

Notas de Álgebra Lineal Computacional

Gabriel Acosta y Santiago Laplagne

Índice general

Preliminares	7
Capítulo 1. Nociones básicas de álgebra lineal	9
1.1. Vectores y matrices	9
1.2. Sistemas lineales de ecuaciones	12
1.3. Espacios vectoriales	18
1.4. Operaciones con matrices	28
1.5. Cambio de base	38
1.6. Transformaciones lineales	40
1.7. Espacios afines	43
Capítulo 2. Números de máquina y número de condición	47
2.1. Cuestiones previas	47
2.2. Bases generales*	48
2.3. Números de máquina	49
2.4. Condición y estabilidad	53
Capítulo 3. Normas y Producto Interno	57
3.1. Normas en \mathbb{K}^n	57
3.2. Producto interno	60
3.3. Normas de matrices	64
3.4. Condición de matrices	68
3.5. Proyecciones y proyectores	71
Capítulo 4. Métodos Directos Para Sistemas Lineales	77
4.1. Sistemas y Factorización de Matrices	77
4.2. Descomposición $LU = A$	77
4.3. Descomposición $LU = PA$ (pivoteo parcial)	82
4.4. Descomposición LDL^* : Cholesky	85
4.5. Ortogonalidad en Métodos Directos	87
Capítulo 5. Diagonalización	89
5.1. Motivación: Google Page Rank	89
5.2. Autovalores y autovectores	90
5.3. Diagonalización	92
5.4. Descomposición de Schur	98
5.5. Autovalores y autovectores de matrices especiales	99
5.6. El método de la potencia	102
5.7. Norma-2 de matrices	107
5.8. El algoritmo QR	111

5.9. Teorema de los círculos de Gershgorin	113
Capítulo 6. Procesos de Markov	115
6.1. Matrices de Markov	117
Capítulo 7. Métodos Iterativos	121
7.1. Un ejemplo: distribución de temperatura	121
7.2. Métodos Iterativos	124
7.3. Algunos métodos clásicos	126
7.4. Método de gradiente (o del descenso más rápido)	131
7.5. Gradiente conjugado	133
7.6. GC y aproximación polinomial	135
Capítulo 8. Descomposición en valores singulares	139
8.1. Motivación: número de condición y autovalores	139
8.2. Introducción	139
8.3. Construcción	141
8.4. Valores singulares y norma-2	143
8.5. Distancia a matrices de menor rango	144
8.6. Aplicaciones	145
Capítulo 9. Cuadrados mínimos	151
9.1. Ajuste lineal	151
9.2. Interpolación polinomial	156
9.3. Regularización	159
Bibliografía	161

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.

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 $\mathbf{v} \in \mathbb{R}^n, \mathbb{C}^n$ se identificarán con matrices *columna* de $n \times 1$. Por ende tendrá sentido $\mathbf{A}\mathbf{v}$, para toda matriz $\mathbf{A} \in \mathbb{R}^{m \times n}, \mathbb{C}^{m \times n}$.
- Con $|\mathbf{v}|$ representamos el vector con las mismas componentes de \mathbf{v} con valor absoluto. Análogamente definimos $|\mathbf{A}|$.
- Dados dos vectores (matrices) $\mathbf{v}_1, \mathbf{v}_2$ ($\mathbf{A}_1, \mathbf{A}_2$) la notación $\mathbf{v}_1 \leq \mathbf{v}_2$ ($\mathbf{A}_1 \leq \mathbf{A}_2$) debe interpretarse componente a componente (lo mismo para $<, >, \geq$).
- Dados un vector (matriz) \mathbf{v} (\mathbf{A}) y una constante c , la notación $\mathbf{v} \leq c$ ($\mathbf{A} \leq c$), indica que todas componentes de \mathbf{v} (\mathbf{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 $\mathbf{v}(2 : 5)$ indica los elementos 2 al 5 incluidos del vector \mathbf{v} ¹. La misma interpretación vale para matrices, por ejemplo, resulta válido para nosotros escribir $\mathbf{A}(2 : 10, 7 : 25)$.

¹Estamos 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

1.1. Vectores y matrices

1.1.1. Vectores. Para $n \in \mathbb{N}$, definimos un vector de n coordenadas como una sucesión de n números reales o complejos $\mathbf{v} = (v_1, \dots, v_n)$. 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 $\bar{z} = a - ib$.
3. $\forall z, w \in \mathbb{C}, \bar{z}\bar{w} = \bar{z}\bar{w}$.
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. $|z|^2 = z\bar{z}$, en particular se observa que si $z \neq 0$, $z\frac{\bar{z}}{|z|^2} = 1$. Llamamos $z^{-1} = \frac{\bar{z}}{|z|^2}$.
7. $\forall \theta \in \mathbb{R}$, se define $e^{i\theta} = \cos(\theta) + i \sin(\theta)$, 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 1.1.1.

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

EJEMPLO 1.1.2.

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

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.]
```

1.1.2. 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 1.1.3.

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

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
```

1.1.3. 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 $\mathbf{v} = (v_1, v_2, \dots, v_n)$ y $\mathbf{u} = (u_1, u_2, \dots, u_n)$,
$$\mathbf{v} + \mathbf{u} = (v_1 + u_1, v_2 + u_2, \dots, v_n + u_n).$$
- Producto de vectores o matrices por un escalar. Si $a \in \mathbb{R}$ o \mathbb{C} y $\mathbf{v} = (v_1, v_2, \dots, v_n)$,
$$a\mathbf{v} = (av_1, av_2, \dots, av_n).$$

