autovalor de $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}$, entonces existe un vector no nulo $\mathbf{v}$ tal que $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}\cdot\mathbf{v}=\lambda\cdot\mathbf{v}$. Entonces $\mathbf{v}^{\mathrm{T}}\cdot\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}\cdot\mathbf{v}=\lambda\cdot\mathbf{v}^{\mathrm{T}}\cdot\mathbf{v}$. Por ser $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}$ semidefinida positiva, $\mathbf{v}^{\mathrm{T}}\cdot\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}\cdot\mathbf{v}\geq 0$, mientras que $\mathbf{v}^{\mathrm{T}}\cdot\mathbf{v}=\|\mathbf{v}\|_2^2>0$, de forma que necesariamente $\lambda\geq 0$.

Por otra parte, tenemos que para todo $\mathbf{x}\in\mathbb{R}^n$

$$\mathbf{A}\cdot\mathbf{x}=\mathbf{0}\quad\Leftrightarrow\quad\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}\cdot\mathbf{x}=\mathbf{0}$$

dado que, trivialmente, si $\mathbf{A}\cdot\mathbf{x}=\mathbf{0}$ entonces $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}\cdot\mathbf{x}=\mathbf{0}$, mientras que si $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}\cdot\mathbf{x}=\mathbf{0}$ entonces $\mathbf{x}^{\mathrm{T}}\cdot\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}\cdot\mathbf{x}=\|\mathbf{A}\cdot\mathbf{x}\|_2^2=0$, lo cual solo sucede si $\mathbf{A}\cdot\mathbf{x}=\mathbf{0}$. Así, $\mathrm{rg}(\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A})=\mathrm{rg}(\mathbf{A})=r$, lo cual significa que $\mathrm{Nul}(\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A})$ tiene dimensión $n-r$; en otras palabras $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}$ tiene exactamente $n-r$ autovectores linealmente independientes asociados al autovalor 0.

Definimos entonces $\{\mathbf{v}_1,\dots,\mathbf{v}_n\}$ como una base ortonormal de autovectores de $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}$, ordenados de modo que los últimos $n-r$ son los asociados al autovalor 0. Veamos que estos $\mathbf{v}_i$ cumplen con las condiciones necesarias.

(i) Para $i=1,\dots,r$, sea $\lambda_i\in\mathbb{R}$ el autovalor de $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}$ asociado al autovector $\mathbf{v}_i$. Sabemos que $\lambda_i>0$. Definimos entonces:

- $\sigma_i=\sqrt{\lambda_i}>0$.

- $\mathbf{u}_i = \dfrac{\mathbf{A}\cdot\mathbf{v}_i}{\sigma_i}$. Trivialmente, puede verse que se satisface la ecuación (1).

Veamos que $\mathbf{u}_1, \ldots, \mathbf{u}_r$ son ortonormales:

- $\|\mathbf{u}_i\|_2^2 = \left(\dfrac{\mathbf{A}\cdot\mathbf{v}_i}{\sigma_i}\right)^{\mathrm{T}} \cdot \left(\dfrac{\mathbf{A}\cdot\mathbf{v}_i}{\sigma_i}\right) = \dfrac{1}{\sigma_i^2}(\mathbf{v}_i^{\mathrm{T}}\cdot\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}\cdot\mathbf{v}_i) = \dfrac{1}{\sigma_i^2}(\mathbf{v}_i^{\mathrm{T}}\cdot\sigma_i^2\cdot\mathbf{v}_i) = \mathbf{v}_i^{\mathrm{T}}\cdot\mathbf{v}_i = \|\mathbf{v}_i\|_2^2 = 1.$
- Si $i \neq j$, entonces $\mathbf{u}_i^{\mathrm{T}} \cdot \mathbf{u}_j = \left(\dfrac{\mathbf{A}\cdot\mathbf{v}_i}{\sigma_i}\right)^{\mathrm{T}} \cdot \left(\dfrac{\mathbf{A}\cdot\mathbf{v}_j}{\sigma_j}\right) = \dfrac{\mathbf{v}_i^{\mathrm{T}}\cdot\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}\cdot\mathbf{v}_j}{\sigma_i\cdot\sigma_j}$
$$= \frac{\mathbf{v}_i^{\mathrm{T}}\cdot\sigma_j^2\cdot\mathbf{v}_j}{\sigma_i\cdot\sigma_j} = \frac{\sigma_j}{\sigma_i}\cdot\mathbf{v}_i^{\mathrm{T}}\cdot\mathbf{v}_j = 0.$$

Resta verificar que se cumple la ecuación (3):

$$\mathbf{A}^{\mathrm{T}}\cdot\mathbf{u}_i = \frac{\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}\cdot\mathbf{v}_i}{\sigma_i} = \frac{\sigma_i^2\cdot\mathbf{v}_i}{\sigma_i} = \sigma_i\cdot\mathbf{v}_i$$

(ii) Para $i = r+1, \ldots, n$, ya vimos que, como $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}\cdot\mathbf{v}_i = \mathbf{0}$, también $\mathbf{A}\cdot\mathbf{v}_i = \mathbf{0}$, por lo que se satisface la ecuación (2).

Aún no definimos $\mathbf{u}_{r+1}, \ldots, \mathbf{u}_m$. Lo hacemos de modo que sean una base ortonormal de $\mathrm{Nul}(\mathbf{A}^{\mathrm{T}})$. Así, se satisface la ecuación (4): $\mathbf{A}^{\mathrm{T}}\cdot\mathbf{u}_i = \mathbf{0}$.

Solo falta ver que $\{\mathbf{u}_1, \ldots, \mathbf{u}_m\}$ es una base ortonormal de $\mathbb{R}^m$. Ahora bien:

- es una base, dado que $\{\mathbf{u}_1, \ldots, \mathbf{u}_r\}$ es una base de $\mathrm{Im}(\mathbf{A})$, $\{\mathbf{u}_{r+1}, \ldots, \mathbf{u}_m\}$ es una base de $\mathrm{Nul}(\mathbf{A}^{\mathrm{T}})$, y $\mathrm{Im}(\mathbf{A}) \oplus \mathrm{Nul}(\mathbf{A}^{\mathrm{T}}) = \mathbb{R}^m$.
- ya vimos que tanto $\{\mathbf{u}_1, \ldots, \mathbf{u}_r\}$ como $\mathrm{Im}(\mathbf{A})$, $\{\mathbf{u}_{r+1}, \ldots, \mathbf{u}_m\}$ son conjuntos ortonormales. Basta ver, entonces, que tomando $i \in \{1, \ldots, r\}$ y $j \in \{r+1, \ldots, m\}$, se cumple que $\mathbf{u}_i \perp \mathbf{u}_j$, es decir, que $\mathbf{u}_i^{\mathrm{T}}\cdot\mathbf{u}_j = 0$. En efecto,

$$\mathbf{u}_i^{\mathrm{T}}\cdot\mathbf{u}_j = \left(\frac{\mathbf{A}\cdot\mathbf{v}_i}{\sigma_i}\right)^{\mathrm{T}}\cdot\mathbf{u}_j = \frac{1}{\sigma_i}\cdot\mathbf{v}_i^{\mathrm{T}}\cdot\mathbf{A}^{\mathrm{T}}\cdot\mathbf{u}_j = \frac{1}{\sigma_i}\cdot\mathbf{v}_i^{\mathrm{T}}\cdot\mathbf{0} = 0.$$

4. Resolución iterativa de sistemas de ecuaciones lineales

4.1. Métodos iterativos

Un **método iterativo** es un procedimiento para la resolución de un problema que se basa en construir una sucesión de elementos $\{x_k\}_{k\in\mathbb{N}}$ tal que $\lim_{k\to\infty} x_k = x^*$, donde x^* es una solución del problema que se quiere resolver. Esto permite obtener un algoritmo que se aproxime progresivamente a una solución calculando, en cada paso, el $k+1$-ésimo término de la sucesión a partir del k-ésimo. De esta forma, mediante sucesivas iteraciones, puede lograrse una aproximación cada vez mejor a una solución del problema.

En el caso de la resolución de un sistema de ecuaciones lineales $\mathbf{A}\cdot\mathbf{x} = \mathbf{b}$ ($\mathbf{A}\in\mathbb{R}^{n\times n}$, $\mathbf{b}\in\mathbb{R}^n$), se busca una sucesión de vectores $\{\mathbf{x}^{(k)}\}_{k\in\mathbb{N}}$ que converja a una solución del sistema, es decir, a un valor $\mathbf{x}^*$ tal que $\mathbf{A}\cdot\mathbf{x}^* = \mathbf{b}$.

En particular, los métodos con los que trabajaremos consisten en reescribir el sistema como

$$\mathbf{x} = \mathbf{c} + \mathbf{T}\cdot\mathbf{x}$$

para ciertos $\mathbf{T}\in\mathbb{R}^{n\times n}$ y $\mathbf{c}\in\mathbb{R}^n$, y definir la iteración

$$\mathbf{x}^{(k)} = \mathbf{c} + \mathbf{T}\cdot\mathbf{x}^{(k-1)}$$

partiendo de algún $\mathbf{x}^{(0)}$ arbitrario. Es claro que si una iteración de este tipo converge a algún vector $\mathbf{x}^*$, dicho vector deberá ser solución del sistema.

4.2. Análisis de convergencia

Llamamos **radio espectral** de una matriz a su autovalor de módulo máximo, es decir

$$\rho(\mathbf{A}) = \text{máx}\{|\lambda| : \lambda \text{ es un autovalor de } \mathbf{A}\}.$$

El radio espectral de una matriz es una cota inferior para cualquier norma matricial inducida $\|\bullet\|$:

$$\|\mathbf{A}\| \geq \rho(\mathbf{A}),$$

dado que siempre es posible tomar un autovector unitario $\mathbf{v}$ asociado al autovalor λ tal que $\rho(\mathbf{A}) = |\lambda|$, en cuyo caso $\|\mathbf{A}\| \geq \|\mathbf{A}\cdot\mathbf{v}\| = \|\lambda\cdot\mathbf{v}\| = |\lambda|\cdot\|\mathbf{v}\| = \rho(\mathbf{A})\cdot 1$.

Una matriz es **convergente** si $\lim_{k\to\infty}\mathbf{A}^k = \mathbf{0}$. Las matrices convergentes se pueden caracterizar exactamente como aquellas matrices cuyo número de condición es menor que 1.

Si consideramos un sistema de ecuaciones de la forma $\mathbf{x} = \mathbf{c} + \mathbf{T}\cdot\mathbf{x}$ y una solución $\mathbf{x}^*$, entonces el método iterativo de la forma $\mathbf{x}^{(k)} = \mathbf{c} + \mathbf{T}\cdot\mathbf{x}^{(k-1)}$ converge a $\mathbf{x}^*$, para cualquier $\mathbf{x}^{(0)}$, si y solo si $\mathbf{T}$ es convergente. Para demostrarlo, consideremos $\mathbf{r}^{(k)} = \mathbf{x}^* - \mathbf{x}^{(k)}$; queremos ver que $\mathbf{r}^{(k)} \xrightarrow{k\to\infty} \mathbf{0}$ si y solo si $\mathbf{T}$ es convergente.

