Notas de Álgebra Lineal Computacional

Gabriel Acosta y Santiago Laplagne

Índice general

Preliminares

En estas notas se presentan los temas de la materia Álgebra Lineal Computacional en la Facultad de Ciencias Exactas y Naturales de la UBA.

bla bla bla

Acerca de la notación: (esto es para amalgamar con lo demás)

- Usaremos letra en **negrita** para representar magnitudes vectoriales o matriciales, y letra común para variables escalares. Para distinguir vectores de matrices, reservamos las letras *mayúsculas* para las segundas y *minúsculas* para los primeros.
- Los vectores $\boldsymbol{v} \in \mathbb{R}^n, \mathbb{C}^n$ se identificarán con matrices *columna* de $n \times 1$. Por ende tendrá sentido $\boldsymbol{A}\boldsymbol{v}$, para toda matriz $\boldsymbol{A} \in \mathbb{R}^{m \times n}, \mathbb{C}^{m \times n}$
- Con $|\boldsymbol{v}|$ representamos el vector con las mismas componentes de $\boldsymbol{v}$ con valor absoluto. Análogamente definimos $|\boldsymbol{A}|$.
- Dados dos vectores (matrices) $\boldsymbol{v}_1, \boldsymbol{v}_2$ $(\boldsymbol{A}_1, \boldsymbol{A}_2)$ la notación $\boldsymbol{v}_1 \leq \boldsymbol{v}_2$ $(\boldsymbol{A}_1 \leq \boldsymbol{A}_2)$ debe interpretarse componente a componente (lo mismo para $<, >, \geq$).
- Dados un vector (matriz) $\boldsymbol{v}$ $(\boldsymbol{A})$ y una constante c, la notación $\boldsymbol{v} \leq c$ $(\boldsymbol{A} \leq c)$, indica que todas componentes de $\boldsymbol{v}$ $(\boldsymbol{A})$ son menores o iguales a c (lo mismo para $<, >, \geq$).
- Utilizamos la notación de los dos puntos ":", compatible con numerosos lenguajes de programación, para identificar rangos de índices en vectores y matrices: por ejemplo $\boldsymbol{v}(2:5)$ indica los elementos 2 al 5 incluidos del vector $\boldsymbol{v}$ [1]. La misma interpretación vale para matrices, por ejemplo, resulta válido para nosotros escribir $\boldsymbol{A}(2:10, 7:25)$.

[1] Estamos manejando índices al estilo Matlab, es decir que comienzan en 1, en otros lenguajes como Python el primer índice del arreglo es 0. Preferimos usar en el texto la primera porque es mas natural con la notación matemática usual.

Capítulo 1

Nociones básicas de álgebra lineal

1. Vectores y matrices

1.1. Vectores. Para $n \in \mathbb{N}$, definimos un vector de n coordenadas como una sucesión de n números reales o complejos $\boldsymbol{v} = (v_1, \ldots, v_n)$. Denotamos $\mathbb{R}^n$ al conjunto de todos los vectores reales de n coordenadas y $\mathbb{C}^n$ al conjunto de todos los vectores complejos de n coordenadas. Ya que la mayoría de los resultados que veremos valen tanto en $\mathbb{R}$ como en $\mathbb{C}$, los enunciamos usando la letra $\mathbb{K}$. En los casos en que debamos restringirnos a $\mathbb{R}$ o $\mathbb{C}$ lo señalaremos explícitamente. Solo haremos uso de definiciones y propiedades elementales de los complejos, antes de continuar recordamos algunas de ellas que aparecerán más adelante

1. $i^2 = -1$.
2. Si $z = a + ib$ el conjugado se define como $\overline{z} = a - ib$.
3. $\forall z, w \in \mathbb{C}$, $\overline{zw} = \overline{z}\,\overline{w}$.
4. El módulo de z, se define como $|z| = \sqrt{a^2 + b^2}$ y representa la distancia del complejo z al origen.
5. $\forall z, w \in \mathbb{C}$, $|zw| = |z||w|$
6. $|z|^2 = z\overline{z}$, en particular se observa que si $z \neq 0$, $z\frac{\overline{z}}{|z|^2} = 1$. Llamamos $z^{-1} = \frac{\overline{z}}{|z|^2}$.
7. $\forall \theta \in \mathbb{R}$, se define $e^{i\theta} = \cos(\theta) + i\,\mathrm{sen}(\theta)$, en particular $|e^{i\theta}| = 1$. Por otro lado, usando trigonometría elemental, se ve que si $|w| = 1$, existe $\theta \in \mathbb{R}$ tal que $w = e^{i\theta}$.
8. Ya que si $z \neq 0$ se tiene que $\left|\frac{z}{|z|}\right| = 1$, del ítem previo se observa que si $z \neq 0$, puede escribirse $z = |z|e^{i\theta}$ con $\theta \in \mathbb{R}$.
9. Todo polinomio a coeficientes en $\mathbb{C}$ de grado $n \geq 1$ tiene exactamente n raíces -contadas con su multiplicidad- en $\mathbb{C}$ (resultado que suele llamarse Teorema Fundamental del Álgebra).

Ejemplo 1.1.

- $\mathbb{R}^2$ es el plano coordenado.
- $\mathbb{R}^3$ es el espacio de 3 dimensiones.

Ejemplo 1.2.

- $\boldsymbol{v} = (1, 2) \in \mathbb{R}^2$.
- $\boldsymbol{u} = (2, -1, \pi, 3/2) \in \mathbb{R}^4$.
- $\boldsymbol{w} = (1 - 2i, 3e^{3i}) \in \mathbb{C}^2$.

En `Python` definimos vectores con el comando `array` (arreglo) del paquete `numpy`.

```
import numpy as np
```

```
v1 = np.array([10, 5, -7, 1])
print(v1)

v2 = np.array([5, 0, 3/2, 2])
print(v2)
```

```
%% v1 =  [10  5 -7  1]
%% v2 =  [5.  0.  1.5 2. ]
```

1.2. Matrices. Denotamos $\mathbb{K}^{m\times n}$ al espacio de matrices de números reales o complejos de m filas y n columnas, que podemos considerar como un vector de $m \times n$ coordenadas agrupadas en m filas de n coordenadas.

Ejemplo 1.3.

1. $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \in \mathbb{R}^{2\times 3}$
2. $\begin{pmatrix} 1 \\ 7 \\ 0.6 \\ -1 \end{pmatrix} \in \mathbb{R}^{4\times 1}$
3. $\begin{pmatrix} 1 & 5+i \\ 7-2i & 0 \\ i & 3 \end{pmatrix} \in \mathbb{C}^{3\times 2}$

Es común considerar a los vectores en $\mathbb{K}^n$ como matrices columna. Es decir, al vector $v = (1, 2, 3)$ lo pensamos como matriz $\begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$ cuando trabajamos con matrices.

En `Python` definimos matrices usando el mismo comando `array` que utilizamos para vectores, pero ingresando cada fila como un vector, es decir para `Python` una matriz es un arreglo de dos dimensiones, que se guarda como un arreglo de arreglos de dimensión 1.

```
A = np.array([[1,2],[3,4]]) # Matriz de 2 x 2
print("A = \n", A)

B = np.array([[1,2,3,4],[7,1,2,-1]]) # Matriz de 4 x 2
print("B = \n", B)

C = np.array([[1], [7], [1/3]]) # Matriz columna de 1 x 3
print("C = \n", C)
```

```
%% A =
%%  [[1 2]
%%  [3 4]]
%% B =
%%  [[ 1  2  3  4]
%%  [ 7  1  2 -1]]
%% C =
%%  [[1.        ]
%%  [7.        ]
```

```
%%  [0.33333333]]
```

Podemos acceder a las casillas de un vector o matriz usando los índices de las casillas, teniendo en cuenta que en Python los índices se numeran siempre empezando en 0.

```
v1 = np.array([1,2,5,10])
print("v1[2] = ", v1[2])

A = np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12]])
print("A[2,3] = ", A[2,3])
```

```
%% v1[2] = 5
%% A[2,3] = 12
```

1.3. Suma y producto por escalar. Tanto en vectores como en matrices podemos realizar las siguientes operaciones, que como veremos más adelante corresponden a operaciones de espacio vectorial:

- Suma de vectores o matrices del mismo tamaño coordenada a coordenada. Si $\boldsymbol{v} = (v_1, v_2, \ldots, v_n)$ y $\boldsymbol{u} = (u_1, u_2, \ldots, u_n)$,
$$\boldsymbol{v} + \boldsymbol{u} = (v_1 + u_1, v_2 + u_2, \ldots, v_n + u_n).$$
- Producto de vectores o matrices por un escalar. Si $a \in \mathbb{R}$ o $\mathbb{C}$ y $\boldsymbol{v} = (v_1, v_2, \ldots, v_n)$,
$$a\boldsymbol{v} = (av_1, av_2, \ldots, av_n).$$

En Python realizamos estas operaciones con los símbolos usuales + y $*$.

```
import numpy as np
v1 = np.array([10, 5, -7, 1])
v2 = np.array([5, 0, 7, 2])
print("v1 + v2 = ", v1 + v2)
```

```
## [15 5 0 3]
```

```
A1 = np.array([[1,2],[3,4]])
print("A1 = \n", A1)

A2 = np.array([[2,7],[1,0]])
print("A2 = \n", A2)

A3 = np.array([[2,7,1,0]])
print("A3 = \n", A3)

print("A1 + A2 = \n", A1 + A2) # Matriz de 2 x 2

# No podemos sumar matrices de distinto tamaño
```

```
#print("A1 + A3 = \n", A1 + A3)
```

```
## A1 =
## [[1 2]
## [3 4]]
## A2 =
## [[2 7]
## [1 0]]
## A3 =
## [2 7 1 0]
## A1 + A2 =
## [[3 9]
## [4 4]]
## [15 5 0 3]
```

2. Sistemas lineales de ecuaciones

2.1. Resolución de sistemas de ecuaciones por triangulación (eliminación gaussiana). La eliminación gaussiana es un método muy eficiente para resolver sistemas de ecuaciones lineales en forma directa (es decir, sin utilizar métodos iterativos que aproximan la solución).

A modo de ejemplo, resolvemos el siguiente sistema de ecuaciones.

$$\begin{cases} x + 5y + 5z = 2 \\ 2x + 2y - 3z = -1 \\ -x - 9y + 2z = 9. \end{cases}$$

A partir del sistema, construimos la **matriz ampliada** de coeficientes y términos independientes:

$$\left(\begin{array}{ccc|c} 1 & 5 & 5 & 2 \\ 2 & 2 & -3 & -1 \\ -1 & -9 & 2 & 9 \end{array}\right).$$

Triangulamos la matriz realizando operaciones de filas. Las operaciones permitidas (que no afectan las soluciones del sistema) son:

- sumarle o restarle a una fila un múltiplo de otra fila,
- intercambiar dos filas entre si.
- multiplicar una fila por un escalar distinto de 0.

El algoritmo de eliminación gaussiana consiste en triangular o escalonar la matriz obteniendo 0's abajo de los elementos de la diagonal, o más generalmente, produciendo que en cada fila la cantidad de 0's iniciales sea mayor que en la fila anterior.

Realizamos las siguientes operaciones:

$$\left(\begin{array}{ccc|c} 1 & 5 & 5 & 2 \\ 2 & 2 & -3 & -1 \\ -1 & -9 & 2 & 9 \end{array}\right) \xrightarrow{f_2-2f_1\to f_2} \left(\begin{array}{ccc|c} 1 & 5 & 5 & 2 \\ 0 & -8 & -13 & -5 \\ -1 & -9 & 2 & 9 \end{array}\right) \to$$
$$\xrightarrow{f_3+f_1\to f_3} \left(\begin{array}{ccc|c} 1 & 5 & 5 & 2 \\ 0 & -8 & -13 & -5 \\ 0 & -4 & 7 & 11 \end{array}\right) \xrightarrow{f_3-\frac{1}{2}f_2\to f_3} \left(\begin{array}{ccc|c} 1 & 5 & 5 & 2 \\ 0 & -8 & -13 & -5 \\ 0 & 0 & \frac{27}{2} & \frac{27}{2} \end{array}\right)$$

Ahora podemos obtener los valores de x, y, z por "sustitución hacia atrás". Primero calculamos $z = 1$, luego $y = -1$ y finalmente $x = 2$. La segunda ecuación queda

$$-8y - 13 \cdot (1) = -5$$

entonces, $y = -1$.

Ejemplo 1.4. Resolvemos escalonando el sistema

$$\begin{cases} x_1 + 2x_2 = -1 \\ 2x_1 + 4x_2 + 3x_3 = 4 \\ 3x_3 = 6. \end{cases}$$

1. Construimos la matriz ampliada
$$\tilde{\boldsymbol{A}} = \left(\begin{array}{ccc|c} 1 & 2 & 0 & -1 \\ 2 & 4 & 3 & 4 \\ 0 & 0 & 3 & 6 \end{array}\right).$$
2. Para conseguir 0's abajo de la casilla $(1, 1)$, realizamos la operación $f_2 - 2f_1 \to f_2$. Obtenemos
$$\tilde{\boldsymbol{A}}_1 = \left(\begin{array}{ccc|c} 1 & 2 & 0 & -1 \\ 0 & 0 & 3 & 6 \\ 0 & 0 & 3 & 6 \end{array}\right).$$
3. Como deseamos tener en cada fila más ceros que en la anterior, debemos ahora obtener un 0 abajo de la casilla $(2, 3)$. Para eso realizamos la operación $f_3 - f_2 \to f_3$. Obtenemos la matriz
$$\tilde{\boldsymbol{A}}_2 = \left(\begin{array}{ccc|c} 1 & 2 & 0 & -1 \\ 0 & 0 & 3 & 6 \\ 0 & 0 & 0 & 0 \end{array}\right),$$
que es una matriz escalonada, por lo que finalizamos el proceso.

Observar que si bien la matriz $\tilde{\boldsymbol{A}}_1$ tiene 0's abajo de la diagonal, no es un matriz escalonada porque las filas 2 y 3 tienen la misma cantidad de 0's iniciales.

Ahora podemos resolver el sistema despejando las ecuaciones que obtuvimos:

$$\begin{cases} x_1 + 2x_2 - x_3 = -1 \\ 3x_3 = 6, \end{cases}$$

de donde despejamos $x_3 = 2$ y $x_1 = -1 - 2x_2 + x_3 = 1 - 2x_2$. El conjunto de soluciones del sistema es

$$S = \{(1 - 2x_2, x_2, 2) : x_2 \in \mathbb{R}\}.$$

En los paquetes comunes de Python no existe un comando para realizar eliminación gaussiana (aunque sí existen comandos para resolver sistemas lineales eficientemente). Más adelante, cuando avancemos con las técnicas de programación y desarrollo de algoritmos, podremos programar nuestro propio programa de eliminación gaussiana. Por el momento, utilizaremos el siguiente programa, extraido de la página https://math.stackexchange.com/questions/3073083/how-to-reduce-matrix-into-row-echelon-form-in-python/3073117.

El nombre de la función es row_echelon, que es el nombre en inglés para una matriz escalonada por filas. Para utilizar esta función, pueden copiar y pegar el código en una celda y ejecutarlo, o grabarlo en un archivo row_echelon.py y cargarlo mediante el comando import row_echelon.

```
def row_echelon(M):
    """ Return Row Echelon Form of matrix A """

    A = M.astype(float)
    # if matrix A has no columns or rows,
    # it is already in REF, so we return itself
    r, c = A.shape
    if r == 0 or c == 0:
        return A

    # we search for non-zero element in the first column
    for i in range(len(A)):
        if A[i,0] != 0:
            break
    else:
        # if all elements in the first column are zero,
        # we perform REF on matrix from second column
        B = row_echelon(A[:,1:])
        # and then add the first zero-column back
        return np.hstack([A[:,:1], B])

    # if non-zero element happens not in the first row,
    # we switch rows
    if i > 0:
        ith_row = A[i].copy()
        A[i] = A[0]
        A[0] = ith_row

    # we divide first row by first element in it
    A[0] = A[0] / A[0,0]
    # we subtract all subsequent rows with first row
    #(it has 1 now as first element)
```

```
    # multiplied by the corresponding element in the first column
    A[1:] -= A[0] * A[1:,0:1]

    # we perform REF on matrix from second row, from second column
    B = row_echelon(A[1:,1:])

    # we add first row and first (zero) column, and return
    return np.vstack([A[:1], np.hstack([A[1:,:1], B]) ])
```

Probamos el programa en el siguiente ejemplo.

```
A = np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12]])
print(row_echelon(A))
```

```
[[1. 2. 3. 4.]
 [0. 1. 2. 3.]
 [0. 0. 0. 0.]]
```

Si queremos armar la matriz ampliada correspondiente a un sistema de ecuaciones y tenemos la matriz $\boldsymbol{A}$ de coeficientes y el vector $\boldsymbol{b}$ de términos independientes, podemos usar el comando `c_` de `numpy` para pegar a $\boldsymbol{A}$ el vector $\boldsymbol{b}$ como última columna.

EJEMPLO 1.5. Resolvemos en `Python` el sistema

$$\begin{cases} x_1 + 5x_2 + 5x_3 = -2 \\ 2x_1 + 2x_2 - 3x_3 = -1 \\ -x_1 - 9x_2 + 2x_3 = 9. \end{cases}$$

```
A = np.array([[1,5,5],[2,2,-3],[-1,-9,2]])
b = np.array([2, -1, 9])
Ab = np.c_[A, b] # Las matrices o vectores van entre corchetes.
print("Ab = \n", Ab)

print("Matriz escalonada: \n", row_echelon(Ab))
```

```
%% Ab =
%%  [[ 1  5  5  2]
%%  [ 2  2 -3 -1]
%%  [-1 -9  2  9]]
%% Matriz escalonada:
%%  [[1.    5.    5.    2.   ]
%%  [0.    1.    1.625 0.625]
%%  [0.    0.    1.    1.   ]]
```

Despejando de abajo hacia arriba, obtenemos

$$\begin{aligned} x_3 &= 1 \\ x_2 &= 0.625 - 1.625x_3 = -1 \\ x_1 &= 2 - 5x_2 - 5x_3 = 2. \end{aligned}$$

2.2. Clasificación de sistemas de ecuaciones. Estudiamos ahora cuántas soluciones puede tener un sistema de ecuaciones a partir de la forma escalonada de la matriz ampliada.

Ejemplo 1.6. Resolvemos el siguiente sistema de ecuaciones:

$$\begin{cases} 5x_1 + 3x_2 = 11 \\ 15x_1 + 9x_2 = 33 \\ 20x_1 + 12x_2 = 44 \end{cases}$$

Construimos la matriz ampliada

$$\left(\begin{array}{cc|c} 5 & 3 & 11 \\ 15 & 9 & 33 \\ 20 & 12 & 44 \end{array}\right)$$

y escalonamos usando `Python`.

```
A = np.array([[5,3,11],[15,9,33],[20,12,44]])
print(row_echelon(A))
```

```
%% [[1.  0.6 2.2]
%%  [0.  0.  0. ]
%%  [0.  0.  0. ]]
```

Vemos que se eliminaron las últimas dos ecuaciones, y nos queda solo una ecuación:

$$x_1 + 0.6x_2 = 2.2$$

de donde podemos despejar $x_1 = 2.2 - 0.6x_2$.

El sistema tiene infinitas soluciones,

$$S = \{(2.2 - 0.6x_2, x_2) : x_2 \in \mathbb{R}\}.$$

Ejemplo 1.7. Consideremos ahora el mismo sistema inicial, modificando un valor en $\boldsymbol{b}$:

$$\begin{cases} 5x_1 + 3x_2 = 11 \\ 15x_1 + 9x_2 = 33 \\ 20x_1 + 12x_2 = 55 \end{cases}$$

Escalonamos la matriz ampliada:

```
A = np.array([[5,3,11],[15,9,33],[20,12,55]])
print(row_echelon(A))
```

```
%% [[1.  0.6 2.2]
%%  [0.  0.  1. ]
%%  [0.  0.  0. ]]
```

En la segunda ecuación, obtuvimos $0x_1 + 0x_2 = 1$. ¡Absurdo! El sistema no tiene solución.

Si al escalonar la matriz ampliada, llegamos a una ecuación

$$0x_1 + \cdots + 0x_n = a \neq 0$$

el sistema no tiene solución. Por el contrario, si en todas las filas que se anulas los coeficientes de las variables, obtenemos también 0 en el término independiente, el sistema admite solución (podemos ir resolviendo hacia atrás).

Obtenemos los siguientes casos para un sistema de m ecuaciones y n incógnitas.

- **Sistema incompatible.** El sistema no tiene solución. Al escalonar obtuvimos una ecuación $0x_1 + \cdots + 0x_n = a \neq 0$. Ejemplo:

$$\left(\begin{array}{ccc|c} 5 & 3 & 2 & 11 \\ 0 & 9 & -1 & 10 \\ 0 & 0 & 0 & 4 \end{array}\right)$$

- **Sistema compatible determinado.** El sistema tiene solución única. Al escalonar la matriz ampliada, obtenemos exactamente n ecuaciones no nulas, y podemos obtener la solución despejando las variables.

$$\left(\begin{array}{ccc|c} 5 & 3 & 2 & 11 \\ 0 & 9 & -1 & 10 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right)$$

- **Sistema compatible indeterminado.** El sistema tiene infinitas soluciones. Al escalonar la matriz ampliada obtenemos menos de n filas no nulas en las primeras n columnas, y el resto de las filas son nulas en todas las columnas. Ejemplo:

$$\left(\begin{array}{ccc|c} 5 & 3 & 2 & 11 \\ 0 & 9 & -1 & 10 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right)$$

Ejercicio 1.1. Escalonar las siguientes matrices y clasificar el sistema en (a) incompatible, (b) compatible determinado, (c) compatible indeterminado.

1. $\begin{cases} 3y - 2z + 3w = 9 \\ 2x + y + w = 5 \\ x - y + z - w = -2 \end{cases}$
2. $\begin{cases} x - 2y = 2 \\ 2x + y = 1 \\ x + 3y = -1 \end{cases}$
3. $\begin{cases} 2x + y - z = 3 \\ x - y + z = 2 \\ 5x + y - z = -5 \end{cases}$

3. Espacios vectoriales

Un espacio vectorial es un conjunto de elementos, llamados *vectores*, que pueden ser sumados entre sí y multiplicados (*escalados*) por números (llamados *escalares*). Llamamos V al conjunto de vectores y $\mathbb{K}$ al conjunto de números.

El conjunto $\mathbb{K}$ debe ser un cuerpo. Esto es, $\mathbb{K}$ posee una suma $+$ y un producto $\cdot$ que satisfacen un conjunto de axiomas que podemos resumir en la siguiente tabla.

	suma	producto
asociatividad	$a+(b+c)=(a+b)+c$	$(a\cdot b)\cdot c = a\cdot(b\cdot c)$
conmutatividad	$a+b=b+a$	$a\cdot b = b\cdot a$
propiedad distributiva	$a\cdot(b+c)=a\cdot b+a\cdot c$	
elemento identidad	$0+a=a=a+0$	$1\cdot a=a=a\cdot 1$
elemento inverso	$a+(-a)=0$	$a\cdot(a^{-1})=1$

En esta materia trabajaremos con $\mathbb{K}=\mathbb{R}$, el cuerpo de los números reales o $\mathbb{K}=\mathbb{C}$ el cuerpo de los números complejos[1].

Ejemplo 1.8. El conjunto $\mathbb{Z}$ de los números enteros no es un cuerpo, porque excepto 1 y -1, los demás números enteros no poseen un inverso entero para la multiplicación. Por ejemplo, el inverso de 2 es $1/2$ que no es un número entero.

Formalmente, un espacio vectorial sobre un cuerpo $\mathbb{K}$ (también llamado $\mathbb{K}$ - espacio vectorial) es un conjunto V y dos operaciones, que satisfacen ciertos axiomas.

Las operaciones definidas en V son:

- Suma de vectores. $+ : V\times V\to V$, a cualquier par $\boldsymbol{v},\boldsymbol{w}$ de vectores le asigna un vector $\boldsymbol{v}+\boldsymbol{w}\in V$.
- Producto por escalar. $\cdot : \mathbb{K}\times V\to V$, a un escalar a y un vector $\boldsymbol{v}\in V$ le asigna un vector $a\boldsymbol{v}\in V$ (no confundir con el producto escalar que dados dos vectores devuelve un escalar que veremos más adelante).

Los axiomas de espacio vectorial son

1. **Asociatividad de la suma.** $\boldsymbol{u}+(\boldsymbol{v}+\boldsymbol{w})=(\boldsymbol{u}+\boldsymbol{v})+\boldsymbol{w}$
2. **Conmutatividad de la suma.** $\boldsymbol{u}+\boldsymbol{v}=\boldsymbol{v}+\boldsymbol{u}$
3. **Elemento neutro de la suma.** Existe un elemento $\mathbf{0}\in V$ tal que $\boldsymbol{v}+\mathbf{0}=\boldsymbol{v}$ para todo $\boldsymbol{v}\in V$.
4. **Inverso para la suma.** Para todo $\boldsymbol{v}\in V$, existe un elemento $-\boldsymbol{v}\in V$, llamado inverso aditivo de $\boldsymbol{v}$, tal que $\boldsymbol{v}+(-\boldsymbol{v})=0$.
5. **Compatibilidad de la multiplicación por escalar con la la multiplicación del cuerpo.** $a(bv)=(ab)\boldsymbol{v}$.
6. **Elemento neutro de la multiplicación por escalar.** $1\boldsymbol{v}=\boldsymbol{v}$, donde 1 es el neutro de $\mathbb{R}$.
7. **Propiedad distributiva de la multiplicación por escalar respecto de la suma de vectores.** $a(\boldsymbol{u}+\boldsymbol{v})=au+av$.

[1]Existen otros ejemplos de cuerpos como los racionales $\mathbb{Q}$ o los enteros módulo un primo p, $\mathbb{Z}_p$.

8. **Propiedad distributiva de la multiplicación por escalar respecto de la suma de escalares.** $(a+b)(\boldsymbol{v}) = av + bv$.

EJEMPLO 1.9. Los siguientes conjuntos son espacios vectoriales definiendo la suma y el producto por escalar de modo "tradicional".

1. $\mathbb{K}^n$, es un $\mathbb{K}$ espacio vectorial.
2. $\mathbb{K}^{m\times n}$, el conjunto de matrices, es un $\mathbb{K}$ espacio vectorial.
3. $\mathbb{K}[x]$ el conjunto de todos los polinomios en una variable a coeficientes reales o complejos.
4. $\mathbb{R}[x]_d$, el conjunto de polinomios en una variable de grado menor o igual que d.
5. $(a_1, a_2, a_3, \dots)$, el conjunto de sucesiones infinitas de números reales o complejos.
6. $\mathbb{K}$ es un $\mathbb{K}$ espacio vectorial. Pero note que $\mathbb{C}$ también es un $\mathbb{R}$ espacio vectorial.

Los siguientes conjuntos *no* son espacios vectoriales.

1. El conjunto de matrices de cualquier tamaño.
2. Los polinomios de grado exactamente d.

APLICACIÓN 1.1. Veamos un ejercicio sencillo de aplicación de espacios vectoriales. Se quiere predecir el consumo en tarjeta de crédito que tendrán los clientes de un banco mediante un modelo lineal. Para eso, se consideran las siguientes *variables explicativas* de los clientes:

1. Sueldo mensual
2. Impuesto a las ganancias pagado el año anterior.
3. Cantidad de integrantes del grupo familiar
4. Puntaje otorgado a la serie "La Casa de Papel"(1 a 5 estrellas)
5. Es fumador (sí / no)
6. Marca de Celular (Samsung / Huawei / Iphone / Otro)

Se quiere definir un espacio vectorial V donde la información de cada cliente sea un vector de V. ¿Qué espacio vectorial utilizaría para este modelo? (Se busca que dos personas con vectores similares tengan comportamiento similar.)

Observamos que los cuatro primeros datos son datos numéricos y podemos representar cada valor con un número real. En las preguntas 3 y 4, nos gustaría utilizar números enteros para representar los valores, pero ya vimos que el conjunto de números enteros no es un cuerpo y por lo tanto no podemos definir un espacio vectorial sobre los enteros.

La quinta pregunta podemos representarla con valores 0 y 1. En este caso al conjunto $\{0, 1\}$ podemos asignarle estructura de cuerpo, pero dado que queremos un único espacio vectorial para representar todas las variables, debemos representar estos valores también como números reales.

Finalmente, para el último dado, podríamos asignarle un valor numérico a cada respuesta: 1 = Samsung, 2 = Huawei, 3 = Iphone, 4 = Otro, sin embargo esto no cumpliría lo que buscamos en nuestro modelo que vectores similares tengan comportamiento similar. Es decir, si usamos esta representación estaríamos indicando que la respuesta Otro es muy similar a Iphone y no tan similar a Samsung, y en principio no hay razón para esa suposición. Por lo tanto es común en este caso usar las llamadas variables indicadoras, que toman valores 0-1. Utilizamos 4 variables 0-1 que valen 1 en caso de que la respuesta corresponda a esa marca.

¿Qué vector $\boldsymbol{v} \in V$ asignaría al cliente con las siguientes características?