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]
```

1.2. Sistemas lineales de ecuaciones

1.2.1. 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 sí.
- multiplicar una fila por un escalar distinto de 0.

El algoritmo de eliminación gaussiana consiste en triangular o escalar 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:

$$\left(\begin{array}{ccc|c} 1 & 5 & 5 & 2 \\ 2 & 2 & -3 & -1 \\ -1 & -9 & 2 & 9 \end{array} \right) \xrightarrow{f_2 - 2f_1 \rightarrow f_2} \left(\begin{array}{ccc|c} 1 & 5 & 5 & 2 \\ 0 & -8 & -13 & -5 \\ -1 & -9 & 2 & 9 \end{array} \right) \rightarrow$$

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

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 1.2.1. Resolvemos escalonando el sistema

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

1. Construimos la matriz ampliada

$$\tilde{\mathbf{A}} = \left(\begin{array}{ccc|c} 1 & 2 & -1 & -1 \\ 2 & 4 & 3 & 4 \\ 0 & 0 & 3 & 6 \end{array} \right).$$

2. Para conseguir 0's abajo de la casilla $(1, 1)$, realizamos la operación $f_2 - 2f_1 \rightarrow f_2$.

Obtenemos

$$\tilde{\mathbf{A}}_1 = \left(\begin{array}{ccc|c} 1 & 2 & -1 & -1 \\ 0 & 0 & 3 & 6 \\ 0 & 0 & 3 & 6 \end{array} \right).$$

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 $f_3 - f_2 \rightarrow f_3$. Obtenemos la matriz

$$\tilde{\mathbf{A}}_2 = \left(\begin{array}{ccc|c} 1 & 2 & -1 & -1 \\ 0 & 0 & 3 & 6 \\ 0 & 0 & 0 & 0 \end{array} \right),$$

que es una matriz escalonada, por lo que finalizamos el proceso.

Observar que si bien la matriz $\tilde{\mathbf{A}}_1$ 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 \mathbf{A} de coeficientes y el vector \mathbf{b} de términos independientes, podemos usar el comando `c_` de numpy para pegar a \mathbf{A} el vector \mathbf{b} como última columna.

EJEMPLO 1.2.2. 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

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

1.2.2. 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 1.2.3. Resolvemos el siguiente sistema de ecuaciones:

$$\left\{ \begin{array}{l} 5x_1 + 3x_2 = 11 \\ 15x_1 + 9x_2 = 33 \\ 20x_1 + 12x_2 = 44 \end{array} \right.$$

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 1.2.4. Consideremos ahora el mismo sistema inicial, modificando un valor en \mathbf{b} :

$$\left\{ \begin{array}{l} 5x_1 + 3x_2 = 11 \\ 15x_1 + 9x_2 = 33 \\ 20x_1 + 12x_2 = 55 \end{array} \right.$$

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 escalaronar 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 anulan 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 escalaronar 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 escalaronar 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 escalaronar 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 1.2.5. Escalaronar 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. $\begin{cases} 2x + y - z = 3 \\ x - y + z = 2 \\ 5x + y - z = -5 \end{cases}$

1.3. 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	$a + (b + c) = (a + b) + c$	$(a \cdot b) \cdot c = a \cdot (b \cdot c)$
commutatividad	$a + b = b + a$	$a \cdot b = b \cdot a$
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 complejos¹.

EJEMPLO 1.3.1. 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. $+ : V \times V \rightarrow V$, a cualquier par \mathbf{v}, \mathbf{w} de vectores le asigna un vector $\mathbf{v} + \mathbf{w} \in V$.
- Producto por escalar. $\cdot : \mathbb{K} \times V \rightarrow V$, a un escalar a y un vector $\mathbf{v} \in V$ le asigna un vector $a\mathbf{v} \in V$ (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.** $\mathbf{u} + (\mathbf{v} + \mathbf{w}) = (\mathbf{u} + \mathbf{v}) + \mathbf{w}$
2. **Commutatividad de la suma.** $\mathbf{u} + \mathbf{v} = \mathbf{v} + \mathbf{u}$
3. **Elemento neutro de la suma.** Existe un elemento $\mathbf{0} \in V$ tal que $\mathbf{v} + \mathbf{0} = \mathbf{v}$ para todo $\mathbf{v} \in V$.
4. **Inverso para la suma.** Para todo $\mathbf{v} \in V$, existe un elemento $-\mathbf{v} \in V$, llamado inverso aditivo de \mathbf{v} , tal que $\mathbf{v} + (-\mathbf{v}) = \mathbf{0}$.
5. **Compatibilidad de la multiplicación por escalar con la multiplicación del cuerpo.** $a(b\mathbf{v}) = (ab)\mathbf{v}$.
6. **Elemento neutro de la multiplicación por escalar.** $1\mathbf{v} = \mathbf{v}$, donde 1 es el neutro de \mathbb{R} .
7. **Propiedad distributiva de la multiplicación por escalar respecto de la suma de vectores.** $a(\mathbf{u} + \mathbf{v}) = a\mathbf{u} + a\mathbf{v}$.

¹Existen 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. $(a + b)(\mathbf{v}) = av + bv$.

EJEMPLO 1.3.2. 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 1.3.3. 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 dato, 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 $\mathbf{v} \in V$ 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 $\mathbf{v} = (80000, 100000, 4, 4, 1, 0, 1, 0, 0)$. 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.

1.3.1. 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. $\forall \mathbf{u}, \mathbf{v} \in U \quad \mathbf{u} + \mathbf{v} \in U$
3. $\forall \mathbf{u} \in U, \forall \alpha \in \mathbb{K} \quad \alpha \mathbf{u} \in U$.

Notar que U es a la vez un espacio vectorial.

EJEMPLO 1.3.4. 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.

1.3.2. Conjuntos de generadores y bases.

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

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

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

EJEMPLO 1.3.5.

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

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

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

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

PROPOSICIÓN 1.3.6. Si los vectores $\{\mathbf{v}_1, \dots, \mathbf{v}_m\}$ son linealmente independientes, cualquier elemento de $U = \langle \mathbf{v}_1, \dots, \mathbf{v}_m \rangle$ se escribe de forma única como combinación lineal de los vectores $\{\mathbf{v}_i : i = 1, \dots, m\}$.

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

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

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

DEFINICIÓN 1.3.7. Si $\mathcal{B} = \{\mathbf{v}_1, \dots, \mathbf{v}_m\} \subset V$ es un conjunto linealmente independiente, decimos que \mathcal{B} es una *base* de $U = \langle \mathbf{v}_1, \dots, \mathbf{v}_m \rangle \subset U$.

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

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

$$\mathbf{w} = a_1\mathbf{v}_1 + \dots + a_m\mathbf{v}_m,$$

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

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

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

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

$$b_1\mathbf{v}_1 + \dots + b_k\mathbf{w} + \dots + b_m\mathbf{v}_m = 0,$$

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 \mathbf{w} obtenemos una escritura de \mathbf{w} en la base \mathcal{B} distinta a la anterior, lo cual también es absurdo.

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

Ahora podemos probar el siguiente resultado.

PROPOSICIÓN 1.3.9. Dado un espacio vectorial V , sea $\mathcal{B} = \{\mathbf{v}_1, \dots, \mathbf{v}_s\}$ 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

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

es una base de V .

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

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

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 \mathbf{v}_j por \mathbf{w}_k obtenemos una nueva base de V .

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

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 1.3.10.

1. \mathbb{R}^n es un espacio vectorial de dimensión n . Una base de \mathbb{R}^n es $\{\mathbf{e}_i : i = 1, \dots, n\}$, con \mathbf{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 1.3.11. 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 1.3.12.

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

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

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 1.3.13. 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

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

y suponemos por simplicidad $a_s \neq 0$. Luego \mathbf{w}_s pertenece al espacio generado por $\{\mathbf{w}_1, \dots, \mathbf{w}_{s-1}\}$, 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 . \square

Notamos también que dado un conjunto de vectores $S = \{\mathbf{v}_1, \dots, \mathbf{v}_s\}$, 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 1.3.14. 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 1.3.13, podemos deducir el siguiente resultado de extensión de una base.

PROPOSICIÓN 1.3.15 (Extensión de una base). Dado un espacio vectorial V de dimensión n y una base $\mathcal{B} = \{\mathbf{v}_1, \dots, \mathbf{v}_s\}$ de un subespacio S , es posible encontrar vectores $\{\mathbf{w}_1, \dots, \mathbf{w}_{n-s}\}$ tales que

$$\{\mathbf{v}_1, \dots, \mathbf{v}_s, \mathbf{w}_1, \dots, \mathbf{w}_{n-s}\}$$

forman una base de V .

DEMOSTRACIÓN. Ejercicio. □

1.3.4. Espacios dados por ecuaciones homogéneas. Dado un sistema de ecuaciones $\mathbf{Ax} = \mathbf{0}$, con $\mathbf{A} \in \mathbb{R}^{m \times n}$ y $\mathbf{0} \in \mathbb{R}^m$ el vector nulo, 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 1.3.16. El sistema

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

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 demás variables. Llamamos *variables dependientes* a las variables que aparecen 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:

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

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:

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

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

1.3.5. 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 1.3.17. 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 plantear 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 $S = \langle \mathbf{v}_1, \dots, \mathbf{v}_s \rangle \subset \mathbb{R}^n$, realizamos los siguientes pasos para obtener ecuaciones que definen a S .

1. Construir la matriz ampliada correspondiente al sistema de ecuaciones

$$(x_1, \dots, x_n) = a_1 \mathbf{v}_1 + \cdots + a_s \mathbf{v}_s.$$

2. Escalonar la matriz.

3. El espacio S queda definido por las ecuaciones correspondientes a los términos de la última columna de las filas en las que todos los coeficientes en las primeras s columnas son nulos.

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

DEFINICIÓN 1.3.18. Dados subespacios S, T de un \mathbb{K} espacio vectorial W , definimos

- **Suma.** $S + T = \{s + t : s \in S, t \in T\}$, el conjunto de todas las sumas posibles entre un elemento de S y un elemento de T .
- **Intersección.** $S \cap T = \{v \in W : v \in S \text{ y } v \in T\}$.

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 1.3.19. Dados los subespacios de \mathbb{R}^4 ,

$$S = \langle (1, 0, -3, 1), (2, 0, 3, -2) \rangle \quad \text{y} \quad T = \{(x_1, x_2, x_3, x_4) : x_1 + x_2 - x_4 = 0, x_2 + x_3 = 0\},$$

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 1.3.20. 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 $\mathcal{B} = \{\mathbf{u}_1, \dots, \mathbf{u}_k\}$ y $\mathcal{B}' = \{\mathbf{v}_1, \dots, \mathbf{v}_n\}$ bases de U y V respectivamente y notemos que si tomamos un $\mathbf{w} \in U + V$ deben existir $\mathbf{u} \in U$ y $\mathbf{v} \in V$ de modo tal que $\mathbf{w} = \mathbf{u} + \mathbf{v}$. 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

$$\mathbf{u} = \sum_{i=1}^k \alpha_i \mathbf{u}_i \quad \mathbf{v} = \sum_{j=1}^n \beta_j \mathbf{v}_j,$$

de donde concluimos que el conjunto

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

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

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

y notamos que en ese caso

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

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

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

y resulta $\alpha_1 = \dots = \alpha_k = \beta_1 = \dots = \beta_n = 0$, 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 $\mathcal{B}'' = \{\mathbf{z}_1, \dots, \mathbf{z}_r\}$ de $U \cap V$ y como $U \cap V \subset U$ gracias a la Proposición 1.3.15, podemos extender \mathcal{B}'' a una base \mathcal{B} de U , $\mathcal{B} = \{\mathbf{z}_1, \dots, \mathbf{z}_r, \mathbf{u}_{r+1}, \dots, \mathbf{u}_k\}$. Análogamente, \mathcal{B}'' se extiende a una base \mathcal{B}' de V , $\mathcal{B}' = \{\mathbf{z}_1, \dots, \mathbf{z}_r, \mathbf{v}_{r+1}, \dots, \mathbf{v}_n\}$. 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 $\tilde{\mathcal{B}} = \{\mathbf{z}_1, \dots, \mathbf{z}_r, \mathbf{u}_{r+1}, \dots, \mathbf{u}_k, \mathbf{v}_{r+1}, \dots, \mathbf{v}_n\}$ es base de $U + V$. En efecto, es trivial ver que genera y para ver que es L.I. escribimos

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

y entonces, por un lado se tiene

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

de donde en particular $\sum_{l=r+1}^n (-\beta_l) \mathbf{v}_l \in U \cap V = \langle \mathbf{z}_1, \dots, \mathbf{z}_r \rangle$, es decir, que $\beta_{r+1} = \beta_{r+2} = \dots = \beta_n = 0$. Volviendo a (1.1), vemos que

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

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

Hemos probado que

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

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. \square

1.4. Operaciones con matrices

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

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

el producto $C = AB$ es la matriz de m filas y p columnas

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

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 C ponemos el producto de la fila i de A con la columna j de 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 1.4.1. 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. $(3 \ 5) \cdot \begin{pmatrix} 4 \\ -3 \end{pmatrix}$
3. $\begin{pmatrix} 4 \\ -3 \end{pmatrix} \cdot (3 \ 5)$
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 $\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C}$, para matrices con los tamaños apropiados para realizar los productos.
- El producto de matrices no es conmutativo. En general, no vale $\mathbf{AB} = \mathbf{BA}$, 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.

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

EJEMPLO 1.4.2.

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

PROPOSICIÓN 1.4.3.

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

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 $\mathbf{v} \in \mathbb{K}^n$, $\mathbf{A} \in \mathbb{K}^{n \times n}$, con $\bar{\mathbf{v}} \in \mathbb{K}^n$ (resp. $\bar{\mathbf{A}} \in \mathbb{K}^{n \times n}$), denotamos el vector (resp. matrix) que tiene los mismos elementos que \mathbf{v} (resp. \mathbf{A}) pero conjugados. Obviamente si $\mathbb{K} = \mathbb{R}$, $\bar{\mathbf{v}} = \mathbf{v}$, $\bar{\mathbf{A}} = \mathbf{A}$.

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

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

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

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

1.4.3. Sistemas lineales y ecuaciones matriciales. Observando el producto del punto 4 del Ejercicio 1.4.1, 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, $\mathbf{A}\mathbf{x} = \mathbf{b}$, donde $\mathbf{A} \in \mathbb{R}^{m \times n}$ es la matriz de coeficientes, \mathbf{x} es un vector o matriz columna de incógnitas y $\mathbf{b} \in \mathbb{R}^n$ es el vector de términos constantes.

Pregunta: ¿cuántas incógnitas y cuántas ecuaciones tiene el sistema $\mathbf{A}\mathbf{x} = \mathbf{b}$ 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]
```