Vale que $\mathbf{r}^{(k)} = \mathbf{x}^* - \mathbf{x}^{(k)} = \left(\mathbf{c} + \mathbf{T}\cdot\mathbf{x}^*\right) - \left(\mathbf{c} + \mathbf{T}\cdot\mathbf{x}^{(k-1)}\right) = \mathbf{T}\cdot\left(\mathbf{x}^* - \mathbf{x}^{(k-1)}\right) = \mathbf{T}\cdot\mathbf{r}^{(k-1)}$ y, por lo tanto,

$$\mathbf{r}^{(k)} = \mathbf{T}^k\cdot\mathbf{r}^{(0)} = \mathbf{T}^k\cdot\left(\mathbf{x}^* - \mathbf{x}^{(0)}\right).$$

Es claro que si $\mathbf{T}$ converge, entonces también lo hace $\mathbf{r}^{(k)}$, independientemente del valor de $\mathbf{x}^{(0)}$. Para ver la vuelta, basta notar que si $\mathbf{r}^{(k)}$ converge para todo $\mathbf{x}^{(0)}$, se puede elegir este último vector de modo que $\mathbf{r}^{(0)}$ sea cualquiera de los vectores canónicos. De esta forma $\mathbf{r}^{(k)} = \mathbf{T}^k\cdot\mathbf{e}_i = \text{col}_i(\mathbf{T})^k \xrightarrow{k\to\infty} \mathbf{0}$, verificando que todos los elementos de $\mathbf{T}$ convergen a 0.

4.3. Método de Jacobi

El **método de Jacobi** es un método que busca resolver un sistema de ecuaciones $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ de forma iterativa, es decir partiendo de un vector inicial $\mathbf{x}^{(0)} = (x_0^{(0)}, \dots, x_n^{(0)})$ e iterando sobre él de modo de obtener aproximaciones cada vez mejores de una solución al sistema.

La idea es proceder, para cada $k = 1, 2, \dots$, de la siguiente forma. Se considera como input de la k-ésima iteración la aproximación $\mathbf{x}^{(k-1)} = (x_0^{(k-1)}, \dots, x_n^{(k-1)})$ producida en la iteración anterior. Luego, se recorren en orden las ecuaciones del sistema

$$a_{i,1} \cdot x_1 + \cdots + a_{i,i} \cdot x_i + \cdots + a_{i,n} \cdot x_n = b_i$$

y, para cada una de ellas, se despeja $x_i^{(k)}$ reemplazando las demás variables por los valores correspondientes de $\mathbf{x}^{(k-1)}$; es decir, se define $x_i^{(k)}$ de modo que

$$a_{i,1} \cdot x_1^{(k-1)} + \cdots + a_{i,i} \cdot x_i^{(k)} + \cdots + a_{i,n} \cdot x_n^{(k-1)} = b_i$$

o, lo que es lo mismo, despejando $x_i^{(k)}$:

$$x_i^{(k)} = \frac{1}{a_{i,i}} \cdot \left(b_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} \left(a_{i,j} \cdot x_j^{(k-1)} \right) \right).$$

Notar que para que la iteración esté bien definida, $\mathbf{A}$ no debe tener ceros en la diagonal.

Escribiendo el algoritmo en forma de pseudocódigo, se tiene

Método de Jacobi

Entrada: $\mathbf{A} \in \mathbb{R}^{n \times n}$ sin ceros en la diagonal, $\mathbf{b}$, $\mathbf{x}^{(0)} \in \mathbb{R}^n$ arbitrarios.
Salida: $\mathbf{x}^*$ solución del sistema $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$.

1 **para** $k = 1, 2, \dots$ **hacer**
2 **para** $i = 1, \dots, n$ **hacer**
3 $x_i^{(k)} \leftarrow \frac{1}{a_{i,i}} \cdot \left(b_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} \left(a_{i,j} \cdot x_j^{(k-1)} \right) \right)$

Resulta evidente que la complejidad de cada iteración es $\mathrm{O}(n^2)$.

El mismo algoritmo puede ser escrito en forma matricial, si se parte de una descomposición $\mathbf{A} = \mathbf{D} - \mathbf{L} - \mathbf{U}$, donde

$$\mathbf{D} = \begin{bmatrix} a_{1,1} & & & \mathbf{0} \\ & a_{2,2} & & \\ & & \ddots & \\ \mathbf{0} & & & a_{n,n} \end{bmatrix}, \quad \mathbf{L} = \begin{bmatrix} & & & \\ -a_{2,1} & & \mathbf{0} & \\ \vdots & \ddots & & \\ -a_{n,1} & \cdots & -a_{n,n-1} & \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} & -a_{1,2} & \cdots & -a_{1,n} \\ & & \ddots & \vdots \\ \mathbf{0} & & & -a_{n-1,n} \\ & & & \end{bmatrix}.$$

Reescribiendo el sistema, y aprovechando que $\mathbf{D}$ es inversible, tenemos que

$$\begin{aligned} \mathbf{A} \cdot \mathbf{x} &= \mathbf{b} && \text{sii} \\ (\mathbf{D} - \mathbf{L} - \mathbf{U}) \cdot \mathbf{x} &= \mathbf{b} && \text{sii} \\ \mathbf{D} \cdot \mathbf{x} - (\mathbf{L} + \mathbf{U}) \cdot \mathbf{x} &= \mathbf{b} && \text{sii} \\ \mathbf{D} \cdot \mathbf{x} &= \mathbf{b} + (\mathbf{L} + \mathbf{U}) \cdot \mathbf{x} && \text{sii} \\ \mathbf{x} &= \mathbf{D}^{-1} \cdot \mathbf{b} + \mathbf{D}^{-1} \cdot (\mathbf{L} + \mathbf{U}) \cdot \mathbf{x} \end{aligned}$$

Podemos observar que

$$\mathbf{x}^{(k)} = \mathbf{D}^{-1} \cdot \mathbf{b} + \mathbf{D}^{-1} \cdot (\mathbf{L} + \mathbf{U}) \cdot \mathbf{x}^{(k-1)}$$

corresponde exactamente a la formulación matricial de la iteración planteada antes, y que de converger, lo hará a una solución de $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$.

Una familia de matrices para la que es posible garantizar la convergencia del método de Jacobi es la de matrices estrictamente diagonal-dominantes por filas. Para verificarlo, se puede ver que la norma infinito de la matriz de la iteración, que acota a su radio espectral, es siempre menor que 1.

4.4. Método de Gauss-Seidel

El método de **Gauss-Seidel** es similar al de Jacobi, pero plantea que en cada iteración del método, al computar el i-ésimo elemento de la nueva solución, se aprovechen los $i-1$ elementos calculados de la misma, en lugar de usar los de la iteración anterior. Como estos elementos representan una mejor aproximación de una solución del sistema, el método de Gauss-Seidel converge de manera más rápida que el de Jacobi.

La iteración queda planteada como

$$x_i^{(k)} = \frac{1}{a_{i,i}} \cdot \left(b_i - \sum_{j=1}^{i-1} \left(a_{i,j} \cdot x_j^{(k)} \right) - \sum_{j=i+1}^{n} \left(a_{i,j} \cdot x_j^{(k-1)} \right) \right),$$

y en forma matricial, aprovechando que $\mathbf{D} - \mathbf{L}$ es inversible,

$$\mathbf{x}^{(k)} = (\mathbf{D} - \mathbf{L})^{-1} \cdot \mathbf{b} + (\mathbf{D} - \mathbf{L})^{-1} \cdot \mathbf{U} \cdot \mathbf{x}^{(k-1)}.$$

La convergencia del método de Gauss-Seidel está garantizada para matrices estrictamente diagonal-dominantes por filas, al igual que con el método de Jacobi. Además, también se puede asegurar su convergencia para matrices simétricas definidas positivas; esto último no es cierto para el método de Jacobi.

Por último, se puede mencionar otra ventaja que presenta el método de Gauss-Seidel sobre el de Jacobi: como solo es necesario tener al mismo tiempo parte de la iteración actual y parte de la anterior, el algoritmo puede realizarse utilizando un único arreglo, es decir, requiere la mitad del espacio.

5. Cálculo de autovalores

El problema de hallar los autovalores de una matriz es recurrente, y tiene múltiples aplicaciones. Una de ellas ya se presentó anteriormente: computar la descomposición en valores singulares de una matriz. También se los utiliza en estadística para realizar análisis de componentes principales sobre conjuntos de datos; permiten diagonalizar matrices, lo cual facilita su exponenciación; y un variado etcétera que convierte su búsqueda en un problema significativo para resolver.

5.1. Método de la potencia

Supongamos que tenemos una matriz $\mathbf{A} \in \mathbb{R}^{n\times n}$ diagonalizable; es decir, $\mathbf{A}$ posee n autovalores reales. Podemos numerar estos autovalores de forma tal que

$$\|\lambda_1\| \geq \|\lambda_2\| \geq \cdots \geq \|\lambda_n\|.$$

Si la primera desigualdad es estricta, es decir, si $\mathbf{A}$ tiene un autovalor simple λ_1 cuyo valor absoluto es mayor que el de todos los demás, decimos que λ_1 es el **autovalor principal** de $\mathbf{A}$. En ese caso, podemos aplicar el **método de la potencia** para obtener su valor.

El método de la potencia es iterativo, y se basa en una idea sumamente sencilla. Consideremos una base de autovectores de $\mathbf{A}$, $\{\mathbf{v}_1, \cdots, \mathbf{v}_n\}$, ordenados de forma tal que cada $\mathbf{v}_i$ está asociado al autovalor λ_i. Como punto de inicio de la iteración, tomaremos *a priori* un vector $\mathbf{x}^{(0)} \in \mathbb{R}^n$ arbitrario. Podemos escribir

$$\mathbf{x}^{(0)} = \alpha_1 \cdot \mathbf{v}_1 + \cdots + \alpha_n \cdot \mathbf{v}_n.$$

En cada iteración, simplemente multiplicaremos a izquierda por $\mathbf{A}$. Es decir,

$$\begin{aligned}
\mathbf{x}^{(k)} = \mathbf{A} \cdot \mathbf{x}^{(k-1)} &= \mathbf{A}^k \cdot \mathbf{x}^{(0)} \\
&= \mathbf{A}^k \cdot (\alpha_1 \cdot \mathbf{v}_1 + \cdots + \alpha_n \cdot \mathbf{v}_n) \\
&= \alpha_1 \cdot \mathbf{A}^k \cdot \mathbf{v}_1 + \cdots + \alpha_n \cdot \mathbf{A}^k \cdot \mathbf{v}_n \\
&= \alpha_1 \cdot \lambda_1^k \cdot \mathbf{v}_1 + \cdots + \alpha_n \cdot \lambda_n^k \cdot \mathbf{v}_n \\
&= \lambda_1^k \cdot \left(\alpha_1 \cdot \mathbf{v}_1 + \alpha_2 \cdot \left(\frac{\lambda_2}{\lambda_1}\right)^k \cdot \mathbf{v}_2 + \cdots + \alpha_n \cdot \left(\frac{\lambda_n}{\lambda_1}\right)^k \cdot \mathbf{v}_n \right).
\end{aligned}$$

Ahora bien, como para todo $i \in \{1, \ldots, n\}$ se cumple que $|\lambda_1| > |\lambda_i|$, entonces

$$\lim_{k\to\infty} \left(\frac{\lambda_i}{\lambda_1}\right)^k = 0.$$

Si llamamos $\mathbf{r}^{(k)} = \dfrac{\mathbf{x}^{(k)}}{\lambda_1^k}$, tenemos que

$$\lim_{k\to\infty} \mathbf{r}^{(k)} = \lim_{k\to\infty} \left(\alpha_1 \cdot \mathbf{v}_1 + \alpha_2 \cdot \left(\frac{\lambda_2}{\lambda_1}\right)^k \cdot \mathbf{v}_2 + \cdots + \alpha_n \cdot \left(\frac{\lambda_n}{\lambda_1}\right)^k \cdot \mathbf{v}_n \right) = \alpha_1 \cdot \mathbf{v}_1.$$

De aquí podemos extraer dos conclusiones, siempre y cuando $\alpha_1 \neq 0$:

- $\lim\limits_{k\to\infty} \dfrac{\|\mathbf{x}^{(k)}\|}{\|\mathbf{x}^{(k-1)}\|} = \lim\limits_{k\to\infty} \dfrac{|\lambda_1^k|}{|\lambda_1^{k-1}|} \cdot \dfrac{\|\mathbf{r}^{(k)}\|}{\|\mathbf{r}^{(k-1)}\|} = |\lambda_1|.$

  Aquí $\|\bullet\|$ puede ser cualquier norma inducida; más aún, se puede considerar cualquier función que respete el producto por un escalar, tome siempre valores no negativos y se anule solo en el cero; una elección sencilla, por ejemplo, es la norma infinito.