1. Sueldo mensual: $80.000

2. Impuesto a las ganancias pagado el año anterior: $100.000
3. Cantidad de integrantes del grupo familiar: 4
4. Puntaje otorgado a la serie "La Casa de Papel"(1 a 5 estrellas): 4 estrellas
5. Es fumador (sí / no): sí
6. Marca de Celular (Samsung / Huawei / Iphone / Otro): Huawei

En base a lo que vimos, asignamos a estas respuestas el vector $\boldsymbol{v} = (80000, 100000, 4, 4, 1, 0, 1, 0, 0)$. Esta asignación que hicimos es muy común cuando construimos modelos lineales. Vemos que la estructura de espacio vectorial es la que nos dice cómo construir el modelo.

3.1. Subespacios vectoriales. Dado V un $\mathbb{K}$ espacio vectorial, un subconjunto $U \subset V$ se llama subespacio de V si verifica

1. $\mathbf{0} \in U$
2. $\forall \boldsymbol{u}, \boldsymbol{v} \in U \ \boldsymbol{u} + \boldsymbol{v} \in U$
3. $\forall \boldsymbol{u} \in U$, $\forall \alpha \in \mathbb{K} \ \alpha \boldsymbol{u} \in U$.

Notar que U es a la vez un espacio vectorial.

Ejemplo 1.10. Algunos ejemplos de subespacios vectoriales son

1. El subconjunto de vectores de $\mathbb{R}^5$ tales que la primera coordenada es 0 es un subespacio vectorial de $\mathbb{R}^5$.
2. El conjunto de matrices diagonales de 3×3 es un subespacio vectorial de $\mathbb{R}^{3\times 3}$.
3. El conjunto de matrices simétricas de $n \times n$ es un subespacio vectorial de $\mathbb{R}^{n\times n}$.
4. El subconjunto de vectores $\mathbb{R}^5$ tales que la primera coordenada es 1 **no** es un subespacio vectorial de $\mathbb{R}^5$.
5. El conjunto de todos los polinomios en $\mathbb{R}[X]$ que se anulan en 1.

3.2. Conjuntos de generadores y bases.

3.2.1. Vectores linealmente independientes. Dado un conjunto $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m\}$ de vectores en $\mathbb{R}^n$, decimos que es un conjunto de vectores linealmente independientes si la única elección de coeficientes para los cuales

$$\sum_{i=1}^{m} a_i \boldsymbol{v}_i = 0$$

es $a_i = 0$ para todo $1 \leq i \leq m$.

Ejemplo 1.11.

- Los vectores $\boldsymbol{v}_1 = (1, 0, 0)$, $\boldsymbol{v}_2 = (0, 1, 0)$ y $\boldsymbol{v}_3 = (0, 0, 1)$ son linealmente independientes.
- Los vectores $\boldsymbol{v}_1 = (1, 0, 1)$, $\boldsymbol{v}_2 = (0, 1, 2)$ y $\boldsymbol{v}_3 = (1, 2, 5)$ son linealmente dependientes, porque $\boldsymbol{v}_3 = \boldsymbol{v}_1 + 2\boldsymbol{v}_2$. O equivalentemente, $\boldsymbol{v}_1 + 2\boldsymbol{v}_2 - \boldsymbol{v}_3 = 0$.

3.2.2. Espacio vectorial generado por vectores. Dado un conjunto de vectores $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m\}$ de un espacio vectorial V, el conjunto de todas las combinaciones lineales de estos vectores

$$\langle \boldsymbol{v}_1, \ldots, \boldsymbol{v}_m \rangle = \{a_1\boldsymbol{v}_1 + \cdots + a_m\boldsymbol{v}_m : a_i \in \mathbb{R}, i = 1, \ldots, m\}$$

es un subespacio U de V llamado el espacio generado por $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m\}$.

PROPOSICIÓN 1.1. Si los vectores $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m\}$ son linealmente independientes, cualquier elemento de $U = \langle \boldsymbol{v}_1, \ldots, \boldsymbol{v}_m \rangle$ se escribe de forma única como combinación lineal de los vectores $\{\boldsymbol{v}_i : i = 1, \ldots, m\}$.

DEMOSTRACIÓN. Supongamos por el absurdo que $\boldsymbol{v}$ puede escribirse de dos formas distintas $\boldsymbol{v} = a_1\boldsymbol{v}_1 + \cdots + a_m\boldsymbol{v}_m = b_1\boldsymbol{v}_1 + \cdots + b_m\boldsymbol{v}_m$, entonces restando obtenemos

$$0 = (a_1 - b_1)\boldsymbol{v}_1 + \cdots + (a_m - b_m)\boldsymbol{v}_m,$$

donde los coeficientes no son todos 0 (porque $(a_1, \ldots, a_m) \neq (b_1, \ldots, b_m)$), y esto contradice la independencia lineal de los vectores $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m\}$. □

DEFINICIÓN 1.1. Si $\mathcal{B} = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m\} \subset V$ es un conjunto linealmente independiente, decimos que $\mathcal{B}$ es una *base* de $U = \langle \boldsymbol{v}_1, \ldots, \boldsymbol{v}_m \rangle \subset U$.

A continuación veremos que si un espacio vectorial V tienen una base con una cantidad finita de elementos, cualquier otra base de V tiene la misma cantidad de elementos.

Comenzamos con el siguiente lema.

PROPOSICIÓN 1.2 (Lema de intercambio). Si $\mathcal{B} = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m\}$ es un base de V y $\boldsymbol{w} \in V$, entonces existe $1 \leq i \leq m$ tal que reemplazando en $\mathcal{B}$ a $\boldsymbol{v}_i$ por $\boldsymbol{w}$, el conjunto resultante sigue siendo una base de V.

DEMOSTRACIÓN. Como $\boldsymbol{w} \in V$ y $\mathcal{B}$ es una base, existen $a_1, \ldots, a_m$ en $\mathbb{K}$ tales que

$$\boldsymbol{w} = a_1\boldsymbol{v}_1 + \cdots + a_m\boldsymbol{v}_m,$$

y existe algún $1 \leq k \leq m$ tal que $a_k \neq 0$. Por lo tanto,

$$\boldsymbol{v}_k = \frac{1}{a_k}\left(\boldsymbol{w} - \sum_{i \neq k} a_i\boldsymbol{v}_i\right),$$

y el conjunto $\{\boldsymbol{v}_1, \ldots, \boldsymbol{w}, \ldots, \boldsymbol{v}_m\}$ es un sistema de generadores de V.

Para ver que forman una base, veamos que son linealmente independientes. Si existe una combinación no nula

$$b_1\boldsymbol{v}_1 + \cdots + b_k\boldsymbol{w} + \cdots + b_m\boldsymbol{v}_m = 0,$$

analizamos dos casos:

- Si $b_k = 0$, encontramos una relación de dependencia lineal en $\mathcal{B}$, lo cual es absurdo porque $\mathcal{B}$ era una base.
- Si $b_k \neq 0$, entonces despejando $\boldsymbol{w}$ obtenemos una escritura de $\boldsymbol{w}$ en la base $\mathcal{B}$ distinta a la anterior, lo cual también es absurdo.

Concluimos que $\{\boldsymbol{v}_1, \ldots, \boldsymbol{w}, \ldots, \boldsymbol{v}_m\}$ es un conjunto linealmente independiente y genera V, por lo tanto es una base de V. □

Ahora podemos probar el siguiente resultado.

PROPOSICIÓN 1.3. Dado un espacio vectorial V, sea $\mathcal{B} = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_s\}$ una base de V de s elementos, y sea $\tilde{\mathcal{B}}$ otra base de V. Luego $\tilde{\mathcal{B}}$ tiene exactamente s elementos.

DEMOSTRACIÓN. Queremos usar el Lema de Intercambio e ir reemplazando uno a uno los vectores de $\mathcal{B}$ por los vectores de $\tilde{\mathcal{B}}$. El único cuidado que tenemos que tener es que vayamos reemplazando en cada paso un vector distinto de $\mathcal{B}$. Para ver que esto siempre es posible, supongamos que ya reemplazamos k vectores y, por simplicidad, supongamos que reemplazamos los primeros k vectores de $\mathcal{B}$ por los primeros k vectores de $\tilde{\mathcal{B}}$.. Es decir, tenemos que

$$\{\boldsymbol{w}_1, \dots, \boldsymbol{w}_k, \boldsymbol{v}_{k+1}, \dots, \boldsymbol{v}_s\}$$

es una base de V.

Ahora queremos reemplazar a $\boldsymbol{w}_{k+1}$ por algún vector $\boldsymbol{v}_i$, $i\,k$. Siguiendo la demostración del Lema de Intercambio, escribimos a $\boldsymbol{w}_{k+1}$ como combinación no nula de los elementos de la base:

$$\boldsymbol{w}_{k+1} = c_1\boldsymbol{w}_1 + \cdots + c_k\boldsymbol{w}_k + c_{k+1}\boldsymbol{v}_{k+1} + \cdots + c_s\boldsymbol{v}_s.$$

No pueden ser todos los coeficientes $c_{k+1}, \dots, c_s$ iguales a 0, porque entonces $\tilde{\mathcal{B}}$ no sería base de V. Luego podemos elegir $j > k$ tal que $c_j \neq 0$, y siguiendo la demostración del Lema de Intercambio, obtenemos que reemplazando a $\boldsymbol{v}_j$ por $\boldsymbol{w}_k$ obtenemos una nueva base de V.

Aplicando este razonamiento inductivamente, concluimos que existen $\{\boldsymbol{w}_1, \dots, \boldsymbol{w}_s\} \subset \tilde{\mathcal{B}}$ que forman una base de V. Por lo tanto $\tilde{\mathcal{B}}$ es exactamente igual a $\{\boldsymbol{w}_1, \dots, \boldsymbol{w}_s\}$, y tiene s elementos. □

Cuando V tiene una base finita, llamamos dimensión de V a la cantidad de elementos de la base, y decimos que V es un espacio de dimensión finita.

EJEMPLO 1.12.

1. $\mathbb{R}^n$ es un espacio vectorial de dimensión n. Una base de $\mathbb{R}^n$ es $\{\boldsymbol{e}_i : i = 1, \dots, n\}$, con $\boldsymbol{e}_i$ el vector que tiene 1 en la entrada i y 0 en todas las demás. Llamamos base canónica a esta base.
2. $\mathbb{R}[x]_d$ es un espacio vectorial de dimensión $d+1$. La base canónica de $\mathbb{R}[x]_d$ es $\{1, x, x^2, \dots, x^d\}$.
3. $\mathbb{R}[x]$ es un espacio vectorial de dimensión infinita.

En esta materia vamos a trabajar sobre espacios de dimensión finita.

PROPOSICIÓN 1.4. Todo $\mathbb{R}$-espacio vectorial de dimensión finita n es isomorfo a $\mathbb{R}^n$ como espacio vectorial (es decir, si solo consideramos las operaciones de espacio vectorial).

EJEMPLO 1.13.

1. Las matrices de $m \times n$ podemos pensarlas como vectores de longitud $m \times n$.
2. Los polinomios de grado d podemos representarlos por sus vectores de coeficientes, de longitud $d+1$. Por ejemplo, en $\mathbb{R}[x]_3$ representamos al polinomio $x^3 - 3x + 2$ por el vector $(1, 0, -3, 2)$.

En general, si $\mathcal{B} = \{\boldsymbol{b}_1, \dots, \boldsymbol{b}_n\}$ es una base de un espacio vectorial V y $\boldsymbol{v} \in V$, vimos que $\boldsymbol{v}$ admite una escritura única como combinación lineal de elementos de $\mathcal{B}$, $\boldsymbol{v} = a_1\boldsymbol{b}_1 + \cdots + a_n\boldsymbol{b}_n$. Podemos representar a $\boldsymbol{v}$ en la base $\mathcal{B}$ como $\boldsymbol{v} = (a_1, \dots, a_n)_{\mathcal{B}}$.

Por ejemplo, $p(x) = x^3 - 3x + 2 = (1, 0, -3, 2)_{\mathcal{B}}$, con $\mathcal{B} = \{x^3, x^2, x, 1\}$.

3.3. Base a partir de sistema de generadores. Como vimos, para un espacio vectorial V de dimensión finita n, cualquier base de V tiene la misma cantidad de elementos n. Esto implica los siguientes resultados directos, cuya demostración dejamos como ejercicio.

- Un conjunto $S \subset V$ con menos de n elementos no puede generar todo V.
- Si $S \subset V$ es un conjunto linealmente independiente de n elementos, S es una base de V.
- Un conjunto $S \subset V$ con más de n elementos no puede ser linealmente independiente.

Más aún, tenemos el siguiente resultado.

PROPOSICIÓN 1.5. Si V es un espacio de dimensión n y $S \subset V$ es un conjunto de generadores de m elementos, con $m > n$, podemos extraer de S un subconjunto $\mathcal{B}$ de n elementos que forma una base de V.

DEMOSTRACIÓN. Veamos que si $m > n$, podemos eliminar algún vector de S y seguir teniendo un sistema de generadores. Como S no puede ser un conjunto linealmente independiente, existe una combinación lineal no nula

$$0 = a_1\boldsymbol{w}_1 + \cdots + a_s\boldsymbol{w}_s,$$

y suponemos por simplicidad $a_s \neq 0$. Luego $\boldsymbol{w}_s$ pertenece al espacio generado por $\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{s-1}\}$, y por lo tanto podemos eliminarlo.

Podemos repetir este proceso inductivamente siempre que la cantidad de vectores resultantes sea mayor que n, hasta llegar a tener un subconjunto de n elementos que generan V y por lo tanto es una base de V. □

Notamos también que dado un conjunto de vectores $S = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_s\}$, podemos obtener una base del espacio vectorial generado por S triangulando el sistema, es decir

1. Colocamos los vectores como filas de una matriz.
2. Triangulamos la matriz utilizando eliminación gaussiana.
3. Tomamos los vectores correspondientes a las filas no nulas de la matriz.

EJEMPLO 1.14. Dado el subespacio $S = \langle(1,4,3,-1),(1,0,1,0),(3,3,2,7),(2,6,0,14),(2,3,1,7)\rangle \subset \mathbb{R}^4$, para obtener una base de S construimos la matriz

$$A = \begin{pmatrix} 1 & 4 & 3 & -1 \\ 1 & 0 & 1 & 0 \\ 3 & 3 & 2 & 7 \\ 2 & 6 & 0 & 14 \\ 2 & 3 & 1 & 7 \end{pmatrix}$$

y triangulamos.

```
A = np.array([[1, 4, 3, -1,], [1, 0, 1, 0], [3, 3, 2, 7], [2, 6, 0,
   14], [2, 3, 1, 7]])
print(row_echelon(A))
```

```
%% [[ 1.    4.    3.   -1.  ]
%%  [ 0.    1.    0.5  -0.25]
%%  [ 0.    0.    1.   -3.1 ]
```

```
%%  [ 0.    0.    0.    0.  ]
%%  [ 0.    0.    0.    0.  ]]
```

Obtenemos la matriz

$$A' = \begin{pmatrix} 1 & 4 & 3 & -1 \\ 0 & 1 & 1/2 & -1/4 \\ 0 & 0 & 1 & -31/10 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Por lo tanto una base de S es $\mathcal{B} = \{(1, 4, 3, -1), (0, 1, 1/2, -1/4), (0, 0, 1, -31/10)\}$.

A partir de la Proposición **??**, podemos deducir el siguiente resultado de extensión de una base.

PROPOSICIÓN 1.6 (Extensión de una base). Dado un espacio vectorial V de dimensión n y una base $\mathcal{B} = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_s\}$ de un subespacio S, es posible encontrar vectores $\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{n-s}\}$ tales que

$$\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_s\, \boldsymbol{w}_1, \ldots, \boldsymbol{w}_{n-s}\}$$

forman una base de V.

DEMOSTRACIÓN. Ejercicio. □

3.4. Espacios dados por ecuaciones homogéneas. Dado un sistema de ecuaciones $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}$, con $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ y $\boldsymbol{b} \in \mathbb{R}^n$, es fácil ver que el conjunto S de soluciones es un subespacio vectorial de $\mathbb{R}^n$.

Por lo tanto, podemos encontrar una base de S como subespacio de $\mathbb{R}^n$. Una forma de encontrar una base es triangular el sistema. Cada fila no nula en la matriz escalonada impone una restricción en el espacio y reduce en uno la dimensión. Es decir, que la dimensión de S es $n - k$ donde k es la cantidad de filas no nulas en la triangulación.

EJEMPLO 1.15. El sistema

$$\begin{cases} x_1 + 2x_2 - x_3 + x_4 = 0 \\ \quad 3x_2 - 2x_3 - x_4 = 0, \end{cases}$$

ya está escalonado. Por lo tanto el espacio de soluciones tiene dimensión $4 - 2 = 2$.

Para encontrar un sistema de generadores, despejamos la primera variable de cada ecuación en el sistema triangulado en función de las demas variables. Llamamos *variables dependientes* a las variables que aparecer como primer variable de alguna fila, y variables libres o independientes a las demás. En este ejemplo, $\{x_1, x_2\}$ son las variables dependientes y $\{x_3, x_4\}$ las variables libres (estos conjuntos dependen del método que utilizamos para resolver el sistema, podemos obtener otros conjuntos de variables dependientes y libres, aunque la cantidad de variables en cada uno será siempre la misma).

Ahora despejamos las variables dependientes en función de las variables libres, de abajo para arriba:

$$\begin{cases} x_2 = \dfrac{2x_3 + x_4}{3} = \dfrac{2}{3}x_3 + \dfrac{1}{3}x_4 \\ x_1 = -2x_2 + x_3 - x_4 = -2\left(\dfrac{2}{3}x_3 + \dfrac{1}{3}x_4\right) + x_3 - x_4 = -\dfrac{1}{3}x_3 - \dfrac{5}{3}x_4, \end{cases}$$

y podemos escribir el conjunto de soluciones en la forma

$$S = \left\{ \left(-\frac{1}{3}x_3 - \frac{5}{3}x_4, \frac{2}{3}x_3 + \frac{1}{3}x_4, x_3, x_4 \right) : x_3, x_4 \in \mathbb{R} \right\}.$$

A partir de esta escritura, es fácil obtener los generadores del subespacio vectorial:

$$\begin{aligned} S &= \left\{ \left(-\frac{1}{3}x_3, \frac{2}{3}x_3, x_3, 0 \right) + \left(-\frac{5}{3}x_4, \frac{1}{3}x_4, 0, x_4 \right) : x_3, x_4 \in \mathbb{R} \}. : x_3, x_4 \in \mathbb{R} \right\} \\ &= \left\{ \left(-\frac{1}{3}, \frac{2}{3}, 1, 0 \right) x_3 + \left(-\frac{5}{3}, \frac{1}{3}, 0, 1 \right) x4 : x_3, x_4 \in \mathbb{R} \}. : x_3, x_4 \in \mathbb{R} \right\}, \end{aligned}$$

y concluimos $S = \langle \left(-\frac{1}{3}, \frac{2}{3}, 1, 0\right), \left(-\frac{5}{3}, \frac{1}{3}, 0, 1\right) \rangle$.

3.5. Ecuaciones a partir de generadores. En la sección anterior, vimos cómo obtener los generadores de un subespacio vectorial dado por ecuaciones homogéneas. En ocasiones es útil realizar el proceso inverso, obtener ecuaciones homogéneas que definen un espacio vectorial dado por generadores.

Veamos cómo hacerlo en un ejemplo.

EJEMPLO 1.16. Hallar las ecuaciones que definen el espacio generado por los vectores $S = \{(1, 0, 2, -1), (3, 1, 0, 2)\}$. Un vector genérico de S es

$$(x_1, x_2, x_3, x_4) = a_1(1, 0, 2, -1) + a_2(3, 1, 0, 2).$$

Podemos platear esta ecuación en forma matricial:

$$\begin{pmatrix} 1 & 3 \\ 0 & 1 \\ 2 & 0 \\ -1 & 2 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}.$$

Ahora buscamos relaciones entre los x_i. Para eso triangulamos la matriz ampliada

$$\left(\begin{array}{cc|c} 1 & 3 & x_1 \\ 0 & 1 & x_2 \\ 2 & 0 & x_3 \\ -1 & 2 & x_4 \end{array} \right).$$

Al realizar la triangulación, obtenemos la matriz

$$\left(\begin{array}{cc|c} 1 & 3 & x_1 \\ 0 & 1 & x_2 \\ 0 & 0 & x_3 - 2x_1 + 3x_2 \\ 0 & 0 & x_4 + x_1 + 5x_2 \end{array} \right),$$

de donde obtenemos las relaciones:

$$\begin{cases} x_3 - 2x_1 + 3x_2 = 0 \\ x_4 + x_1 + 5x_2 = 0 \end{cases}$$

y estas son las ecuaciones que definen nuestro espacio vectorial, es decir,

$$S = \{(x_1, x_2, x_3, x_4) : -2x_1 + 3x_2 + x_3 = 0, x_1 + 5x_2 + x_4 = 0\}.$$

En general, dados generadores de un espacio vectorial $S = \langle \boldsymbol{v}_1, \ldots, \boldsymbol{v}_s \rangle \subset \mathbb{R}^n$, realizamos los siguientes pasos para obtener ecuaciones que definen a S.

1. Construir la matriz ampliada correspondiente al sistema de ecuaciones
$$(x_1, \ldots, x_n) = a_1\boldsymbol{v}_1 + \cdots + a_s\boldsymbol{v}_s.$$
2. Escalonar la matriz.
3. El espacio S queda definido por las ecuaiones correspondientes a los términos de la última columna de las filas en las que todos los coeficientes en las primeras s columnas son nulos.

3.6. Suma e intersección de subespacios. Como aplicación de las últimas dos secciones veamos cómo calcular sumas e intersecciones de subespacios vectoriales.

DEFINICIÓN 1.2. Dados subespacios S, T de un $\mathbb{K}$ espacio vectorial W, definimos
- **Suma.** $S + T = \{\boldsymbol{s} + \boldsymbol{t} : \boldsymbol{s} \in S, \boldsymbol{t} \in T\}$, el conjunto de todas las sumas posibles entre un elemento de S y un elemento de T.
- **Intersección.** $S \cap T = \{\boldsymbol{v} \in W : \boldsymbol{v} \in S \text{ y } \boldsymbol{v} \in T\}$.

Queda como ejercicio verificar que estos dos conjuntos son subespacios vectoriales de W.

Concentremos un poco nuestra atención en el caso en que $W = \mathbb{K}^n$. Dependiendo si S y T vienen dados por generadores o ecuaciones puede ser más fácil o más difícil calcular estos subespacios. Podemos resumir los métodos de la siguiente forma (dejamos como ejercicio verificar la correctitud de los métodos).

- Si S y T están dados por generadores, $S + T$ está generado por la unión de todos los generadores.
- Si S y T están dados por ecuaciones, $S \cap T$ está generado por la unión de todas las ecuaciones.
- Si S y/o T están dados por ecuaciones, calculamos generadores de ambos y tomamos la unión.
- Si S y/o T están dados por generadores, calculamos ecuaciones de ambos y tomamos la unión de las ecuaciones.

Con métodos similares podemos realizar otras operaciones entre subespacios. Por ejemplo, ¿cómo podemos determinar si $S \subset T$? Ahora el caso más simple es si S está dado por generadores y T por ecuaciones. En este caso, alcanza verificar si todos los generadores de S son elementos de T verificando si cumplen las ecuaciones que definen a T.

EJERCICIO 1.2. Dados los subespacios de $\mathbb{R}^4$, $S = \langle (1, 0, -3, 1), (2, 0, 3, -2) \rangle$ y $T = \{(x_1, x_2, x_3, x_4) : x_1 + x_2 - x_4 = 0, x_2 + x_3 = 0\}$, hallar generadores de $S + T$ y $S \cap T$. Determinar si $S \subset T$.

Es simple determinar la relación entre las dimensiones de los espacios S,U, su suma y su intersección. De eso trata el siguiente resultado.

PROPOSICIÓN 1.7. Sean U y V dos subespacios de un $\mathbb{K}$ espacio vectorial de dimensión finita. Entonces
$$dim(U + V) = dim(U) + dim(V) - dim(U \cap V).$$

DEMOSTRACIÓN. Probemos primero el caso en que $U \cap V = \{\mathbf{0}\}$. En ese caso, $dim(U \cap V) = 0$, por lo cual basta ver que $dim(U+V) = dim(U) + dim(V)$.

Para ello consideramos $\mathcal{B} = \{\boldsymbol{u}_1, \cdots, \boldsymbol{u}_k\}$ y $\mathcal{B}' = \{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n\}$ bases de U y V respectivamente y notemos que si tomamos un $\boldsymbol{w} \in U+V$ deben existir $\boldsymbol{u} \in U$ y $\boldsymbol{v} \in V$ de modo tal que $\boldsymbol{w} = \boldsymbol{u} + \boldsymbol{v}$. En particular, como $\mathcal{B}$ y $\mathcal{B}'$ son bases de U y V, deben existir escalares $\{\alpha_i\}_{1 \leq i \leq k}$ y $\{\beta_j\}_{1 \leq j \leq n}$ de modo tal que

$$\boldsymbol{u} = \sum_{i_1}^{k} \alpha_i \boldsymbol{u}_i \qquad \boldsymbol{v} = \sum_{j=1}^{n} \beta_j \boldsymbol{v}_j,$$

de donde concluímos que el conjunto

$$\tilde{\mathcal{B}} = \mathcal{B} \cup \mathcal{B}' = \{\boldsymbol{u}_1, \cdots, \boldsymbol{u}_k, \boldsymbol{v}_1, \cdots, \boldsymbol{v}_n\}$$

es un sistema de generadores de $U+V$. Decimos que $\tilde{\mathcal{B}}$ es L.I. con lo cual resultaría una base y por ende $dim(U+V) = \#\mathcal{B} + \#\mathcal{B}' = dim(U) + dim(V)$ como queríamos probar. Para ver que $\tilde{\mathcal{B}}$ es L.I. suponemos que para cierta combinación lineal, se tiene

$$\sum_{i_1}^{k} \alpha_i \boldsymbol{u}_i + \sum_{j=1}^{n} \beta_j \boldsymbol{v}_j = \mathbf{0},$$

y notamos que en ese caso

$$U \ni \sum_{i_1}^{k} \alpha_i \boldsymbol{u}_i = \sum_{j=1}^{n} (-\beta_j) \boldsymbol{v}_j \in V,$$

lo que dice que ambos miembros (que son iguales) pertenecen tanto a U como a V y por ende a $U \cap V = \{\mathbf{0}\}$. Luego

$$\sum_{i_1}^{k} \alpha_i \boldsymbol{u}_i = \mathbf{0} = \sum_{j=1}^{n} (-\beta_j) \boldsymbol{v}_j,$$

y resulta $\alpha_1 = \cdots = \alpha_k = \beta_1 = \cdots = \beta_n = 0$, indicando que $\tilde{\mathcal{B}}$ es L.I.

Resta probar el caso en que $U \cap V \neq \{\mathbf{0}\}$. En este caso tomamos una base $\mathcal{B}'' = \{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_r\}$ de $U \cap V$ y como $U \cap V \subset U$ gracias a la Proposición **??**, podemos extender $\mathcal{B}''$ a una base $\mathcal{B}$ de U, $\mathcal{B} = \{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_r, \boldsymbol{u}_{r+1}, \cdots, \boldsymbol{u}_k\}$. Análogamente, $\mathcal{B}''$ se extiende a una base $\mathcal{B}'$ de V, $\mathcal{B}' = \{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_r, \boldsymbol{v}_{r+1}, \cdots, \boldsymbol{v}_n\}$. Notemos, antes de proseguir, que las dimensiones de los espacios involucrados son r,k y n para $U \cap V$, U y V respectivamente. Decimos que $\tilde{\mathcal{B}} = \{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_r, \boldsymbol{u}_{r+1}, \cdots, \boldsymbol{u}_k, \boldsymbol{v}_{r+1}, \cdots, \boldsymbol{v}_n\}$ es base de $U+V$. En efecto, es trivial ver que genera y para ver que es L.I. escribimos

$$\sum_{i=1}^{r} \gamma_i \boldsymbol{z}_i + \sum_{j=r+1}^{n} \alpha_j \boldsymbol{u}_j + \sum_{l=r+1}^{k} \beta_l \boldsymbol{v}_l = \mathbf{0},$$

y entonces, por un lado se tiene

$$U \ni \sum_{i=1}^{r} \gamma_i \boldsymbol{z}_i + \sum_{j=r+1}^{k} \alpha_j \boldsymbol{u}_j = \sum_{l=r+1}^{n} (-\beta_l) \boldsymbol{v}_l \in V, \tag{1.1}$$

de donde en particular $\sum_{l=r+1}^{n}(-\beta_l)\boldsymbol{v}_l \in U \cap V = \langle \boldsymbol{z}_1, \cdots, \boldsymbol{z}_r \rangle$, es decir, que $\beta_{r+1} = \beta_{r+2} = \cdots = \beta_n = 0$. Volviendo a (**??**), vemos que

$$\sum_{i=1}^{r} \gamma_i \boldsymbol{z}_i + \sum_{j=r+1}^{k} \alpha_j \boldsymbol{u}_j = \mathbf{0},$$

y entonces $\gamma_1 = \gamma_2 = \cdots = \gamma_r = \alpha_{r+1} = \cdots = \alpha_k = 0$ porque $\mathcal{B}$ es base y por lo tanto L.I.