1.4.4. 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 "independientes" tenemos.

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

PROPOSICIÓN 1.4.4. 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 1.4.5.

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

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 1.4.6. 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
```

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

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

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

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

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

1.4.5. Matrices especiales.

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

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

EJERCICIO 1.4.7. 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]]
```

1.4.5.2. *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 1.4.8. Para toda $\mathbf{B} \in \mathbb{R}^{n \times n}$, $I_n \cdot \mathbf{B} = \mathbf{B} \cdot I_n = \mathbf{B}$.

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 ]]
```

1.4.5.3. *Mas Matrices especiales.*

- **Triangular superior:** Una matriz $\mathbf{U} \in \mathbb{R}^{n \times n}$ 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 $\mathbf{L} \in \mathbb{R}^{n \times n}$ 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:** $\mathbf{A} \in \mathbb{R}^{n \times n}$ es simétrica si $\mathbf{A}^T = \mathbf{A}$.
- **Matriz Hermitiana:** $\mathbf{A} \in \mathbb{C}^{n \times n}$ es Hermitiana si $\mathbf{A}^* = \mathbf{A}$.

- **Matriz ortogonal:** \mathbf{A} es ortogonal si $\mathbf{AA}^T = \mathbf{A}^T\mathbf{A} = \mathbf{I}_n$.
- **Matriz unitaria:** $\mathbf{A} \in \mathbb{C}^{n \times n}$ es unitaria si $\mathbf{AA}^* = \mathbf{A}^*\mathbf{A} = \mathbf{I}_n$.

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

$$\mathbf{AB} = \mathbf{I}_n = \mathbf{BA},$$

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

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

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

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

EJEMPLO 1.4.9. Calculamos la inversa de la matriz

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

Podemos resolver los sistemas $\mathbf{Ab}_j = \mathbf{e}_j$, $1 \leq j \leq 3$, simultáneamente considerando una matriz ampliada con los 3 vectores \mathbf{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 \mathbf{B} . Por ejemplo, la última fila será $[-0.20.40.2]$. Para las filas anteriores, podemos realizar ahora la

triangulación hacia arriba:

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

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

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

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

$$\mathbf{Ax} = \mathbf{b}$$

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

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 $\mathbf{AX} = \mathbf{I}_n$).

```
# 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.