Por lo tanto, el cociente entre las normas de dos iteraciones sucesivas del método de las potencias converge al valor absoluto del autovalor principal de $\mathbf{A}$.

- Para k lo suficientemente grande, $\mathbf{x}^{(k)} \approx \lambda_1^k \cdot \alpha_1 \cdot \mathbf{v}_1$, que es múltiplo de $\mathbf{v}_1$ y, por lo tanto, es un autovector principal de $\mathbf{A}$.

En general, al aplicar el método de las potencias, puede resultar convienente ir normalizando $\mathbf{x}^{(k)}$ en cada paso para evitar que sus componentes alcancen valores extremos, lo cual provocaría que se pierda precisión.

La única hipótesis que el método requiere sobre $\mathbf{x}^{(0)}$ es que α_1, su componente en la dirección del autovector principal, no sea nula. Esto suele ser difícil de garantizar, justamente porque no se conoce dicho autovector. La solución suele ser elegir $\mathbf{x}^{(0)}$ de manera arbitraria y modificar esta elección en caso de que el método no converja.

El método se puede completar con una técnica conocida como **deflación**, que permite, una vez conocido λ_1, seguir obteniendo autovalores de $\mathbf{A}$ en orden de valor absoluto decreciente. Consiste en definir

$$\mathbf{A}' = \mathbf{A} - \lambda_1 \cdot \mathbf{u}_1 \cdot \mathbf{u}_1^{\mathrm{T}},$$

donde $\mathbf{u}_1$ es un autovector unitario asociado a λ_1. La matriz $\mathbf{A}'$ tiene autovalores $0, \lambda_2, \ldots, \lambda_n$, por lo que si $\lambda_2 > \lambda_3$, puede volver a aplicarse el método de la potencia.

5.2. Método de la potencia inversa

El **método de la potencia inversa** es una variante del método de la potencia que permite, dada una matriz inversible $\mathbf{A}$, encontrar su autovalor (y autovector asociado) de módulo mínimo, si el mismo existe y tiene multiplicidad simple. Se basa en el hecho de que, si los autovalores de $\mathbf{A}$ son

$$\lambda_1, \ldots, \lambda_n,$$

con $|\lambda_1| < |\lambda_i|$ para todo $i \in \{2, \ldots, n\}$, entonces los autovalores de $\mathbf{A}^{-1}$ son

$$\lambda_1^{-1}, \ldots, \lambda_n^{-1},$$

con $|\lambda_1^{-1}| > |\lambda_i^{-1}|$ para todo $i \in \{2, \ldots, n\}$. Por lo tanto, basta con aplicar el método de las potencias sobre $\mathbf{A}^{-1}$ para obtener $|\lambda_1^{-1}|$.

Una variante interesante del método de la potencia inversa permite, dado un valor $\mu \in \mathbb{R}$, encontrar el autovalor de $\mathbf{A}$ más cercano a μ. Consiste en aplicar el método de la potencia sobre la matriz $(\mathbf{A} - \mu \cdot \mathbf{I})^{-1}$, que tiene como autovalores a

$$(\lambda_1 - \mu)^{-1}, \ldots, (\lambda_n - \mu)^{-1};$$

el autovalor λ_i de $\mathbf{A}$ que minimiza la distancia con μ es también el que maximiza el valor de $(\lambda_i - \mu)^{-1}$.

6. Interpolación polinómica

El problema de **interpolación** consiste en, dado un conjunto de puntos $(x_0, f(x_0)), \ldots, (x_n, f(x_n))$, encontrar una función I tal que para todo $i \in \{0, \ldots, n\}$ se cumpla que $I(x_i) = f(x_i)$. Este problema tiene múltiples aplicaciones; resulta útil, por ejemplo:

- cuando se tiene una serie de muestras discretas de alguna función y se busca reconstruir la función a partir de las mismas, por ejemplo, para aumentar la granularidad de las muestras agregando valores intermedios (esto es útil en el caso de funciones que se encuentran tabuladas), o para realizar una representación gráfica.
- cuando se quiere estimar el valor de una función en un punto desconocido a partir de los valores que toma en otros puntos.
- para evaluar, derivar, integar, etc. una versión más simple de una función complicada; por ejemplo, obteniendo un polinomio que tenga un comportamiento similar.
- en aplicaciones relacionadas con procesamiento multimedia, como redimensionamiento de imágenes, *slow motion* en videos, etc.

La forma más común de interpolación es la **interpolación polinómica**, donde la función interpolante es un polinomio, o una combinación de polinomios. Esto se debe principalmente a que los polinomios son una clase de funciones muy estudiada, y tienen múltiples propiedades deseables; por ejemplo, son sencillos de evaluar, derivar e integrar.

6.1. Interpolación con polinomio único

6.1.1. Polinomio interpolador de Lagrange

Dado un conjunto de puntos $(x_0, f(x_0)), \ldots, (x_n, f(x_n))$, el **polinomio interpolador de Lagrange** en $x_0, \ldots, x_n$ se define como

$$P(X) := \sum_{k=0}^{n} f(x_k) \cdot L_k(X) \qquad \text{donde} \qquad L_k(X) := \prod_{\substack{j=0 \\ j \neq k}}^{n} \frac{(X - x_j)}{(x_k - x_j)}.$$

Podemos observar que el polinomio de Lagrange interpola correctamente los puntos, dado que para todos $k, i \in \{0, \ldots, n\}$,

$$L_k(x_i) = \begin{cases} 1 & \text{si } k = i \\ 0 & \text{si } k \neq i \end{cases}$$

de donde, para todo $i \in \{0, \ldots, n\}$, $P(x_i) = \sum_{\substack{k=0 \\ k \neq i}}^{n} f(x_k) \cdot L_k(x_i) + f(x_i) \cdot L_i(x_i) = f(x_i)$.

Una propiedad interesante del polinomio interpolador de Lagrange es que, si $x_0, \ldots, x_n \in [a, b]$ y $f \in \mathcal{C}^{n+1}([a, b])$, entonces podemos dar, para cualquier punto intermedio entre las muestras utilizadas, una aproximación del **error** cometido en la interpolación (es decir, de la diferencia entre el resultado de evaluar en dicho punto el polinomio obtenido y el valor correspondiente a la función f que produjo los valores a interpolar) en términos de la $n+1$-ésima derivada de f. En particular, para todo $x \in [a, b]$ existe $\xi_x \in [a, b]$ tal que

$$f(x) = P(x) + \frac{f^{(n+1)}(\xi_x)}{(n+1)!} \cdot \prod_{i=0}^{n} (x - x_i)$$

Otra propiedad destacada del polinomio de Lagrange es su **unicidad**: es el único polinomio de grado menor o igual que n que interpola un conjunto de puntos $(x_0, f(x_0)), \ldots, (x_n, f(x_n))$. En efecto, sea P el polinomio interpolador de Lagrange en los puntos $x_0, \ldots, x_n$, y supongamos que existe otro polinomio Q, de grado menor o igual que n, tal que, para todo $i \in \{0, \ldots, n\}$, $Q(x_i) = f(x_i)$. Ahora bien, la función $Q(x) \in \mathcal{C}^{n+1}([x_0, x_n])$, con $Q^{(n+1)}(x) \equiv 0$, y su polinomio interpolador de Lagrange en los los puntos $x_0, \ldots, x_n$ es precisamente P. Así, podemos aplicar la fórmula del error, de donde, para todo $x \in [x_0, x_n]$,

$$Q(x) = P(x) + \frac{Q^{(n+1)}(\xi_x)}{(n+1)!} \cdot \prod_{i=0}^{n}(x - x_i) = P(x), \quad \text{para algún } \xi_x \in [x_0, x_n].$$

6.1.2. Diferencias divididas

La forma de **diferencias divididas** es una manera alternativa de escribir el polinomio interpolador de Lagrange. Presenta una ventaja importante sobre la forma expuesta arriba; si se tiene el polinomio interpolador en una serie de puntos $x_0, \ldots, x_n$ y se agrega un nuevo punto x_{n+1}, puede obtenerse un nuevo polinomio interpolador a partir del anterior, sin necesidad de recomputarlo por completo.

Partimos de la siguiente definición recursiva:

- La **diferencia dividida de orden 0** en x_j es, para $j = 0, \ldots, n$:

$$f[x_j] := f(x_j).$$

- La **diferencia dividida de orden 1** en x_j es, para $j = 0, \ldots, n-1$:

$$f[x_j, x_{j+1}] := \frac{f(x_{j+1}) - f(x_j)}{x_{j+1} - x_j} = \frac{f[x_{j+1}] - f[x_j]}{x_{j+1} - x_j}.$$

- La **diferencia dividida de orden 2** en x_j es, para $j = 0, \ldots, n-2$:

$$f[x_j, x_{j+1}, x_{j+2}] := \frac{f[x_{j+1}, x_{j+2}] - f[x_j, x_{j+1}]}{x_{j+2} - x_j}.$$

- En general, la **diferencia dividida de orden k** en x_j es, para $j = 0, \ldots, n-k$:

$$f[x_j, \ldots, x_{j+k}] := \frac{f[x_{j+1}, \ldots, x_{j+k}] - f[x_j, \ldots, x_{j+k-1}]}{x_{j+k} - x_j}.$$

Afirmamos que, si P es el polinomio interpolador para los puntos $x_0, \ldots, x_n$, entonces

$$P(X) = \sum_{i=0}^{n} f[x_0, \ldots, x_i] \left(\prod_{j=0}^{i-1} (X - x_j) \right)$$

es decir,

$$\begin{aligned} P(X) = f[x_0] + f[x_0, x_1](X - x_0) + f[x_0, x_1, x_2](X - x_0)(X - x_1) \\ + \cdots + f[x_0, x_1, \ldots, x_n](X - x_0)(X - x_1)\ldots(X - x_n). \end{aligned}$$

Este resultado, que se conoce a veces como **forma de Newton** del polinomio interpolador, puede demostrarse por inducción en la cantidad de puntos interpolados. El polinomio queda escrito en una serie de términos que dependen de una cantidad incremental de los puntos interpolados; más específicamente, el término i-ésimo del polinomio depende solo de los valores $x_0, \ldots, x_i$. Así, para agregar un nuevo punto a interpolar, basta con agregar un nuevo término al polinomio. Este

término, además, se puede calcular de forma eficiente, ya que su coeficiente es la diferencia dividida $f[x_0, \ldots, x_{n+1}]$, que, gracias a la estructura recursiva de las diferencias divididas, se puede computar reutilizando resultados anteriores.