Hemos probado que

$$\tilde{\mathcal{B}} = \{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_r, \boldsymbol{u}_{r+1}, \cdots, \boldsymbol{u}_k, \boldsymbol{v}_{r+1}, \cdots, \boldsymbol{v}_n\}$$

es una base de $U + V$, por lo tanto

$$dim(U + V) = r + (k - r) + (n - r) = k + n - r = dim(U) + dim(V) - dim(U \cap V),$$

como queríamos probar. □

4. Operaciones con matrices

4.1. Multiplicación de matrices. Dadas matrices $A \in \mathbb{K}^{m\times n}$ y $B \in \mathbb{K}^{n\times p}$

$$\boldsymbol{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}, \quad \boldsymbol{B} = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1p} \\ b_{21} & b_{22} & \cdots & b_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{np} \end{pmatrix}$$

el producto $\boldsymbol{C} = \boldsymbol{AB}$ es la matriz de m filas y p columnas

$$\boldsymbol{C} = \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1p} \\ c_{21} & c_{22} & \cdots & c_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mp} \end{pmatrix} \in \mathbb{K}^{m\times p},$$

con

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{in}b_{nj} = \sum_{k=1}^{n} a_{ik}b_{kj}$$

(en la casilla (i, j) de $\boldsymbol{C}$ ponemos el producto de la fila i de $\boldsymbol{A}$ con la columna j de $\boldsymbol{B}$) .

Si multiplicamos dos matrices o vectores en `Python` con $*$ ...

```
A1 = np.array([[1,2],[3,4]])
A2 = np.array([[1,0],[0,1]])
print(A1)
print(A2)
print(A1 * A2)
```

```
%% A1 =
%%  [[1 2]
%%  [3 4]]
%% A2 =
%%  [[1 0]
%%  [0 1]]
%% A1 * A2 =
%%  [[1 0]
%%  [0 4]]
```

La operación $*$ en `Python` calcula el producto coordenada a coordenada. Esta operación se llama producto de Hadamard, y es útil en muchos casos, pero no es el producto usual de matrices.

Para el producto usual de matrices usamos en Python el símbolo @.

```
print("A1 @ A2 = \n", A1 @ A2)   # Producto usual de matrices
```

```
%% A1 @ A2 =
```

```
%%  [[1 2]
%%  [3 4]]
```

¡Revisar siempre que estemos usando el producto correcto en `Python`!

EJERCICIO 1.3. Realizar (a mano) los siguientes productos de matrices:

1. $\begin{pmatrix} 3 & 5 \\ 2 & -9 \end{pmatrix} \cdot \begin{pmatrix} 1 & -1 \\ 0 & 2 \end{pmatrix}$
2. $\begin{pmatrix} 3 & 5 \end{pmatrix} \cdot \begin{pmatrix} 4 \\ -3 \end{pmatrix}$
3. $\begin{pmatrix} 4 \\ -3 \end{pmatrix} \cdot \begin{pmatrix} 3 & 5 \end{pmatrix}$
4. $\begin{pmatrix} 3 & 5 \\ 2 & -9 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix}$

Observaciones:

- El producto de matrices es asociativo. Es decir $\boldsymbol{A}(\boldsymbol{BC}) = (\boldsymbol{AB})\boldsymbol{C}$, para matrices con los tamaños apropiados para realizar los productos.
- El producto de matrices no es conmutativo. En general, no vale $\boldsymbol{AB} = \boldsymbol{BA}$, ni siquiera en el caso de matrices cuadradas. Lo verificamos en `Python`.

```
A = np.array([[1, 2], [2, 3]])
B = np.array([[2, 5], [1, 4]])
print("A @ B = \n", A @ B)
print("B @ A = \n", B @ A)
```

```
A @ B =
 [[ 4 13]
 [ 7 22]]
B @ A =
 [[12 19]
 [ 9 14]]
```

- Si bien vimos que el conjunto de matrices $\mathbb{K}^{m\times n}$ es un espacio vectorial, el producto de matrices no es una operación de espacio vectorial.

4.2. Matrices transpuesta y conjugada. La matriz transpuesta de $\boldsymbol{A} = (a_{ij}) \in \mathbb{R}^{m\times n}$ es la matriz que se obtienen a partir de $\boldsymbol{A}$ intercambiando filas por columnas, es decir $\boldsymbol{A}^T = (a_{ji}) \in \mathbb{R}^{n\times m}$.

EJEMPLO 1.17.

- Si $\boldsymbol{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{pmatrix}$, $\boldsymbol{A}^T = \begin{pmatrix} a_{11} & a_{21} \\ a_{12} & a_{22} \\ a_{13} & a_{23} \end{pmatrix}$.
- Si $\boldsymbol{A} = \begin{pmatrix} 5 & 7 \\ 3 & -1 \end{pmatrix}$, $\boldsymbol{A}^T = \begin{pmatrix} 5 & 3 \\ 7 & -1 \end{pmatrix}$.

PROPOSICIÓN 1.8.

- $(\boldsymbol{A}^T)^T = \boldsymbol{A}$.
- $(\boldsymbol{AB})^T = \boldsymbol{B}^T\boldsymbol{A}^T$.

En `Python` calculamos la transpuesta de una matriz con el comando `transpose` de `numpy`.

```
A = np.array([[5,7],[3,-1]])
print(np.transpose(A))
```

```
## [[ 5  3]
##  [ 7 -1]]
```

En lo que sigue resulta de utilidad definir lo siguiente: consideremos $\boldsymbol{v} \in \mathbb{K}^n$, $\boldsymbol{A} \in \mathbb{K}^{n\times n}$, con $\overline{\boldsymbol{v}} \in \mathbb{K}^n$ (resp. $\overline{\boldsymbol{A}} \in \mathbb{K}^{n\times n}$), denotamos el vector (resp. matrix) que tiene los mismos elementos que $\boldsymbol{v}$ (resp. $\boldsymbol{A}$) pero conjugados. Obviamente si $\mathbb{K} = \mathbb{R}$, $\overline{\boldsymbol{v}} = \boldsymbol{v}$, $\overline{\boldsymbol{A}} = \boldsymbol{A}$.

La siguiente definición generaliza a la traspuesta en las matrices y a la conjugación en los escalares.

Dada $\boldsymbol{A} \in \mathbb{K}^{n\times m}$, la *matriz conjugada* de $\boldsymbol{A}$, denotada con $\boldsymbol{A}^*$ se define como $\boldsymbol{A}^* = \overline{\boldsymbol{A}^T}$.

Notar que si $a \in \mathbb{K}^{1\times 1}$ (o sea, es un escalar, $a \in \mathbb{K}$) entonces $a^* = \overline{a}$. La conjugación hereda las propiedades de la traspuesta:

- $(\boldsymbol{A}^*)^* = \boldsymbol{A}$
- Si $\mathbb{K} = \mathbb{R}$, $\boldsymbol{A}^* = \boldsymbol{A}^T$.
- $(\boldsymbol{AB})^* = \boldsymbol{B}^*\boldsymbol{A}^*$
- $(a\boldsymbol{A} + b\boldsymbol{B})^* = \overline{a}\boldsymbol{A}^* + \overline{b}\boldsymbol{B}^*$.

4.3. Sistemas lineales y ecuaciones matriciales. Observando el producto del punto **??** del Ejercicio **??**, vemos que podemos plantear el sistema de ecuaciones lineales

$$\begin{cases} 3x + 5y = 4 \\ 2x - 9y = 15. \end{cases}$$

en forma matricial

$$\begin{pmatrix} 3 & 5 \\ 2 & -9 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 4 \\ 15 \end{pmatrix},$$

o en general, $\boldsymbol{Ax} = \boldsymbol{b}$, donde $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ es la matriz de coeficientes, $\boldsymbol{x}$ es un vector o matriz columna de incógnitas y $\boldsymbol{b} \in \mathbb{R}^n$ es el vector de términos constantes.

Pregunta: ¿cuántas incógnitas y cuántas ecuaciones tiene el sistema $\boldsymbol{Ax} = \boldsymbol{b}$ con los tamaños dados?

Para resolver sistemas de ecuaciones en `Python` podemos usar el comando `solve` (vamos a volver a esto más adelante).

```
A = np.array([[3,2],[5,-4]])
b = np.array([22, -11])
print(np.linalg.solve(A, b))
```

```
## [3.  6.5]
```

4.4. Rango de una matriz. Para saber si un sistema de ecuaciones puede tener solución única, infinitas soluciones o ninguna solución, calculamos el *rango* de la matriz A. Pensando las filas de una matriz como ecuaciones, el rango de la matriz nos indica cuantas ecuaciones ïndependientes"tenemos.

El rango de una matriz es la máxima cantidad de filas o columnas linealmente independientes que posee la matriz.

Proposición 1.9. La máxima cantidad de filas linealmente independientes de una matriz coincide con la máxima cantidad de columnas linealmente independientes.

Por lo tanto, no es necesario distinguir rango-fila de rango-columna, y hablamos simplemente de *rango*.

Ejemplo 1.18.

1. $\text{rango}\left(\begin{pmatrix} 1 & 4 & 7 \\ 0 & 2 & 6 \end{pmatrix}\right) = 2$
2. $\text{rango}\left(\begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 2 \\ 1 & 2 & 5 \end{pmatrix}\right) = 1$

4.4.1. ¿Cómo calcular el rango? Para calcular el rango de una matriz, triangulamos la matriz y contamos cuántas filas no nulas quedan.

Proposición 1.10. El espacio vectorial generado por las filas de una matriz antes o después de triangular es el mismo.

En `Python`, calculamos el rango de una matriz con el comando `matrix_rank` de `numpy.linalg`.

```
A = np.array([[1,0,1],[0,1,2],[1,2,5]])
print("Rango de A = ", np.linalg.matrix_rank(A))

B = np.array([[1,2,3],[4,5,6],[7,8,9]])
print("Rango de B = ", np.linalg.matrix_rank(B))
```

```
## Rango de A =  2
## Rango de B =  2
```

4.4.2. Sistemas de ecuaciones y rango de matrices. Si $\text{rango}(\boldsymbol{A})$ es igual a la cantidad de filas, el sistema $\boldsymbol{Ax} = \boldsymbol{b}$ siempre tiene solución. Todas las ecuaciones son independientes entre sí.

Si la matriz $\boldsymbol{A}$ es cuadrada y $\text{rango}(\boldsymbol{A})$ es igual a la cantidad de filas, decimos que $\boldsymbol{A}$ tiene rango máximo. En este caso, el sistema $\boldsymbol{Ax} = \boldsymbol{b}$ tiene solución única para todo $\boldsymbol{b}$.

4.4.3. Rango de un producto de matrices. Dadas matrices $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ y $\boldsymbol{B} \in \mathbb{R}^{n\times p}$,

1. $\text{rango}(\boldsymbol{A}) = \text{rango}(\boldsymbol{A}^T) = \text{rango}(\boldsymbol{A}^T\boldsymbol{A}) = \text{rango}(\boldsymbol{A}\boldsymbol{A}^T)$

2. $\text{rango}(\boldsymbol{AB}) \leq \text{mín}\{\text{rango}(\boldsymbol{A}), \text{rango}(\boldsymbol{B})\}$.
3. Si $\text{rango}(\boldsymbol{B}) = n$, entonces $\text{rango}(\boldsymbol{AB}) = \text{rango}(\boldsymbol{A})$.
4. Si $\boldsymbol{B} \in \mathbb{R}^{n\times n}$, y $\boldsymbol{B}$ tiene rango máximo, entonces $\text{rango}(\boldsymbol{AB}) = \text{rango}(\boldsymbol{A})$.

Idea de las demostración: las filas de $\boldsymbol{AB}$ son combinaciones de las filas de $\boldsymbol{B}$ y las columnas de $\boldsymbol{AB}$ son combinaciones de las columnas de $\boldsymbol{A}$.

4.5. Matrices especiales.

4.5.1. Matriz diagonal. Una matriz $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ se dice diagonal si $a_{ij} = 0$ para todo $i \neq j$, es decir:

$$\boldsymbol{A} = \begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{pmatrix}$$

Ejercicio 1.4. Calcular a mano pero sin hacer muchas cuentas:

1. $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}^3$,
2. $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{pmatrix}$,
3. $\begin{pmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}$.

```
# Usamos el comando diag para definir matrices diagonales
D = np.diag(np.array([1,2,3]))
print("D = \n", D)
print("D^2 = \n", D @ D)
```

```
%% D =
%%  [[1 0 0]
%%  [0 2 0]
%%  [0 0 3]]
%% D^2 =
%%  [[1 0 0]
%%  [0 4 0]
%%  [0 0 9]]
```

4.5.2. Matriz identidad. Para cada $n \in \mathbb{N}$, la matriz $I_n \in \mathbb{R}^{n\times n}$ es la matriz diagonal con $a_{ii} = 1$ para todo $1 \leq i \leq n$ y $a_{ij} = 0$ para $i \neq j$, es decir:

$$I_n = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}$$

PROPOSICIÓN 1.11. Para toda $\boldsymbol{B} \in \mathbb{R}^{n\times n}$, $I_n \cdot \boldsymbol{B} = \boldsymbol{B} \cdot I_n = \boldsymbol{B}$.

Podemos usar el comando `eye` de `numpy` para definir una matriz identidad.

```
id = np.eye(3)

# Creamos una matriz de 3 x 3 con números aleatorios entre 0 y 1
A = np.random.rand(3,3)
print("A = \n", A)
print("id * A = \n", id @ A)
```

```
%% A =
%%  [[0.91836939 0.31101594 0.48474114]
%%  [0.00093026 0.37622722 0.91214465]
%%  [0.6293026  0.20801675 0.6855896 ]]
%% id * A =
%%  [[0.91836939 0.31101594 0.48474114]
%%  [0.00093026 0.37622722 0.91214465]
%%  [0.6293026  0.20801675 0.6855896 ]]
```

4.5.3. Mas Matrices especiales.

- **Triangular superior:** Una matriz $\boldsymbol{U} \in \mathbb{R}^{n\times n}$ se llama triangular superior si $a_{ij} = 0$ para todo $i > j$ (es decir, todas las entradas abajo de la diagonal son 0).
- **Triangular inferior:** Una matriz $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ se llama triangular inferior si $a_{ij} = 0$ para todo $i > j$ (es decir, para todas las entradas arriba de la diagonal son 0).
- **Matriz simétrica:** $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es simétrica si $\boldsymbol{A}^T = \boldsymbol{A}$.
- **Matriz Hermitiana:** $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ es Hermitiana si $\boldsymbol{A}^* = \boldsymbol{A}$.
- **Matriz ortogonal:** $\boldsymbol{A}$ es ortogonal si $\boldsymbol{A}\boldsymbol{A}^T = \boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{I}_n$.
- **Matriz unitaria:** $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ es unitaria si $\boldsymbol{A}\boldsymbol{A}^* = \boldsymbol{A}^*\boldsymbol{A} = \boldsymbol{I}_n$.

4.5.4. Inversa de una matriz. Dada una matriz $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, si existe una matriz $\boldsymbol{B} \in \mathbb{R}^{n\times n}$ tal que

$$\boldsymbol{A}\boldsymbol{B} = \boldsymbol{I}_n = \boldsymbol{B}\boldsymbol{A},$$

decimos que $\boldsymbol{A}$ es inversible y $\boldsymbol{B}$ es la inversa de $\boldsymbol{A}$. Si $\boldsymbol{A}$ es inversible, la inversa es única y la notamos $\boldsymbol{A}^{-1}$. Si $\boldsymbol{A}$ no es inversible, decimos que $\boldsymbol{A}$ es singular.

Para calcular la inversa de una matriz podemos aplicar eliminación gaussiana, observando que encontrar una matriz B tal que $\boldsymbol{AB} = \boldsymbol{I}_n$ es equivalente a resolver los n sistemas de ecuaciones

$$\boldsymbol{A}\begin{pmatrix} b_{1j} \\ b_{2j} \\ \vdots \\ b_{nj} \end{pmatrix} = \boldsymbol{e}_j,$$

donde $\boldsymbol{e}_j$ es el j-ésimo vector canónico, que coinicide con la j-ésima columna de la matriz $\boldsymbol{I}_n$.

Ejemplo 1.19. Calculamos la inversa de la matriz

$$\boldsymbol{A} = \begin{pmatrix} 2 & 0 & -2 \\ 1 & -1 & 1 \\ 0 & 2 & 1 \end{pmatrix}.$$

Podemos resolver los sistemas $\boldsymbol{Ab}_j = \boldsymbol{e}_j$, $1 \leq j \leq 3$, simultáneamente considerando una matriz ampliada con los 3 vectores $\boldsymbol{e}_j$ a la derecha:

$$\left(\begin{array}{ccc|ccc} 2 & 0 & -2 & 1 & 0 & 0 \\ 1 & -1 & 1 & 0 & 1 & 0 \\ 0 & 2 & 1 & 0 & 0 & 1 \end{array}\right).$$

```
A = np.array([[1, 4, 3, -1,], [1, 0, 1, 0], [3, 3, 2, 7], [2, 6, 0,
   14], [2, 3, 1, 7]])
print(row_echelon(A))
```

```
%% [[ 2.  0. -2.  1.  0.  0.]
%%  [ 1. -1.  1.  0.  1.  0.]
%%  [ 0.  2.  1.  0.  0.  1.]]
%% Matriz escalonada:
%%  [[ 1.   0.  -1.   0.5  0.   0. ]
%%  [ 0.   1.  -2.   0.5 -1.  -0. ]
%%  [ 0.   0.   1.  -0.2  0.4  0.2]]
```

La función `row_echelon` que estamos utilizando coloca 1's como primer elemento no nulo de cada fila, dividiendo cada fila por el escalar apropiado.

A partir de la matriz escalonada podemos ahora encontrar fácilmente las casillas b_{ij} de $\boldsymbol{B}$. Por ejemplo, la última fila será $[-0.20.40.2]$. Para las filas anteriores, podemos realizar ahora la triangulación hacia arriba:

$$\left(\begin{array}{ccc|ccc} 1. & 0. & -1. & 0.5 & 0. & 0. \\ 0. & 1. & -2. & 0.5 & -1. & 0. \\ 0. & 0. & 1. & -0.2 & 0.4 & 0.2 \end{array}\right) \xrightarrow[f_1+f_3\to f_1]{f_2+2f_3\to f_2} \left(\begin{array}{ccc|ccc} 1. & 0. & 0. & 0.5 & 0.4 & 0.2 \\ 0. & 1. & 0. & 0.5 & 0.2 & 0.4. \\ 0. & 0. & 1. & -0.2 & 0.4 & 0.2 \end{array}\right).$$

Ahora la matriz que estamos buscando es exactamente la matriz formada por las últimas 3 columnas de la matriz ampliada.

$$\boldsymbol{B} = \begin{pmatrix} 0.3 & 0.4 & 0.2 \\ 0.1 & 0.2 & 0.4. \\ -0.2 & 0.4 & 0.2 \end{pmatrix}.$$

Lo verificamos en Python.

```
B = np.array([[0.3, 0.4, 0.2], [0.1, -0.2, 0.4], [-0.2, 0.4, 0.2]])
print(A @ B)
```

```
%% [[ 1.00000000e+00  0.00000000e+00  0.00000000e+00]
%%  [-2.77555756e-17  1.00000000e+00  0.00000000e+00]
%%  [ 0.00000000e+00  0.00000000e+00  1.00000000e+00]]
```

APLICACIÓN 1.2. Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es inversible, la solución del sistema

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

es $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$.

Para calcular la inversa en Python usamos el comando inv del paquete numpy.linalg (o podemos usar el comando solve del mismo paquete para resolver la ecuación matricial $\boldsymbol{A}\boldsymbol{X} = \boldsymbol{I}_n$).

```
# Probando...
A = np.array([[1,7],[2,3]])
print("A = \n", A)

A_inv = np.linalg.inv(A)
print("A^(-1) = \n", A_inv)

print("A * A^(-1) = \n", A @ A_inv)

print("La solución de AX = Id_2 es")
print(np.linalg.solve(A, np.eye(2)))
```

```
## A =
##  [[1 7]
##  [2 3]]
## A^(-1) =
##  [[-0.27272727  0.63636364]
##  [ 0.18181818 -0.09090909]]
## A * A^(-1) =
##  [[1. 0.]
##  [0. 1.]]
## La solución de AX = Id_2 es
##  [[-0.27272727  0.63636364]
##  [ 0.18181818 -0.09090909]]
```

Nota: En la práctica, es mejor resolver un sistema por eliminación gaussiana que invirtiendo la matriz.

4.5.5. Traza de una matriz. La traza de una matriz *cuadrada* es la suma de los elementos de la diagonal.

Si $\boldsymbol{A} = (a_{ij}) \in \mathbb{R}^{n\times n}$, $\operatorname{tr}(\boldsymbol{A}) = a_{11} + a_{22} + \cdots + a_{nn} = \sum_{i=1}^{n} a_{ii}$.

PROPOSICIÓN 1.12. Si $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n\times n}$:

- $\operatorname{tr}(\boldsymbol{A}^T) = \operatorname{tr}(\boldsymbol{A})$
- $\operatorname{tr}(\boldsymbol{A} + \boldsymbol{B}) = \operatorname{tr}(\boldsymbol{A}) + \operatorname{tr}(\boldsymbol{B})$.

Calculamos la traza en `Python` con el comando `trace` de `numpy`.

```
A = np.array([[1,5,1],[-6,7,8],[0,9,-2]])
print("A = \n", A)
print("Traza de A = ", tr(A)
```

```
%% A =
%%  [[ 1  5  1]
%%  [-6  7  8]
%%  [ 0  9 -2]]
%% Traza de A =  6
```

4.6. Determinante de una matriz. El determinante de una matriz *cuadrada* se puede definir recursivamente por la siguiente fórmula

$$\det(A) = \begin{cases} a_{11} & \text{si } n = 1 \\ \sum_{j=1}^{n} (-1)^{i+j} a_{ij} M_{ij} & \text{si } n > 1, \end{cases}$$

donde M_{ij} es la matrix de $(n-1)\times(n-1)$ que se obtiene al eliminar la fila i y la columna j de A.

EJEMPLO 1.20.

1. $\det \begin{pmatrix} a & b \\ c & d \end{pmatrix} = ad - bc$
2. $\det \begin{pmatrix} 1 & 2 & 3 \\ 3 & 0 & 7 \\ 0 & -1 & 4 \end{pmatrix} = \det \begin{pmatrix} 0 & 7 \\ -1 & 4 \end{pmatrix} - 3 \det \begin{pmatrix} 2 & 3 \\ -1 & 4 \end{pmatrix} = 7 - 3 \cdot 11 = -26$

PROPOSICIÓN 1.13. Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ y f_i, $1 \le i \le n$, son las filas de $\boldsymbol{A}$, las siguientes operaciones en la matriz modifican el determinante:

- transponer la matriz: $\det(\boldsymbol{A}^T) = \det(\boldsymbol{A})$

- multiplicar una fila por un escalar: $\det \begin{pmatrix} — & f_1 & — \\ & \vdots & \\ — & f_{i-1} & — \\ — & \alpha f_i & — \\ — & f_{i+1} & — \\ & \vdots & \\ — & f_n & — \end{pmatrix} = \alpha \det(A)$
- sumar un múltiplo de una fila a otra fila: $\det \begin{pmatrix} — & f_1 & — \\ & \vdots & \\ — & f_{i-1} & — \\ — & f_i + \beta f_j & — \\ — & f_{i+1} & — \\ & \vdots & \\ — & f_n & — \end{pmatrix} = \det(A)$
- intercambiar dos filas: $\det \begin{pmatrix} — & f_1 & — \\ & \vdots & \\ — & f_i & — \\ & \vdots & \\ — & f_j & — \\ & \vdots & \\ — & f_n & — \end{pmatrix} = -\det \begin{pmatrix} — & f_1 & — \\ & \vdots & \\ — & f_j & — \\ & \vdots & \\ — & f_i & — \\ & \vdots & \\ — & f_n & — \end{pmatrix}$
- multiplicar toda la matriz por un escalar: $\det(k\boldsymbol{A}) = k^n \det(\boldsymbol{A})$, para $k \in \mathbb{R}$.
- multiplicar toda la matriz por (-1): $\det(-\boldsymbol{A}) = (-1)^n \det(\boldsymbol{A})$.

En base a esta propiedad podemos demostrar fácilmente las siguientes propiedades.

COROLARIO 1.1.

- Si $\boldsymbol{A}$ tiene dos filas iguales, $\det(\boldsymbol{A}) = 0$.
- Si $\boldsymbol{A}$ tiene una fila de ceros, $\det(\boldsymbol{A}) = 0$.
- $\det(\boldsymbol{A}) = 0$ si y solo si $\boldsymbol{A}$ es singular.

PROPOSICIÓN 1.14. Si $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n \times n}$,

$$\det(\boldsymbol{AB}) = \det(\boldsymbol{A}) \det(\boldsymbol{B}).$$

Calculamos determinantes en `Python` con el comando `det` de `numpy.linalg`.

```
A = np.array([[1,5,0],[-1,-3,8],[0,1,-2]])
B = np.array([[0,4,2],[2,6,1],[1,5,-1]])
print("det(A) = ", np.linalg.det(A));
print("det(B) = ", np.linalg.det(B));
print("det(AB) = ", np.linalg.det(A @ B));

# No hay relación con los determinantes de A y B
```

```
print("det(A + B) = ", np.linalg.det(A + B));

M = np.array([[1,2,3],[4,5,6],[7,8,9]])
print("det(M) = ", np.linalg.det(M));
```

```
%% det(A) =  -12.0
%% det(B) =  19.999999999999996
%% det(AB) =  -240.0000000000002
%% det(A + B) =  51.0
%% det(M) =  -9.51619735392994e-16
```

EJERCICIO 1.5. Para $\boldsymbol{A} = \begin{pmatrix} 3 & 2 & -1 \\ 3 & 0 & 7 \\ 0 & -1 & 4 \end{pmatrix}$, probar que el sistema $\boldsymbol{Ax} = \boldsymbol{b}$ tiene solución única para todo $\boldsymbol{b} \in \mathbb{R}^3$

4.6.1. Determinante de una matriz triangular superior o inferior. Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es triangular inferior o superior su determinante es igual al producto de los elementos de la diagonal.

$$\det \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 0 & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{pmatrix} = \det \begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} = \prod_{i=1}^{n} a_{ii}$$

5. Cambio de base

Dado un espacio vectorial V de dimensión n llamamos base canónica de V al conjunto $\mathcal{E} = \{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n\}$ de vectores canónicos, $\boldsymbol{e}_i(j) = \begin{cases} 1 & \text{si } i = j \\ 0 & \text{si } i \neq j \end{cases}$, donde $\boldsymbol{e}_i(j)$ es la j-ésima coordenada del vector $\boldsymbol{e}_i$.

Dada otra base $\mathcal{B}$ de V, queremos saber cómo escribir en la base $\mathcal{B}$ a un vector $\boldsymbol{v}$ del cual conocemos sus coordenadas en la base $\mathcal{E}$. Y viceversa, cómo escribir en la base canónica a un vector del cual conocemos sus coordenadas en la base $\mathcal{B}$.

5.0.1. Cambio de base de $\mathcal{B}$ a $\mathcal{E}$. Analizamos un ejemplo. Consideramos la base de $\mathbb{R}^3$

$$\mathcal{B} = \{(1, 2, 5), (0, 1, 7), (0, 0, -1)\}$$

y el vector

$$\boldsymbol{v} = (3, -2, 1)_{\mathcal{B}}.$$

Para averiguar las coordenadas de $\boldsymbol{v}$ en la base canónica, hacemos la cuenta

$$\boldsymbol{v} = 3(1, 2, 5) + (-2)(0, 1, 7) + 1(0, 0, -1) = (3, 4, 0)$$

(en general omitimos el subíndice $\mathcal{E}$ cuando escribimos un vector en las coordenadas de la base canónica).

Podemos escribirlo matricialmente:

$$\boldsymbol{v} = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 5 & 7 & -1 \end{pmatrix} \begin{pmatrix} 3 \\ -2 \\ -1 \end{pmatrix} = \begin{pmatrix} 3 \\ 4 \\ 0 \end{pmatrix}.$$

Observamos que la matriz $\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 5 & 7 & -1 \end{pmatrix}$ posee los vectores de $\mathcal{B}$ como columnas y es la matriz que nos permite pasar un vector en coordenadas de una base $\mathcal{B}$ a la base canónica. Llamamos $\boldsymbol{C}_{\mathcal{BE}}$ a esta matriz. Obtenemos

$$\boldsymbol{v}_{\mathcal{E}} = \boldsymbol{C}_{\mathcal{BE}} \boldsymbol{v}_{\mathcal{B}}.$$

Ejercicio 1.6.

1. Dada la base $\mathcal{B} = \{(1,3,7), (4,0,1), (5,-7,0)\}$ de $\mathbb{R}^3$ y el vector $\boldsymbol{v}$ de $\mathbb{R}^3$ de coordenadas $(1,2,-5)$ en la base $\mathcal{B}$, hallar las coordenadas de $\boldsymbol{v}$ en la base canónica.
2. Dada la base $\mathcal{B} = \{1, (x-1), (x-1)^2\}$ de $\mathbb{R}[x]_2$ y el polinomio $p(x)$ de coordenadas $(7,-2,1)$ en la base $\mathcal{B}$, hallar las coordenadas de p en la base canónica de $\mathbb{R}[x]_2$. Observar que esto equivale a escribir como suma de potencias de x a un polinomio que viene dado como suma de potencias de $(x-1)$.