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

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

PROPOSICIÓN 1.4.11. Si $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times n}$:

- $\text{tr}(\mathbf{A}^T) = \text{tr}(\mathbf{A})$
- $\text{tr}(\mathbf{A} + \mathbf{B}) = \text{tr}(\mathbf{A}) + \text{tr}(\mathbf{B})$.

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

```

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

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

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

EJEMPLO 1.4.12.

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

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

- multiplicar una fila por un escalar: $\det \begin{pmatrix} & f_1 & - \\ & \vdots & - \\ & f_{i-1} & - \\ & \alpha f_i & - \\ & f_{i+1} & - \\ & \vdots & - \\ & f_n & - \end{pmatrix} = \alpha \det(\mathbf{A})$

- sumar un múltiplo de una fila a otra fila: $\det \begin{pmatrix} & f_1 & - \\ & \vdots & - \\ & f_{i-1} & - \\ & f_i + \beta f_j & - \\ & f_{i+1} & - \\ & \vdots & - \\ & f_n & - \end{pmatrix} = \det(\mathbf{A})$

- 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: $\det(k\mathbf{A}) = k^n \det(\mathbf{A})$, para $k \in \mathbb{R}$.
- multiplicar toda la matriz por (-1) : $\det(-\mathbf{A}) = (-1)^n \det(\mathbf{A})$.

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

COROLARIO 1.4.14.

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

PROPOSICIÓN 1.4.15. Si $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times n}$,

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

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) = 19.99999999999996
%% det(AB) = -240.0000000000002
%% det(A + B) = 51.0
%% det(M) = -9.51619735392994e-16

```

EJERCICIO 1.4.16. Para $\mathbf{A} = \begin{pmatrix} 3 & 2 & -1 \\ 3 & 0 & 7 \\ 0 & -1 & 4 \end{pmatrix}$, probar que el sistema $\mathbf{Ax} = \mathbf{b}$ tiene solución única para todo $\mathbf{b} \in \mathbb{R}^3$

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

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

1.5. Cambio de base

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

Dada otra base \mathcal{B} de V , queremos saber cómo escribir en la base \mathcal{B} a un vector \mathbf{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} .

1.5.0.1. 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

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

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

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

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

Podemos escribirlo matricialmente:

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

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 $\mathbf{C}_{\mathcal{B}\mathcal{E}}$ a esta matriz. Obtenemos

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

EJERCICIO 1.5.1.

1. Dada la base $\mathcal{B} = \{(1, 3, 7), (4, 0, 1), (5, -7, 0)\}$ de \mathbb{R}^3 y el vector \mathbf{v} de \mathbb{R}^3 de coordenadas $(1, 2, -5)$ en la base \mathcal{B} , hallar las coordenadas de \mathbf{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 $(x-1)$.

1.5.0.2. Cambio de base de \mathcal{B} a \mathcal{E} . Ahora queremos escribir a $\mathbf{v} = (3, 7, -1)_{\mathcal{E}}$ 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 $\mathbf{C}_{\mathcal{B}\mathcal{E}}$. Como \mathcal{B} es una base, la matriz $\mathbf{C}_{\mathcal{B}\mathcal{E}}$ es inversible (¿por qué?).

Por lo tanto,

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

Obtenemos que la matriz de cambio de base de la base canónica \mathcal{E} a una base \mathcal{B} es $\mathbf{C}_{\mathcal{E}\mathcal{B}} = \mathbf{C}_{\mathcal{B}\mathcal{E}}^{-1}$.

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

1. Construimos la matriz $C_{\mathcal{B}\mathcal{E}} = \begin{pmatrix} | & | & & | \\ b_1 & b_2 & \cdots & b_n \\ | & | & & | \end{pmatrix}$.
2. Para pasar un vector en base \mathcal{B} a base canónica, usamos $\mathbf{v}_{\mathcal{E}} = C_{\mathcal{B}\mathcal{E}}\mathbf{v}_{\mathcal{B}}$.
3. Para pasar un vector en base canónica a la base \mathcal{B} , definimos $C_{\mathcal{E}\mathcal{B}} = (C_{\mathcal{B}\mathcal{E}})^{-1}$ y usamos $\mathbf{v}_{\mathcal{B}} = C_{\mathcal{E}\mathcal{B}}\mathbf{v}_{\mathcal{E}}$.

1.6. Transformaciones lineales

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

$$T_{\mathbf{A}}(\mathbf{v}) = \mathbf{Av}.$$

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

EJEMPLO 1.6.1. Si $\mathbf{A} = \begin{pmatrix} 1 & 3 & 0 \\ 2 & -1 & -4 \end{pmatrix} \in \mathbb{R}^{2 \times 3}$ y $\mathbf{v} = (1, 0, -2)$, entonces

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

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

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

Entonces podemos escribir

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

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

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

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

EJEMPLO 1.6.3. 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 \mathbf{A} son las imágenes de los vectores canónicos \mathbf{e}_i , $1 \leq i \leq n$.

PROPOSICIÓN 1.6.4. Si $\mathcal{B} = \{\mathbf{v}_1, \dots, \mathbf{v}_n\}$ es una base de V , podemos definir una transformación lineal definiendo la imagen de cada elemento de la base.

Para un elemento $\mathbf{v} \in V$, si $\mathbf{v} = a_1\mathbf{v}_1 + \cdots + a_n\mathbf{v}_n$, entonces $f(\mathbf{v}) = a_1f(\mathbf{v}_1) + \cdots + a_nf(\mathbf{v}_n)$.

EJEMPLO 1.6.5. $\mathcal{B} = \{(1, 2), (2, -1)\}$ es una base de \mathbb{R}^2 . Si $f : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ es una transformación lineal, $f(1, 2) = (0, 1)$ y $f(2, -1) = (1, 3)$, entonces

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

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

DEFINICIÓN 1.6.6. El subespacio de \mathbb{K}^m

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

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

El subespacio de \mathbb{K}^n

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

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

Propiedad. $\text{rango}(\mathbf{A}) = \dim(\text{im}(\mathbf{A}))$.

TEOREMA 1.6.7 (Teorema de la dimensión.). Dada una matriz $\mathbf{A} \in \mathbb{K}^{m \times n}$,

$$n = \dim(\ker(\mathbf{A})) + \dim(\text{im}(\mathbf{A}))$$

DEMOSTRACIÓN. Utilizamos la Proposición 1.3.15 de extensión de bases. Suponemos que el núcleo de \mathbf{A} tiene dimensión t y consideramos una base $\{\mathbf{u}_1, \dots, \mathbf{u}_t\} \subset \mathbb{K}^n$ del núcleo de \mathbf{A} . Extendemos dicha base a una base de \mathbb{R}^n :

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

Veamos que

$$\{\mathbf{Aw}_1, \dots, \mathbf{Aw}_{n-t}\}$$

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

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

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

Como $\mathbf{Au}_i = \mathbf{0}$ para $1 \leq i \leq t$, obtenemos que

$$\mathbf{Av} = +b_1\mathbf{Aw}_1 + \cdots + b_{n-t}\mathbf{Aw}_{n-t},$$

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

Supongamos $c_1\mathbf{Aw}_1 + \cdots + c_{n-t}\mathbf{Aw}_{n-t} = 0$. Luego

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

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

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

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

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

EJEMPLO 1.6.8. Dada la matriz $\mathbf{A} = \begin{pmatrix} 1 & 0 & -2 & 1 \\ 0 & 2 & -1 & -2 \\ 2 & 1 & 1 & 1 \end{pmatrix}$,

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

Respuestas

1. $\text{im}(\mathbf{A}) \subset \mathbb{R}^3$ y $\ker(\mathbf{A}) \subset \mathbb{R}^4$.
2. $\text{im}(\mathbf{A}) = \langle (1, 0, 2), (0, 2, 1), (-2, -1, 1), (1, -2, 1) \rangle$. 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 $\dim(\text{im}(\mathbf{A})) = 3$, porque quedan 3 filas no nulas.
3. Para resolver el sistema $\mathbf{Ax} = 0$, triangulamos \mathbf{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

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

y $\dim(\ker(\mathbf{A})) = 1$.