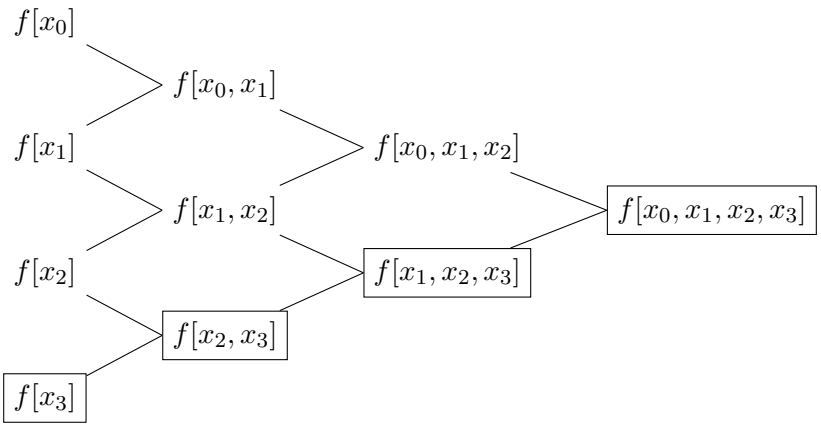

Figura 2: Diferencias divididas que es necesario computar para agregar un cuarto punto a un conjunto de tres puntos ya interpolados.

6.1.3. Método de Neville

El polinomio interpolador para los puntos $x_1, \ldots, x_n$ admite también una formulación recursiva, en términos de dos polinomios que interpolan, cada uno, $n-1$ puntos. Más específicamente, para cualesquiera $0 \leq i, j \leq n$, con $i \neq j$, se tiene que

$$P(X) = \frac{(X - x_j) \cdot P_j(X) - (X - x_i) \cdot P_i(X)}{x_i - x_j},$$

donde P_i denota al polinomio interpolador para los puntos $x_0, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$, y P_j al polinomio interpolador para $x_0, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n$.

No es difícil ver que este polinomio interpola correctamente todos los puntos:

- En x_i, tenemos que $P(x_i) = \frac{(x_i - x_j) \cdot P_j(x_i) - (x_i - x_i) \cdot P_i(x_i)}{x_i - x_j} = P_j(x_i) = f(x_i)$.
- Análogamente, en x_j, tenemos que $P(x_j) = \frac{(x_j - x_j) \cdot P_j(x_j) - (x_j - x_i) \cdot P_i(x_j)}{x_i - x_j} = P_i(x_j) = f(x_j)$.
- En x_k, con $k \neq i, j$, tenemos que $P(x_k) = \frac{(x_k - x_j) \cdot P_j(x_k) - (x_k - x_i) \cdot P_i(x_k)}{x_i - x_j} = f(x_k) \cdot \frac{(x_k - x_j) - (x_k - x_i)}{x_i - x_j} = f(x_k)$.

Esta escritura recursiva da origen al **método de Neville**, que es otra manera de construir polinomios interpoladores de forma incremental, permitiendo obtener un polinomio interpolador para $n+1$ puntos a partir de uno que interpola un subconjunto de n de estos puntos.

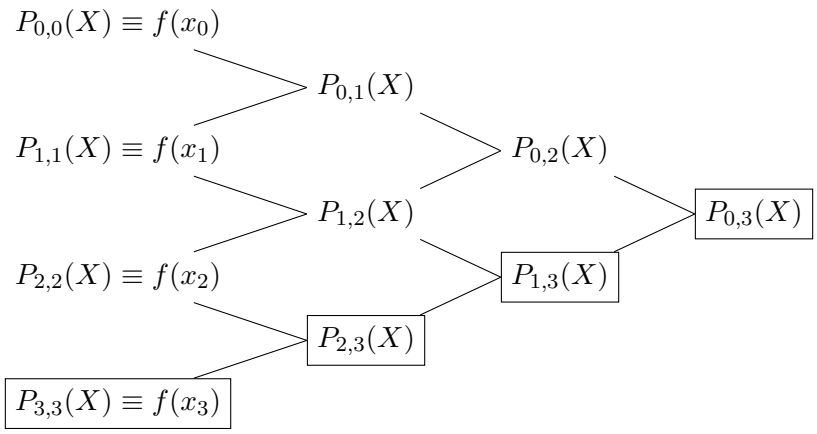

Figura 3: Extensión de un polinomio que interpola los puntos $x_0, \ldots, x_3$ para interpolar también el punto x_4. La notación $P_{i,j}$ indica, para $0 \leq i \leq j \leq 4$, el polinomio interpolador en los puntos $x_i, x_{i+1}, \ldots, x_j$.

6.2. Interpolación fragmentaria

Cuando se busca interpolar una cantidad de muestras de cierto tamaño, el polinomio interpolador empieza a dar cuenta de algunos problemas. Esto se debe a que en general, a medida que crece la cantidad de valores a interpolar, aumenta también el grado del polinomio interpolador. El inconveniente es que los polinomios de grado elevado tienden a oscilar demasiado, lo cual suele no ajustarse bien al comportamiento de la función que produjo las muestras y, por lo tanto es un comportamiento poco deseado al interpolar.

Una alternativa ante esta situación es la **interpolación fragmentaria**, que consiste en utilizar polinomios distintos, de menor grado, en diferentes fragmentos del intervalo donde se quiere interpolar. El resultado ya no será un polinomio, sino una función compuesta de muchos polinomios "pegados" en los extremos.

En general, buscaremos definir n polinomios distintos, $S_1, \ldots, S_n$, uno para cada par de puntos consecutivos entre los valores a interpolar. La idea es que estos polinomios cumplan $S_i(x_{i-1}) = f(x_{i-1})$ y $S_i(x_i) = f(x_i)$, para $i = 1, \ldots, n$. Luego, podremos construir la función

$$S(x) := \begin{cases} S_1(x) & \text{si } x \in [x_0, x_1) \\ S_2(x) & \text{si } x \in [x_1, x_2) \\ \quad \vdots & \\ S_n(x) & \text{si } x \in [x_{n-1}, x_n]. \end{cases}$$

6.2.1. Interpolación fragmentaria lineal

La forma más sencilla de interpolación fragmentaria es la **interpolación lineal**, donde cada uno de los S_i es un polinomio de grado menor o igual que 1, que interpola correctamente en los puntos x_{i-1} y x_i[5]. Los S_i se definen, para $i = 1, \ldots, n$, como

$$S_i(X) := f(x_{i-1}) + \frac{f(x_i) - f(x_{i-1})}{x_i - x_{x-1}} \cdot (X - x_{i-1}).$$

[5]Cada S_i es, así, el polinomio interpolador de Lagrange para los puntos x_{i-1}, x_i.

La interpolación lineal es sencilla, fácil de calcular (incluso en forma manual) y en muchas ocasiones resulta suficiente para el problema que se busca resolver. Sin embargo, la función que se obtiene no es derivable en los puntos x_i, donde presenta un aspecto "puntiagudo" que no resulta natural. Este problema puede resolverse utilizando polinomios de mayor grado.

6.2.2. Interpolación fragmentaria cuadrática

Si cada S_i se define para ser un polinomio **cuadrático**, se tienen más parámetros para definir, que se pueden aprovechar para que la función S obtenida sea derivable en todo su dominio. La idea es definir los S_i en la forma

$$S_i(X) := a_i(X - x_i)^2 + b_i(X - x_i) + c_i$$

y determinar valores para los a_i, b_i y c_i de modo que se cumplan las siguientes condiciones:

(i) S es interpoladora:
- $S_i(x_{i-1}) = f(x_{i-1})$, para $i = 1, \ldots, n$ (n ecuaciones).
- $S_n(x_n) = f(x_n)$ (1 ecuación).

(ii) S es continua: $S_i(x_i) = f(x_i)$, para $i = 1, \ldots, n-1$ ($n-1$ ecuaciones).

(iii) S es derivable: $S_i'(x_i) = S_{i+1}'(x_i)$, para $i = 1, \ldots, n-1$ ($n-1$ ecuaciones).

Se tiene un sistema de $3n$ incógnitas y $3n-1$ ecuaciones, lo cual deja una única ecuación libre para pedir alguna propiedad adicional. En general, esta se utiliza para controlar el comportamiento de la derivada en alguno de los extremos x_0 o x_n, con el inconveniente de que se obtiene una solución asimétrica, ya que es imposible pedir condiciones simultáneamente sobre los dos extremos y mantener la certeza de que el sistema resulta compatible.

6.2.3. *Splines* cúbicos

La utilización de polinomios **cúbicos** para definir cada S_i resuelve el problema recién mencionado, y permite a su vez obtener una función interpoladora que es dos veces derivable en todo su dominio. Para lograrlo, los S_i se definen en la forma

$$S_i(X) := a_i(X - x_i)^3 + b_i(X - x_i)^2 + c_i(X - x_i) + d_i$$

y los valores para cada a_i, b_i, c_i y d_i se determinan de modo tal que:

(i) S es interpoladora:
- $S_i(x_{i-1}) = f(x_{i-1})$, para $i = 1, \ldots, n$ (n ecuaciones).
- $S_n(x_n) = f(x_n)$ (1 ecuación).

(ii) S es continua: $S_i(x_i) = f(x_i)$, para $i = 1, \ldots, n-1$ ($n-1$ ecuaciones).

(iii) S es derivable: $S_i'(x_i) = S_{i+1}'(x_i)$, para $i = 1, \ldots, n-1$ ($n-1$ ecuaciones).

(iv) S es dos veces derivable: $S_i''(x_i) = S_{i+1}''(x_i)$, para $i = 1, \ldots, n-1$ ($n-1$ ecuaciones).

El sistema que resulta tiene $4n$ incógnitas y $4n-2$ ecuaciones, con lo que pueden agregarse dos nuevas ecuaciones, para condicionar el comportamiento en los puntos frontera, x_0 y x_n. Esto puede hacerse de dos formas alternativas:

(v) (a) Frontera sujeta:
- $S_1'(x_0) = f'(x_0)$.

- $S_n'(x_n) = f'(x_n)$.

(b) Frontera natural: $S_1''(x_0) = S_n''(x_n) = 0$.

En ambos casos, el sistema de ecucaciones que se obtiene es tridiagonal y estrictamente diagonal dominante. Esto permite resolverlo computacionalmente de forma eficiente, y asegura que siempre tiene solución única.

7. Cuadrados mínimos lineales

Un problema muy frecuente en distintos ámbitos consiste en, dados un conjunto de puntos del plano $(x_1, y_1), \ldots, (x_m, y_m)$ y una familia de funciones $\mathcal{F}$ determinadas por ciertos parámetros, hallar cuál de estas funciones cuyo gráfico se ajusta mejor (es decir, "pasa más cerca") a dichos puntos. Esto permite, por ejemplo, considerar los resultados de un experimento y predecir los valores que se obtendrían para distintos valores de una variable involucrada.

Las familias de funciones que tendremos en cuenta serán siempre de la forma

$$\mathcal{F} = \{a_1 \cdot \varphi_1 + \cdots + a_n \cdot \varphi_n : a_1, \ldots, a_n \in \mathbb{R}\}$$

donde $\varphi_1, \ldots, \varphi_n$ son funciones reales fijas. A modo de ejemplo, podemos considerar la familia de funciones lineales (en cuyo caso tendremos dos parámetros), de funciones cuadráticas (donde habrá tres parámetros), etc.