5.0.2. Cambio de base de $\mathcal{B}$ a $\mathcal{E}$. Ahora queremos escribir a $\boldsymbol{v} = (3,7,-1)_{\mathcal{E}}$ en la base $\mathcal{B}$. Para esto, necesitamos resolver el sistema de ecuaciones

$$(3,7,-1) = \sum_{i=1}^{3} a_i b_i = a_1(1,2,5) + a_2(0,1,7) + a_3(0,0,-1)$$

Esto nos da un sistema de 3 ecuaciones y 3 incógnitas, que podemos plantear en forma matricial:

$$\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 5 & 7 & -1 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} 3 \\ 7 \\ -1 \end{pmatrix}.$$

Observamos que la matriz $\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 5 & 7 & -1 \end{pmatrix}$ de coeficientes posee los vectores de $\mathcal{B}$ como columnas, es decir, es la matriz $\boldsymbol{C}_{\mathcal{BE}}$. Como $\mathcal{B}$ es una base, la matriz $\boldsymbol{C}_{\mathcal{BE}}$ es inversible (¿por qué?).

Por lo tanto,

$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 5 & 7 & -1 \end{pmatrix}^{-1} \begin{pmatrix} 3 \\ 7 \\ -1 \end{pmatrix} = (\boldsymbol{C}_{\mathcal{BE}})^{-1} \begin{pmatrix} 3 \\ 7 \\ -1 \end{pmatrix}.$$

Obtenemos que la matriz de cambio de base de la base canónica $\mathcal{E}$ a una base $\mathcal{B}$ es $\boldsymbol{C}_{\mathcal{EB}} = \boldsymbol{C}_{\mathcal{BE}}$.

En resumen: Tenemos una base $\mathcal{B} = \{b_1, \ldots, b_n\}$ de un espacio vectorial V de dimensión n.

1. Construimos la matriz $\boldsymbol{C}_{\mathcal{BE}} = \begin{pmatrix} | & | & & | \\ \boldsymbol{b}_1 & \boldsymbol{b}_2 & \cdots & \boldsymbol{b}_n \\ | & | & & | \end{pmatrix}$.
2. Para pasar un vector en base $\mathcal{B}$ a base canónica, usamos $\boldsymbol{v}_{\mathcal{E}} = \boldsymbol{C}_{\mathcal{BE}} \boldsymbol{v}_{\mathcal{B}}$.

3. Para pasar un vector en base base canónica a la base $\mathcal{B}$, definimos $\boldsymbol{C}_{\mathcal{E}\mathcal{B}} = (\boldsymbol{C}_{\mathcal{B}\mathcal{E}})^{-1}$ y usamos $\boldsymbol{v}_{\mathcal{B}} = \boldsymbol{C}_{\mathcal{E}\mathcal{B}}\boldsymbol{v}_{\mathcal{E}}$.

6. Transformaciones lineales

6.1. Matrices y transformaciones lineales. Una matriz $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ define una función $T_{\boldsymbol{A}} : \mathbb{R}^n \to \mathbb{R}^m$ dada por

$$T_{\boldsymbol{A}}(\boldsymbol{v}) = \boldsymbol{A}\boldsymbol{v}.$$

(tomando a $\boldsymbol{v}$ como matriz columna).

EJEMPLO 1.21. Si $\boldsymbol{A} = \begin{pmatrix} 1 & 3 & 0 \\ 2 & -1 & -4 \end{pmatrix} \in \mathbb{R}^{2\times 3}$ y $\boldsymbol{v} = (1, 0, -2)$, entonces

$$T_{\boldsymbol{A}}(\boldsymbol{v}) = \begin{pmatrix} 1 & 3 & 0 \\ 2 & -1 & -4 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ -2 \end{pmatrix} = \begin{pmatrix} 1 \\ 10 \end{pmatrix}$$

Si multiplicamos la matriz $\boldsymbol{A}$ por un vector genérico $\boldsymbol{v} = (x_1, x_2, x_3) \in \mathbb{R}^3$, obtenemos

$$T_{\boldsymbol{A}}(\boldsymbol{v}) = \begin{pmatrix} 1 & 3 & 0 \\ 2 & -1 & -4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1x_1 + 3x_2 + 0x_3 \\ 2x_1 - x_2 - 4x_3 \end{pmatrix}.$$

Entonces podemos escribir

$$T_{\boldsymbol{A}}(x_1, x_2, x_3) = (x_1 + 3x_2, 2x_1 - x_2 - 4x_3).$$

Ejercicio. Escribir a la función $f(x_1, x_2) = (3x_1, x_2 - x_1, x_2 + 2x_1)$ de $\mathbb{R}^2$ en $\mathbb{R}^3$ en forma matricial.

Estas funciones reciben el nombre de *transformaciones lineales*.

DEFINICIÓN 1.3. Dados dos espacios vectoriales V y W, una función $f : V \to W$ se llama **transformación lineal** si cumple:

1. $f(\boldsymbol{v} + \boldsymbol{v}') = f(\boldsymbol{v}) + f(\boldsymbol{v}')$ para todos $\boldsymbol{v}, \boldsymbol{v}' \in V$.
2. $f(a \cdot \boldsymbol{v}) = a \cdot f(\boldsymbol{v})$ para todos $a \in \mathbb{R}$ y $\boldsymbol{v} \in V$.

El producto de matrices verifica $\boldsymbol{A}(\boldsymbol{v} + \boldsymbol{w}) = \boldsymbol{A}\boldsymbol{v} + \boldsymbol{A}\boldsymbol{w}$ y $\boldsymbol{A}(a\boldsymbol{v}) = a(\boldsymbol{A}\boldsymbol{v})$, por lo tanto se cumplen los axiomas de transformación lineal.

EJEMPLO 1.22. La función $f(x_1, x_2) = (3x_1 + 1, x_2 - x_1, x_2 + 3x_1)$ **no** es una transformación lineal de $\mathbb{R}^2$ en $\mathbb{R}^3$. Por ejemplo, $f(2\cdot(1, 1)) = f(2, 2) = (7, 0, 8)$ y $2f(1, 1) = 2(4, 0, 4) = (8, 0, 8)$.

Observación. Las columnas de $\boldsymbol{A}$ son las imágenes de los vectores canónicos $\boldsymbol{e}_i$, $1 \le i \le n$.

PROPOSICIÓN 1.15. Si $\mathcal{B} = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ es una base de V, podemos definir una transformación lineal definiendo la imagen de cada elemento de la base.

Para un elemento $\boldsymbol{v} \in V$, si $\boldsymbol{v} = a_1\boldsymbol{v}_1 + \cdots + a_n\boldsymbol{v}_n$, entonces $f(\boldsymbol{v}) = a_1 f(\boldsymbol{v}_1) + \cdots + a_n f(\boldsymbol{v}_n)$.

EJEMPLO 1.23. $\mathcal{B} = \{(1, 2), (2, -1)\}$ es una base de $\mathbb{R}^2$. Si $f : \mathbb{R}^2 \to \mathbb{R}^2$ es una transformación lineal, $f(1, 2) = (0, 1)$ y $f(2, -1) = (1, 3)$, entonces

$$f(3, 1) = f((1, 2) + (2, -1)) = f(1, 2) + f(2, -1) = (0, 1) + (1, 3) = (1, 4).$$

6.2. Imagen y núcleo de matrices y transformaciones lineales. En base a esto, hacemos las siguientes definiciones

DEFINICIÓN 1.4. El subespacio de $\mathbb{K}^m$

$$\operatorname{im}(A) = \{\boldsymbol{b} \in \mathbb{K}^m : \boldsymbol{b} = \boldsymbol{A}\boldsymbol{x} \text{ para algún } \boldsymbol{x} \in \mathbb{K}^n\}$$

es la imagen de $\boldsymbol{A}$ (o rango de $\boldsymbol{A}$, pero trae confusión con el rango que ya definimos). La imagen de una matriz $\boldsymbol{A}$ está generada por las columnas de $\boldsymbol{A}$ (¿por qué?).

El subespacio de $\mathbb{K}^n$

$$\ker(\boldsymbol{A}) = \{\boldsymbol{x} \in \mathbb{K}^n : \boldsymbol{A}\boldsymbol{x} = 0\}$$

es el núcleo o kernel de $\boldsymbol{A}$. Podemos calcular el núcleo de $\boldsymbol{A}$ resolviendo el sistema homogéneo $\boldsymbol{A}\boldsymbol{x} = 0$.

Propiedad. $\operatorname{rango}(\boldsymbol{A}) = \dim(\operatorname{im}(\boldsymbol{A}))$.

TEOREMA 1.1 (Teorema de la dimensión.). Dada una matriz $\boldsymbol{A} \in \mathbb{K}^{m\times n}$,

$$n = \dim(\ker(\boldsymbol{A})) + \dim(\operatorname{im}(\boldsymbol{A}))$$

DEMOSTRACIÓN. Utilizamos la Proposición **??** de extensión de bases. Suponemos que el núcleo de $\boldsymbol{A}$ tiene dimensión t y consideramos una base $\{\boldsymbol{u}_1, \dots, \boldsymbol{u}_t\} \subset \mathbb{K}^n$ del núcleo de $\boldsymbol{A}$. Extendemos dicha base a una base de $\mathbb{R}^n$:

$$\mathcal{B} = \{\boldsymbol{u}_1, \dots, \boldsymbol{u}_t, \boldsymbol{w}_1, \dots, \boldsymbol{w}_{n-t}\}.$$

Veamos que

$$\{\boldsymbol{A}\boldsymbol{w}_1, \dots, \boldsymbol{A}\boldsymbol{w}_{n-t}\}$$

forman una base de $\operatorname{im}(A)$.

Dado un vector $v \in \mathbb{R}^n$, podemos escribirlo como combinación de elementos de la base:

$$v = a_1\boldsymbol{u}_1 + \cdots + a_t\boldsymbol{u}_t + b_1\boldsymbol{w}_1 + \cdots + b_{n-t}\boldsymbol{w}_{n-t}.$$

Como $\boldsymbol{A}\boldsymbol{u}_i = \boldsymbol{0}$ para $1 \le i \le t$, obtenemos que

$$\boldsymbol{A}\boldsymbol{v} = +b_1\boldsymbol{A}\boldsymbol{w}_1 + \cdots + b_{n-t}\boldsymbol{A}\boldsymbol{w}_{n-t},$$

y por lo tanto $\{\boldsymbol{A}\boldsymbol{w}_1, \dots, \boldsymbol{A}\boldsymbol{w}_{n-t}\}$ es un conjunto de generadores de $\operatorname{im}(A)$. Solo falta ver que es un conjunto linealmente independiente.

Supongamos $c_1\boldsymbol{A}\boldsymbol{w}_1 + \cdots + c_{n-t}\boldsymbol{A}\boldsymbol{w}_{n-t} = 0$. Luego

$$\boldsymbol{A}(c_1\boldsymbol{w}_1 + \cdots + c_{n-t}\boldsymbol{w}_{n-t}) = 0,$$

es decir $c_1\boldsymbol{w}_1 + \cdots + c_{n-t}\boldsymbol{w}_{n-t} \in \ker(A)$.

Por lo tanto, existen d_i, $1 \le i \le t$ tales que

$$c_1\boldsymbol{w}_1 + \cdots + c_{n-t}\boldsymbol{w}_{n-t} = d_1\boldsymbol{u}_1 + \cdots + d_t\boldsymbol{u}_t.$$

Pero los vectores de $\mathcal{B}$ forman una base de $\mathbb{R}^n$, por lo tanto debe ser $c_i = 0$ para todo $1 \le i \le n-t$, y concluimos que los vectores $\boldsymbol{A}\boldsymbol{w}_1, \dots, \boldsymbol{A}\boldsymbol{w}_{n-t}$ son linealmente independientes. □

EJEMPLO 1.24. Dada la matriz $\boldsymbol{A} = \begin{pmatrix} 1 & 0 & -2 & 1 \\ 0 & 2 & -1 & -2 \\ 2 & 1 & 1 & 1 \end{pmatrix}$,

1. los conjuntos $\mathrm{im}(\boldsymbol{A})$ y $\ker(\boldsymbol{A})$, ¿son subespacios de qué espacios vectoriales?
2. hallar generadores de $\mathrm{im}(\boldsymbol{A})$ y calcular $\dim(\mathrm{im}(\boldsymbol{A}))$.
3. hallar generadores de $\ker(\boldsymbol{A})$ y calcular $\dim(\ker(\boldsymbol{A}))$.
4. verificar si se cumple el teorema de la dimensión.

Respuestas

1. $\mathrm{im}(\boldsymbol{A}) \subset \mathbb{R}^3$ y $\ker(\boldsymbol{A}) \subset \mathbb{R}^4$.
2. $\mathrm{im}(\boldsymbol{A}) = \langle (1,0,2), (0,2,1), (-2,-1,1), (1,-2,1) \rangle$. Para calcular la dimensión, triangulamos.

```
A = np.array([[1, 0, -2, 1], [0, 2, -1, -2], [2, 1, 1, 1]])
At = np.transpose(A)
print(At)

At_echelon = row_echelon(At)
print("Matriz A transpuesta escalonada:\n", At_echelon)
```

```
%% [[ 1  0  2]
%%  [ 0  2  1]
%%  [-2 -1  1]
%%  [ 1 -2  1]]
%% Matriz A transpuesta escalonada:
%%  [[1.  0.  2. ]
%%  [0.  1.  0.5]
%%  [0.  0.  1. ]
%%  [0.  0.  0. ]]
```

   Obtenemos que $\dim(\mathrm{im}(\boldsymbol{A})) = 3$, porque quedan 3 filas no nulas.
3. Para resolver el sistema $\boldsymbol{Ax} = 0$, triangulamos $\boldsymbol{A}$:

```
A = np.array([[1, 0, -2, 1], [0, 2, -1, -2], [2, 1, 1, 1]])
A_echelon = row_echelon(A)
print("Matriz A escalonada:\n", A_echelon)
```

```
Matriz A escalonada:
 [[ 1.   0.  -2.   1. ]
 [ 0.   1.  -0.5 -1. ]
 [ 0.   0.   1.   0. ]]
```

   Obtenemos $x_3 = 0$, $x_2 - x_4 = 0$ y $x_1 + x_4 = 0$. Podemos despejar todas las variables x_1, x_2 y x_3 en función de x_4:

$$\begin{aligned} x_1 &= -x_4 \\ x_2 &= x_4 \\ x_3 &= 0 \end{aligned}$$

Luego, las soluciones del sistema son $\{(-x_4, x_4, 0, x_4) : x_4 \in \mathbb{R}\} = \{x_4(-1, 1, 0, 1) : x_4 \in \mathbb{R}\}$. Obtenemos

$$\ker(\boldsymbol{A}) = \langle(-1, 1, 0, 1)\rangle$$

y $\dim(\ker(A)) = 1$.

En `Python` podemos calcular una base del núcleo de una matriz con el comando `null_space` del paquete `scipy.linalg`. Para calcular una solución particular usamos eliminación gaussiana.

4. Tenemos $\dim(\ker(\boldsymbol{A})) + \dim(\mathrm{im}(\boldsymbol{A})) = 1 + 3 = 4$, se cumple el teorema.

7. Espacios afines

Dada $\boldsymbol{A} \in \mathbb{K}^{m\times n}$, ya vimos que las soluciones de un sistema homogéneo de ecuaciones

$$\boldsymbol{Ax} = 0$$

forman un subespacio de $\mathbb{K}^n$. Veamos qué pasa para un sistema no-homogéneo

$$\boldsymbol{Ax} = \boldsymbol{b}, \text{ con } \boldsymbol{b} \in \mathbb{K}^n.$$

Ejemplo 1.25. Resolvemos el siguiente sistema de ecuaciones:

$$\begin{cases} 5x_1 + 3x_2 = 11 \\ 15x_1 + 9x_2 = 33 \\ 20x_1 + 12x_2 = 44 \end{cases}$$

Construimos la matriz ampliada

$$\left(\begin{array}{cc|c} 5 & 3 & 11 \\ 15 & 9 & 33 \\ 20 & 12 & 44 \end{array}\right)$$

y escalonamos usando `Python`.

```
A = np.array([[5,3,11],[15,9,33],[20,12,44]])
print(row_echelon(A))
```

```
%% [[1.  0.6 2.2]
%%  [0.  0.  0. ]
%%  [0.  0.  0. ]]
```

Vemos que se eliminaron las últimas dos ecuaciones, y nos queda solo una ecuación:

$$x_1 + 0.6x_2 = 2.2$$

de donde podemos despejar $x_1 = 2.2 - 0.6x_2$.

Podemos entonces escribir el conjunto de soluciones en función de x_2:

$$\begin{aligned} S &= \{(2.2 - 0.6x_2, x_2) : x_2 \in \mathbb{R}\} \\ &= \{(2.2, 0) + (-0.6x_2, x_2) : x_2 \in \mathbb{R}\} \\ &= \{(2.2, 0) + (-0.6, 1)x_2 : x_2 \in \mathbb{R}\} \end{aligned}$$

Esto no es un subespacio de $\mathbb{R}^2$ (por ejemplo, $(0,0) \notin S$), pero es un subespacio "corrido": el conjunto

$$\{(-0.6, 1)x_2 : x_2 \in \mathbb{R}\} = \langle(-0.6, 1)\rangle,$$

es un subespacio de $\mathbb{R}^2$ y los puntos de S se obtienen sumandole el vector $(2.2, 0)$ a cualquier punto de $\langle(-0.6, 1)\rangle$, es decir,

$$S = (2.2, 0) + \langle(-0.6, 1)\rangle$$

Estos conjuntos se llaman **espacios lineales afines**, se obtienen trasladando un subespacio vectorial en la dirección de un vector del espacio. El espacio vectorial podría estar generado por varios vectores, es decir, puede tener cualquier dimensión.

Observamos:

- $(2.2, 0)$ es una solución del sistema no-homogéneo $\boldsymbol{Ax} = \boldsymbol{b}$.
- El subespacio $\langle(-0.6, 1)\rangle$ es el conjunto de soluciones del sistema homogéneo $\boldsymbol{Ax} = 0$.

y esto vale en general:

PROPOSICIÓN 1.16. Dado el sistema $\boldsymbol{Ax} = \boldsymbol{b}$, si $\tilde{\boldsymbol{x}}$ es una solución particular del sistema, el conjunto de todas las soluciones del sistema es

$$S = \{\tilde{\boldsymbol{x}} + \boldsymbol{v} : \boldsymbol{v} \text{ una solución cualquiera del sistema homogéneo } \boldsymbol{Ax} = 0\}.$$

Idea de la demostración: si $\boldsymbol{x}, \boldsymbol{y}$ son soluciones del sistema no homogéneo, entonces

$$\boldsymbol{Ax} - \boldsymbol{Ay} = \boldsymbol{b} - \boldsymbol{b} = 0,$$

Por lo tanto $\boldsymbol{z} = \boldsymbol{x} - \boldsymbol{y}$ es solución del sistema homogéneo y $\boldsymbol{x} = \boldsymbol{y} + \boldsymbol{z}$.

EJEMPLO 1.26. Resolver el sistema de ecuaciones

$$\begin{cases} 3x + 2y + 6z = 12 \\ x - 3y + z = 10. \end{cases}$$

En `Python` podemos calcular una base del núcleo de una matriz con el comando `null_space` del paquete `scipy.linalg`. Para calcular una solución particular usamos eliminación gaussiana.

```
import scipy.linalg
A = np.array([[3, 2, 6],[1, -2, 1]])
print("Núcleo de A: \n", scipy.linalg.null_space(A))
```

```
Núcleo de A:
 [[-0.85359507]
 [-0.18291323]
 [ 0.48776861]]
```

```
b = np.array([12, 10])
Ab = np.c_[A,b]
print("Matriz ampliada escalonada: \n", row_echelon(Ab))
```

```
%% Matriz ampliada escalonada:
%%  [[ 1.          0.66666667  2.          4.        ]
%%  [ 0.          1.          0.375      -2.25      ]]
```

Obtenemos las ecuaciones

$$\begin{cases} x + 2/3y + 2z = 4 \\ \quad y + 0.375z = -2.25, \end{cases}$$

y tomando $z = 1$ obtenemos la solución $(x, y, z) = (3.75, -2.625, 1)$.

Concluimos que el conjunto de soluciones es

$$S = (3.75, -2.625, 1) + \langle(-0.8535951, -0.1829132, 0.4877686)\rangle.$$

Capítulo 2

Números de Máquina y Número de Condición

1. Cuestiones previas

En el *álgebra lineal computacional* (o en el análisis numérico en general) interesan no solo los algoritmos sino su *confiabilidad* y su *costo computacional*. La primera cuestión aparece porque en la práctica no es posible en general trabajar con precisión infinita (lo que hace imprescindible el estudio de los errores numéricos y su propagación). La segunda cuestión tiene que ver con el número de operaciones involucradas en el algoritmo (*complejidad del algoritmo*) cuestión que impacta en el tiempo de ejecución[1].

Comenzamos entonces definiendo de manera informal los números de máquina.

Los números de máquina, son los que la máquina puede representar de forma exacta.

Notemos que -por su propia definición- podemos sospechar que los números de máquina son finitos y en consecuencia la *mayoría* de los números reales no podrán almacenarse sin cometer errores. Vamos a tratar de precisar este punto en lo que sigue. Vamos a asumir que trabajaremos con número reales o complejos, sin embargo como para almacenar un número complejo basta almacenar su parte real e imaginaria podemos enfocarnos en el caso real.

Consideremos un número cualquiera, por ejemplo $x = 125.4$. Estamos habituados a trabajar en base 10 lo que significa que estamos pensando a x en la forma

$$x = 1\,10^2 + 2\,10^1 + 5\,10^0 + 4\,10^{-1}.$$

Para número real (positivo) diferente, la escritura

$$\tilde{b}_k\tilde{b}_{k-1}...\tilde{b}_0.\tilde{b}_{-1}\tilde{b}_{-2}...$$

con $0 \leq \tilde{b}_i \leq 9$ y $\tilde{b}_k \neq 0$, la asociamos con

$$x = \sum_{-\infty}^{k} \tilde{b}_i 10^i.$$

La escritura hecha de este modo es única, salvo desarrollos periódicos del número 9 como ocurre en el caso

$$0.999... = 1,$$

en el que el mismo número, en este caso $x = 1$, puede escribirse de dos modos distintos. En este contexto, el número 10 se denomina la base y los $\tilde{b}_i$ los dígitos del desarrollo.

[1] Otras consideraciones son posibles: por ejemplo si puede o no paralelizarse.

2. Bases generales*

En general todo número $x \in \mathbb{R}$ $x \neq 0$, puede representarse en una base $b \in \mathbb{N}$, $b \geq 2$, de la forma[2]

$$(x)_b = sg(x)\tilde{b}_k\tilde{b}_{k-1}...\tilde{b}_0.\tilde{b}_{-1}\tilde{b}_{-2}... \tag{2.1}$$

donde los "dígitos" verifican $0 \leq \tilde{b}_i \leq b-1$, $sg(x)$ el signo de x y donde por convención numeramos los índices para que el punto -o coma- se ubique entre $\tilde{b}_0$ y $\tilde{b}_{-1}$.

Asumamos que x es positivo para no lidiar con el signo. La representación (**??**) se obtiene de la escritura de x como

$$x = \tilde{b}_k b^k + \tilde{b}_{k-1}b^{k-1} + ... + \tilde{b}_0 b^0 + \tilde{b}_{-1}b^{-1} + \tilde{b}_{-2}b^{-2} + ...$$

que es única si descartamos los desarrollos infinitos de la forma $\tilde{b}_j = b-1$ para todo $j \leq l$ con $l \in \mathbb{Z}$ fijo.

Para aclarar este punto recordemos la expresión cerrada de la geométrica

$$\sum_{j=0}^{m} \beta^j = \frac{\beta^{m+1}-1}{\beta-1},$$

válida para $\beta \neq 1$. Notemos que en caso de tener un x con un desarrollo de la forma

$$x = \sum_{i=-\infty}^{l} (b-1)b^i,$$

podemos reescribirlo como

$$x = (b-1)b^l\left(\sum_{i=-\infty}^{0} b^i\right) = (b-1)b^l\frac{b}{b-1} = b^{l+1},$$

lo que indica dos representaciones alternativas para el mismo x. Como ejemplo si $l=0$ tenemos

$$10 = (x)_b = (b-1).(b-1)(b-1)(b-1)\cdots$$

Continuemos asumiendo que $x > 0$. El uso del punto *fijo* en (**??**) determina la división entre su *parte entera*[3] $[x] \in \mathbb{Z}$ y la diferencia $0 \leq x - [x] < 1$. En efecto, por un lado, para todo $m \in \mathbb{N}$

$$\sum_{0}^{m} \tilde{b}_i b^i \in \mathbb{Z}$$

y

$$0 \leq \sum_{-\infty}^{-1} \tilde{b}_i b^i < \sum_{-\infty}^{-1} \tilde{b}_i(b-1) = (b-1)\frac{1}{b-1} = 1.$$

Por otro lado, esta representación tradicional no da una idea rápida del orden de magnitud de un número. Por esta razón se suele trabajar con una versión *normalizada* eligiendo ubicar el punto justo a la izquierda de $\tilde{b}_k$ (i.e. el primer dígito no nulo de x). Así, un $0 \neq x \in \mathbb{R}$, puede escribirse (renombramos los dígitos eliminando el tilde y los índices)

[2] $(x)_b$ se lee x en la base b.

[3] La parte entera de x, denotada $[x]$ se define como el mayor entero menor o igual a x.

$$(x)_b = sg(x)\, 0.b_0b_1b_2...\, b^e \tag{2.2}$$

donde, como antes, $0 \leq b_i \leq b-1$ y $e \in \mathbb{Z}$. Una vez más, esta representación es única si asumimos que $b_0 \neq 0$, y que en el caso $b_i = b-1$ para $i \leq l$, $b_{i+1} < b-1$ convenimos en tomar

$$(x)_b = sg(x)\, 0.b_0b_1...(b_{i-1}+1)\, b^{e+1}.$$

Suele llamarse notación científica normalizada a la escritura (**??**). La expresión $0.b_0b_1b_2...$ se llama *mantisa* y e *exponente*. Con esta normalización se ve a simple vista que $b^e < x \leq b^{e+1}$.

Observemos que el exponente es un entero que admite asimismo una representación en la misma base b

$$e = sg(e)\, c_lc_{l-1}...c_0$$

donde una vez mas $c_l \neq 0$ y $e = sg(e)\, \sum_0^l c_ib^i$.

3. Números de Máquina

En una máquina, la memoria dedicada para el almacenamiento de los números es limitada dedicando cierta cantidad de dígitos, digamos $m \geq 1$, para la mantisa y cierta cantidad, digamos $E \geq 1$, para el exponente. Los números que pueden representarse de forma exacta con esas limitaciones se denominan *números de máquina*.

La cantidad de números de máquina es obviamente finita. De hecho podemos calcular fácilmente su cantidad: considerando que $b_0 \neq 0$ tenemos $b-1$ posibilidades para la elección del b_0, y b posibilidades para cada uno de los restantes b_i, $1 \leq i \leq m-1$. La cantidad diferente de mantisas es entonces $(b-1)b^{m-1}$ (para el número cero se utiliza un símbolo especial). Análogamente hay b^E distintos exponentes. Considerando los respectivos signos de la mantisa y el exponente, habrá $4(b-1)b^{m+E-1}$ diferentes números de máquina.

El mayor exponente para una máquina dada es el número

$$E_{max} = \sum_{j=0}^{E-1}(b-1)b^j = \frac{b^E-1}{b-1}(b-1) = b^E - 1$$

y la mayor mantisa

$$M_{max} = \sum_{j=1}^{m}(b-1)b^{-j} = \frac{(b-1)}{b}\sum_{j=0}^{m-1} b^{-j} = \frac{(b-1)}{b}\frac{(b^{-m}-1)b}{1-b} = 1 - b^{-m},$$

lo que da

$$M_{max}^{E_{max}} = \left(1 - b^{-m}\right) b^{b^E-1},$$

como mayor número de máquina (análogamente se obtiene el menor como $-M_{max}^{E_{max}}$).

Cualquier número mayor a $M_{max}^{E_{max}}$ producirá un desborde *overflow*. El *menor* número *positivo* de máquina es obviamente $b^{-E_{max}-1}$. Todo número x, $0 < x < b^{-E_{max}-1}$ genera un desborde por debajo *underflow*.

FIGURA 1. Algunos números positivos de máquina con $b = 2$ y $m = 4$. Notar que los números son equiespaciados únicamente entre potencias sucesivas de 2, y que la cantidad de números de máquina en esos rangos se mantiene constante.

Escalas de los números de máquina: En las máquinas que utilizamos habitualmente $b = 2$ y $m = 52$ y $E = 10$. Esto es, se trabaja en base 2 y se utilizan 64 bits para almacenar un número (esta formato se denomina *doble precisión*[a]). El término *bit* significa dígito binario (admite solo dos caracteres: 0 y 1). De los 64 bits disponibles, dos se reservan para los signos de la mantisa y el exponente.
En este caso, considerando que $2^{10} = 1024 \sim 10^3$:

$$M_{max}^{E_{max}} = (1 - 2^{-52})2^{2^{10}-1} \sim 2^{1000} \sim (2^{10})^{100} \sim 10^{300}.$$

Esta máquina puede trabajar con números del orden de 10^{300} sin overflow. En la escala pequeña puede trabajar (haciendo una cuenta similar para $b^{-E_{max}-1}$) con números del orden 10^{-300}. Estas cantidades son razonables para describir las mayoría de las magnitudes que aparecen en las disciplinas científicas. Notemos que la mayor y menor escala se deciden básicamente en términos del tamaño del exponente y *no del tamaño de la mantisa*. Aumentar los bits de la mantisa no mejora las escalas (i.e. no aumenta el rango de valores extremales de nuestra máquina).