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

1.7. Espacios afines

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

$$\mathbf{A}\mathbf{x} = 0$$

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

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \text{ con } \mathbf{b} \in \mathbb{K}^n.$$

EJEMPLO 1.7.1. 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 $\mathbf{A}\mathbf{x} = \mathbf{b}$.
- El subespacio $\langle(-0.6, 1)\rangle$ es el conjunto de soluciones del sistema homogéneo $\mathbf{A}\mathbf{x} = 0$.

y esto vale en general:

PROPOSICIÓN 1.7.2. Dado el sistema $\mathbf{A}\mathbf{x} = \mathbf{b}$, si $\tilde{\mathbf{x}}$ es una solución particular del sistema, el conjunto de todas las soluciones del sistema es

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

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

$$\mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{y} = \mathbf{b} - \mathbf{b} = 0,$$

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

EJEMPLO 1.7.3. 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

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

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

2.1. 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 ejecución¹.

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

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

Para número real (positivo) diferente, la escritura

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

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

$$0.999\dots = 1,$$

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.

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

2.2. 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 forma²

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

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 (2.1) se obtiene de la escritura de x como

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

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

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

Continuemos asumiendo que $x > 0$. El uso del punto *fijo* en (2.1) 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 .

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

$$(2.2) \quad (x)_b = sg(x) 0.b_0 b_1 b_2 \dots b^e$$

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

$$(x)_b = sg(x) 0.b_0 b_1 \dots (b_{i-1} + 1) b^{e+1}.$$

Suele llamarse notación científica normalizada a la escritura (2.2). La expresión $0.b_0 b_1 b_2 \dots$ 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

$$e = sg(e) c_l c_{l-1} \dots c_0$$

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

2.3. 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

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

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*.

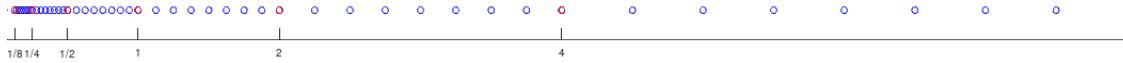


FIGURA 1. Algunos números positivos de máquina con $b = 2$ y $m = 4$. Notar que los números son equiespaciados únicamente entre potencias sucesivas de 2, y que la cantidad de números de máquina en esos rangos se mantiene constante.

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 (admiten 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).

^aEn *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 más 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 (2.2) 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 las 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:

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

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

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

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

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

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 $0 \leq i \leq m - 1$

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

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 $x_1 \leq x \leq x_2$ (i.e. x esta dentro de las escalas manejadas por la máquina) entonces

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

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

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

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

resulta

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

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

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

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). Naturalmente 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

$$x \rightarrow fl(x) \quad y \rightarrow fl(y)$$

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

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

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

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

por lo que

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

y en definitiva

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

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 relativo⁵. Como es bien sabido, eso no es posible si *restamos* números similares⁶. 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

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

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 similares⁷ 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

$$(2.3) \quad 1 \oplus \varepsilon > 1,$$

y

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

⁴Notemos 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?.

⁵Cuando 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.

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

⁷Ver 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 (2.3). 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

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

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*.

2.4. 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 hablaremos 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 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 $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ puede definirse el número de condición de este problema de la forma^a

^aUn ejemplo concreto y elemental de mal condicionamiento es, como hemos visto, la resta de números similares: si escribimos $f : \mathbb{R}^2 \rightarrow \mathbb{R}$, $f(x, y) = x - y$, 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 $\operatorname{tg}(x)$ en un $x_0 < \pi/2$, $x_0 \sim \pi/2$ vemos que el número de condición

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

cumple que

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

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

En los años 50, Wilkinson experimentaba con las primeras computadoras y rápidamente comenzó a observar fenómenos de inestabilidades y mal condicionamiento⁸. Una cosa que notó, al probar un algoritmo de aproximación de raíces, 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 raíces $10, 11, \dots, 19$ se transforman en 5 pares complejos conjugados. Las 18 y 19 en complejos de la forma $19.5 \dots \pm 19.5 \dots i$, es decir que una perturbación de orden 2^{-23} genera variaciones de orden 1 .⁹ Mostrando que el problema de calcular raíces 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 más 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

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

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 términos de la serie

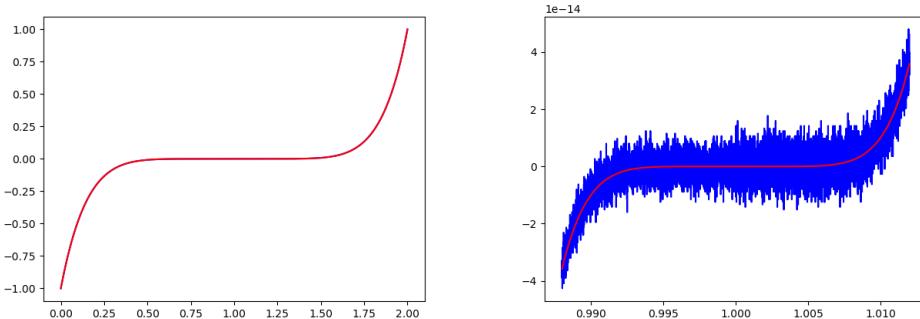
$$e^x = 1 + x + \frac{1}{2!}x^2 + \frac{1}{3!}x^3 \dots$$

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

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

⁸En sus propias palabras: “The cosy relationship that mathematics enjoyed with polynomials suffered a severe setback in the early fifties where electronic computers came into general use. Speaking for myself I regard it as the most traumatic experience in my career as a numerical analyst” [7]

⁹A 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

$$e^{-12} \sim 6.1442123533282097586823081788055323112239893148882529755\dots \cdot 10^{-6}$$

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 2 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

3.1. 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:

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

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

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

lo cual nos da una expresión aplicable a los complejos.

DEFINICIÓN 3.1.1. Una norma de un K -espacio vectorial es una función $\|\cdot\| : V \rightarrow \mathbb{R}_{\geq 0}$ que cumple las siguientes propiedades:

1. $\|a\mathbf{v}\| = |a|\|\mathbf{v}\|$, para $a \in K$ y $\mathbf{v} \in V$.
2. Si $\|\mathbf{v}\| = 0$, entonces $\mathbf{v} = 0$.
3. $\|\mathbf{u} + \mathbf{v}\| \leq \|\mathbf{u}\| + \|\mathbf{v}\|$, para todo $\mathbf{u}, \mathbf{v} \in V$ (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 3.1.2 (Desigualdad de Cauchy-Schwarz). Dados $\mathbf{u}, \mathbf{v} \in \mathbb{K}^n$,

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

DEMOSTRACIÓN. Asumamos primero que $K = \mathbb{R}$. 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 raíz 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 complejos¹

$$\left| \sum_i u_i \bar{v}_i \right| \leq \sum_i |u_i| |\bar{v}_i| = \sum_i |u_i| |v_i|,$$

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

OBSERVACIÓN 3.1.3. Notar que la desigualdad de Cauchy-Schwarz vale también tomando a la izquierda $\sum_i \bar{u}_i v_i$, puesto que $\sum_i \bar{u}_i v_i = \overline{\sum_i u_i \bar{v}_i}$ y el módulo de un complejo es igual al de su conjugado.