Existen varios criterios para cuantificar cómo se ajusta una función f a un determinado conjunto de puntos. Por ejemplo, tres de ellos son:

- Minimizar el *máximo error* (*minimax*) entre cada uno de los puntos y el gráfico de la función, es decir, considerar como métrica

  $$\max_{1 \le i \le m} |f(x_i) - x_i| .$$

  Este criterio tiene la desventaja ser muy susceptible a la presencia de *outliers*.

- Minimizar el *error absoluto* entre los puntos y el gráfico de la función:

  $$\sum_{i=1}^{m} |f(x_i) - x_i| .$$

  Este método es mucho más estable ante *outliers*, pero tiene la complicación de que se busca minimizar una función que no es diferenciable en el origen.

- Minimizar el *error cuadrático*:

  $$\sum_{i=1}^{m} (f(x_i) - x_i)^2 ,$$

  que es el criterio que vamos a utilizar. Se trata de un método que da un mayor peso relativo a los *outliers*, pero sin permitir que dominen la métrica; además, tiene la ventaja de ser diferenciable en todo punto.

Entonces, queremos hallar valores para los parámetros $a_1, \ldots, a_n$ que realicen el mínimo

$$\min_{a_1, \ldots, a_n} \sum_{i=1}^{m} \left(a_1 \cdot \varphi_1(x_i) + \cdots + a_n \cdot \varphi_n(x_i) - y_i\right)^2$$

o, considerando,

$$\mathbf{A} = \begin{bmatrix} \varphi_1(x_1) & \varphi_2(x_1) & \cdots & \varphi_n(x_1) \\ \varphi_1(x_2) & \varphi_2(x_2) & \cdots & \varphi_n(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_1(x_m) & \varphi_2(x_m) & \cdots & \varphi_n(x_m) \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

buscamos hallar $\mathbf{x} \in \mathbb{R}^m$ que realice el mínimo

$$\min_{\mathbf{x}} \|\mathbf{A} \cdot \mathbf{x} - \mathbf{b}\|_2^2 .$$

A este último problema se lo conoce como el problema de **cuadrados mínimos lineales**: hallar, dadas $\mathbf{A} \in \mathbb{R}^{m \times n}$ y $\mathbf{b} \in \mathbb{R}^m$, un vector $\mathbf{x} \in \mathbb{R}^n$ que minimice la distancia entre $\mathbf{A} \cdot \mathbf{x}$ y $\mathbf{b}$.

Desde el punto de vista geométrico, existe una interpretación intuitiva para este problema. Para todo $\mathbf{x} \in \mathbb{R}^n$, el producto $\mathbf{A} \cdot \mathbf{x}$ describe una combinación lineal de las columnas de $\mathbf{A}$; en otras palabras, $\mathbf{x} \in \mathrm{Im}(\mathbf{A})$, el subespacio generado por las columnas de $\mathbf{A}$. De todos los elementos de este subespacio, buscamos aquel que se encuentre a distancia mínima de $\mathbf{b}$. Este elemento siempre existe y es único: se trata de la proyección ortogonal de $\mathbf{b}$ sobre el subespacio $\mathrm{Im}(\mathbf{A})$, es decir, $\mathrm{proy}_{\mathrm{Im}(\mathbf{A})}(\mathbf{b})$.

Más formalmente, recordemos que siempre es posible escribir $\mathbf{b}$ como $\mathbf{b} = \mathbf{b_1} + \mathbf{b_2}$, con

- $\mathbf{b_1} = \mathrm{proy}_{\mathrm{Im}(\mathbf{A})}(\mathbf{b}) \in \mathrm{Im}(\mathbf{A})$, y
- $\mathbf{b_2} = \mathrm{proy}_{\mathrm{Im}(\mathbf{A})^\perp}(\mathbf{b}) \in \mathrm{Im}(\mathbf{A})^\perp$, el complemento ortogonal de la imagen de $\mathbf{A}$.

Entonces, es posible demostrar que $\mathbf{x}$ realiza el mínimo buscado, y por lo tanto es solución del problema de cuadrados mínimos lineales, si y solo si $\mathbf{A} \cdot \mathbf{x} = \mathbf{b_1}$.

Del razonamiento anterior pueden extraerse tres conclusiones:

(i) Dado que $\mathbf{b_1} \in \mathrm{Im}(\mathbf{A})$, seguro existe $\mathbf{x}$ tal que $\mathbf{A} \cdot \mathbf{x} = \mathbf{b_1}$. Por lo tanto, el problema de cuadrados mínimos lineales siempre tiene solución.

(ii) Como $\mathbf{b_1}$ es único, la solución es única si y solo si hay una única forma de escribir a $\mathbf{b_1}$ como combinación lineal de las columnas de $\mathbf{A}$. Esto equivale a pedir que las columnas de $\mathbf{A}$ sean linealmente independientes, o que $\mathbf{A}$ sea de rango columna completo ($\mathrm{rg}(\mathbf{A}) = n$).

(iii) Si el sistema $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ tiene alguna solución $\mathbf{x}^*$, entonces $\mathbf{b} \in \mathrm{Im}(\mathbf{A})$, por lo que $\mathbf{b_1} = \mathbf{b}$, y por lo tanto $\mathbf{x}^*$ también será solución del problema de cuadrados mínimos lineales. Sin embargo, lo interesante de este problema es que tiene solución *incluso* cuando el sistema $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$ no la tiene, es decir, cuando está *sobredeterminado* (tiene ecuaciones que "se contradicen"). En estos casos, obviamente, el resultado hallado no será una solución de $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$, pero sí será la mejor aproximación posible según el criterio de minimizar el error cuadrático.

7.1. Ecuaciones normales

Una de las formas de encontrar la solución de un problema de cuadrados mínimos es resolviendo el siguiente sistema de ecuaciones, que recibe el nombre de **ecuaciones normales**:

$$\mathbf{A}^{\mathrm{T}} \cdot \mathbf{A} \cdot \mathbf{x} = \mathbf{A}^{\mathrm{T}} \cdot \mathbf{b}.$$

Para verificar que estas ecuaciones resuelven el problema, es necesario tener en cuenta en primer lugar que $\mathrm{Nul}(\mathbf{A}^{\mathrm{T}}) = \mathrm{Im}(\mathbf{A})^\perp$; es decir, multiplicar a izquierda por $\mathbf{A}^{\mathrm{T}}$ anula cualquier elemento que sea ortogonal a $\mathrm{Im}(\mathbf{A})$. Para comprender esto, basta notar que al tomar un vector $\mathbf{v}$ y multiplicarlo a izquierda por $\mathbf{A}^{\mathrm{T}}$, se está computando la suma de los productos internos entre $\mathbf{v}$ y cada una de las filas de $\mathbf{A}^{\mathrm{T}}$, que son las columnas de $\mathbf{A}$; si $\mathbf{v}$ es ortogonal a $\mathrm{Im}(\mathbf{A})$, entonces todos estos productos serán nulos.

Si, con este hecho en mente, observamos el lado izquierdo de la ecuación anterior, y escribimos $\mathbf{b} = \mathbf{b_1} + \mathbf{b_2}$ como antes, tenemos que

$$\mathbf{A}^{\mathrm{T}} \cdot \mathbf{b} = \mathbf{A}^{\mathrm{T}} \cdot (\mathbf{b_1} + \mathbf{b_2}) = \mathbf{A}^{\mathrm{T}} \cdot \mathbf{b_1} + \mathbf{A}^{\mathrm{T}} \cdot \mathbf{b_2} = \mathbf{A}^{\mathrm{T}} \cdot \mathbf{b_1}.$$

Por lo tanto, las ecuaciones normales equivalen a

$$\mathbf{A}^{\mathrm{T}} \cdot \mathbf{A} \cdot \mathbf{x} = \mathbf{A}^{\mathrm{T}} \cdot \mathbf{b_1}.$$

Esto quiere decir que cualquier solución de cuadrados mínimos, que debe satisfacer $\mathbf{A} \cdot \mathbf{x} = \mathbf{b_1}$, también será solución de las ecuaciones normales. Para ver que cualquier solución de las ecuaciones normales es una solución de cuadrados mínimos, consideremos $\mathbf{x}$ tal que $\mathbf{A}^{\mathrm{T}} \cdot \mathbf{A} \cdot \mathbf{x} = \mathbf{A}^{\mathrm{T}} \cdot \mathbf{b_1}$; en tal caso $\mathbf{A}^{\mathrm{T}} \cdot (\mathbf{A} \cdot \mathbf{x} - \mathbf{b_1}) = 0$, por lo que

$$\mathbf{A} \cdot \mathbf{x} - \mathbf{b_1} \in \mathrm{Nul}(\mathbf{A}^{\mathrm{T}}) = \mathrm{Im}(\mathbf{A})^{\perp}.$$

Por otra parte, como $\mathbf{A} \cdot \mathbf{x} \in \mathrm{Im}(\mathbf{A})$ y $\mathbf{b_1} \in \mathrm{Im}(\mathbf{A})$, necesariamente

$$\mathbf{A} \cdot \mathbf{x} - \mathbf{b_1} \in \mathrm{Im}(\mathbf{A}).$$

El único vector que está simultaneamente en $\mathrm{Im}(\mathbf{A})$ y en $\mathrm{Im}(\mathbf{A})^{\perp}$ es 0. Luego $\mathbf{A} \cdot \mathbf{x} - \mathbf{b_1} = 0$, por lo que $\mathbf{A} \cdot \mathbf{x} = \mathbf{b_1}$ y entonces $\mathbf{x}$ es una solución de cuadrados mínimos.

A modo de curiosidad, las ecuaciones normales son exactamente las que se obtienen si se calculan las derivadas parciales de la expresión del error cuadrático en función de cada una de las componentes de $\mathbf{x}$ y se las iguala a cero.

El sistema de ecuaciones que se obtiene al formular las ecuaciones normales tiene la buena propiedad de que su matriz, $\mathbf{A}^{\mathrm{T}} \cdot \mathbf{A}$, es semi-definida positiva, con lo cual puede usarse la factorización de Cholesky para resolverlo de forma eficiente.

No obstante, las ecuaciones normales también presentan un inconveniente: no garantizan la estabilidad numérica. El número de condición de la matriz $\mathbf{A}$, que ya de por sí puede no ser bueno, se eleva al cuadrado al computar $\mathbf{A}^{\mathrm{T}} \cdot \mathbf{A}$. Esto motiva la utilización de otros métodos más estables, que aprovechan algunas de las factorizaciones matriciales estudiadas anteriormente.