[a]En *precisión simple* se usan 32 bits, con $E = 7$ y $m = 23$.

Si un número $x \in \mathbb{R}$ no es de máquina, se puede elegir el número de máquina mas cercano para representarlo (*redondeo*) o, por ejemplo, el inmediato inferior (*truncado*). Dado un número real x denominamos $fl(x)$ al número de máquina elegido para representarlo. Se utiliza fl para indicar que estamos trabajando en *punto flotante*. El hecho de "mover" el punto de acuerdo a (**??**) acarrea una consecuencia muy importante a la hora de truncar o redondear el número y es la de mantener uniforme el *error relativo* a la hora de almacenar x a lo largo de todas la escalas. Esta idea es central y no podría lograrse eligiendo números de máquina equiespaciados.

Errores Relativos y Absolutos En una magnitud escalar x_v, el error absoluto se define como

$$e_{abs} = |x_a - x_v|,$$

donde x_a es el valor aproximado de x_v. En general (y para $x_v \neq 0$) estaremos interesados en el error relativo

$$e_{rel} = \frac{|x_a - x_v|}{|x_v|},$$

es decir en el tamaño del error absoluto respecto del tamaño del valor exacto x_v. Para magnitudes vectoriales o matriciales reemplazamos el módulo por una norma.

Distribución de los números de máquina: Los números de máquina no se distribuyen de manera uniforme. Describamos dos números de máquina consecutivos $x_1 < x_2$ asumiendo, para simplificar, que $0 < x_1$ y que se escribe:

$$(x_1)_b = 0.b_0b_1b_2...b_{m-1}\, b^e.$$

Un momento de reflexión nos dice que el próximo número de máquina (salvo que haya overflow) será

$$(x_2)_b = 0.b_0b_1b_2...(b_{m-1}+1)\, b^e$$

salvo que $b_{m-1} = b-1$ en cuyo caso

$$(x_2)_b = 0.b_0b_1b_2...(b_{m-2}+1)0\, b^e$$

salvo que también $b_{m-2} = b-1$ en cuyo caso habrá que repetir el argumento y eventualmente -en caso de que $b_i = b-1$ para todo $0 \le i \le m-1$

$$(x_2)_b = 0.1b^{e+1}.$$

En cualquier caso observamos que

$$x_2 - x_1 = b^{-m}b^e$$

no es constante al variar el exponente e, pero sí lo es para cada exponente fijo dado.

La consecuencia mas notable de la distribución de los números de máquina es que si $x \in \mathbb{R}$, y existen $x_1 < x_2$ números de máquina tales que $x_1 \le x \le x_2$ (i.e. x esta dentro de las escalas manejadas por la máquina) entonces

$$|x - fl_{redond}(x)| \le \frac{1}{2}b^{e-m}$$

$$|x - fl_{trunc}(x)| \le b^{e-m}$$

para los casos de redondeo y truncado respectivamente.

Error relativo y precisión de la máquina: Nos concentraremos en el caso de redondeo, por lo que asumiremos de aquí en mas que $fl = fl_{redond}$. Queremos acotar el error relativo al almacenar un número $x \neq 0$. Observando que

$$b^{e-1} = |0.1\, b^e| \le |x|,$$

resulta

$$|\frac{x - fl(x)}{x}| \le \frac{1}{2}b^{1-m}.$$

Notar que el error relativo depende *del tamaño de la mantisa*. En el caso de base $b = 2$ y $m = 52$ mencionado antes resulta:

$$|\frac{x - fl(x)}{x}| \le 2^{-52} \sim 10^{-16}.$$

este número, propio de cada máquina, se denomina: precisión de la máquina o épsilon de la máquina y se denota con ε.
En el caso del ejemplo, en terminos simples, dice que nuestra máquina almacena 16 dígitos exactos (en base 10). Naturalemente sería ideal conservar esta precisión a lo largo de las operaciones que debamos realizar a partir de $fl(x)$. Eso, como veremos a continuación, no es posible en general.

Como acabamos de ver, en general ocurre que $x \neq fl(x)$[4]. . Sin embargo podemos garantizar que la diferencia verifica

$$x - fl(x) = \mu_x x,$$

con

$$\frac{|x - fl(x)|}{|x|} = |\mu_x| \leq \varepsilon.$$

Veamos entonces que ocurre con estos errores al efectuar alguna operación sencilla como por ejemplo sumar. Vamos a distinguir la operación realizada por la máquina con el símbolo $\oplus$.

Pretendemos calcular $x + y$, de modo simplificado la máquina realiza las siguientes operaciones: primero

$$x \to fl(x) \qquad y \to fl(y)$$

luego suma $fl(x) + fl(y)$ y finalmente almacena el resultado

$$fl(x) + fl(y) \to fl(fl(x) + fl(y)).$$

Cómo se comportará el error resultante?. Por una lado tenemos:

$$fl(x) = x(1 + \mu_x), \qquad fl(y) = y(1 + \mu_y y),$$

por lo que

$$fl(fl(x) + fl(y)) = (fl(x) + fl(y))(1 + \mu_z)$$

y en definitiva

$$x \oplus y = fl(fl(x) + fl(y)) = (x(1 + \mu_x) + y(1 + \mu_y y))(1 + \mu_z).$$

Vemos entonces que *si* $0 < x, y$ es posible acotar

$$|x \oplus y - (x + y)| \leq (x + y)2\mu + O(\varepsilon^2)$$

con $|\mu| \leq \varepsilon$. Es decir que hemos de algún modo preservado el tamaño del error relativo[5] Como es bien sabido, eso no es posible si *restamos* números similares[6]. Un ejemplo elemental es el siguiente: tomemos $b = 10$, $m = 4$, la precisión es $\varepsilon = \frac{1}{2}10^{-3} = 0.0005$. Si $x = 125,49$ e $y = 125,31$ tenemos respectivamente $x - y = 0.18$ $x \ominus y = 0.2$ por lo que el error relativo es

$$|\frac{x - y - x \ominus y}{x - y}| = \frac{0.02}{0.18} \sim 0.11,$$

es decir que a pesar de que $\varepsilon \sim 10^{-3}$ el error en la cuenta anterior es del 11 %. El ejemplo anterior muestra que en una simple operación podemos perder dígitos significativos (de hecho nuestra máquina no ha acertado ningún dígito de la solución). Como regla general deben evitarse restas de números similares[7] para evitar la denominada *cancelación catastrófica*.

Es fácil construir ejemplos de números de máquina $0 < x < y$ en los cuales no solo $x+y \neq x \oplus y$ sino que, por ejemplo, $x \oplus y = y$. En particular es fácil ver que

$$1 \oplus \varepsilon > 1, \tag{2.3}$$

y

$$1 \oplus \varepsilon/2 = 1$$

[4]Notemos que incluso números muy sencillos como $\frac{1}{10}$ no son de máquina y la introducción de un error de redondeo es inevitable. Evalúe en Python la expresión $0.1 + 0.1 + 0.1! = 0.3$, qué obtiene?.

[5]Cuando los errores se acumulan de forma aditiva, como en este caso, estamos en un buen escenario porque harían falta una enorme cantidad de operaciones para deteriorar el error inicial.

[6]Verifique que la cuenta anterior no puede repetirse si los signos de x e y difieren.

[7]Ver la guía de ejercicios para mas ejemplos

lo que da una posible definición alternativa del ϵ como el menor número de máquina con la propiedad (**??**). Otras cuestión que aparece entonces en la aritmética de la máquina es la pérdida de la propiedad asociativa, ya que usando los comentarios previos vemos que por ejemplo

$$(1 \oplus \varepsilon/2) \oplus \varepsilon/2 = 1 \neq 1 \oplus (\varepsilon/2 + \varepsilon/2).$$

Hay diversas fuentes de errores computacionales que pueden estropear completamente el resultado de los algoritmos. En este sentido hay dos conceptos que nombraremos tangencialmente ya que no son objeto central de este curso: la *condición* y la *estabilidad*.

4. Condición y Estabilidad

De manera muy general, un *problema bien condicionado* es aquél que reacciona benignamente con los errores en los datos. Es decir, que sus soluciones no varían demasiado al no variar demasiado los datos. Por el contrario, si un problema está muy mal condicionado, no lograremos resolverlo con mucha precisión aunque limitemos los errores en los datos (salvo que trabajemos con precisión infinita). La noción de condición es algo *intrínseco del problema* y está mas allá de nuestro algoritmo de resolución. Por otra parte, cuando los problemas están bien condicionados tenemos esperanza de resolverlos con precisión siempre que nuestro algoritmo no incremente desproporcionadamente los errores inherentes a los datos. En este caso hablaramos de *algoritmos estables* y por el contrario, de *algoritmos inestables* si no cumplen con este criterio. La estabilidad entonces es algo *intrínseco del algoritmo*.

Es posible dar una expresión precisa para la noción de condición, a través del llamado *número de condición*. Consideremos el problema de evaluar una función en el valor $x_0 \neq 0$. Si por alguna razón (error de medición o redondeo) modificamos el dato a evaluar x_0 en un una cierta magnitud pequeña h entonces el error relativo que cometeremos es (asumiendo que la función es de continuamente diferenciable)

$$\frac{f(x_0+h) - f(x_0)}{f(x_0)} = \frac{hf'(\eta)}{f(x_0)},$$

para cierto η intermedio entre x_0 y $x_0 + h$. Esto indica que para $h \sim 0$

$$\frac{hf'(\eta)}{f(x_0)} \sim \frac{x_0 f'(x_0)}{f(x_0)} \frac{h}{x_0}$$

el error relativo en los datos $\frac{h}{x_0}$ se magnifica en nuestro problema en un factor

$$\frac{|x_0||f'(x_0)|}{|f(x_0)|},$$

llamado el *número de condición* de evaluar f en x_0. Siguiendo un razonamiento similar vemos que para la versión $f : \mathbb{R}^n \to \mathbb{R}^m$ puede definirse el número de condición de este problema de la forma[a]

[a]Un ejemplo concreto y elemental de mal condicionamiento es, como hemos visto, la resta de números similares: si escribimos $f : \mathbb{R}^2 \to \mathbb{R}$, $f(x, y) = x - y$, se tiene $Df(x_0, y_0) = (1, -1)$, $\|(1, -1)\|_\infty = 2$, $\|(x_0, y_0)\|_\infty = max\{|x_0|, |y_0|\}$, luego $\frac{\|x_0\|\|Df(x_0)\|}{\|f(x_0)\|} \sim \frac{1}{\|x_0 - y_0\|}$.

Como ejemplo de lo anterior si queremos evaluar $tg(x)$ en un $x_0 < \pi/2$, $x_0 \sim \pi/2$ vemos que el número de condición

$$\frac{x_0}{sen(x_0)cos(x_0)},$$

cumple que

$$\lim_{x_0 \to \pi/2^-} \frac{x_0}{sen(x_0)cos(x_0)} = +\infty.$$

En particular si elegimos x_0 tal que $\frac{x_0}{sen(x_0)cos(x_0)} \sim 10^{16}$ no esperamos tener ningún dígito significativo en nuestro cálculo de $tg(x_0)$.

En los años 50, Wilkinson experimentaba con las primeras computadoras y rápidamente comenzó a observar fenomenos de inestabilidades y mal condicionamiento[8]. Una cosa que notó, al probar un algoritmo de aproximación de raices, es que si al polinomio

$$p(x) = \prod_{i=1}^{20} (x - i)$$

se le perturba el coeficiente que acompaña a x^{19} (cuyo valor es 210) en 2^{-23} las raíces mayores a 7 sufren considerables modificaciones. En particular las raices $10, 11, \ldots 19$ se transforman en 5 pares complejos conjugados. Las 18 y 19 en complejos de la forma $19.5 \cdots \pm 19.5 \ldots i$, es decir que una perturbación de orden 2^{-23} genera variaciones de orden 1. [9] Mostrando que el problema de calcular raices está muy mal condicionado.

Ejemplos de algoritmos inestables abundan. La inestabilidad es muy frecuente y se hace notar muy rápidamente porque los resultados obtenidos difieren notoriamente de lo esperado. Los casos mas extremos aparecen en procesos que deben iterarse en los cuales el error se acumula exponencialmente en vez de aditivamente. Sin embargo hay casos muy simples que pueden ejemplificarse.

Mas arriba vimos que restar números similares es un problema mal condicionado y por eso debe evitarse o tratarse con cautela.

Supongamos que queremos evaluar e^{-12}. Usando lo explicado mas arriba, podemos ver que se trata de un problema bien condicionado. Si utilizamos un algoritmo basado en la serie convergente

$$e^{-x} = 1 - x + \frac{1}{2!}x^2 - \frac{1}{3!}x^3 \cdots$$

para una evaluación directa en $x = -12$ obtenemos, sumando los primeros 50 términos de la serie:

$$e^{-12} \sim 6.144189436702122 \cdot 10^{-6}.$$

Si por otro lado modificamos el algoritmo calculado primero e^{12} sumando los primeros 50 terminos de la serie

$$e^{x} = 1 + x + \frac{1}{2!}x^2 + \frac{1}{3!}x^3 \cdots$$

y luego invirtiendo $1/e^{12}$ obtenemos

$$e^{-12} \sim 6.144212353328213 \cdot 10^{-6}.$$

[8] En sus propias palabras: "The cosy relationship that mathematics enjoyed with polynomials suffered a severe setback in the early fifities where electronic computers came into general use. Speaking for myself I regard it as the most traumatic experience in my carrer as a numerical analyst" [?]

[9] A pesar de esto, se puede probar que las raíces dependen de forma continua respecto de los coeficientes.

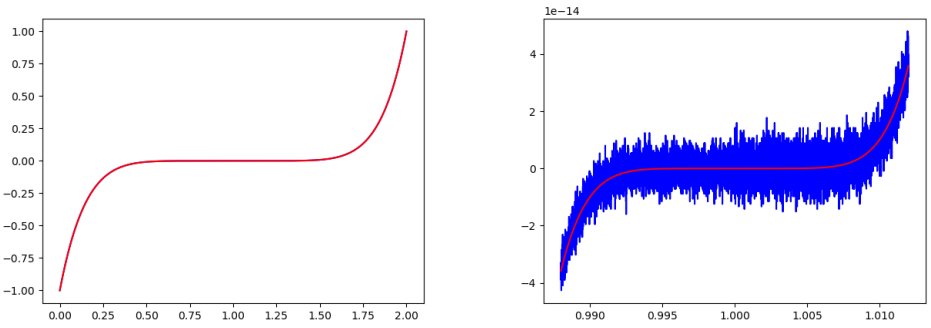


FIGURA 2. Gráficos de $p(x)$

La pregunta es a priori, cuál de las dos respuestas es mas confiable. Sin duda el segundo método es mas estable. La razón es que en el cómputo de la serie alternada, muchos terminos "grandes" deben cancelarse mágicamente para producir el resultado "pequeño" e^{-12}. Los errores relativos pequeños son proporcionalmente grandes en términos del resultado final. De hecho

$$e^{-12} \sim 6.144212353328209758682308178805532311223989314888252975 5... \cdot 10^{-6}$$

es decir que solo obtuvimos 4 dígitos correctos con nuestro primero algoritmo pero 14 con el segundo. El segundo método es mucho mas estable que el primero. Puede probar los calculos previos con el algoritmo:

```
import numpy as np
import math
v=np.arange(0,50)
resulI=0.0
resulE=0.0
for i in v:
        resulI=resulI+1/math.factorial(i)*(-12.0)**i
        resulE=resulE+1/math.factorial(i)*(12.0)**i
resulE=1/resulE
print(resulInes)
print(resulEs)
```

Los problemas que emergen debido a la precisión limitada de las máquinas dan lugar a muchos comportamientos inesperados. El ejemplo que damos a continuación, debido a Cleve Moler, es particularmente sencillo y sigue las consideraciones del ejemplo previo. Queremos evaluar $p(x) = (x-1)^7$ para posteriormente graficarlo. En el primer algoritmo usamos la expresión cerrada $(x-1)^7$ y en el segundo método la expresión equivalente

$$p(x) = x^7 - 7x^6 + 21x^5 - 35x^4 + 35x^3 - 21x^2 + 7x - 1.$$

En la Figura **??** a la izquierda vemos ambos gráficos perfectamente superpuestos (azul para el primer método tapado por el rojo del segundo método). Sin embargo en la misma figura a la derecha se muestra un zoom cerca de la raíz del polinomio.

```
import numpy as np
import matplotlib.pyplot as plt
x=np.linspace(0,2,200)
plt.plot(x,x**7-7*x**6+21*x**5-35*x**4+35*x**3-21*x**2+7*x-1,'b', x,(
   x-1)**7,'r')
plt.show()

# Sin embargo a la derecha vemos el resultado de samplear en una
#escala mas fina alrededor de la raíz

x=np.linspace(0.988,1.012,10000)
plt.plot(x,x**7-7*x**6+21*x**5-35*x**4+35*x**3-21*x**2+7*x-1,'b', x,(
   x-1)**7,'r')
plt.show()
```

Las nociones de condición y estabilidad que hemos comentado de modo superficial son temas transversales a toda el área del análisis numérico. Como nos restringiremos a cuestiones de álgebra lineal solo hemos pretendido dar una idea somera de sus implicaciones generales. En el capítulo siguiente retomaremos la cuestión de la condición en el ámbito matricial.

Capítulo 3

Normas y Producto Interno

1. Normas en $\mathbb{K}^n$

La distancia del punto $(x, y) \in \mathbb{R}^2$ al origen puede calcularse por el teorema de Pitágoras

$$z = \sqrt{x^2 + y^2}.$$

Generalizando esa fórmula a espacios de cualquier dimensión, definimos una norma vectorial:

$$\|\boldsymbol{v}\|_2 = \sqrt{v_1^2 + \cdots + v_n^2},$$

Esta norma suele llamarse norma-2 o norma euclídea. Esta norma, a su vez, puede escribirse equivalentemente como

$$\|\boldsymbol{v}\|_2 = \sqrt{|v_1|^2 + \cdots + |v_n|^2},$$

lo cual nos da una expresión aplicable a los complejos.

Definición 3.1. Una norma de un K-espacio vectorial es una función $\|\cdot\| : V \to \mathbb{R}_{\geq 0}$ que cumple las siguientes propiedades:

1. $\|a\boldsymbol{v}\| = |a|\|\boldsymbol{v}\|$, para $a \in \mathbb{K}$ y $\boldsymbol{v} \in V$.
2. Si $\|\boldsymbol{v}\| = 0$, entonces $\boldsymbol{v} = 0$.
3. $\|\boldsymbol{u} + \boldsymbol{v}\| \leq \|\boldsymbol{u}\| + \|\boldsymbol{v}\|$, para todo $\boldsymbol{u}, \boldsymbol{v} \in V$ (desigualdad triangular)

Es fácil ver que la norma-2 cumple las primeras dos propiedades. La tercera propiedad puede probarse usando la siguiente desigualdad clásica.

Proposición 3.1 (Desigualdad de Cauchy-Schwarz). Dados $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{K}^n$,

$$\left|\sum_{i=1}^{n} \overline{u}_i v_i\right| \leq \|\boldsymbol{u}\|_2 \|\boldsymbol{v}\|_2.$$

Demostración. Asumamos primero que $\mathbb{K} = \mathbb{R}$. Consideremos el siguiente polinomio de grado 2 en la variable x

$$p(x) = (u_1 x + v_1)^2 + \cdots + (u_n x + v_n)^2 = \left(\sum_{i=1}^{n} u_i^2\right) x^2 + 2\left(\sum_{i=1}^{n} u_i v_i\right) x + \sum_{i=1}^{n} v_i^2.$$

Como p es una suma de cuadrados, $p(x) \geq 0$ para todo $x \in \mathbb{R}$ tiene o bien un raiz real doble o raíces complejas. Escribiendo $p(x) = ax^2 + bx + c$ vemos que debe ser entonces $b^2 - 4ac \leq 0$. Es decir,

$$4\left(\sum_i u_i v_i\right)^2 - 4\left(\sum_i u_i^2\right)\left(\sum_i v_i^2\right) \leq 0,$$

y eliminando el factor 4 y despejando, obtenemos la desigualdad buscada.

Si $\mathbb{K} = \mathbb{C}$ notamos, por la desigualdad triangular en los complejos[1]

$$\left|\sum_i u_i\overline{v_i}\right| \leq \sum_i |u_i||\overline{v_i}| = \sum_i |u_i||v_i|,$$

y la desigualdad sale ahora usando el caso real con los vectores $(|u_1|, \cdots, |u_n|)$,$(|v_1|, \cdots, |v_n|)$. □

Observación 3.1. Notar que la desigualdad de Cauchy-Schwarz vale también tomando a la izquierda $\sum_i \overline{u_i}v_i$, puesto que $\sum_i \overline{u_i}v_i = \overline{\sum_i u_i\overline{v_i}}$ y el módulo de un complejo es igual al de su conjugado.

Corolario 3.1. Para todo $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{K}^n$ vale, $\|\boldsymbol{u} + \boldsymbol{v}\|_2 \leq \|\boldsymbol{u}\|_2 + \|\boldsymbol{v}\|_2$.

Demostración. La hacemos en $\mathbb{C}$ y en $\mathbb{R}$ sale como caso particular.

$$\|\boldsymbol{u} + \boldsymbol{v}\|_2^2 = \sum_{i=1}^n |u_i + v_i|^2 = \sum_{i=1}^n (u_i + v_i)\overline{(u_i + v_i)}$$

$$= \sum_{i=1}^n u_i\overline{u_i} + \sum_{i=1}^n (u_i\overline{v_i} + v_i\overline{u_i}) + \sum_{i=1}^n v_i\overline{v_i} = \|\boldsymbol{u}\|_2^2 + \sum_{i=1}^n (u_i\overline{v_i} + v_i\overline{u_i}) + \|\boldsymbol{v}\|_2^2,$$

los términos entre paréntesis son reales (un número complejo -eventualmente complejo- mas su conjugado), luego

$$\sum_{i=1}^n (u_i\overline{v_i} + v_i\overline{u_i}) \leq \left|\sum_{i=1}^n (u_i\overline{v_i} + v_i\overline{u_i})\right| \leq \left|\sum_{i=1}^n u_i\overline{v_i}\right| + \left|\sum_{i=1}^n v_i\overline{u_i}\right| \leq 2\|\boldsymbol{u}\|_2\|\boldsymbol{v}\|_2,$$

donde hemos usado Cauchy-Schwarz en la última desigualdad. Luego, se tiene

$$\|\boldsymbol{u} + \boldsymbol{v}\|_2^2 \leq \|\boldsymbol{u}\|^2 + \|\boldsymbol{v}\|^2 + 2\|\boldsymbol{u}\|_2\|\boldsymbol{v}\|_2 = \left(\|\boldsymbol{u}\|_2 + \|\boldsymbol{v}\|_2\right)^2,$$

lo que termina la demostración tomando raíz cuadrada □

Como generalización de la norma-2, están las normas$-p$, definidas también en $\mathbb{K}^n$:

- Norma-1: $\|\boldsymbol{v}\|_1 = |v_1| + \cdots + |v_n|$
- Norma-infinito: $\|\boldsymbol{v}\|_\infty = \text{máx}\{|v_1|, \ldots, |v_n|\}$
- Norma$-p$: $\|\boldsymbol{v}\|_p = (|v_1|^p + \cdots + |v_n|^p)^{1/p}$

En el siguiente gráfico podemos ver la diferencia entre las 3 normas más usuales en $\mathbb{R}^2$. En cada gráfico están representados todos los puntos con norma igual a 1 bajo la norma respectiva.

[1] Si escribimos $z_1 = a_1 + ib_1, z_2 = a_2 + ib_2 \in \mathbb{C}$, resulta $|z_1 + z_2| \leq |z_1| + |z_2|$ sí y solo sí

$$(a_1 + a_2)^2 + (b_1 + b_2)^2 \leq \left(\sqrt{a_1^2 + b_1^2} + \sqrt{a_2^2 + b_2^2}\right)^2,$$

desarrollar el cuadrado y ver que eso ocurre sí y solo sí $0 \leq (a_1b_2 - a_2b_1)^2$ que obviamente siempre es válido.

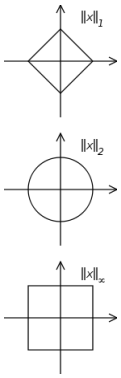


FIGURA 1. "Círculos" de radio 1 en diferentes normas. Es decir $\boldsymbol{v} \in \mathbb{R}^2$ tales que $\|\boldsymbol{v}\| = 1$.

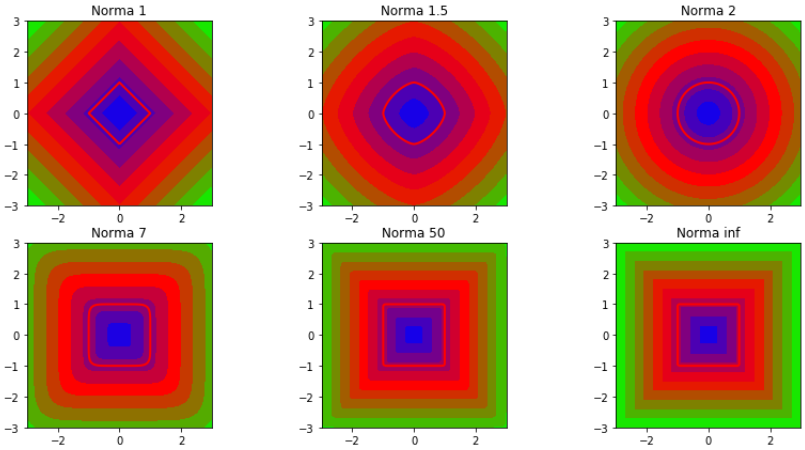


FIGURA 2. Curvas de nivel de diferentes normas$-p$.

En la siguiente figura observamos distintas curvas de nivel[2] para distintas normas.

EJERCICIO 3.1. Examinando los gráficos de las curvas de nivel de las normas$-p$ se puede intuir que $\lim_{p\to+\infty} \|\boldsymbol{v}\|_p = \|\boldsymbol{v}\|_\infty$. De una demostración de este hecho.

Para nosotros va a ser relevante medir *errores* en espacios vectoriales, para ello recordemos que dados $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n$ $\|\boldsymbol{u} - \boldsymbol{v}\|_2$ da la distancia entre ellos. En ocasiones, sin embargo, vamos a notar que es mas sencillo trabajar con una norma diferente a la norma-2, por lo que resulta natural ver que relación hay entre las diferentes normas-p. Puede probarse muy fácilmente lo siguiente.

EJERCICIO 3.2. Verificar que para todo $\boldsymbol{x} \in \mathbb{K}^n$

1. $\|\boldsymbol{x}\|_2 \leq \|\boldsymbol{x}\|_1 \leq \sqrt{n}\|\boldsymbol{x}\|_2$
2. $\|\boldsymbol{x}\|_\infty \leq \|\boldsymbol{x}\|_2 \leq \sqrt{n}\|\boldsymbol{x}\|_\infty$
3. $\|\boldsymbol{x}\|_\infty \leq \|\boldsymbol{x}\|_1 \leq n\|\boldsymbol{x}\|_\infty$

[2]Las curvas en las cuales la norma respectiva permanece constante.

Ese resultado es un caso particular de uno mucho mas general. Primero definamos *equivalencia de normas*.

DEFINICIÓN 3.2. Sean $\|\cdot\|$ y $\|\cdot\|_*$ dos normas en un $\mathbb{K}$-espacio vectorial V. Decimos que son equivalentes si existen dos constantes $0 < c, C$ tales que para todo $\boldsymbol{x} \in V$

$$c\|\boldsymbol{x}\|_* \leq \|\boldsymbol{x}\| \leq C\|\boldsymbol{x}\|_*$$

Vale lo siguiente,

PROPOSICIÓN 3.2. En un $\mathbb{K}$-espacio vectorial de dimensión finita todas la normas son equivalentes[3].

Este resultado nos es útil en muchos contextos. Veamos primero la siguiente definición.

DEFINICIÓN 3.3. Decimos que una sucesión de vectores $\{v_n\}_{n\in\mathbb{N}}$ converge bajo una norma $\|\cdot\|$ a un vector v si

$$\|v_n - v\| \to 0 \quad \text{cuando } n \to \infty.$$

Como consecuencia de la Proposición **??**, la convergencia en una norma cualquiera implica la convergencia en todas las normas.

2. Producto Interno

DEFINICIÓN 3.4. Dados dos vectores $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{K}^n$ definimos[a] el producto interno (canónico) por la fórmula[b]

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \boldsymbol{u}^* \boldsymbol{v} \in \mathbb{K},$$

es decir

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \overline{u_1} v_1 + \overline{u_2} v_2 + \cdots + \overline{u_n} v_n = \sum_{i=1}^{n} \overline{u_i} v_i.$$

[a] Recordemos que asociamos los vectores con las matrices columna.

[b] En muchos casos se suele conjugar los coeficientes del $\boldsymbol{v}$ en la sumatoria en vez de los de $\boldsymbol{u}$ en la definición de $\langle \boldsymbol{u}, \boldsymbol{v} \rangle$. Para nuestros intereses no cambia en absoluto la definición utilizada que por lo demas coinciden sobre $\mathbb{R}$.

Tenemos las siguientes propiedades inmediatas.