COROLARIO 3.1.4. Para todo $\mathbf{u}, \mathbf{v} \in \mathbb{K}^n$ vale, $\|\mathbf{u} + \mathbf{v}\|_2 \leq \|\mathbf{u}\|_2 + \|\mathbf{v}\|_2$.

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

$$\begin{aligned} \|\mathbf{u} + \mathbf{v}\|_2^2 &= \sum_{i=1}^n |u_i + v_i|^2 = \sum_{i=1}^n (u_i + v_i)(\bar{u}_i + \bar{v}_i) \\ &= \sum_{i=1}^n u_i \bar{u}_i + \sum_{i=1}^n (u_i \bar{v}_i + v_i \bar{u}_i) + \sum_{i=1}^n v_i \bar{v}_i = \|\mathbf{u}\|_2^2 + \sum_{i=1}^n (u_i \bar{v}_i + v_i \bar{u}_i) + \|\mathbf{v}\|_2^2, \end{aligned}$$

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

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

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

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

lo que termina la demostración tomando raíz cuadrada \square

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

- Norma-1: $\|\mathbf{v}\|_1 = |v_1| + \dots + |v_n|$
- Norma-infinito: $\|\mathbf{v}\|_\infty = \max\{|v_1|, \dots, |v_n|\}$
- Norma- p : $\|\mathbf{v}\|_p = (|v_1|^p + \dots + |v_n|^p)^{1/p}$

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.

¹Si 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 $0 \leq (a_1 b_2 - a_2 b_1)^2$ que obviamente siempre es válido.

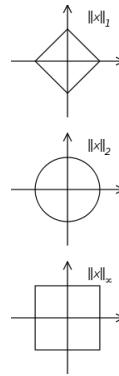


FIGURA 1. “Círculos” de radio 1 en diferentes normas. Es decir $\mathbf{v} \in \mathbb{R}^2$ tales que $\|\mathbf{v}\| = 1$.

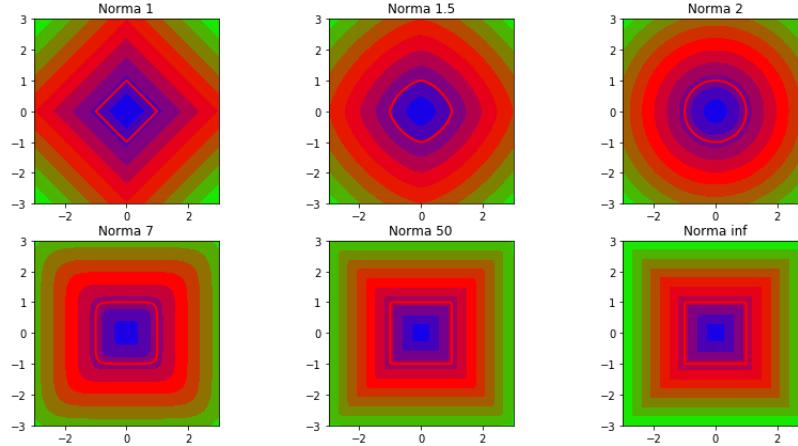


FIGURA 2. Curvas de nivel de diferentes normas $-p$.

En la siguiente figura observamos distintas curvas de nivel² para distintas normas.

EJERCICIO 3.1.5. Examinando los gráficos de las curvas de nivel de las normas $-p$ se puede intuir que $\lim_{p \rightarrow +\infty} \|\mathbf{v}\|_p = \|\mathbf{v}\|_\infty$. De una demostración de este hecho.

Para nosotros va a ser relevante medir *errores* en espacios vectoriales, para ello recordemos que dados $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ $\|\mathbf{u} - \mathbf{v}\|_2$ 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 3.1.6. Verificar que para todo $\mathbf{x} \in \mathbb{K}^n$

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

²Las 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 3.1.7. 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 $\mathbf{x} \in V$

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

Vale lo siguiente,

PROPOSICIÓN 3.1.8. En un \mathbb{K} -espacio vectorial de dimensión finita todas la normas son equivalentes³.

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

DEFINICIÓN 3.1.9. Decimos que una sucesión de vectores $\{v_n\}_{n \in \mathbb{N}}$ converge bajo una norma $\|\cdot\|$ a un vector v si

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

Como consecuencia de la Proposición 3.1.8, la convergencia en una norma cualquiera implica la convergencia en todas las normas.

3.2. Producto interno

DEFINICIÓN 3.2.1. Dados dos vectores $\mathbf{u}, \mathbf{v} \in \mathbb{K}^n$ definimos^a el producto interno (canónico) por la fórmula^b

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

es decir

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

^aRecordemos que asociamos los vectores con las matrices columna.

^bEn muchos casos se suele conjugar los coeficientes del \mathbf{v} en la sumatoria en vez de los de \mathbf{u} en la definición de $\langle \mathbf{u}, \mathbf{v} \rangle$. Para nuestros intereses no cambia en absoluto la definición utilizada que por lo demás coinciden sobre \mathbb{R} .

Tenemos las siguientes propiedades inmediatas.

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

OBSERVACIÓN 3.2.2. Notar que para $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{K}^n$, $\mathbf{w} \in \mathbb{K}^m$ $a, b \in \mathbb{K}$, $\mathbf{A} \in \mathbb{K}^{n \times m}$, resulta

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

^aPuesto que $\langle \mathbf{A}\mathbf{w}, \mathbf{x} \rangle = (\mathbf{A}\mathbf{w})^*\mathbf{x} = \mathbf{w}^*\mathbf{A}^*\mathbf{x} = \langle \mathbf{w}, \mathbf{A}^*\mathbf{x} \rangle$.

EJEMPLO 3.2.3. $\langle \mathbf{x} + \mathbf{y}, \mathbf{x} + \mathbf{y} \rangle = \langle \mathbf{x}, \mathbf{x} + \mathbf{y} \rangle + \langle \mathbf{y}, \mathbf{x} + \mathbf{y} \rangle = \langle \mathbf{x}, \mathbf{x} \rangle + \langle \mathbf{x}, \mathbf{y} \rangle + \langle \mathbf{y}, \mathbf{x} \rangle + \langle \mathbf{y}, \mathbf{y} \rangle = \|\mathbf{x}\|^2 + 2\langle \mathbf{x}, \mathbf{y} \rangle + \|\mathbf{y}\|^2$

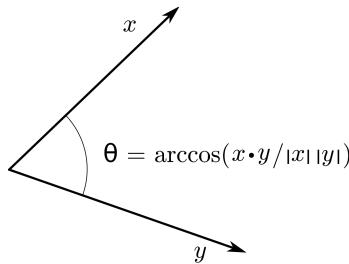
Sean $\mathbf{x}, \mathbf{y} \in \mathbb{R}^2$, $\mathbf{x} \neq \mathbf{0} \neq \mathbf{y}$, $\mathbf{x} = (a_1, a_2)$, $\mathbf{y} = (b_1, b_2)$. Podemos escribir $\mathbf{x} = \|\mathbf{x}\| \frac{\mathbf{x}}{\|\mathbf{x}\|}$ y como $\frac{\mathbf{x}}{\|\mathbf{x}\|}$ está en el círculo de radio 1 vemos que existe un único α , $0 \leq \alpha < 2\pi$ tal que $\frac{\mathbf{x}}{\|\mathbf{x}\|} = (\cos(\alpha), \sin(\alpha))$. Es decir, $\mathbf{x} = \|\mathbf{x}\|_2(\cos(\alpha), \sin(\alpha))$. Análogamente, podemos escribir $\mathbf{y} = \|\mathbf{y}\|_2(\cos(\beta), \sin(\beta))$ y en particular⁴

$$\langle \mathbf{x}, \mathbf{y} \rangle = \|\mathbf{x}\|_2 \|\mathbf{y}\|_2 (\cos(\alpha) \cos(\beta) + \sin(\alpha) \sin(\beta)) = \|\mathbf{x}\|_2 \|\mathbf{y}\|_2 \cos(\alpha - \beta),$$

lo que dice que puede escribirse

$$(3.1) \quad \cos(\theta) = \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\|_2 \|\mathbf{y}\|_2},$$

con θ el ángulo⁵ entre \mathbf{x} y \mathbf{y} .