7.2. Resolución con factorización QR

Un método para resolver el problema de cuadrados mínimos es utilizar la factorización QR. Recordemos que toda matriz admite una factorización QR. Si escribimos $\mathbf{A} = \mathbf{Q} \cdot \mathbf{R}$, la expresión que buscamos minimizar se puede reescribir como

$$\|\mathbf{A} \cdot \mathbf{x} - \mathbf{b}\|_2^2 = \|\mathbf{Q} \cdot \mathbf{R} \cdot \mathbf{x} - \mathbf{b}\|_2^2 .$$

Pero como $\mathbf{Q}$ es una matriz ortogonal, multiplicar por $\mathbf{Q}^{\mathrm{T}}$ no altera la norma 2, por lo que resulta que

$$\|\mathbf{A} \cdot \mathbf{x} - \mathbf{b}\|_2^2 = \left\|\mathbf{Q}^{\mathrm{T}} \cdot (\mathbf{Q} \cdot \mathbf{R} \cdot \mathbf{x} - \mathbf{b})\right\|_2^2 . = \left\|\mathbf{R} \cdot \mathbf{x} - \mathbf{Q}^{\mathrm{T}} \cdot \mathbf{b}\right\|_2^2 .$$

En definitiva, nuestro problema se convierte en hallar $\mathbf{x}$ que realice el mínimo

$$\min_{\mathbf{x}} \left\|\mathbf{R} \cdot \mathbf{x} - \mathbf{Q}^{\mathrm{T}} \cdot \mathbf{b}\right\|_2^2 .$$

Para hallar este mínimo hay que proceder de una forma ligeramente distinta en base a dos casos, que dependen de $k = \mathrm{rg}(\mathbf{A})$. En ambos casos es necesario tener en cuenta que, como $\mathbf{Q}$ es inversible, entonces $\mathrm{rg}(\mathbf{R}) = \mathrm{rg}(\mathbf{A}) = k$.

(i) Si $\mathbf{A}$ es de rango columna completo, podemos escribir:

$$\mathbf{R} = \left[\begin{array}{c} \mathbf{R_1} \\ \hline \mathbf{0} \end{array}\right] \qquad \mathbf{Q}^{\mathrm{T}} \cdot \mathbf{b} = \left[\begin{array}{c} \mathbf{c} \\ \hline \mathbf{d} \end{array}\right]$$

donde $\mathbf{R_1} \in \mathbb{R}^{n\times n}$ es triangular superior, $\mathbf{c} \in \mathbb{R}^n$ y $\mathbf{d} \in \mathbb{R}^{m-n}$.

Entonces, resulta que:

$$\min_{\mathbf{x}} \left\|\mathbf{R}\cdot\mathbf{x} - \mathbf{Q}^{\mathrm{T}}\cdot\mathbf{b}\right\|_2^2 = \min_{\mathbf{x}} \left\| \left[\begin{array}{c} \mathbf{R_1}\cdot\mathbf{x} \\ \hline \mathbf{0} \end{array}\right] - \left[\begin{array}{c} \mathbf{c} \\ \hline \mathbf{d} \end{array}\right] \right\|_2^2 = \min_{\mathbf{x}} \left\| \left[\begin{array}{c} \mathbf{R_1}\cdot\mathbf{x} - \mathbf{c} \\ \hline -\mathbf{d} \end{array}\right] \right\|_2^2$$

$$= \min_{\mathbf{x}} \|\mathbf{R_1}\cdot\mathbf{x} - \mathbf{c}\|_2^2 + \|\mathbf{d}\|_2^2$$

Basta con minimizar el primer término de la expresión, ya que es el único que depende de $\mathbf{x}$. En efecto, el mínimo se alcanza si dicho término se anula, es decir, si $\mathbf{x}$ es solución del sistema

$$\mathbf{R_1}\cdot\mathbf{x} = \mathbf{c},$$

que siempre tiene solución única por ser $\mathbf{R_1}$ cuadrada y de rango completo.

(ii) Si $\mathbf{A}$ no es de rango columna completo, es decir, $k < n$, necesitaremos que la factorización QR haya sido obtenida con *pivoteo de columnas*; esto quiere decir, que haya sido construida de forma tal que las columnas incompletas de $\mathbf{R}$ se encuentren todas al final. En general, se permutan las columnas de $\mathbf{R}$ de forma tal que $|r_{1,1}| \geq |r_{2,2}| \geq \cdots \geq |r_{n,n}|$. Así,

$$\mathbf{A} = \mathbf{Q}\cdot\mathbf{R}\cdot\mathbf{P},$$

donde $\mathbf{P}$ es la matriz de permutación correspondiente al pivoteo. Por lo tanto, el mínimo buscado será

$$\min_{\mathbf{x}} \left\|\mathbf{R}\cdot\mathbf{P}\cdot\mathbf{x} - \mathbf{Q}^{\mathrm{T}}\cdot\mathbf{b}\right\|_2^2 = \min_{\mathbf{x}} \left\|\mathbf{R}\cdot\tilde{\mathbf{x}} - \mathbf{Q}^{\mathrm{T}}\cdot\mathbf{b}\right\|_2^2$$

donde $\tilde{\mathbf{x}} = \mathbf{P}\cdot\mathbf{x}$. Resolveremos el sistema para $\tilde{\mathbf{x}}$; terminado el proceso, deberemos tener en cuenta que las soluciones halladas tendrán permutadas sus componentes.

Podemos escribir, entonces:

$$\mathbf{R} = \left[\begin{array}{cc} \mathbf{R_1} & \mathbf{R_2} \\ \hline \mathbf{0} & \mathbf{0} \end{array}\right] \qquad \mathbf{Q}^{\mathrm{T}}\mathbf{b} = \left[\begin{array}{c} \mathbf{c} \\ \hline \mathbf{d} \end{array}\right] \qquad \tilde{\mathbf{x}} = \left[\begin{array}{c} \mathbf{x_1} \\ \hline \mathbf{x_2} \end{array}\right]$$

donde $\mathbf{R_1} \in \mathbb{R}^{k\times k}$ es triangular superior, $\mathbf{R_2} \in \mathbb{R}^{k\times n-k}$, $\mathbf{c}, \mathbf{x_1} \in \mathbb{R}^k$ y $\mathbf{d}, \mathbf{x_2} \in \mathbb{R}^{n-k}$.

De lo anterior,

$$\min_{\mathbf{x}} \left\|\mathbf{R}\cdot\mathbf{x} - \mathbf{Q}^{\mathrm{T}}\cdot\mathbf{b}\right\|_2^2 = \min_{\mathbf{x}} \left\| \left[\begin{array}{c} \mathbf{R_1}\cdot\mathbf{x_1} + \mathbf{R_2}\cdot\mathbf{x_2} - \mathbf{c} \\ \hline -\mathbf{d} \end{array}\right] \right\|_2^2$$

$$= \min_{\mathbf{x}} \|\mathbf{R_1}\cdot\mathbf{x_1} + \mathbf{R_2}\cdot\mathbf{x_2} - \mathbf{c}\|_2^2 + \|\mathbf{d}\|_2^2$$

De nuevo, esta expresión alcanza el mínimo cuando el primer término es nulo. Las soluciones son infinitas: $\mathbf{x_2}$ puede fijarse libremente en cualquier $\mathbf{x_2^*}$, y luego se determina $\mathbf{x_1}$ como la única solución del sistema

$$\mathbf{R_1} \cdot \mathbf{x_1} = \mathbf{c} - \mathbf{R_2} \cdot \mathbf{x_2^*}.$$

Como ya se mencionó cuando se habló de la factorización QR como método para resolver sistemas de ecuaciones lineales, se trata de un método numéricamente muy estable, debido a que la matriz $\mathbf{Q}$ es ortogonal. Sin embargo, al utilizarlo con matrices de rango incompleto, se vuelve necesario incorporar el pivoteo; esto complejiza el algoritmo y lo vuelve poco estable. Además, hallar el rango de $\mathbf{A}$ puede resultar difícil debido a errores de redondeo.

7.3. Resolución con descomposición en valores singulares

Otra posibilidad para resolver el problema de cuadrados mínimos es utilizar la descomposición en valores singulares de $\mathbf{A} = \mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{V}^{\mathrm{T}}$. En este caso, como $\mathbf{U}$ es ortogonal, multiplicar por $\mathbf{U}^{\mathrm{T}}$ no modifica la norma 2, y tenemos que:

$$\|\mathbf{A} \cdot \mathbf{x} - \mathbf{b}\|_2^2 = \left\|\mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{V}^{\mathrm{T}} \cdot \mathbf{x} - \mathbf{b}\right\|_2^2 = \left\|\mathbf{\Sigma} \cdot \mathbf{V}^{\mathrm{T}} \cdot \mathbf{x} - \mathbf{U}^{\mathrm{T}} \cdot \mathbf{b}\right\|_2^2.$$

Si llamamos $\mathbf{V}^{\mathrm{T}} \cdot \mathbf{x} = \mathbf{y}$, como $\mathbf{V}^{\mathrm{T}}$ es inversible, tenemos un sistema de ecuaciones determinado: si encontramos un valor que nos sirva para $\mathbf{y}$, podemos determinar cuál debe ser el valor de $\mathbf{x}$. Entonces, sustituyendo, obtenemos que

$$\|\mathbf{A} \cdot \mathbf{x} - \mathbf{b}\|_2^2 = \left\|\mathbf{\Sigma} \cdot \mathbf{y} - \mathbf{U}^{\mathrm{T}} \cdot \mathbf{b}\right\|_2^2,$$

y, por lo tanto, el problema de minimización a resolver es

$$\min_{\mathbf{x}} \left\|\mathbf{\Sigma} \cdot \mathbf{y} - \mathbf{U}^{\mathrm{T}} \cdot \mathbf{b}\right\|_2^2.$$

Volvemos a separar en dos casos según $k = \mathrm{rg}(\mathbf{A}) = \mathrm{rg}(\mathbf{\Sigma})$.

(i) Si $\mathbf{A}$ es de rango columna completo, podemos escribir:

$$\mathbf{\Sigma} = \left[\begin{array}{ccc} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \\ \hline & \mathbf{0} & \end{array}\right] \qquad \mathbf{U}^{\mathrm{T}} \cdot \mathbf{b} = \left[\begin{array}{c} \mathbf{c} \\ \hline \mathbf{d} \end{array}\right]$$

donde $\mathbf{c} \in \mathbb{R}^n$ y $\mathbf{d} \in \mathbb{R}^{m-n}$. Así,

$$\begin{aligned} \min_{\mathbf{y}} \left\|\mathbf{\Sigma} \cdot \mathbf{y} - \mathbf{U}^{\mathrm{T}} \cdot \mathbf{b}\right\|_2^2 &= \min_{\mathbf{y}} \left\| \left[\begin{array}{c} \sigma_1 \cdot y_1 \\ \vdots \\ \sigma_n \cdot y_n \\ \hline \mathbf{0} \end{array}\right] - \left[\begin{array}{c} \mathbf{c} \\ \hline \mathbf{d} \end{array}\right] \right\|_2^2 \\ &= \min_{\mathbf{y}} \left\| \begin{bmatrix} \sigma_1 \cdot y_1 - c_1 \\ \vdots \\ \sigma_n \cdot y_n - c_n \end{bmatrix} \right\|_2^2 + \|\mathbf{d}\|_2^2 \end{aligned}$$

Para alcanzar el mínimo basta con anular el primer término, lo cual sucede si y solo si se toma $\mathbf{y} = \left(\frac{c_1}{\sigma_1}, \ldots, \frac{c_n}{\sigma_n}\right)$.