[3] Las constantes que relacionan las normas dependen de la dimensión n, típicamente se deterioran si $n \to \infty$

OBSERVACIÓN 3.2. Notar que para $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathbb{K}^n$, $\boldsymbol{w} \in \mathbb{K}^m$ $a, b \in \mathbb{K}$, $\boldsymbol{A} \in \mathbb{K}^{n\times m}$, resulta

1. $\langle \boldsymbol{x}, \boldsymbol{y}\rangle = \overline{\langle \boldsymbol{y}, \boldsymbol{x}\rangle}$,
2. $\langle \boldsymbol{x}, a\boldsymbol{y} + b\boldsymbol{z}\rangle = a\langle \boldsymbol{x}, \boldsymbol{y}\rangle + b\langle \boldsymbol{x}, \boldsymbol{z}\rangle$, y $\langle a\boldsymbol{y} + b\boldsymbol{z}, \boldsymbol{x}\rangle = \overline{a}\langle \boldsymbol{y}, \boldsymbol{x}\rangle + \overline{b}\langle \boldsymbol{z}, \boldsymbol{x}\rangle$.
3. $\langle \boldsymbol{A}\boldsymbol{w}, \boldsymbol{x}\rangle = \langle \boldsymbol{w}, \boldsymbol{A}^*\boldsymbol{x}\rangle$[a]
4. $\langle a\boldsymbol{x}, a\boldsymbol{y}\rangle = \overline{a}a\langle \boldsymbol{x}, \boldsymbol{y}\rangle = |a|^2\langle \boldsymbol{x}, \boldsymbol{y}\rangle$.
5. Utilizando el producto interno, la norma-2 de un vector se puede definir por la fórmula $\|\boldsymbol{v}\| = \sqrt{\langle \boldsymbol{v}, \boldsymbol{v}\rangle}$.
6. La desigualdad de Cauchy-Schwarz se escribe $|\langle \boldsymbol{v}, \boldsymbol{w}\rangle| \leq \|\boldsymbol{v}\|_2\|\boldsymbol{w}\|_2$,

[a]Puesto que $\langle \boldsymbol{A}\boldsymbol{w}, \boldsymbol{x}\rangle = (\boldsymbol{A}\boldsymbol{w})^*\boldsymbol{x} = \boldsymbol{w}^*\boldsymbol{A}^*\boldsymbol{x} = \langle \boldsymbol{w}, \boldsymbol{A}^*\boldsymbol{x}\rangle$.

EJEMPLO 3.1. $\langle \boldsymbol{x}+\boldsymbol{y}, \boldsymbol{x}+\boldsymbol{y}\rangle = \langle \boldsymbol{x}, \boldsymbol{x}+\boldsymbol{y}\rangle + \langle \boldsymbol{y}, \boldsymbol{x}+\boldsymbol{y}\rangle = \langle \boldsymbol{x}, \boldsymbol{x}\rangle + \langle \boldsymbol{x}, \boldsymbol{y}\rangle + \langle \boldsymbol{y}, \boldsymbol{x}\rangle + \langle \boldsymbol{y}, \boldsymbol{y}\rangle = \|\boldsymbol{x}\|^2 + 2\langle \boldsymbol{x}, \boldsymbol{y}\rangle + \|\boldsymbol{y}\|^2$

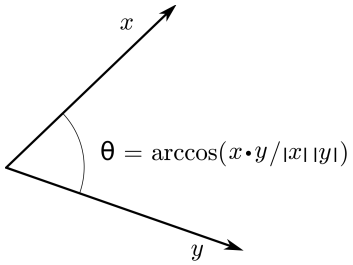
Sean $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^2$, $\boldsymbol{x} \neq \boldsymbol{0} \neq \boldsymbol{y}$, $\boldsymbol{x} = (a_1, a_2)$, $\boldsymbol{y} = (b_1, b_2)$. Podemos escribir $\boldsymbol{x} = \|\boldsymbol{x}\|\frac{\boldsymbol{x}}{\|\boldsymbol{x}\|}$ y como $\frac{\boldsymbol{x}}{\|\boldsymbol{x}\|}$ está en el círculo de radio 1 vemos que existe un único α, $0 \leq \alpha < 2\pi$ tal que $\frac{\boldsymbol{x}}{\|\boldsymbol{x}\|} = (\cos(\alpha), \operatorname{sen}(\alpha))$. Es decir, $\boldsymbol{x} = \|\boldsymbol{x}\|_2(\cos(\alpha), \operatorname{sen}(\alpha))$. Análogamente, podemos escribir $\boldsymbol{y} = \|\boldsymbol{y}\|_2(\cos(\beta), \sin(\beta))$ y en particular[4]

$$\langle \boldsymbol{x}, \boldsymbol{y}\rangle = \|\boldsymbol{x}\|_2\|\boldsymbol{y}\|_2\left(\cos(\alpha)\cos(\beta) + \operatorname{sen}(\alpha)\operatorname{sen}(\beta)\right) = \|\boldsymbol{v}\|_2\|\boldsymbol{w}\|_2\cos(\alpha - \beta),$$

lo que dice que puede escribirse

$$\cos(\theta) = \frac{\langle \boldsymbol{x}, \boldsymbol{y}\rangle}{\|\boldsymbol{v}\|_2\|\boldsymbol{w}\|_2}, \tag{3.1}$$

con θ el ángulo[5] entre $\boldsymbol{x}$ y $\boldsymbol{y}$.



En $\mathbb{R}^n$ ocurre lo mismo. Ya que $-1 \leq \frac{\langle \boldsymbol{x}, \boldsymbol{y}\rangle}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|} \leq 1$, (ver (6) de la Observación **??**) se define el ángulo θ entre $\boldsymbol{x}$ e $\boldsymbol{y}$ a través de (**??**).

Las normas nos han permitido incorporar al *álgebra* (el espacio vectorial) el concepto de distancia, que nos permitirá introducir la noción de convergencia y una forma de medir errores. El producto escalar, por su parte, nos permite trabajar con ángulos. En particular con la idea de perpendicularidad u ortogonalidad que resulta fundamental en el desarrollo de algoritmos estables.

[4]Recordar: $cos(\alpha+\beta) = cos(\alpha)cos(\beta) - sen(\alpha)sen(\beta)$, $sen(\alpha+\beta) = sen(\alpha)cos(\beta) + cos(\alpha)sen(\beta)$.

[5]Considerando que $\cos(\mu) = \cos(2\pi - \mu)$ siempre puede elegirse $0 \leq \theta < \pi$.

De (??) vemos que $\theta = \pi/2$ si y solo si $\boldsymbol{v}^*\boldsymbol{w} = 0$, de ahí la siguiente definición que la extendemos a los complejos.

DEFINICIÓN 3.5. Dados $\boldsymbol{v}, \boldsymbol{w} \in \mathbb{K}^n$, no nulos. Decimos que son ortogonales sí y solo sí $\boldsymbol{v}^*\boldsymbol{w} = \langle \boldsymbol{v}, \boldsymbol{w} \rangle = 0$. En este caso también usaremos la notación $\boldsymbol{v} \perp \boldsymbol{w}$.

DEFINICIÓN 3.6. Una colección de vectores $\{\boldsymbol{v}_i\}_{1 \leq i \leq k} \subset \mathbb{K}^n$ se dice *ortogonal* si $\langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle = \delta_i^j$. Si además $\|\boldsymbol{v}_i\|_2 = 1$ para todo $i \leq i \leq k$ decimos que el conjunto es *ortonormal.*

Los conjuntos ortogonales tienen muchas propiedades que pueden explotarse, tanto teóricamente como en la práctica.

Veamos una de ellas de importancia fundamental. Sea $\mathcal{B} = \{\boldsymbol{v}\}_{1 \leq i \leq n} \subset \mathbb{K}^n$ una *base* ortonormal de $\mathbb{K}^n$. Dado $\boldsymbol{w} \in \mathbb{K}^n$ queremos conocer las coordenadas de $\boldsymbol{w}$ en la base $\mathcal{B}$. Esto se reduce a hallar los $\{\alpha_i\}_{1 \leq i \leq n} \subset \mathbb{K}$ tales que

$$\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i \boldsymbol{v}_i, \tag{3.2}$$

que como sabemos implica resolver un sistema de $n \times n$. Sin embargo en este caso particular, vemos que para cada $1 \leq j \leq n$ el producto escalar

$$\boldsymbol{v}_j^* \boldsymbol{w} = \sum_{i=1}^{n} \alpha_i \boldsymbol{v}_j^* \boldsymbol{v}_i = \alpha_j, \tag{3.3}$$

permite "despejar" α_i por el módico costo de un producto escalar[6] Desde el punto de vista matricial, sea $\boldsymbol{Q} \in \mathbb{K}^{n \times n}$, que tiene por columnas los $\boldsymbol{v}_i$. Resulta que el problema anterior se puede escribir de modo matricial

$$\boldsymbol{w} = \boldsymbol{Q} \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix},$$

y de la expresión de mas arriba vemos que

$$\boldsymbol{Q}^* \boldsymbol{w} = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix}.$$

Esto no expresa mas que el hecho de que invertir $\boldsymbol{Q}$ se reduce a conjugar (trasponer si $\mathbb{K} = \mathbb{R}$) lo que a su vez es inmediato considerando que sus columnas son ortonomales[7] y así

$$[\boldsymbol{Q}^* \boldsymbol{Q}]_{ij} = \boldsymbol{v}_i^* \boldsymbol{v}_j = \delta_i^j.$$

Estas matrices tienen un nombre particular.

DEFINICIÓN 3.7. Una matriz $\boldsymbol{Q} \in \mathbb{K}^{n \times n}$ se dice *unitaria* (suele llamarse *ortogonal* cuando $\mathbb{K} = \mathbb{R}$) si $\boldsymbol{Q}^*\boldsymbol{Q} = \boldsymbol{I}$ ($\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}$ si $\mathbb{K} = \mathbb{R}$).

OBSERVACIÓN 3.3. $\boldsymbol{Q}$ es unitaria (resp. ortogonal) si y solo si $\boldsymbol{Q}^*$ es unitaria (resp. ortogonal).

[6] Esto es $2n - 1$ operaciones, n productos y $n - 1$ sumas.

[7] El costo total de resolver el sistema es $n(2n - 1) \sim O(2n^2)$ operaciones.

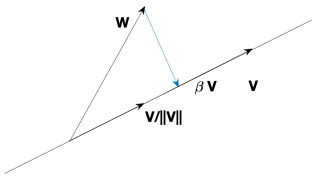


FIGURA 3. Proyección de $\boldsymbol{w}$ sobre la recta generada por $\boldsymbol{v}$.

Estudiemos ahora el problema de calcular la proyección *ortogonal* de un vector $\boldsymbol{w}$ sobre la recta generada por otro vector $\boldsymbol{v} \neq \boldsymbol{0}$ (ver Figura **??**). Es decir, que buscamos un vector $\beta\boldsymbol{v}$ tal que $(\beta\boldsymbol{v} - \boldsymbol{w}) \perp \boldsymbol{v}$, esto equivale a $(\beta\boldsymbol{v} - \boldsymbol{w})^*\boldsymbol{v} = 0$, lo que indica que $\overline{\beta} = \frac{\boldsymbol{w}^*\boldsymbol{v}}{\|\boldsymbol{v}\|_2^2}$, es decir

$$\beta = \frac{\boldsymbol{v}^*\boldsymbol{w}}{\|\boldsymbol{v}\|_2^2}.$$

Si $\|\boldsymbol{v}\| = 1$ la expresión coincide con la de los α calculados en (**??**), lo que nos da una interpretación alternativa de (**??**). Esto es, llamando $\boldsymbol{P_v}(\boldsymbol{w})$ a la proyección ortogonal de $\boldsymbol{w}$ sobre el subespacio generado por $\boldsymbol{v}$, vemos que

$$\boldsymbol{P_v}(\boldsymbol{w}) = \frac{\boldsymbol{v}^*\boldsymbol{w}}{\|\boldsymbol{v}\|_2^2}\boldsymbol{v}, \tag{3.4}$$

y en particular (**??**) toma la forma

$$\boldsymbol{w} = \sum_{1 \leq i \leq n} \boldsymbol{P}_{\boldsymbol{v}_i}(\boldsymbol{w}).$$

De los argumentos previos vemos que *en una base ortonormal*, las proyecciones ortogonales de un vector $\boldsymbol{w}$ sobre los elementos de la base nos dan sus coordenadas en esa base. Eso es justamente a lo que estamos acostumbrados en la base canónica.

Otro hecho de gran importancia es la generalización del teorema de Pitágoras en cualquier dimensión si tenemos una base ortonormal. Esto es, de la expresión (**??**), tenemos que el cuadrado de la longitud de $\boldsymbol{w}$ (el cuadrado de la "hipotenusa") es, por un lado

$$\|\boldsymbol{w}\|_2^2 = \boldsymbol{w}^*\boldsymbol{w}$$

y por el otro

$$(\sum_{i=1}^{n} \alpha_i \boldsymbol{v}_i)^*(\sum_{i=1}^{n} \alpha_i \boldsymbol{v}_i) = (\sum_{i=1}^{n} \overline{\alpha_i}\boldsymbol{v}_i^*)(\sum_{i=1}^{n} \alpha_i \boldsymbol{v}_i) = \sum_{i,j=1}^{n} \overline{\alpha_j}\alpha_i \boldsymbol{v}_j^*\boldsymbol{v}_i = \sum_{i=1}^{n} \overline{\alpha_i}\alpha_i = \sum_{i=1}^{n} |\alpha_i|^2,$$

es decir el cuadrado de la suma de las longitudes de las coordenadas (los "catetos"). Para resaltar este hecho que usaremos en lo sucesivo, lo recuadramos,

OBSERVACIÓN 3.4. En toda base ortonormal $\mathcal{B} = \{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n\}$, si

$$\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i \boldsymbol{v}_i,$$

entonces

$$\|\boldsymbol{w}\|^2 = \sum_{i=1}^{n} |\alpha_i|^2.$$

Mas adelante (Sección **??**) volveremos sobre el tema de las proyecciones.

Para finalizar la sección de producto interno definimos

DEFINICIÓN 3.8. Una matriz $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ se dice *Hermitiana* si $\boldsymbol{A} = \boldsymbol{A}^*$. En el caso de que $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, $\boldsymbol{A}$ se dice *simétrica*.

Si $\boldsymbol{A} \in \mathbb{K}^{n\times n}$ es Hermitiana, entonces, para todo $\boldsymbol{v} \in \mathbb{K}^n$, $\boldsymbol{v}^*\boldsymbol{A}\boldsymbol{v} \in \mathbb{R}$. En efecto, por un lado, debido a las dimensiones de los elementos que aparecen en el producto, se tiene $z = \boldsymbol{v}^*\boldsymbol{A}\boldsymbol{v} \in \mathbb{K}$, por otro lado puesto que

$$\overline{\boldsymbol{v}^*\boldsymbol{A}\boldsymbol{v}} = (\boldsymbol{v}^*\boldsymbol{A}\boldsymbol{v})^* = \boldsymbol{v}^*\boldsymbol{A}^*(\boldsymbol{v}^*)^* = \boldsymbol{v}^*\boldsymbol{A}\boldsymbol{v},$$

resulta de la primera y ultima expresión de la cadena de igualdades que $\overline{z} = z$, es decir $z \in \mathbb{R}$.

DEFINICIÓN 3.9. Una matriz $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ ($\boldsymbol{A} \in \mathbb{R}^{n\times n}$) se dice *definida positiva* si es Hermitiana (simétrica) y además $\boldsymbol{v}^*\boldsymbol{A}\boldsymbol{v} > 0$ para todo $\boldsymbol{v} \neq 0$. Si por otro lado $\boldsymbol{v}^*\boldsymbol{A}\boldsymbol{v} \geq 0$ para todo $\boldsymbol{v}$, la matriz se dice *semi definida* positiva.

3. Normas de matrices

Dada una matriz $A \in \mathbb{K}^{n\times m}$, se la puede pensar pensar como un vector de $\mathbb{K}^{nm}$ y en consecuencia se aplicaría a las matrices todo lo que hemos desarrollado para normas vectoriales. Sin embargo, salvo casos excepcionales[8], esta normas no son de gran utilidad en el contexto de las matrices. Esto es debido a que nos interesa estudiarlas desde la perspectiva de las trasformaciones lineales asociadas. Por esta razón se introducen las normas subordinadas.

Dada una matriz $\boldsymbol{A} \in \mathbb{K}^{n\times m}$, y un par de normas vectoriales $\|\cdot\|_n, \|\cdot\|_m$ en $\mathbb{K}^n$ y $\mathbb{K}^m$ respectivamente, definimos

$$\|\boldsymbol{A}\|_{n,m} = \max_{\boldsymbol{0}\neq\boldsymbol{x}\in\mathbb{K}^m} \frac{\|\boldsymbol{A}\boldsymbol{x}\|_n}{\|\boldsymbol{x}\|_m} = \max_{\boldsymbol{x}\in\mathbb{K}^m, \|\boldsymbol{x}\|_m=1} \|\boldsymbol{A}\boldsymbol{x}\|_n.$$

Un caso usual en este curso es que la matriz sea cuadrada $\boldsymbol{A} \in \mathbb{K}^{n\times m}$ y en ese caso usamos una sola norma vectorial $\|\cdot\|$

[8]Como la norma de Frobenius que veremos mas adelante.

$$\|\boldsymbol{A}\| = \max_{\boldsymbol{0}\neq\boldsymbol{x}\in\mathbb{K}^m} \frac{\|\boldsymbol{A}\boldsymbol{x}\|}{\|\boldsymbol{x}\|} = \max_{\boldsymbol{x}\in\mathbb{K}^m, \|\boldsymbol{x}\|=1} \|\boldsymbol{A}\boldsymbol{x}\|. \tag{3.5}$$

Estas normas matriciales se dicen *subordinadas* a la norma vectorial $\|\cdot\|$ que estamos utilizando en el espacio $\mathbb{K}^n$. De su definición vemos que la norma (subordinada) de una matriz mide en qué proporción puede aumentar como máximo la norma de un vector $\boldsymbol{x}$ al multiplicarlo por $\boldsymbol{A}$.

EJERCICIO 3.3. Sea $\boldsymbol{A} \in \mathbb{K}^{n\times n}$ y $\|\cdot\|$ una norma en $\mathbb{K}^n$, entonces (**??**) define una norma.

EJERCICIO 3.4. Sea $\boldsymbol{I} \in \mathbb{K}^{n\times n}$ la identidad, entonces $\|\boldsymbol{I}\| = 1$ para toda norma subordinada (siempre que pensemos $\mathbb{K}^n$ con la misma norma vectorial como espacio de partida y llegada de la transformación $\boldsymbol{I} : \mathbb{K}^n \to \mathbb{K}^n$).

También resulta inmediato de la definición que para todo $\boldsymbol{x}$

$$\|\boldsymbol{A}\boldsymbol{x}\| \leq \|\boldsymbol{A}\|\|\boldsymbol{x}\|. \tag{3.6}$$

De aquí se obtiene la siguiente propiedad. Para todo par de matrices[9] $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{K}^{n\times n}$, y toda *norma subordinada*

$$\|\boldsymbol{A}\boldsymbol{B}\| \leq \|\boldsymbol{A}\|\|\boldsymbol{B}\|. \tag{3.7}$$

En efecto, dos aplicaciones sucesivas de (**??**) da

$$\|\boldsymbol{A}\boldsymbol{B}\| = \max_{\|\boldsymbol{x}\|=1} \|\boldsymbol{A}\boldsymbol{B}\boldsymbol{x}\| \leq \max_{\|\boldsymbol{x}\|=1} \|\boldsymbol{A}\|\|\boldsymbol{B}\boldsymbol{x}\| \leq \max_{\|\boldsymbol{x}\|=1} \|\boldsymbol{A}\|\|\boldsymbol{B}\|\|\boldsymbol{x}\| = \|\boldsymbol{A}\|\|\boldsymbol{B}\|,$$

que prueba (**??**). Aplicando sucesivamente (**??**), vemos que para toda matriz cuadrada y para todo $k \in \mathbb{N}$, vale[10]

$$\|\boldsymbol{A}^k\| \leq \|\boldsymbol{A}\|^k. \tag{3.8}$$

En algunos pocos casos es posible calcular explícitamente la expresión de $\|\boldsymbol{A}\|$.

EJEMPLO 3.2. Consideremos en $\mathbb{K}^n$ la norma $\|\cdot\|_\infty$. Para $\boldsymbol{A} \in \mathbb{K}^{n\times n}$ se tiene[11]

$$\|\boldsymbol{A}\|_\infty = \max_{1\leq i\leq n} \{\sum_{j=1}^{n} |a_{ij}|\}.$$

[9]Como se desprende de la cuenta, también vale para matrices rectangulares, siempre que pueda hacerse el producto $\boldsymbol{A}\boldsymbol{B}$.

[10]Que ocurre si $k \in \mathbb{Z}$ con $k < 0$?. Por ejemplo, cuando $\boldsymbol{A}$ es invertible sabemos darle sentido a $\boldsymbol{A}^{-1}$ como la inversa de $\boldsymbol{A}$. Por lo tanto $\boldsymbol{I} = \boldsymbol{A}\boldsymbol{A}^{-1}$ de donde $1 \leq \|\boldsymbol{A}\|\|\boldsymbol{A}^{-1}\|$ (hemos usado aquí el Ejercicio **??**) y de ahí $\|\boldsymbol{A}\|^{-1} \leq \|\boldsymbol{A}^{-1}\|$. Es decir al revés que en (**??**). Piense que ocurre en general. Defina primero $\boldsymbol{A}^k$ con $k < 0$. Como ayuda pregúntese que será por ejemplo $\boldsymbol{A}^{-3}$?. Vea que puede pensarlo como $\boldsymbol{A}^{-3} := \left(\boldsymbol{A}^{-1}\right)^3$, es decir $\boldsymbol{A}^{-3} = \boldsymbol{A}^{-1}\boldsymbol{A}^{-1}\boldsymbol{A}^{-1}$. Pero otro lado $\boldsymbol{A}^3\left(\boldsymbol{A}^{-1}\boldsymbol{A}^{-1}\boldsymbol{A}^{-1}\right) = \boldsymbol{I}$, lo que dice que la primera definición coincide con $(\boldsymbol{A}^3)^{-1}$. Use estas consideraciones para responder la pregunta.

[11]También vale la demostración en el caso de matrices rectangulares.

Demostración. Lo vemos para $\mathbb{K} = \mathbb{R}$. Debemos calcular

$$\|\boldsymbol{A}\|_\infty = \max_{\boldsymbol{x}\in\mathbb{R}^n, \|\boldsymbol{x}\|_\infty=1} \|\boldsymbol{A}\boldsymbol{x}\|_\infty.$$

Llamemos k a la fila donde ser realiza el máximo, es decir

$$\max_{1\le i\le n}\{\sum_{j=1}^n |a_{ij}|\} = \sum_{j=1}^n |a_{kj}|,$$

(si hay más de una tomamos cualquiera). Podemos suponer que $\sum_{j=1}^n |a_{kj}| > 0$ de otro modo el resultado es inmediato. Llamemos $\boldsymbol{z}$ al vector de los signos de los elementos de esa fila $\boldsymbol{z} = (sg(a_{k,1}), \cdots, sg(a_{k,n})$. Si algún $a_{k,i} = 0$ tomamos $sg(a_{k,1}) = 0$. Como la fila no es nula, $\boldsymbol{z} \neq \boldsymbol{0}$. Luego $\|\boldsymbol{z}\|_\infty = \max_{1\le i\le n}\{|z_i|\} = 1$, y haciendo

$$\|\boldsymbol{A}\|_\infty = \max_{\boldsymbol{x}\in\mathbb{R}^n, \|\boldsymbol{x}\|_\infty=1} \|\boldsymbol{A}\boldsymbol{x}\|_\infty \ge \|\boldsymbol{A}\boldsymbol{z}\|_\infty \ge \sum_{j=1}^n a_{i,j} sg(a_{i,j}) = \sum_{j=1}^n |a_{i,j}| = \max_{1\le i\le n}\{\sum_{j=1}^n |a_{ij}|\},$$

lo que prueba una desigualdad. Para a otra desigualdad, sea $\boldsymbol{x} \in \mathbb{R}^n$ tal que $\|\boldsymbol{x}\|_\infty = 1$, luego para todo $1 \le j \le n$ $|x_j| \le \|\boldsymbol{x}\|_\infty = 1$ y entonces, para cualquier elemento i del vector $\boldsymbol{A}\boldsymbol{x}$

$$|[\boldsymbol{A}\boldsymbol{x}]_i| = |\sum_{j=1}^n a_{i,j}x_j| \le \sum_{j=1}^n |a_{i,j}| \le \max_{1\le i\le n}\{\sum_{j=1}^n |a_{ij}|\},$$

lo que da la otra desigualdad y demuestra el caso $\mathbb{K} = \mathbb{R}$. El caso $\mathbb{K} = \mathbb{C}$ sale del mismo modo recordando que si $z \in \mathbb{C}$ y $z = |z|e^{i\theta}$, se tiene que $ze^{-i\theta} = |z|$ y $|e^{i\theta}| = 1$. □

Análogamente para la norma-1 se tiene

Ejemplo 3.3. Consideremos en $\mathbb{K}^n$ la norma $\|\cdot\|_1$. Para $\boldsymbol{A} \in \mathbb{K}^{n\times n}$ se tiene[12]

$$\|\boldsymbol{A}\|_1 = \max_{1\le j\le n}\{\sum_{i=1}^n |a_{ij}|\}.$$

Un caso de mucha importancia para nosotros es la norma matricial subordinada a la norma-2. Comencemos calculando un caso elemental: la norma-2 de una matriz unitaria/ortogonal (Definición **??**). Supongamos que $\boldsymbol{Q} \in \mathbb{K}^{n\times n}$ y $\boldsymbol{Q}^*\boldsymbol{Q} = \boldsymbol{I}$, luego, para todo$\boldsymbol{x} \in \mathbb{K}^n$

$$\|\boldsymbol{Q}\boldsymbol{x}\|_2^2 = (\boldsymbol{Q}\boldsymbol{x})^*\boldsymbol{Q}\boldsymbol{x} = \boldsymbol{x}^*\boldsymbol{x} = \|\boldsymbol{x}\|_2^2,$$

es decir que

$$\|\boldsymbol{Q}\boldsymbol{x}\|_2 = \|\boldsymbol{x}\|_2, \ \forall \boldsymbol{x} \in \mathbb{K}^n. \tag{3.9}$$

Vemos que las matrices unitarias son *isometrías*, es decir, no cambian las longitudes. Tampoco cambian los ángulos ya que preservan el producto interno, propiedad que resaltamos a continuación

$$\langle \boldsymbol{Q}\boldsymbol{v}, \boldsymbol{Q}\boldsymbol{w}\rangle = (\boldsymbol{Q}\boldsymbol{v})^*\boldsymbol{Q}\boldsymbol{w} = \boldsymbol{v}^*\boldsymbol{w} = \langle \boldsymbol{v}, \boldsymbol{w}\rangle. \tag{3.10}$$

[12]También vale en el caso de matrices rectangulares.

Estas propiedades definen a los *movimientos rígidos*. De (**??**), tenemos

$$\|\boldsymbol{Q}\|_2 = \sup_{\boldsymbol{x}\neq\boldsymbol{0}} \frac{\|\boldsymbol{Q}\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} = 1,$$

lo que les da su nombre a las matrices unitarias.

Ejercicio 3.5. Verificar que la matriz

$$\boldsymbol{A} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix}$$

es una matriz ortogonal para cualquier valor de α. Interpretar geométricamente la transformación lineal $\boldsymbol{A}\boldsymbol{x}$.

Como ya hemos señalado, las normas subordinadas son las mas útiles. Sin embargo la norma euclídea aplicada a matrices -que es no subordinada- tiene importantes aplicaciones y un nombre propio.

Definición 3.10. Dada $\boldsymbol{A} \in \mathbb{K}^{n\times m}$ definimos la norma de Frobenius

$$\|\boldsymbol{A}\|_F = \sqrt{\sum_{1\leq i\leq n} \sum_{i\leq j\leq m} |a_{i,j}|^2}$$

Notar que

$$\|\boldsymbol{A}\|_F = \sqrt{tr(\boldsymbol{A}^T\boldsymbol{A})},$$

donde tr indica el operador traza (la suma de los elementos de la diagonal). Las normas no subordinadas de matrices no suelen cumplir con la desigualdad (**??**). La norma de Frobenius es una excepción, gracias a la desigualdad de Cauchy-Schwarz. En efecto, dados[13] $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{K}^{n\times n}$, consideremos el elemento i, j de la matriz producto

$$|[\boldsymbol{A}\boldsymbol{B}]_{i,j}|^2 \leq |(\sum_{l=1}^{n} \boldsymbol{A}_{i,l}\boldsymbol{B}_{l,j})|^2 \leq (\sum_{1=1}^{n} \boldsymbol{A}_{i,l}^2)(\sum_{1=1}^{n} \boldsymbol{B}_{l,j}^2),$$

donde hemos usado la Proposición **??** en la última desigualdad. Sumando en i y en j se tiene el resultado deseado

$$\|\boldsymbol{A}\boldsymbol{B}\|_F^2 = \sum_{1\leq i,j\leq n} |[\boldsymbol{A}\boldsymbol{B}]_{i,j}|^2 \leq \sum_{1\leq i\leq n} \sum_{1\leq j\leq n} (\sum_{1=1}^{n} \boldsymbol{A}_{i,l}^2)(\sum_{1=1}^{n} \boldsymbol{B}_{l,j}^2) = \|\boldsymbol{A}\|_F^2 \|\boldsymbol{B}\|_F^2,$$

esto es

$$\|\boldsymbol{A}\boldsymbol{B}\|_F \leq \|\boldsymbol{A}\|_F \|\boldsymbol{B}\|_F.$$

Cerramos esta sección con propiedades de las normas de matrices unitarias (algunas las hemos ya probado pero las ponemos en recuadro para recordarlas). De todas la matrices, las unitarias poseen propiedades especiales que las hacen muy favorables para su uso en los algoritmos.