En \mathbb{R}^n ocurre lo mismo. Ya que $-1 \leq \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\| \|\mathbf{y}\|} \leq 1$, (ver (6) de la Observación 3.2.2) se define el ángulo θ entre \mathbf{x} e \mathbf{y} a través de (3.1).

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.

⁴Recordar: $\cos(\alpha + \beta) = \cos(\alpha)\cos(\beta) - \sin(\alpha)\sin(\beta)$, $\sin(\alpha + \beta) = \sin(\alpha)\cos(\beta) + \cos(\alpha)\sin(\beta)$.

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

De (3.1) vemos que $\theta = \pi/2$ si y solo si $\mathbf{v}^* \mathbf{w} = 0$, de ahí la siguiente definición que la extendemos a los complejos.

DEFINICIÓN 3.2.4. Dados $\mathbf{v}, \mathbf{w} \in \mathbb{K}^n$, no nulos. Decimos que son ortogonales sí y solo sí $\mathbf{v}^* \mathbf{w} = \langle \mathbf{v}, \mathbf{w} \rangle = 0$. En este caso también usaremos la notación $\mathbf{v} \perp \mathbf{w}$.

DEFINICIÓN 3.2.5. Una colección de vectores $\{\mathbf{v}_i\}_{1 \leq i \leq k} \subset \mathbb{K}^n$ se dice *ortogonal* si $\langle \mathbf{v}_i, \mathbf{v}_j \rangle = 0$ para $i \neq j$. Si además $\|\mathbf{v}_i\|_2 = 1$ para todo $1 \leq i \leq k$ (es decir, $\langle \mathbf{v}_i, \mathbf{v}_j \rangle = \delta_i^j$ para todo $1 \leq i, j \leq n$) 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 $\mathcal{B} = \{\mathbf{v}_i\}_{1 \leq i \leq n} \subset \mathbb{K}^n$ una *base ortonormal* de \mathbb{K}^n . Dado $\mathbf{w} \in \mathbb{K}^n$ queremos conocer las coordenadas de \mathbf{w} en la base \mathcal{B} . Esto se reduce a hallar los $\{\alpha_i\}_{1 \leq i \leq n} \subset \mathbb{K}$ tales que

$$(3.2) \quad \mathbf{w} = \sum_{i=1}^n \alpha_i \mathbf{v}_i,$$

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

$$(3.3) \quad \mathbf{v}_j^* \mathbf{w} = \sum_{i=1}^n \alpha_i \mathbf{v}_j^* \mathbf{v}_i = \alpha_j,$$

permite “despejar” α_i por el módico costo de un producto escalar⁶ Desde el punto de vista matricial, sea $\mathbf{Q} \in \mathbb{K}^{n \times n}$, que tiene por columnas los \mathbf{v}_i . Resulta que el problema anterior se puede escribir de modo matricial

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

y de la expresión de mas arriba vemos que

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

Esto no expresa mas que el hecho de que invertir \mathbf{Q} se reduce a conjugar (trasponer si $\mathbb{K} = \mathbb{R}$) lo que a su vez es inmediato considerando que sus columnas son ortonormales⁷ y así

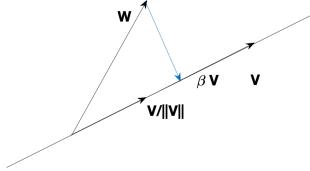
$$[\mathbf{Q}^* \mathbf{Q}]_{ij} = \mathbf{v}_i^* \mathbf{v}_j = \delta_i^j.$$

Estas matrices tienen un nombre particular.

DEFINICIÓN 3.2.6. Una matriz $\mathbf{Q} \in \mathbb{K}^{n \times n}$ se dice *unitaria* (suele llamarse *ortogonal* cuando $\mathbb{K} = \mathbb{R}$) si $\mathbf{Q}^* \mathbf{Q} = \mathbf{I}$ ($\mathbf{Q}^T \mathbf{Q} = \mathbf{I}$ si $\mathbb{K} = \mathbb{R}$).

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

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

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

OBSERVACIÓN 3.2.7. \mathbf{Q} es unitaria (resp. ortogonal) si y solo si \mathbf{Q}^* es unitaria (resp. ortogonal).

Estudiemos ahora el problema de calcular la proyección *ortogonal* de un vector \mathbf{w} sobre la recta generada por otro vector $\mathbf{v} \neq \mathbf{0}$ (ver Figura 3). Es decir, que buscamos un vector $\beta\mathbf{v}$ tal que $(\beta\mathbf{v} - \mathbf{w}) \perp \mathbf{v}$, esto equivale a $(\beta\mathbf{v} - \mathbf{w})^*\mathbf{v} = 0$, lo que indica que $\bar{\beta} = \frac{\mathbf{w}^*\mathbf{v}}{\|\mathbf{v}\|_2^2}$, es decir

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

Si $\|\mathbf{v}\| = 1$ la expresión coincide con la de los α calculados en (3.3), lo que nos da una interpretación alternativa de (3.2). Esto es, llamando $\mathbf{P}_v(\mathbf{w})$ a la proyección ortogonal de \mathbf{w} sobre el subespacio generado por \mathbf{v} , vemos que

$$(3.4) \quad \mathbf{P}_v(\mathbf{w}) = \frac{\mathbf{v}^*\mathbf{w}}{\|\mathbf{v}\|_2^2}\mathbf{v},$$

y en particular (3.2) toma la forma

$$\mathbf{w} = \sum_{1 \leq i \leq n} \mathbf{P}_{v_i}(\mathbf{w}).$$

De los argumentos previos vemos que *en una base ortonormal*, las proyecciones ortogonales de un vector \mathbf{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 (3.2), tenemos que el cuadrado de la longitud de \mathbf{w} (el cuadrado de la “hipotenusa”) es, por un lado

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

y por el otro

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

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 3.2.8. En toda base ortonormal $\mathcal{B} = \{\mathbf{v}_1, \dots, \mathbf{v}_n\}$, si

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

entonces

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

Mas adelante volveremos sobre el tema de las proyecciones.

Para finalizar la sección de producto interno definimos

DEFINICIÓN 3.2.9. Una matriz $\mathbf{A} \in \mathbb{C}^{n \times n}$ se dice *Hermitiana* si $\mathbf{A} = \mathbf{A}^*$. En el caso de que $\mathbf{A} \in \mathbb{R}^{n \times n}$, \mathbf{A} se dice *simétrica*.

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

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

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

DEFINICIÓN 3.2.10. Una matriz $\mathbf{A} \in \mathbb{C}^{n \times n}$ ($\mathbf{A} \in \mathbb{R}^{n \times n}$) se dice *definida positiva* si es Hermitiana (simétrica) y además $\mathbf{v}^* \mathbf{A} \mathbf{v} > 0$ para todo $\mathbf{v} \neq 0$. Si por otro lado $\mathbf{v}^* \mathbf{A} \mathbf{v} \geq 0$ para todo \mathbf{v} , la matriz se dice *semi definida* positiva.