(ii) Si $\mathbf{A}$ no es de rango columna completo ($k < n$), solo las primeras k entradas de la diagonal de $\mathbf{\Sigma}$ son no nulas. Escribimos entonces:

$$\mathbf{U}^{\mathrm{T}} \cdot \mathbf{b} = \left[\begin{array}{c} \mathbf{c} \\ \hline \mathbf{d} \end{array}\right]$$

con $\mathbf{c} \in \mathbb{R}^k$ y $\mathbf{d} \in \mathbb{R}^{m-k}$. Ahora,

$$\begin{aligned} \min_{\mathbf{y}} \left\| \mathbf{\Sigma} \cdot \mathbf{y} - \mathbf{U}^{\mathrm{T}} \cdot \mathbf{b} \right\|_2^2 &= \min_{\mathbf{y}} \left\| \left[\begin{array}{c} \sigma_1 \cdot y_1 \\ \vdots \\ \sigma_k \cdot y_k \\ \hline \mathbf{0} \end{array}\right] - \left[\begin{array}{c} \mathbf{c} \\ \hline \mathbf{d} \end{array}\right] \right\|_2^2 \\ &= \min_{\mathbf{y}} \left\| \begin{bmatrix} \sigma_1 \cdot y_1 - c_1 \\ \vdots \\ \sigma_k \cdot y_k - c_k \end{bmatrix} \right\|_2^2 + \|\mathbf{d}\|_2^2 \end{aligned}$$

De nuevo, para alcanzar el mínimo, debe anularse el primer término. Existen infinitas soluciones, las cuales se logran tomando $\mathbf{y} = \left(\frac{c_1}{\sigma_1}, \ldots, \frac{c_k}{\sigma_k}, y_{k+1}, \ldots, y_n\right)$, con $y_{k+1}, \ldots, y_n \in \mathbb{R}$ cualesquiera. Una posibilidad es tomarlos a todos iguales a 0, lo cual minimiza la norma de la solución $\mathbf{x}$ que se encontrará luego.

En ambos casos, una vez hallado $\mathbf{y}$, resta resolver el sistema $\mathbf{V}^{\mathrm{T}}\mathbf{x} = \mathbf{y}$ para hallar la solución $\mathbf{x}$ al problema de cuadrados mínimos.

Utilizar SVD para resolver el problema es más costoso que hacerlo mediante QR, si bien existen optimizaciones posibles, como no computar la matriz $\mathbf{U}$ de forma explícita. Más allá de las consideraciones de eficiencia, en casos de matrices de rango incompleto, su estabilidad numérica hace claramente preferible emplear SVD por encima de QR.

8. Ceros de funciones

Como ya hemos visto, a la hora de resolver sistemas de ecuaciones lineales, existe una gran variedad de métodos entre los cuales elegir. No obstante, si bien las ecuaciones lineales permiten modelar, o al menos aproximar, una inmensa cantidad de problemas, en muchas ocasiones se vuelve necesario resolver ecuaciones que no son lineales. Si bien para algunos casos particulares existen resoluciones analíticas, esto no es cierto en el caso general; a modo de ejemplo, para los polinomios de grado cinco o superior, es posible demostrar que no existen fórmulas algebraicas cerradas que permitan obtener sus raíces. Esto hace que se trate de un problema complejo, y que depende en gran medida de características particulares de la ecuación que se busca resolver.

En general, una **ecuación** no (necesariamente) lineal **en una variable real** es una expresión de la forma $f(x) = g(x)$, donde la variable x representa un valor desconocido. Para resolver la ecuación, se busca para esta variable un valor $x^* \in \mathbb{R}$ que satisfaga la igualdad.

Si definimos $h(x) = f(x) - g(x)$, lo que buscamos ahora es un valor x^* satisfaga $h(x^*) = 0$, es decir, un **cero** de la función h. A continuación, estudiaremos algunos métodos iterativos que permiten hallar ceros de funciones en el caso general. Es decir, dada una función f, buscaremos definir una sucesión $\{x_k\}_{k\in\mathbb{N}} \in \mathbb{R}^{\mathbb{N}}$ de modo que $\lim_{k\to\infty} x_k = x^*$, con $f(x^*) = 0$.

8.1. Orden de convergencia

Sea $\{x_k\}_{k\in\mathbb{N}}$ una sucesión tal que $\lim_{k\to\infty} x_k = x^*$. Decimos que $\{x_k\}_{k\in\mathbb{N}}$ tiene **orden de convergencia** p si

$$\lim_{k\to\infty} \frac{|x_{k+1} - x^*|}{(|x_k - x^*|)^p} = c > 0.$$

Informalmente, esto se puede interpretar de la siguiente forma: tomando valores suficientemente grandes de k, se cumple que $|x_{k+1} - x^*| \approx c \cdot (|x_k - x^*|)^p$ para algún $c > 0$.

- Si $p = 1$, decimos que la convergencia es **lineal**.
- Si $p = 2$, decimos que la convergencia es **cuadrática**.
- Si $1 < p < 2$, decimos que la convergencia es **superlineal**.

El orden de convergencia nos permite estimar la velocidad con la que las aproximaciones arrojadas por el método analizado convergen al valor buscado. Un orden de convergencia más alto indica una convergencia más rápida.

Contamos además con el siguiente criterio para comparar el orden de convergencia de dos sucesiones: sean $\{\alpha_k\}_{k\in\mathbb{N}}$, $\{\beta_k\}_{k\in\mathbb{N}}$ dos sucesiones tales que $\alpha_k \xrightarrow{k\to\infty} \alpha$ y $\beta_k \xrightarrow{k\to\infty} 0$. Si existe algún $k_0 \in \mathbb{N}$ tal que, para todo $k \geq k_0$, $|\alpha_k - \alpha| \leq |\beta_k|$, entonces α_k se acerca a α al menos tan rápidamente como β_k se acerca a 0. En tal caso, si $\{\beta_k\}_{k\in\mathbb{N}}$ tiene orden de convergencia p, decimos que $\{\alpha_k\}_{k\in\mathbb{N}}$ tiene orden de convergencia al menos p.

8.2. Criterios de parada

Es necesario contar con un criterio que permita decidir cuándo la aproximación ya es lo suficientemente buena, para así detener la ejecución del algoritmo. Esto se conoce como **criterio de parada**. Algunos de los criterios más comúnmente utilizados en algoritmos para la búsqueda de ceros de funciones son:

(i) Establecer una cantidad fija de iteraciones. Es el criterio más sencillo, pero es insensible a las características del método usado y no permite decidir *a priori* la precisión de los resultados.

(ii) Fijar un valor $\varepsilon > 0$ y parar cuando $|x_{k+1} - x_k| < \varepsilon$, es decir, cuando el ritmo de convergencia sea lo suficientemente lento. Si bien es un criterio más sofisticado, puede dar resultados erróneos. Por ejemplo, considerando la sucesión $x_k = \sum_{i=0}^{k} \frac{1}{k} = 1 + \frac{1}{2} + \cdots + \frac{1}{k}$, se tiene que $|x_{k+1} - x_k| = \frac{1}{k+1} \xrightarrow{k\to\infty} 0$. Luego, para cualquier valor de ε, el criterio terminará y arrojará un resultado supuestamente cercano al límite de la sucesión, que en realidad es divergente.

(iii) Fijar un valor $\varepsilon > 0$ y parar cuando $\frac{|x_{k+1}-x_k|}{|x_k|} < \varepsilon$. La idea, en este caso, es testear el error relativo de la aproximación. Sufre de problemas similares al criterio anterior, pero es un buen candidato a utilizar en la ausencia de información adicional.

(iv) Fijar un valor $\varepsilon > 0$ y parar cuando $f(x_k) < \varepsilon$. También puede dar falsos positivos, ya que f puede tomar valores arbitrariamente cercanos a 0 sin que esto indique la cercanía de una raíz.

(v) Fijar un valor $\varepsilon > 0$ y parar cuando $|f(x_{k+1}) - f(x_k)| < \varepsilon$.

(vi) Fijar un valor $\varepsilon > 0$ y parar cuando $\frac{|f(x_{k+1})-f(x_k)|}{|f(x_k)|} < \varepsilon$.

Como puede verse, todos estos criterios tienen casos patológicos en los que arrojan resultados falsos. Por este motivo, la elección del criterio de parada debe hacerse teniendo en cuenta las características del problema a resolver. Además, es posible emplearlos de forma combinada; por ejemplo, es común establecer un límite fijo de iteraciones incluso aunque se use un criterio de parada distinto, para evitar la posibilidad de caer en un *loop* infinito si la sucesión diverge.

8.3. Método de la bisección

Consideremos una función continua $f : [a, b] \to \mathbb{R}$ tal que $f(a) \cdot f(b) < 0$; es decir, cuyos valores en los extremos tienen signos opuestos. Entonces, por el teorema de Bolzano, la función tiene algún cero en el intervalo (a, b), es decir, existe $x^* \in (a, b)$ tal que $f(x^*) = 0$.

El **método de la bisección** permite, bajo dichas condiciones, encontrar este valor x^*. La idea es similar a la búsqueda binaria: se comienza por dividir el intervalo (a, b) en dos mitades. Necesariamente el valor x^* buscado estará en una de estas mitades; se identifica de cuál de las dos se trata, y se repite el procedimiento en el nuevo intervalo. De esta forma, con cada iteración, se obtiene un intervalo de menor longitud que contiene a la raíz buscada, lo cual permite aproximarla con precisión arbitraria.

El pseudocódigo del algoritmo es el siguiente (debe ser completado con algún criterio de parada).

Algoritmo de la bisección

Entrada: $a, b \in \mathbb{R}$, y $f : [a, b] \to \mathbb{R}$ tal que $f(a) \cdot f(b) < 0$
Salida: una aproximación de una raíz $x^* \in (a, b)$ de f

```
a_0 ← a
b_0 ← b
para k = 0, 1, 2, ... hacer
    c_k ← (a_k + b_k)/2
    si f(c_k) = 0 entonces
        devolver c_k
    si f(c_k) · f(a_k) < 0 entonces
        a_{k+1} ← a_k
        b_{k+1} ← c_k
    en otro caso
        a_{k+1} ← c_k
        b_{k+1} ← b_k
```

Es claro que, en el método de la bisección, $c_k \xrightarrow{k\to\infty} x^*$ con $f(x^*) = 0$. Más aún, para todo $k \in \mathbb{N}$, tanto x^* como c_k pertenecen al intervalo $[a_{k+1}, b_{k+1}]$, y como la longitud de estos intervalos se reduce a la mitad en cada iteración, tenemos que

$$|c_k - x^*| \leq |b_{k+1} - a_{k+1}| = \left|\frac{b-a}{2^{k+1}}\right|$$

es decir, el método de la bisección converge a x^* al menos tan rápidamente como la sucesión $\left\{\frac{b-a}{2^{k+1}}\right\}_{k\in\mathbb{N}}$ converge a 0. Esta última tiene un orden de convergencia lineal, lo cual nos permite afirmar que el método de la bisección se aproxima al menos linealmente a una raíz de f.

8.4. Algoritmos de punto fijo

Dada una función $g : [a, b] \to \mathbb{R}$, se llama **punto fijo** de g a un valor $p \in [a, b]$ tal que $g(p) = p$.

El problema de encontrar ceros de funciones está fuertemente relacionado con la búsqueda de puntos fijos. En efecto, dada una función f, existen diversas formas de definir una función g de modo tal que las raíces de f sean los puntos fijos de g. Por ejemplo, para $\lambda \neq 0$, se puede definir la función $g(x) = x + \lambda \cdot f(x)$. Así, para cualquier valor x^*,

$$g(x^*) = \lambda \cdot f(x^*) + x^* = x^* \qquad \text{sii} \qquad f(x^*) = 0$$

por lo que x^* será un punto fijo de g si y solo si es una raíz de f.