[13] El siguiente argumento vale también para matrices rectangulares.

PROPOSICIÓN 3.3. Sea $\boldsymbol{A} \in \mathbb{K}^{n\times m}$, $\boldsymbol{Q} \in \mathbb{K}^{n\times n}$, $\boldsymbol{Q}$ unitaria/ortogonal, resulta

1. $\forall \boldsymbol{v} \in \mathbb{K}^n \ \|\boldsymbol{Q}\boldsymbol{v}\|_2 = \|\boldsymbol{v}\|_2$
2. $\|\boldsymbol{Q}\|_2 = 1$
3. $\|\boldsymbol{Q}\boldsymbol{A}\|_2 = \|\boldsymbol{A}\|_2$
4. $\|\boldsymbol{Q}\boldsymbol{A}\|_F = \|\boldsymbol{A}\|_F$

DEMOSTRACIÓN. Las propiedades (1) y (2) ya las hemos probado.

Para (3), notemos que por (1) se tiene

$$\|\boldsymbol{Q}\boldsymbol{A}\|_2 = sup_{\boldsymbol{v},\|\boldsymbol{v}\|_2=1}\|\boldsymbol{Q}\boldsymbol{A}\boldsymbol{v}\|_2 = sup_{\boldsymbol{v},\|\boldsymbol{v}\|=1}\|\boldsymbol{A}\boldsymbol{v}\|_2 = \|\boldsymbol{A}\|_2.$$

Finalmente, para (4) escribimos

$$\|\boldsymbol{Q}\boldsymbol{A}\|_F^2 = tr((\boldsymbol{Q}\boldsymbol{A})^*(\boldsymbol{Q}\boldsymbol{A})) = tr(\boldsymbol{A}^*\boldsymbol{A}) = \|\boldsymbol{A}\|_F^2.$$

□

4. Condicion de Matrices

Sea $\boldsymbol{A} \in \mathbb{R}^{n\times m}$, queremos estudiar en más detalle la condición de evaluar $\boldsymbol{A}\boldsymbol{v}$ (pensado como transformación lineal). Notemos que podemos hacerlo con bastante en general aprovechando la linealidad. En efecto, para una perturbación arbitraria $\mathbf{h}$ de $\boldsymbol{v}$ tenemos que el error relativo de evaluar en $\boldsymbol{v}$ (asumiendo que $\boldsymbol{v} \notin Nu(\boldsymbol{A})$) puede escribirse

$$\frac{\|\boldsymbol{A}(\boldsymbol{v}+\mathbf{h}) - \boldsymbol{A}\boldsymbol{v}\|}{\|\boldsymbol{A}\boldsymbol{v}\|} = \frac{\|\boldsymbol{A}\mathbf{h}\|}{\|\boldsymbol{A}\boldsymbol{v}\|},$$

y este valor, respecto del error relativo en $\boldsymbol{v}$ resulta

$$\frac{\frac{\|\boldsymbol{A}\mathbf{h}\|}{\|\boldsymbol{A}\boldsymbol{v}\|}}{\frac{\|\boldsymbol{v}\|}{\|\mathbf{h}\|}} = \frac{\|\boldsymbol{A}\mathbf{h}\|}{\|\mathbf{h}\|}\frac{\|\boldsymbol{v}\|}{\|\boldsymbol{A}\boldsymbol{v}\|} \leq \|\boldsymbol{A}\|\frac{\|\boldsymbol{v}\|}{\|\boldsymbol{A}\boldsymbol{v}\|}.$$

El número que está a la derecha es independiente de $\mathbf{h}$ y podemos usarlo como representante de la condición de evaluar $\boldsymbol{A}$ en $\boldsymbol{v}$ y de hecho vemos que coincide con la expresión sugerida para la condición en el caso general (i.e. $\frac{\|x_0\|\|Df(x_0)\|}{\|f(x_0)\|}$). Si quisieramos una expresión independiente de $\boldsymbol{v}$ que podríamos hacer?. En el caso en que $n = m$ y $\boldsymbol{A}$ sea invertible, podemos llamar $\boldsymbol{w} = \boldsymbol{A}\boldsymbol{v}$ y el cociente $\frac{\|\boldsymbol{v}\|}{\|\boldsymbol{A}\boldsymbol{v}\|}$ puede acotarse idependiente de $\boldsymbol{v}$

$$\frac{\|\boldsymbol{v}\|}{\|\boldsymbol{A}\boldsymbol{v}\|} = \frac{\|\boldsymbol{A}^{-1}\boldsymbol{w}\|}{\|\boldsymbol{w}\|} \leq \|\boldsymbol{A}^{-1}\|.$$

En definitiva hallamos un número que mayora a la condición de evaluar en $\boldsymbol{v}$ para todo $\boldsymbol{v}$ y este es

$$\|\boldsymbol{A}\|\frac{\|\boldsymbol{v}\|}{\|\boldsymbol{A}\boldsymbol{v}\|} \leq \|\boldsymbol{A}\|\|\boldsymbol{A}^{-1}\|.$$

Notemos que argumentando con $\boldsymbol{A}^{-1}$, vemos que el mismo número mayora a la condición de evaluar $\boldsymbol{A}^{-1}\boldsymbol{w}$. Esto motiva la siguiente definición.

DEFINICIÓN 3.11. (Condición de una matriz) Dada $\boldsymbol{A} \in \mathbb{K}^{n\times n}$ invertible definimos la condición de $\boldsymbol{A}$, $\kappa(\boldsymbol{A})$, asociada a una norma subordinada $\|\cdot\|$, como

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

Si $\boldsymbol{A}$ no es invertible definimos $\kappa(\boldsymbol{A}) = +\infty$.

Que ocurre si $\boldsymbol{A} \in \mathbb{K}^{n\times m}$? podremos definir algo similar?. Volveremos sobre este caso mas adelante cuando definamos la *pseudoinversa*.

OBSERVACIÓN 3.1. Observemos que

1. La condición de una matriz depende de la norma matricial elegida. Sin embargo, como en dimensión finita todas las normas son equivalentes las condiciones son equivalentes.
2. Para todo $\mathbf{0} \neq \alpha \in \mathbb{K}$, $\kappa(\alpha\boldsymbol{A}) = \kappa(\boldsymbol{A})$.
3. $\kappa(\boldsymbol{I}) = 1$ para toda norma.
4. Como consecuencia del ítem previo, para toda norma, $1 \leq \kappa(\boldsymbol{A})$.
5. Para toda $\boldsymbol{Q}$ unitaria/ortogonal $\kappa(\boldsymbol{Q}) = 1$. (ver item (2) en Proposición **??**)
6. $\kappa(\boldsymbol{A}) = \kappa(\boldsymbol{A}^{-1})$

Consideremos por un segundo los problemas que pueden aparecer cuando resolvemos un sistema

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

con $\boldsymbol{b} \neq \mathbf{0}$ y $\boldsymbol{A}$ invertible. Por un lado tenemos que considerar que tanto $\boldsymbol{A}$ como $\boldsymbol{b}$ se almacenaran con errores. En general entonces estaremos resolviendo otro problema

$$(\boldsymbol{A} + \Delta\boldsymbol{A})(\boldsymbol{x} + \Delta\boldsymbol{x}) = \boldsymbol{b} + \Delta\boldsymbol{b}.$$

Asumiendo que procedemos a resolver el problema con aritmética exacta (no hay otras fuentes de error) nos interesa ver el impacto en el error $\Delta\boldsymbol{x}$ debido a los errores $\Delta\boldsymbol{A}, \Delta\boldsymbol{b}$.

Tenemos, usando que $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$

$$\Delta\boldsymbol{A}\boldsymbol{x} + \boldsymbol{A}\Delta\boldsymbol{x} + \Delta\boldsymbol{A}\Delta\boldsymbol{x} = \Delta\boldsymbol{b}.$$

Procediendo *informalmente*, despreciamos el término de "segundo orden" $\Delta\boldsymbol{A}\Delta\boldsymbol{x}$ y resulta

$$\Delta\boldsymbol{A}\boldsymbol{x} + \boldsymbol{A}\Delta\boldsymbol{x} \sim \Delta\boldsymbol{b}.$$

de donde

$$\Delta\boldsymbol{x} \sim \boldsymbol{A}^{-1}\Delta\boldsymbol{b} - \boldsymbol{A}^{-1}\Delta\boldsymbol{A}\boldsymbol{x},$$

y usando que $\|\boldsymbol{A}\| \neq 0$,

$$\|\Delta\boldsymbol{x}\| \lesssim \|\boldsymbol{A}^{-1}\|\|\Delta\boldsymbol{b}\| + \|\boldsymbol{A}^{-1}\|\Delta\boldsymbol{A}\|\|\boldsymbol{x}\| = \|\boldsymbol{A}^{-1}\|\frac{\|\Delta\boldsymbol{b}\|}{\|\boldsymbol{b}\|}\|\boldsymbol{b}\| + \|\boldsymbol{A}^{-1}\|\frac{\|\Delta\boldsymbol{A}\|}{\|\boldsymbol{A}\|}\|\boldsymbol{A}\|\|\boldsymbol{x}\|,$$

usando que $\|\boldsymbol{b}\| \leq \|\boldsymbol{A}\|\|\boldsymbol{x}\|$ y diviendo por $\|\boldsymbol{x}\| \neq \mathbf{0}$ tenemos

$$\frac{\|\Delta\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \lesssim \kappa(\boldsymbol{A})\left(\frac{\|\Delta\boldsymbol{b}\|}{\|\boldsymbol{b}\|} + \frac{\|\Delta\boldsymbol{A}\|}{\|\boldsymbol{A}\|}\right).$$

Lo que dice que el error relativo generado en la solución se acota en terminos de los errores relativos en los datos por la condición de la matriz.

Observar que en caso de ser $\Delta \boldsymbol{A} = \boldsymbol{0}$ (no hay error en la matriz) podemos reemplazar $\lesssim$ con $\leq$ y queda

$$\frac{\|\Delta \boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leq \kappa(\boldsymbol{A}) \frac{\|\Delta \boldsymbol{b}\|}{\|\boldsymbol{b}\|}.$$

De hecho, escribiendo $\boldsymbol{A}^{-1}\boldsymbol{b} = \boldsymbol{x}$, y $\boldsymbol{A}^{-1}(\boldsymbol{b} + \Delta \boldsymbol{b}) = \boldsymbol{x} + \Delta \boldsymbol{x}$ se puede argumentar con $\boldsymbol{A}^{-1}$ y por (6) de la Observación **??**, se ve que

$$\frac{\|\Delta \boldsymbol{b}\|}{\|\boldsymbol{b}\|} \leq \kappa(\boldsymbol{A}) \frac{\|\Delta \boldsymbol{x}\|}{\|\boldsymbol{x}\|}.$$

Resumiendo

$$\frac{1}{\kappa(\boldsymbol{A})} \frac{\|\Delta \boldsymbol{b}\|}{\|\boldsymbol{b}\|} \leq \frac{\|\Delta \boldsymbol{x}\|}{\|\boldsymbol{x}\|} \leq \kappa(\boldsymbol{A}) \frac{\|\Delta \boldsymbol{b}\|}{\|\boldsymbol{b}\|}.$$

EJERCICIO 3.6. Verifique con ejemplos que las desigualdades previas se alcanzan (Sug.: use matrices diagonales adecuadas).

Teniendo en cuenta el ejercicio previo, vemos que la cota superior puede alcanzarse en la práctica. En particular nos dice de un modo simplificado que podemos perder $log_{10}(\kappa(\boldsymbol{A}))$ dígitos significativos al resolver el sistema[14].

El siguiente resultado da una idea cualitativa de $\kappa(\boldsymbol{A})$ como el recíproco de la distancia relativa de $\boldsymbol{A}$ a las matrices singulares.

TEOREMA 3.1. $\boldsymbol{A} \in \mathbb{R}^{n \times n}$, se tiene,

$$\frac{1}{\kappa(\boldsymbol{A})} = \inf_{\boldsymbol{B} \text{singular}} \frac{\|\boldsymbol{A} - \boldsymbol{B}\|}{\|\boldsymbol{A}\|}, \tag{3.11}$$

y en donde asumimos $\frac{1}{\kappa(\boldsymbol{A})} = 0$ en caso de que $\boldsymbol{A}$ no sea invertible.

DEMOSTRACIÓN. Si $\boldsymbol{A}$ es no invertible el resultado es evidente. Supongamos entonces que $\boldsymbol{A}$ es invertible. Probaremos únicamente que vale el $\leq$ en (**??**).

Sea $B \in \mathbb{R}^{n \times n}$ una matriz singular cualquiera. Tomemos $\boldsymbol{0} \neq \boldsymbol{v} \in Ker(\boldsymbol{B})$, y definamos $\boldsymbol{0} \neq \boldsymbol{w} = \boldsymbol{A}\boldsymbol{v}$. Se tiene

$$0 \neq \|\boldsymbol{w}\| = \|(\boldsymbol{A} - \boldsymbol{B})\boldsymbol{v}\| \leq \|\boldsymbol{A} - \boldsymbol{B}\| \|\boldsymbol{A}^{-1}\boldsymbol{w}\| \leq \|\boldsymbol{A} - \boldsymbol{B}\| \|\boldsymbol{A}^{-1}\| \|\boldsymbol{w}\|,$$

dividiendo por $\|\boldsymbol{A}\| \|\boldsymbol{A}^{-1}\| \|\boldsymbol{w}\| \neq 0$ se obtiene

$$\frac{1}{\kappa(\boldsymbol{A})} \leq \frac{\|\boldsymbol{A} - \boldsymbol{B}\|}{\|\boldsymbol{A}\|},$$

y por ser $\boldsymbol{B}$ arbitrario

$$\frac{1}{\kappa(\boldsymbol{A})} \leq \inf_{\boldsymbol{B} \text{singular}} \frac{\|\boldsymbol{A} - \boldsymbol{B}\|}{\|\boldsymbol{A}\|},$$

□

[14]Por ejemplo, si $\kappa(\boldsymbol{A}) \sim 10^{16}$ no tenemos la garantía de tener ni un dígito correcto en nuestro resultado.

El resultado anterior dice que un número condición grande significa cercanía -relativa- a las matrices singulares. Es importante remarcar que el determinante no es la herramienta adecuada para medir esa cercanía, pues el ítem (2) de la Observación **??** dice que el número de condición no cambia por múltiplos de una matriz sin embargo $\det(\alpha \boldsymbol{A}) = \alpha^n \det(\boldsymbol{A})$ que tiende a cero si $\alpha \to 0$.

Capítulo 4

Métodos Directos Para Sistemas Lineales

1. Sistemas y Factorización de Matrices

Comenzaremos asumiendo que trabajaremos con matrices cuadradas, i.e. $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ (o mas en general $\mathbb{K}^{n\times n}$ ya que la mayoría de las consideranciones se aplican a los complejos).

Recordemos que los vectores $\boldsymbol{v} \in \mathbb{K}^n$ los identificamos con matrices columna de $n \times 1$ de donde tiene sentido $\boldsymbol{Av}$.

Factorizar siginifica descomponer una expresión en factores de algún modo mas simples o que nos otorguen algún beneficio teórico o práctico. La factorización LU consiste en escribir una matriz invertible $\boldsymbol{A}$ como producto de dos factores: uno triangular inferior $\boldsymbol{L}$ y otro triangular superior $\boldsymbol{U}$, es decir

$$\boldsymbol{L} = \begin{pmatrix} * & 0 & \cdots & 0 \\ * & * & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ * & * & \cdots & * \end{pmatrix}, \boldsymbol{U} = \begin{pmatrix} * & * & \cdots & * \\ 0 & * & \cdots & * \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & * \end{pmatrix}$$

y

$$\boldsymbol{A} = \boldsymbol{LU}.$$

Un beneficio inmediato de esta factorización es que para resolver un sistema

$$\boldsymbol{Ax} = \boldsymbol{b},$$

basta resolver

$$\boldsymbol{Ly} = \boldsymbol{b},$$
$$\boldsymbol{Ux} = \boldsymbol{y},$$

y cada sistema triangular puede resolverse por sustitución (regresiva o progresiva) en $\sim n^2$ operaciones[1].

Ejercicio 4.1. Escriba un algoritmo para resolver sistemas triangulares con orden n^2.

2. Descomposición $\boldsymbol{LU} = \boldsymbol{A}$

La factorización LU es un subproducto del método de eliminación de Gauss. Dada la matriz

$$\boldsymbol{A} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix}$$

[1]De todos modos el costo de la factorización es de orden $\sim \frac{2}{3}n^3$ como veremos mas adelante.

y suponiendo que $a_{1,1}$, denominado *pivot*, es no nulo es posible operar sobre las filas de $\boldsymbol{A}$ para generar ceros debajo del elemento $a_{1,1}$. Esto se hace fila a fila multiplicando la fila 1 de $\boldsymbol{A}$ por $-\frac{a_{i,1}}{a_{1,1}}$ (lo cual es posible ya que hemos asumido $a_{1,1} \neq 0$) y sumando ese resultado a la fila i. En términos matriciales, eso equivale a multiplicar a izquierda la matriz $\boldsymbol{A}$ por el multiplicador $\boldsymbol{M}_1$

$$\boldsymbol{M}_1 = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ -\frac{a_{2,1}}{a_{1,1}} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -\frac{a_{n,1}}{a_{1,1}} & 0 & \cdots & 1 \end{pmatrix}$$

El efecto neto del producto resulta

$$\boldsymbol{M}_1\boldsymbol{A} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ 0 & a_{2,2}^{(1)} & a_{2,3}^{(1)} & \cdots & a_{2,n}^{(1)} \\ 0 & a_{3,2}^{(1)} & a_{3,3}^{(1)} & \cdots & a_{3,n}^{(1)} \\ \vdots & \vdots & \ddots & \vdots & \\ 0 & a_{n,2}^{(1)} & a_{n,3}^{(1)} & \cdots & a_{n,n}^{(1)} \end{pmatrix}$$

donde se destaca con un superíndice (1) una nueva matriz $\boldsymbol{A}^{(1)} \in \mathbb{R}^{(n-1)\times(n-1)}$

$$\boldsymbol{A}^{(1)} = \begin{pmatrix} a_{2,2}^{(1)} & \cdots & a_{2,n}^{(1)} \\ \vdots & \vdots & \ddots \\ a_{n,2}^{(1)} & \cdots & a_{n,n}^{(1)} \end{pmatrix}$$

que en caso de tener el pivot $a_{2,2}^{(1)} \neq 0$ admite un nuevo paso en la iteración tomando

$$\boldsymbol{M}_2 = \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ 0 & -\frac{a_{3,2}^{(1)}}{a_{2,2}^{(1)}} & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \\ 0 & -\frac{a_{n,2}^{(1)}}{a_{2,2}^{(1)}} & 0 & \cdots & 1 \end{pmatrix}$$

de donde

$$\boldsymbol{M}_2\boldsymbol{M}_1\boldsymbol{A} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ 0 & a_{2,2}^{(1)} & a_{2,3}^{(1)} & \cdots & a_{2,n}^{(1)} \\ 0 & 0 & a_{3,3}^{(2)} & \cdots & a_{3,n}^{(2)} \\ \vdots & \vdots & \ddots & \vdots & \\ 0 & 0 & a_{n,3}^{(2)} & \cdots & a_{n,n}^{(2)} \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ 0 & a_{2,2}^{(1)} & a_{2,3}^{(1)} & \cdots & a_{2,n}^{(1)} \\ 0 & 0 & & & \\ \vdots & \vdots & & \boldsymbol{A}^{(2)} & \\ 0 & 0 & & & \end{pmatrix},$$

con $\boldsymbol{A}^{(2)} \in \mathbb{R}^{(n-2)\times(n-2)}$. Este procedimiento, dado que cada $\boldsymbol{M}_i$ es triangular, se denomina *triangulación por matrices triangulares*. Si algoritmo no se detiene, es decir que cada elemento pivot $a_{k+1,k+1}^{(k)} \neq 0$, para todo $1 \leq k \leq n-1$, se obtiene una matriz triangular superior $\boldsymbol{U}$

invertible

$$\boldsymbol{M}_{n-1}\boldsymbol{M}_{n-2}\cdots\boldsymbol{M}_1\boldsymbol{A} = \boldsymbol{U} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} & \cdots & a_{1,n} \\ 0 & a_{2,2}^{(1)} & a_{2,3}^{(1)} & a_{2,4}^{(1)} & \cdots & a_{2,n}^{(1)} \\ 0 & 0 & a_{3,3}^{(2)} & a_{3,4}^{(2)} & \cdots & a_{3,n}^{(2)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & a_{n,n}^{(n-1)} \end{pmatrix}$$

Observemos que la productoria

$$\boldsymbol{M}_{n-1}\cdots\boldsymbol{M}_2\boldsymbol{M}_1 = \boldsymbol{M},$$

es una matriz triangular inferior, dado que es producto de matrices triangulares inferiores. La inversa de $\boldsymbol{M}$, es por lo tanto triangular inferior. La llamaremos $\boldsymbol{L} = \boldsymbol{M}^{-1}$ y por ende

$$\boldsymbol{A} = \boldsymbol{LU}.$$

El problema que aparece inmediatamente es que no hemos obtenido $\boldsymbol{L}$ constructivamente (en el sentido de que aún falta invertir $\boldsymbol{M}$). Sin embargo, un hecho notable facilita ese trabajo.

Observemos que cada $\boldsymbol{M}_i$ tiene la forma

$$\boldsymbol{M}_i = \boldsymbol{I} - \boldsymbol{z}_i\boldsymbol{e}_i^T, \tag{4.1}$$

donde $\boldsymbol{e}_i$ representa el $i-$ésimo canónico y $\boldsymbol{z}_i$ es de la forma

$$\boldsymbol{z}_i = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ z_{i+1}^{(i)} \\ \vdots \\ z_n^{(i)} \end{pmatrix},$$

con $z_j^{(i)} = \frac{a_{j,i}^{(i-1)}}{a_{i,i}^{(i-1)}}$, $i+1 \leq j \leq n$.

LEMA 4.1. Sea $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n$, tales que $\boldsymbol{v}^T\boldsymbol{u} \neq 1$ entonces $\boldsymbol{I} - \boldsymbol{uv}^T$ es invertible y además

$$(\boldsymbol{I} - \boldsymbol{uv}^T)^{-1} = \boldsymbol{I} + \frac{\boldsymbol{uv}^T}{1 - \boldsymbol{v}^T\boldsymbol{u}}.$$

DEMOSTRACIÓN.

$$(\boldsymbol{I} - \boldsymbol{uv}^T)(\boldsymbol{I} + \frac{\boldsymbol{uv}^T}{1 - \boldsymbol{v}^T\boldsymbol{u}}) = \boldsymbol{I} + \boldsymbol{uv}^T(\frac{1}{1 - \boldsymbol{v}^T\boldsymbol{u}} - 1) - \frac{\boldsymbol{uv}^T\boldsymbol{uv}^T}{1 - \boldsymbol{v}^T\boldsymbol{u}} = \boldsymbol{I}.$$

□

Considerando el lema anterior y observando que en (**??**) se tiene $\boldsymbol{e}_i^T\boldsymbol{z}_i = 0$, resulta [2]

$$\boldsymbol{M}_i^{-1} = \boldsymbol{I} + \boldsymbol{z}_i\boldsymbol{e}_i^T,$$

[2]Es decir que el costo de invertir $\boldsymbol{M}_i$ es casi nulo (apenas un cambio de signos debajo de la diagonal).

de donde

$$\boldsymbol{L} = \boldsymbol{M}_1^{-1}\boldsymbol{M}_2^{-1}\cdots\boldsymbol{M}_{(n-1)}^{-1} = (\boldsymbol{I} + \boldsymbol{z}_1\boldsymbol{e}_1^T)(\boldsymbol{I} + \boldsymbol{z}_2\boldsymbol{e}_2^T)\cdots(\boldsymbol{I} + \boldsymbol{z}_{n-1}\boldsymbol{e}_{n-1}^T).$$

Ademas de la sencillez en el cómputo de $\boldsymbol{M}_i^{-1}$ hay otro hecho "afortunado" que permite expresar $\boldsymbol{L}$ sin cálculos adicionales. Como $\boldsymbol{e}_i^T\boldsymbol{z}_k = 0$ no solo para $k = i$ sino para todo $i \leq k \leq n$, se puede desarrollar la expresión para $\boldsymbol{L}$ y obtener

$$\boldsymbol{L} = \boldsymbol{I} + \sum_{i=1}^{n-1} \boldsymbol{z}_i\boldsymbol{e}_i^T,$$

o dicho de otra forma,

$$\boldsymbol{L} = \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ \frac{a_{2,1}}{a_{1,1}} & 1 & \cdots & 0 & 0 \\ \frac{a_{3,1}}{a_{1,1}} & \frac{a_{3,2}^{(1)}}{a_{2,2}^{(1)}} & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & 1 & \vdots \\ \frac{a_{n,1}}{a_{1,1}} & \frac{a_{n,2}^{(1)}}{a_{2,2}^{(1)}} & \cdots & \frac{a_{n,n-1}^{(n-1)}}{a_{n-1,n-1}^{(n-1)}} & 1 \end{pmatrix},$$

i.e., basta con almacenar los multiplicadores en sus respectivos lugares cambiando los signos.

Un supuesto básico para que la descomposición LU pueda llevarse a cabo es que todos los pivots sean no nulos. Eso no puede garantizarse bajo la única hipótesis de que $det(\boldsymbol{A}) \neq 0$ como se ve en tomando por ejemplo

$$\boldsymbol{A} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Sin embargo puede garantizarse si se requiere que todos lo menores de la matriz $\boldsymbol{A}$, sean invertibles.

PROPOSICIÓN 4.1. Sea $\boldsymbol{A} \in \mathbb{K}^{n\times n}$, entonces existe factorización LU de $\boldsymbol{A}$ con $\boldsymbol{L}, \boldsymbol{U}$ *invertibles* sí y solo sí para todo $1 \leq k \leq n$, $\det(\boldsymbol{A}(1:k, 1:k)) \neq 0$.

DEMOSTRACIÓN. Veamos primero la implicación $\Leftarrow$:

Como $a_{11} = A(1,1)$ resulta $a_{11} \neq 0$ y el primer pivot es no nulo. LLevado adelante el proceso de eliminación, vemos que se puede escribir

$$\boldsymbol{A} = \boldsymbol{M}_1^{-1}\boldsymbol{U}_1$$

donde

$$\boldsymbol{M}_1^{-1}\boldsymbol{U}_1 = \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ \frac{a_{2,1}}{a_{1,1}} & 1 & \cdots & 0 & 0 \\ \frac{a_{3,1}}{a_{1,1}} & 0 & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & 1 & \vdots \\ \frac{a_{n,1}}{a_{1,1}} & 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ 0 & a_{2,2}^{(1)} & a_{2,3}^{(1)} & \cdots & a_{2,n}^{(1)} \\ 0 & 0 & & & \\ \vdots & \vdots & & \boldsymbol{A}^{(2)} & \\ 0 & 0 & & & \end{pmatrix}$$

Por lo tanto

$$\boldsymbol{A}(1:2,1:2) = \begin{pmatrix} 1 & 0 \\ \frac{a_{2,1}}{a_{1,1}} & 1 \end{pmatrix} \begin{pmatrix} a_{1,1} & a_{1,2} \\ 0 & a_{2,2}^{(1)} \end{pmatrix}$$

y como por hipótesis $0 \neq det(\boldsymbol{A}(1:2,1:2))$ tenemos $det(\boldsymbol{A}(1:2,1:2)) = a_{1,1}a_{2,2}^{(1)} \neq 0$, por lo tanto el segundo pivot no se anula. La demostración se sigue por inducción.

Para la implicación $\Rightarrow$: Notamos que si $\boldsymbol{A} = \boldsymbol{LU}$ con $\boldsymbol{L}$ y $\boldsymbol{U}$ invertibles, deben ser $l_{kk} \neq 0 \neq u_{kk}$ para todo $1 \leq k \leq n$ ya que son triangulares. En particular $\det(L(1:k,1:k) \neq 0 \neq \det(U(1:k,1:k))$ y por ser una de ellas triangular inferior y la otra superior, resulta $A(1:k,1:k) = L(1:k,1:k)U(1:k,1:k)$ de donde $\det(A(1:k,1:k)) \neq 0$. □

Para ver otro criterio necesitamos una definición.

DEFINICIÓN 4.1. Una matriz $\boldsymbol{A} \in \mathbb{R}^{n\times n}, \mathbb{C}^{n\times n}$ se dice *diagonal dominante* (*estrictamente diagonal dominante*) y se denota DD (EDD) si y solo si para todo i, $1 \leq i \leq n$

$$\sum_{1\leq j\leq n, j\neq i} |a_{i,j}| \leq (<)|a_{i,i}|.$$

Y la siguiente proposición (que puede refinarse en algunos casos como veremos mas adelante).

PROPOSICIÓN 4.2. Sea $\boldsymbol{A} \in \mathbb{R}^{n\times n}, \mathbb{C}^{n\times n}$ EDD entonces $\boldsymbol{A}$ es invertible.

DEMOSTRACIÓN. Supongamos que no. Existe entonces $\boldsymbol{0} \neq \boldsymbol{v} \in \mathbb{C}^n$ tal que $\boldsymbol{Av} = \boldsymbol{0}$. Sea i, $1 \leq i \leq n$ tal que $0 \neq |v_i| = \|\boldsymbol{v}\|_\infty$, entonces

$$\sum_{j=1}^{n} a_{i,j}v_j = 0,$$

de donde

$$|a_{i,i}| \leq \sum_{j=1, j\neq i}^{n} |a_{i,j}| \frac{|v_j|}{|v_i|} \leq \sum_{j=1, j\neq i}^{n} |a_{i,j}|,$$

lo que contradice la EDD. □

PROPOSICIÓN 4.3. Si $\boldsymbol{A}$ es EDD entonces, trabajando con aritmética exacta, el proceso de eliminación de Gauss no produce pivots nulos.

DEMOSTRACIÓN. Ejercicio. □

OBSERVACIÓN 4.1. Las matrices DD no son una artificio teórico, aparecen frecuentemente en la práctica como por ejemplo en la discretización de ecuaciones diferenciales.

Aún en el caso en que no aparezcan pivots nulos, el algoritmo de eliminación no delvoverá en general los verdaderos factores $\boldsymbol{L}$ y $\boldsymbol{U}$, sino una aproximación de ellos. Mas explícitamente podemos enunciar el siguiente teorema (ver [**?**]).

TEOREMA 4.1. Sea $\boldsymbol{A} \in \mathbb{K}^{n\times n}$, si el algoritmo LU no produce pivots nulos,la factorización LU producida por la máquina verifica

$$\tilde{\boldsymbol{L}}\tilde{\boldsymbol{U}} = \boldsymbol{A} + \Delta\boldsymbol{A}$$

donde $|\Delta\boldsymbol{A}| \lesssim 2(n-1)\varepsilon(|\boldsymbol{A}| + |\tilde{\boldsymbol{L}}||\tilde{\boldsymbol{U}}|)$.

OBSERVACIÓN 4.2. El Teorema **??**, sugiere controlar el tamaño de los factores $\tilde{\boldsymbol{L}}, \tilde{\boldsymbol{U}}$. En efecto, si el producto $|\tilde{\boldsymbol{L}}||\tilde{\boldsymbol{U}}|$ es muy grande no tendremos control (al menos teórico) sobre el error. El *pivoteo parcial* permite garantizar al menos el control $|\boldsymbol{L}| \leq 1$ a una costo razonable. Si bien, esto no necesariamente implica que $|\tilde{\boldsymbol{L}}||\tilde{\boldsymbol{U}}|$ sea pequeño, resulta suficiente en la mayoría de los casos prácticos (ver sin embargo la Observación **??**).

EJERCICIO 4.2. Muestre el el número de operaciones necesarias para realizar la factorización LU es $\mathcal{O}(\frac{2n^3}{3})$

El número de elementos de una matriz de $n \times n$ es obviamente n^2. El número mínimo de operaciones que un método de resolución puede efectuar debe ser de orden mayor o igual a n^2. Todos los métodos directos clásicos son de orden cúbico. Esto presenta problemas para matrices muy grandes ya que en ciertas aplicaciones es posible tener un gran número de incógnitas.

Otro problema importante es el *rellenado* que produce el algoritmo. Esto es, aunque $\boldsymbol{A}$ sea *rala* -cosa que afortunadamente ocurre en muchas aplicaciones, como en ecuaciones diferenciales- los factores $\boldsymbol{L}, \boldsymbol{U}$ pueden no serlo.

EJERCICIO 4.3. Cuanta memoria ocuparía una matriz de $10^6 \times 10^6$ llena?.

3. Descomposición $\boldsymbol{LU} = \boldsymbol{PA}$ (pivoteo parcial)

Una matriz de permutaciones $\boldsymbol{P} \in \mathbb{K}^{n\times n}$ es una matriz identidad con las filas permutadas. Alternativamente puede verse como una matriz cuyas filas están conformadas por los elementos de la base canónica no necesariamente ordenados. Es decir,

$$\boldsymbol{P} = \begin{pmatrix} \boldsymbol{e}_{i_1}^T \\ - - - - \\ \boldsymbol{e}_{i_2}^T \\ \vdots \\ - - - - - \\ \boldsymbol{e}_{i_n}^T \end{pmatrix}$$

donde $i_1, i_2, \cdots i_n \in \{1, 2, \cdots n\}$ son todos diferentes. Alternativamente podemos pensar una matriz de permutaciones como la identidad con las columnas permutadas

$$\boldsymbol{P} = \begin{pmatrix} \boldsymbol{e}_{i_1} & | & \boldsymbol{e}_{i_2} & | & \cdots | \boldsymbol{e}_{i_n} \end{pmatrix}.$$

Notemos que si

$$\boldsymbol{v} \in \mathbb{K}^n,$$

$$\boldsymbol{Pv} = \begin{pmatrix} v_{i_1} \\ v_{i_2} \\ \vdots \\ v_{i_n} \end{pmatrix}$$

$$\boldsymbol{v}^T \boldsymbol{P} = \left(v_{i_1}, v_{i_2}, \cdots v_{i_n}\right).$$

Resumamos las siguientes propiedades de las permutaciones.

- Dada $\boldsymbol{A} \in \mathbb{K}^{n\times n}$, $\boldsymbol{PA}$ coincide con $\boldsymbol{A}$ pero con las filas permutadas y $\boldsymbol{AP}$ coincide con $\boldsymbol{A}$ pero con las columnas permutadas.
- Dada $\boldsymbol{x} \in \mathbb{K}^n$, $\boldsymbol{Px}$ coincide con $\boldsymbol{x}$ pero con los elementos permutados.
- $\det(\boldsymbol{P}) = \pm 1$
- $\boldsymbol{P}^{-1} = \boldsymbol{P}^T$
- Si $\boldsymbol{P}_1$ y $\boldsymbol{P}_2$ son permutaciones entonces $\boldsymbol{P} = \boldsymbol{P}_1\boldsymbol{P}_2$ también lo es.

Si durante la eliminación de Gauss hallamos un pivot nulo, podemos intercambiar filas para continuar con el algoritmo (siempre que haya algún elemento no nulo con el cual proseguir). Mas aún, aunque el correspondiente pivot sea no nulo, podemos, casi al mismo costo, tomar el pivot mas grande (en valor absoluto). Esa operación la podemos representar matricialmente multiplicando por una adecuada permutación.

El procedimiento sería el siguiente: tomemos el elemento con mayor valor absoluto en la primera columna de $\boldsymbol{A}$. Digamos que este es $a_{k,1}$ (si $\det(\boldsymbol{A}) \neq 0$ se tiene que $a_{k,1} \neq 0$). Permutemos la fila k con la fila 1, lo cual se hace con una matriz $\boldsymbol{P}_1$. Es decir

$$\boldsymbol{P}_1\boldsymbol{A} = \begin{pmatrix} a_{k,1} & a_{k,2} & \cdots & a_{k,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix},$$

a partir de aquí construimos el multiplicador, $\boldsymbol{M}_1$ de modo tal que

$$\boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{A} = \begin{pmatrix} a_{k,1} & a_{k,2} & \cdots & a_{k,n} \\ 0 & & & \\ \vdots & & \boldsymbol{A}^{(1)} & \\ 0 & & & \end{pmatrix}.$$

Llegado a este punto, repetimos el procedimiento con la primer columna de $\boldsymbol{A}^{(1)}$ que eventualmente requerirá otra matriz $\boldsymbol{P}_2$ antes de la construcción del nuevo multiplicador $\boldsymbol{M}_2$. En definitiva,

$$\boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1}\cdots\boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{A} = \boldsymbol{U}, \tag{4.2}$$

se obtiene una nueva matriz triangular superior $\boldsymbol{U}$, y donde $\boldsymbol{M}_i = \boldsymbol{I} - \boldsymbol{z}_i \boldsymbol{e}_i^T$ con

$$\boldsymbol{z}_i = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ z_{i+1}^{(i)} \\ \vdots \\ z_n^{(i)} \end{pmatrix},$$

$|\boldsymbol{z}_i| \leq 1$ (i.e. los multplicadores son menores o iguales a 1 en valor absoluto). El problema ahora es que en la forma (**??**) no parece verse el modo de reconstruir los factores $\boldsymbol{L}$ y $\boldsymbol{P}$ de manera sencilla. Sin embargo un hecho inesperadamente favorable resuelve esta cuestión. Notemos que por el orden de las iteraciones en el paso k solo permutaremos en busca del mayor pivot (en caso de ser necesario) filas desde la k a la n porque las filas previas ya han sido procesadas. Eso indica que $\boldsymbol{e}_j^T \boldsymbol{P}_k = \boldsymbol{e}_j^T$ para todo $j < k$. En particular, si bien *no es cierto* que $\boldsymbol{P}_k$ conmuta con $\boldsymbol{M}_j$ sí es cierto, *para todo* $j < k$, que

$$\boldsymbol{P}_k \boldsymbol{M}_j = \boldsymbol{P}_k(\boldsymbol{I} - \boldsymbol{z}_j \boldsymbol{e}_j^T) = (\boldsymbol{I} - \boldsymbol{P}_k \boldsymbol{z}_j \boldsymbol{e}_j^T)\boldsymbol{P}_k = \tilde{\boldsymbol{M}}_j \boldsymbol{P}_k,$$

es decir que podemos pasar la permutación a la derecha permutando adecuadamente los multiplicadores de la matriz $\boldsymbol{M}_j$. Lo importante es que esta nueva matriz, denotada con $\tilde{\boldsymbol{M}}_j$ tiene la misma estructura que $\boldsymbol{M}_j$. Así que volviendo a (**??**) tenemos

$$\boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1} \cdots \boldsymbol{M}_1 \boldsymbol{P}_1 \boldsymbol{A} = \tilde{\boldsymbol{M}}_{n-1} \cdots \tilde{\boldsymbol{M}}_1 \boldsymbol{P}_{n-1} \cdots \boldsymbol{P}_1 \boldsymbol{A} = \boldsymbol{U}.$$

Argumentando como en el caso son pivoteo (usando que la estructura de las $\tilde{\boldsymbol{M}}_i$ es igual a la de las de las $\boldsymbol{M}_i$) y llamando $\boldsymbol{P} = \boldsymbol{P}_{n-1} \cdots \boldsymbol{P}_1$ se llega a

$$\boldsymbol{P}\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}.$$

PROPOSICIÓN 4.4. Trabajando con aritmética exacta se tiene que si $det(\boldsymbol{A}) \neq 0$ el algoritmo de pivoteo parcial no tiene pivots nulos.

OBSERVACIÓN 4.3. El proceso de pivoteo parcial garantiza que $|\boldsymbol{L}| \leq 1$ pero no se tiene un control demasiado beningno sobre el tamaño de $|\boldsymbol{U}|$, como se ve en este ejemplo

$$\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 1 \\ -1 & 1 & 0 & \cdots & 0 & 1 \\ -1 & 0 & 1 & 0 & \cdots & 1 \\ \vdots & \vdots & \vdots & \ddots & & \vdots \\ -1 & 0 & 0 & 0 & \cdots & 1 \end{pmatrix},$$

puede verse que $\|\boldsymbol{U}\|_\infty \geq 2^n \|\boldsymbol{A}\|_\infty$. Este ejemplo sin embargo parece ser singular en el sentido de que en los casos genéricos el crecimiento de $|\boldsymbol{U}|$ estaría controlado por $\sqrt{n}$ [**?**] [a]. Esta particularidad parece (junto con el hecho de que su costo es la mitad que el de la factorización QR que veremos mas adelante) haber mantenido el algoritmo de eliminación como uno de los más populares entre los métodos directos.

[a] Al presente no parece haber una demostración rigurosa de esto.

Controlar el tamaño del ambos factores $\boldsymbol{L}$ y $\boldsymbol{U}$ es posible. La idea se denomina *pivoteo completo*, esto implica hacer permutaciones de filas y de columnas para tomar siempre el mayor pivot posible. Así en el paso inicial se busca el elemento $a_{k,l}$ con máximo valor absoluto y se permutan la fila y columna 1 con la fila y columna k y l respectivamente. En términos matriciales eso requiere de dos matrices de permutaciones $\boldsymbol{P}_1, \tilde{\boldsymbol{P}}_1$ antes de aplicar el paso de eliminación. Es decir

$$\boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{A}\tilde{\boldsymbol{P}}_1 = \begin{pmatrix} * & * & \cdots & * \\ 0 & & & \\ \vdots & & \boldsymbol{A}^{(1)} & \\ 0 & & & \end{pmatrix}.$$

Repitiendo el procedimiento se obtiene $\boldsymbol{U}$

$$\boldsymbol{M}_{n-1}\boldsymbol{P}_{n-1}\cdots\boldsymbol{M}_1\boldsymbol{P}_1\boldsymbol{A}\tilde{\boldsymbol{P}}_1\cdots\tilde{\boldsymbol{P}}_{n-1} = \boldsymbol{U},$$

y argumentando como antes se obtienen ahora dos matrices de permutaciones $\boldsymbol{P}, \tilde{\boldsymbol{P}}$ tales que

$$\boldsymbol{P}\boldsymbol{A}\tilde{\boldsymbol{P}} = \boldsymbol{L}\boldsymbol{U}.$$

El pivoteo completo no se lleva a cabo en general debido a su alto costo (calcule el número de comparaciones que debe realizar en cada) ya que hay otros métodos estables mas económicos. Wilkinson probó lo siguiente

TEOREMA 4.2. En aritmética exacta, para el algoritmo de pivoteo completo se tiene
$$\|\boldsymbol{U}\|_\infty \leq \sqrt{n}(2.3^{1/2}.4^{1/3}\cdots n^{1/(n-1)})^{1/2}\|\boldsymbol{A}\|_\infty$$

EJERCICIO 4.4. Estimar numéricamente el crecimiento del factor $\sqrt{n}(2.3^{1/2}.4^{1/3}\cdots n^{1/(n-1)})^{1/2}$.

4. Descomposición $\boldsymbol{LDL}^*$: Cholesky

En muchos casos es posible explotar la estructura de las matrices a la hora de resolver sistemas. Dicho sea esto a la hora de ahorrar computos o mejorar la estabilidad de los algoritmos.

PROPOSICIÓN 4.5. Si $\boldsymbol{A}$ es definida positiva entonces es invertible y también lo son todos sus menores.

DEMOSTRACIÓN. En efecto, de no ser invertible, existiría $\boldsymbol{0} \neq \boldsymbol{v}$ tal que $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{0}$ lo que contradice la definida positividad ya que para ese $\boldsymbol{v}$ se tendría $\boldsymbol{v}^*\boldsymbol{A}\boldsymbol{v} = 0$.

Respecto de los menores el resultado se sigue del anterior. En efecto, sea k fijo, $1 \leq k \leq n$ y tomemos vectores $\boldsymbol{v} \in \mathbb{K}^n$ de la forma $(\tilde{\boldsymbol{v}}, 0, \cdots, 0)$ con $\tilde{\boldsymbol{v}} \in \mathbb{K}^k$ arbitrario. Resulta

$$0 < \boldsymbol{v}^*\boldsymbol{A}\boldsymbol{v} = \tilde{\boldsymbol{v}}^*\boldsymbol{A}(1:k, 1:k)\tilde{\boldsymbol{v}},$$

y de ahí la definida positividad de $\boldsymbol{A}(1:k, 1:k)$ y por ende su invertibilidad. □

Si $\boldsymbol{A}$ es definida positiva admite una descomposición $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}$ (gracias a las Proposiciones **??** y **??**) y podemos escribir $\boldsymbol{U} = \boldsymbol{D}\tilde{\boldsymbol{L}}^*$, con $\tilde{\boldsymbol{L}}$ triangular inferior con 1 en la diagonal y $\boldsymbol{D}$ una matriz diagonal. De ahí resulta

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{D}\tilde{\boldsymbol{L}}^* = \tilde{\boldsymbol{L}}\boldsymbol{D}^*\boldsymbol{L}^*,$$

entonces

$$\boldsymbol{D}(\boldsymbol{L}^{-1}\tilde{\boldsymbol{L}})^* = \boldsymbol{L}^{-1}\tilde{\boldsymbol{L}}\boldsymbol{D}^*,$$

y como el lado izquierdo de esta ecuación es triangular superior y el derecho triangular inferior deberán ser ambos diagonales. Luego, debe serlo también $\boldsymbol{L}^{-1}\tilde{\boldsymbol{L}}$. Por construcción, $\tilde{\boldsymbol{L}}_{i,i} = 1 = \boldsymbol{L}_{i,i}$, para todo $1 \leq i \leq n$. Debido a la primera igualdad se tiene en particular que $\boldsymbol{L}_{i,i}^{-1} = 1$. De aquí resulta $\boldsymbol{L}^{-1}\tilde{\boldsymbol{L}} = \boldsymbol{I}$, es decir $\boldsymbol{L} = \tilde{\boldsymbol{L}}$ y además $\boldsymbol{D} = \boldsymbol{D}^*$. En particular se obtiene la factorización

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^*.$$

Como además $\boldsymbol{A}$ es definida positiva, entonces $\boldsymbol{D} > 0$, pues para todo $\boldsymbol{v} \neq 0$

$$0 < \boldsymbol{v}^*\boldsymbol{A}\boldsymbol{v} = \boldsymbol{v}^*\boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^*\boldsymbol{v} = (\boldsymbol{L}^*\boldsymbol{v})^*\boldsymbol{D}\boldsymbol{L}^*\boldsymbol{v},$$

de donde, llamando $\boldsymbol{w} = \boldsymbol{L}^*\boldsymbol{v}$ y usando que $\boldsymbol{L}$ (y por ende $\boldsymbol{L}^*$) es invertible, resulta la definida positividad de $\boldsymbol{D}$, lo que en este caso, por ser diagonal, equivale a $\boldsymbol{D} > 0$. Esto nos garantiza la existencia de la factorización de Cholesky:

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{D}^{\frac{1}{2}}\boldsymbol{D}^{\frac{1}{2}}\boldsymbol{L}^* = \boldsymbol{C}\boldsymbol{C}^*,$$

donde $\boldsymbol{C}$ es triangular inferior con elementos diagonales positivos. La construcción de $\boldsymbol{C}$ puede hacerse desde la factorización LU, sin embargo ese modo mas costoso y menos estable que el algoritmo que describiremos debajo. Antes, sin embargo, veamos cómo sería la base de un algoritmo (garantizando que pueda ejecutarse sin interrupciones).

Asociemos el algoritmo con una función $chol()$[3]. Es decir $chol(\boldsymbol{A})$ devuelve el factor $\boldsymbol{C}$.

Ya que $\boldsymbol{A} = \boldsymbol{C}\boldsymbol{C}^*$, con $\boldsymbol{C}$ triangular inferior $c_{i,i} > 0$, (de hecho, ya probamos la existencia de esta factorización). Intentaremos "despejar" $\boldsymbol{C}$.

Escribimos

$$\boldsymbol{C} = \begin{pmatrix} c_{1,1} & | & \boldsymbol{0} \\ - & - & - \\ \boldsymbol{C}(2:n,1) & | & \boldsymbol{C}(2:n,2:n) \end{pmatrix}$$

donde $\boldsymbol{C}(2:n,2:n)$ tiene las mismas características que $\boldsymbol{C}$. Sabiendo que

$$\begin{pmatrix} a_{1,1} & | & \boldsymbol{A}(2:n,1)^* \\ - & - & - \\ \boldsymbol{A}(2:n,1) & | & \boldsymbol{A}(2:n,2:n) \end{pmatrix} = \tag{4.3}$$

$$\begin{pmatrix} c_{1,1} & | & \boldsymbol{0} \\ - & - & - \\ \boldsymbol{C}(2:n,1) & | & \boldsymbol{C}(2:n,2:n) \end{pmatrix} \begin{pmatrix} c_{1,1} & | & \boldsymbol{C}(2:n,1)^* \\ - & - & - \\ \boldsymbol{0} & | & \boldsymbol{C}(2:n,2:n)^* \end{pmatrix}$$

se tiene, respectivamente, que

$$\textbf{Paso 1} \qquad c_{1,1}^2 = a_{1,1} \qquad \rightarrow \qquad c_{1,1} := \sqrt{a_{1,1}},$$

$$\textbf{Paso 2} \quad \boldsymbol{A}(2:n,1) = c_{1,1}\boldsymbol{C}(2:n,1) \qquad \rightarrow \qquad \boldsymbol{C}(2:n,1) := \boldsymbol{A}(2:n,1)/c_{1,1},$$

lo cual nos permitió "despejar" las incógnitas $\boldsymbol{C}(1:n,1)$, puesto que $a_{1,1} > 0$ por hipótesis, lo que permite elegir $c_{1,1} > 0$. Ahora consideramos

$$\textbf{Paso 3} \quad \boldsymbol{A}(2:n,2:n) = \boldsymbol{C}(1:n,1)\boldsymbol{C}(1:n,1)^* + \boldsymbol{C}(2:n,2:n)\boldsymbol{C}(2:n,2:n)^* \rightarrow$$
$$\boldsymbol{C}(2:n,2:n) := chol(\boldsymbol{A}(2:n,2:n) - \boldsymbol{C}(1:n,1)\boldsymbol{C}(1:n,1)^*),$$

[3]En Python *np.linalg.cholesky*() en Matlab *chol*().

de donde el problema de resolver $chol(\boldsymbol{A}(1:n,1:n))$ para una matriz de $n\times n$ se redujo a resolver $chol(\boldsymbol{A}(2:n,2:n)-\boldsymbol{C}(2:n,1)\boldsymbol{C}(2:n,1)^*)$ para una matriz de tamaño $(n-1)\times(n-1)$, lo que inductivamente (siempre que la nueva matriz sea definida positiva) nos lleva al problema trivial de factorizar un número real positivo (matriz de 1×1 definida positiva) como el cuadrado de otro positivo. Veamos entonces la siguiente

PROPOSICIÓN 4.6. Si $\boldsymbol{A}$ es SDP entonces $\boldsymbol{A}(2:n,2:n)-\boldsymbol{C}(2:n,1)\boldsymbol{C}(2:n,1)^*$, donde $\boldsymbol{C}=(2:n,1)=\boldsymbol{A}(2:n,1)/\sqrt{a_{1,1}}$, es DP.

DEMOSTRACIÓN. Claramente $\boldsymbol{A}(2:n,2:n)-\boldsymbol{C}(2:n,1)\boldsymbol{C}(2:n,1)^*$ es Hermitiana (o simétrica, en el caso real). Por otro lado, y por hipótesis, para todo $\mathbf{0}\neq\boldsymbol{u}=(u_1,\boldsymbol{v})\in\mathbb{C}^n$ se tiene

$$0<\boldsymbol{u}^*\boldsymbol{A}\boldsymbol{u}=\begin{pmatrix}u_1\\-\\\boldsymbol{v}\end{pmatrix}^*\begin{pmatrix}a_{1,1}&|&\boldsymbol{A}(2:n,1)^*\\-&-&-\\\boldsymbol{A}(2:n,1)&|&\boldsymbol{A}(2:n,2:n)\end{pmatrix}\begin{pmatrix}u_1\\-\\\boldsymbol{v}\end{pmatrix}=$$

$$|u_1|^2a_{1,1}+u_1^*\boldsymbol{A}(2:n,1)^*\boldsymbol{v}+u_1\boldsymbol{v}^*\boldsymbol{A}(2:n,1)+\boldsymbol{v}^*\boldsymbol{A}(2:n,2:n)\boldsymbol{v}.$$

Elijamos $u_1=-\frac{\boldsymbol{A}(2:n,1)^*\boldsymbol{v}}{a_{1,1}}$ y resulta para todo $\boldsymbol{v}\neq\mathbf{0}$,

$$0<\boldsymbol{v}^*\left(\boldsymbol{A}(2:n,2:n)-\boldsymbol{C}(2:n,1)\boldsymbol{C}(2:n,1)^*\right)\boldsymbol{v},$$

lo que completa la demostración. □

En forma de algoritmo, Cholesky puede escribirse del siguiente modo,

for j= 1:n
  $a_{j,j}:=\sqrt{a_{j,j}}$
  for i=j+1:n
    $a_{i,j}:=a_{i,j}/a_{j,j}$
  for k=j+1:n
    for i=k:n
    $a_{i,k}:=a_{i,k}-a_{i,j}a_{k,j}$

en donde se utliliza solo la información de la parte triangular inferior de $\boldsymbol{A}$ (la parte superior es redundante) y en donde, además, se almacenan los datos de la matriz $\boldsymbol{C}$.

5. Ortogonalidad en Métdos Directos

Como hemos mencionado, el método de eliminación se basa en triangulación por matrices triangulares (los multiplicadores). El crecimiento de los multiplicadores y de la matriz triangular superior resultante podría ser problemático. Por esa razón lo ideal sería realizar operaciones con matrices de norma controlada (idealmente de norma 1). Esos algoritmos son posibles y se basan en ideas de ortogonalidad.

Dada una matriz $\boldsymbol{A} \in \mathbb{K}^{m\times n}$, con columnas $\{\boldsymbol{a}_1, \boldsymbol{a}_2, \cdots, \boldsymbol{a}_n\}$ linealmente independientes, podemos considerar los subespacios anidados generados por sus columnas:

$$< \boldsymbol{a}_1 > \subseteq < \boldsymbol{a}_1, \boldsymbol{a}_2 > \subseteq \cdots \subseteq < \boldsymbol{a}_1, \boldsymbol{a}_2, \cdots, \boldsymbol{a}_k > .^4$$

En varias aplicaciones es de interés encontrar generadores *ortonormales* de esos subespacios. Esto es, una colección $\{\boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_k\}$, tales que $\boldsymbol{q}_i^* \boldsymbol{q}_j = \delta_i^j$ y

$$\langle \boldsymbol{a}_1, \boldsymbol{a}_2, \cdots, \boldsymbol{a}_l \rangle = \langle \boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_l \rangle$$

para todo l. Matricialmente, y observando los subespacios anidados, es obvio que esto equivale a hallar una matriz $\boldsymbol{R} \in \mathbb{C}^{n\times n}$ triangular superior

$$\boldsymbol{R} = \begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} & \cdots & r_{1,n} \\ 0 & r_{2,2} & r_{2,3} & \cdots & r_{2,n} \\ 0 & 0 & r_{3,3} & \cdot & r_{3,n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & r_{n,n} \end{pmatrix}$$

tal que

$$\begin{pmatrix} | & | & | & | \\ | & | & | & | \\ \boldsymbol{a}_1 & \boldsymbol{a}_2 & \cdots & \boldsymbol{a}_n \\ | & | & | & | \\ | & | & | & | \end{pmatrix} = \begin{pmatrix} | & | & | & | \\ | & | & | & | \\ \boldsymbol{q}_1 & \boldsymbol{q}_2 & \cdots & \boldsymbol{q}_n \\ | & | & | & | \\ | & | & | & | \end{pmatrix} \begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} & \cdots & r_{1,n} \\ 0 & r_{2,2} & r_{2,3} & \cdots & r_{2,n} \\ 0 & 0 & r_{3,3} & \cdot & r_{3,n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & r_{n,n} \end{pmatrix},$$

de donde se observa que tanto los $\boldsymbol{q}_i$ (elegidos ortonomales) como los $r_{i,j}$ pueden obtenerse desde las ecuaciones

$$\begin{aligned} \boldsymbol{a}_1 &= r_{1,1}\boldsymbol{q}_1 \\ \boldsymbol{a}_2 &= r_{1,2}\boldsymbol{q}_1 + r_{2,2}\boldsymbol{q}_2 \\ &\vdots \\ \boldsymbol{a}_n &= r_{1,n}\boldsymbol{q}_1 + r_{2,n}\boldsymbol{q}_2 + \cdots r_{n,n}\boldsymbol{q}_n \end{aligned}$$

por ejemplo, a través de Gram-Schmidt. Como hemos mencionado, sin embargo, ese algoritmo no es estable[5] y suelen utilizarse en cambio métodos alternativos. Sin embargo para propósitos teóricos funciona perfectamente.

Notemos que en caso de que $\boldsymbol{Q}$ resulte cuadrada ($\boldsymbol{Q} \in \mathbb{K}^{n\times n}$), será una matriz unitaria (ortogonal en caso de ser una matriz real). Antes de continuar conviene observar ciertas propiedas de las matrices unitarias.

[4] Si las columnas son *l.i.* las inclusiones son obviamente estrictas, pero no en el caso general.

[5] Hay una versión modificada utilizando proyectores que veremos mas adelante.

Bibliografía

[1] J. W. Demmel Applied Numerical Linear Algebra SIAM, 1996.

[2] T. Eirola, O. Nevanlinna, Numerical Linear Algebra, Iterative Methods, Lecture Notes, Mat. 1.175, Institute of Mathematics, Helsinki Univ. of Technolgy, 2003.

[3] G. Golub, C.F. Van Loan, Matrix Computations, 3rd. Ed., The Johns Hopkins University Press, 1996.

[4] J. M. Ortega, Numerical Analysis: A second course. SIAM, 1990.

[5] L. N. Trefethen, D.Bau III, Numerical linear algebra, SIAM 1997.

[6] R. S. Varga, Matrix Iterative Analysis, Prentice-Hall, 1962.

[7] J. Wilkinson *The perfidious polynomial* Studies in Numerical Analysis, pp. 1-28, MAA Stud. Math., 24, 1984.

[8] R. Horn, C. Johnson *Matrix Analysis*, Cambridge University Press, 1990.

[9] S. Lang *Algebra*, Springer, 2002.