La ventaja de transformar el problema de esta manera radica en que existe una idea sumamente sencilla que puede aplicarse para buscar los puntos fijos de una función a través de un método iterativo. Considerando una función $g : [a, b] \to \mathbb{R}$ cualquiera y fijando algún punto inicial $x_0 \in [a, b]$, se puede definir la sucesión $\{x_k\}_{k\in\mathbb{N}}$ con

$$x_{k+1} = g(x_k)$$

Si existe $\lim_{k\to\infty} x_k = x^*$, tomando límite de ambos lados en la ecuación anterior, resulta que

$$x^* = \lim_{k\to\infty} x_{k+1} = \lim_{k\to\infty} g(x_k) = g(x^*)$$

por lo que x^* deberá ser un punto fijo de g.

Queda por determinar bajo qué condiciones es posible asegurar que este método converge, y que lo hace en forma eficiente; a continuación se enuncian algunos resultados que resultan útiles en este sentido. Sea una función $g : [a, b] \to [a, b]$. Se cumplen:

(i) Si g es continua, entonces g tiene al menos un punto fijo en $[a, b]$.

(ii) Si, además, g es derivable en (a, b) y existe alguna constante c, con $0 < c < 1$, tal que $(\forall\, x \in (a, b))\ |g'(x)| \leq c$, entonces el punto fijo es único.

(iii) Bajo estas condiciones, para cualquier $x_0 \in [a, b]$, la sucesión dada por $x_{k+1} = g(x_k)$ converge al único punto fijo de g en $[a, b]$.

En cuanto a la velocidad de convergencia, podemos afirmar que:

(iv) Bajo las hipótesis del punto (iii) anterior (es decir, $g : [a, b] \to [a, b]$ continua, derivable en (a, b), con $|g'(x)| \leq c < 1$ para algún valor c), pueden extraerse cotas que relacionan el error de aproximación cometido por el algoritmo en la iteración k-ésima con el valor de c^k. Esto permite deducir que, cuanto menor sea la cota que pueda imponerse a la derivada de la función g, más rápida será la convergencia garantizada por el algoritmo.

(v) Si se tiene una función $g \in \mathcal{C}^r([a,b])$ (es decir, g es r veces derivable en (a,b) con derivada continua) y un valor de x_0 tales que la iteración $x_{k+1} = g(x_k)$ converge a un punto fijo $x^* \in (a,b)$, y se cumple que

- $g'(x^*) = g''(x^*) = \cdots = g^{(r-1)}(x^*) = 0$, y
- $g^{(r)}(x^*) \neq 0$,

entonces el orden de convergencia de $\{x_k\}_{k\in\mathbb{N}}$ es r.

A continuación, veremos cómo pueden aprovecharse estos resultados para convertir el problema de hallar ceros de una función en un problema de punto fijo, asegurando que la sucesión obtenida converge y que lo hace de manera rápida.

8.5. Método de Newton

El **método de Newton** es un algoritmo para la búsqueda de raíces de funciones, se basa en plantear una iteración de punto fijo que, bajo ciertas hipótesis, converge con orden al menos cuadrático.

El método requiere que se considere una función $f \in \mathcal{C}^2([a,b])$, y que podamos calcular los valores de f y de su derivada. Se comienza con un valor $x_0 \in ([a,b])$ y, en cada iteración, se utiliza la derivada de f para aproximar el comportamiento de f en un entorno del valor actual y, de esa forma, ir acercándose de manera iterativa a una raíz de la función.

En concreto, buscaremos un punto fijo de la función

$$g(x) = x - \frac{f(x)}{f'(x)},$$

con lo cual la iteración de punto fijo quedará definida por

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}.$$

Si queremos utilizar el algoritmo para encontrar una raíz $x^* \in ([a,b])$ de f, será necesario que g esté bien definida en un entorno de x^*, es decir, que $f'(x^*) \neq 0$. En ese caso, resulta que

$$g(x^*) = x^* - \frac{f(x^*)}{f'(x^*)} = x^*,$$

es decir, x^* es efectivamente un punto fijo de g.

Además, podemos notar que, en ese caso, se cumple que

$$g'(x^*) = \frac{f(x^*) \cdot f''(x^*)}{f'(x^*)^2} = 0.$$

Como g' es continua, existe un entorno $(x^* - \delta, x^* + \delta)$ de x^* en el cual $|g'(x)| \leq c$ con $0 < c < 1$. Se puede demostrar que la restricción de $g : [x^* - \delta, x^* + \delta] \to [x^* - \delta, x^* + \delta]$ está bien definida. Teniendo en cuenta los resultados previos, esto nos permite afirmar que, en dicho entorno, el método converge a la raíz x^* y lo hace con un orden de convergencia al menos cuadrático.

Cabe destacar que la convergencia solo puede asegurarse dentro de un entorno, tal vez reducido, de x^*. Esto hace que el método solo funcione correctamente si se parte de una aproximación más o menos buena de una raíz de la función. Una solución común a este problema, cuando no se cuenta con dicha aproximación, es obtenerla en primer lugar ejecutando algunas iteraciones del método de la bisección.

El método tiene una interpretación geométrica muy intuitiva, que observarse en la Figura 4. Partiendo de un punto x_0, se considera la recta tangente a f en el punto $(x_0, f(x_0))$. El punto x_1 se define como la intersección entre esta recta y el eje de abscisas. El proceso se repite con cada iteración, arrojando cada vez una aproximación más cercana a la raíz x^* buscada.

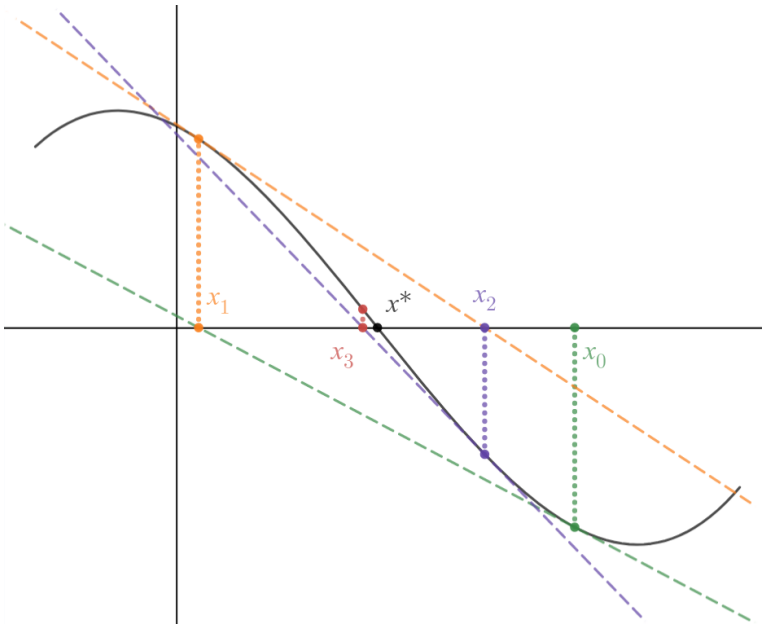

Figura 4: Interpretación geométrica del método de Newton.

Se trata de un método que, bajo las condiciones adecuadas, converge a una buena velocidad. Sin embargo, tiene dos principales desventajas. Una de ellas ya se mencionó anteriormente, y es la necesidad de conocer de antemano una aproximación relativamente buena de la raíz buscada para poder asegurar la convergencia; más aún si se desea que la convergencia sea rápida, ya que el orden cuadrático puede resultar excesivamente lento si se comienza demasiado lejos de la raíz. La otra desventaja es la necesidad de computar, en cada paso, el valor de la derivada de f, lo cual puede ser difícil, computacionalmente costoso o directamente imposible.

8.6. Método de la secante

El **método de la secante** es otra iteración de punto fijo que es muy similar método de Newton, pero en lugar de considerar la derivada de f en el punto actual, la aproxima mediante la recta secante que pasa por el punto actual y el punto anterior. Es decir, se considera que

$$f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}.$$

Como cada iteración tendrá en cuenta los dos puntos anteriores, se debe comenzar determinando dos puntos iniciales, x_0 y x_1. Luego, la iteración se define como

$$x_{k+1} = x_k - \frac{f(x_k)}{\frac{f(x_k)-f(x_{k-1})}{x_k - x_{k-1}}} = x_k - \frac{f(x_k)\cdot(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}.$$

La Figura 5 ilustra la interpretación geométrica de este método.

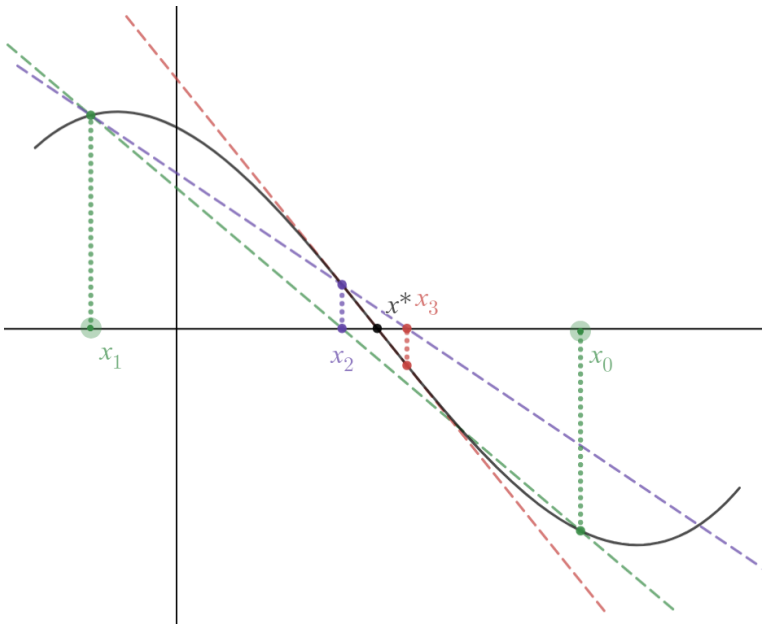

Figura 5: Interpretación geométrica del método de la secante.

El método de la secante converge un poco más lentamente que el método de Newton. Sin embargo, puede demostrarse que su orden orden de convergencia es supralineal: más exactamente, es $\varphi = \frac{1+\sqrt{5}}{2}$, la razón áurea. Además, presenta la gran ventaja de no requerir el cómputo de ninguna derivada.

Uno de los inconvenientes que surge a la hora de utilizar este método tiene que ver con la pérdida de dígitos significativos (o *cancelación catastrófica*) que se produce al tener que computar diferencias entre valores que, con cada iteración, se encuentran cada vez más cerca; el método *regula falsi* presenta una alternativa que no padece de este problema.

8.7. Método *regula falsi*

El **método *regula falsi*** es una variante del método de la bisección, que incorpora la idea principal del método de la secante. Al igual que en el método de la bisección, se comienza con dos puntos iniciales donde f tiene distinto signo; en cada paso se divide el intervalo en dos y se pasa a trabajar con la parte en cuyos extremos f tiene diferente signo. Sin embargo, en lugar de dividir al intervalo por su punto medio, se utiliza la intersección entre el eje de abscisas y la recta que pasa por los dos puntos anteriores.

Este método converge con un orden lineal, de la misma forma que el método de la bisección, pero resulta más rápido en términos prácticos. Comparado con el método de la secante, si bien es más lento, reduce la posibilidad de que se produzcan cancelaciones catastróficas: los valores de f con los que se trabaja siempre tienen signos opuestos, con lo cual las operaciones que se realizan pueden pensarse como sumas en lugar de diferencias.