3.3. Normas de matrices

Dada una matriz $\mathbf{A} \in \mathbb{K}^{n \times m}$, se la puede 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 excepcionales⁸, estas 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 $\mathbf{A} \in \mathbb{K}^{n \times m}$, y un par de normas vectoriales $\|\cdot\|_n, \|\cdot\|_m$ en \mathbb{K}^n y \mathbb{K}^m respectivamente, definimos

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

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

⁸Como la norma de Frobenius que veremos mas adelante.

$$(3.5) \quad \|\mathbf{A}\| = \max_{\mathbf{0} \neq \mathbf{x} \in \mathbb{K}^m} \frac{\|\mathbf{Ax}\|}{\|\mathbf{x}\|} = \max_{\mathbf{x} \in \mathbb{K}^m, \|\mathbf{x}\|=1} \|\mathbf{Ax}\|.$$

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 \mathbf{x} al multiplicarlo por \mathbf{A} .

EJERCICIO 3.3.1. Sea $\mathbf{A} \in \mathbb{K}^{n \times n}$ y $\|\cdot\|$ una norma en \mathbb{K}^n , entonces (3.5) define una norma.

EJERCICIO 3.3.2. Sea $\mathbf{I} \in \mathbb{K}^{n \times n}$ la identidad, entonces $\|\mathbf{I}\| = 1$ para toda norma subordinada (siempre que pensemos \mathbb{K}^n con la misma norma vectorial como espacio de partida y llegada de la transformación $\mathbf{I} : \mathbb{K}^n \rightarrow \mathbb{K}^n$).

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

$$(3.6) \quad \|\mathbf{Ax}\| \leq \|\mathbf{A}\| \|\mathbf{x}\|.$$

De aquí se obtiene la siguiente propiedad. Para todo par de matrices⁹ $\mathbf{A}, \mathbf{B} \in \mathbb{K}^{n \times n}$, y toda *norma subordinada*

$$(3.7) \quad \|\mathbf{AB}\| \leq \|\mathbf{A}\| \|\mathbf{B}\|.$$

En efecto, dos aplicaciones sucesivas de (3.6) da

$$\|\mathbf{AB}\| = \max_{\|\mathbf{x}\|=1} \|\mathbf{ABx}\| \leq \max_{\|\mathbf{x}\|=1} \|\mathbf{A}\| \|\mathbf{Bx}\| \leq \max_{\|\mathbf{x}\|=1} \|\mathbf{A}\| \|\mathbf{B}\| \|\mathbf{x}\| = \|\mathbf{A}\| \|\mathbf{B}\|,$$

que prueba (3.7). Aplicando sucesivamente (3.7), vemos que para toda matriz cuadrada y para todo $k \in \mathbb{N}$, vale¹⁰

$$(3.8) \quad \|\mathbf{A}^k\| \leq \|\mathbf{A}\|^k.$$

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

EJEMPLO 3.3.3. Consideremos en \mathbb{K}^n la norma $\|\cdot\|_\infty$. Para $\mathbf{A} \in \mathbb{K}^{n \times n}$ se tiene¹¹

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

⁹Como se desprende de la cuenta, también vale para matrices rectangulares, siempre que pueda hacerse el producto \mathbf{AB} .

¹⁰Que ocurre si $k \in \mathbb{Z}$ con $k < 0$? Por ejemplo, cuando \mathbf{A} es invertible sabemos darle sentido a \mathbf{A}^{-1} como la inversa de \mathbf{A} . Por lo tanto $\mathbf{I} = \mathbf{AA}^{-1}$ de donde $1 \leq \|\mathbf{A}\| \|\mathbf{A}^{-1}\|$ (hemos usado aquí el Ejercicio 3.3.2) y de ahí $\|\mathbf{A}\|^{-1} \leq \|\mathbf{A}^{-1}\|$. Es decir al revés que en (3.8). Piense que ocurre en general. Defina primero \mathbf{A}^k con $k < 0$. Como ayuda pregúntese que será por ejemplo \mathbf{A}^{-3} ? Vea que puede pensarla como $\mathbf{A}^{-3} := (\mathbf{A}^{-1})^3$, es decir $\mathbf{A}^{-3} = \mathbf{A}^{-1} \mathbf{A}^{-1} \mathbf{A}^{-1}$. Pero otro lado $\mathbf{A}^3 (\mathbf{A}^{-1} \mathbf{A}^{-1} \mathbf{A}^{-1}) = \mathbf{I}$, lo que dice que la primera definición coincide con $(\mathbf{A}^3)^{-1}$. Use estas consideraciones para responder la pregunta.

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

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

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

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

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

(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 \mathbf{z} al vector de los signos de los elementos de esa fila $\mathbf{z} = (sg(a_{k,1}), \dots, sg(a_{k,n}))$. Si algún $a_{k,i} = 0$ tomamos $sg(a_{k,1}) = 0$. Como la fila no es nula, $\mathbf{z} \neq \mathbf{0}$. Luego $\|\mathbf{z}\|_\infty = \max_{1 \leq i \leq n} \{|z_i|\} = 1$, y haciendo

$$\|\mathbf{A}\|_\infty = \max_{\mathbf{x} \in \mathbb{R}^n, \|\mathbf{x}\|_\infty=1} \|\mathbf{Ax}\|_\infty \geq \|\mathbf{Az}\|_\infty \geq \sum_{j=1}^n a_{i,j} sg(a_{i,j}) = \sum_{j=1}^n |a_{i,j}| = \max_{1 \leq i \leq n} \left\{ \sum_{j=1}^n |a_{ij}| \right\},$$

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

$$|[\mathbf{Ax}]_i| = \left| \sum_{j=1}^n a_{i,j} x_j \right| \leq \sum_{j=1}^n |a_{i,j}| \leq \max_{1 \leq i \leq n} \left\{ \sum_{j=1}^n |a_{ij}| \right\},$$

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 $|e^{i\theta}| = 1$. \square

Análogamente para la norma-1 se tiene

EJEMPLO 3.3.4. Consideremos en \mathbb{K}^n la norma $\|\cdot\|_1$. Para $\mathbf{A} \in \mathbb{K}^{n \times n}$ se tiene¹²

$$\|\mathbf{A}\|_1 = \max_{1 \leq j \leq n} \left\{ \sum_{i=1}^n |a_{ij}| \right\}.$$

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 3.2.6). Supongamos que $\mathbf{Q} \in \mathbb{K}^{n \times n}$ y $\mathbf{Q}^* \mathbf{Q} = \mathbf{I}$, luego, para todo $\mathbf{x} \in \mathbb{K}^n$

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

es decir que

$$(3.9) \quad \|\mathbf{Qx}\|_2 = \|\mathbf{x}\|_2, \quad \forall \mathbf{x} \in \mathbb{K}^n.$$

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

$$(3.10) \quad \langle \mathbf{Qv}, \mathbf{Qw} \rangle = (\mathbf{Qv})^* \mathbf{Qw} = \mathbf{v}^* \mathbf{w} = \langle \mathbf{v}, \mathbf{w} \rangle.$$

¹²También vale en el caso de matrices rectangulares.

Estas propiedades definen a los *movimientos rígidos*. De (3.9), tenemos

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

lo que les da su nombre a las matrices unitarias.

EJERCICIO 3.3.5. Verificar que la matriz

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

es una matriz ortogonal para cualquier valor de α . Interpretar geométricamente la transformación lineal \mathbf{Ax} .

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 3.3.6. Dada $\mathbf{A} \in \mathbb{K}^{n \times m}$ definimos la norma de Frobenius

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

Notar que

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

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 (3.7). La norma de Frobenius es una excepción, gracias a la desigualdad de Cauchy-Schwarz. En efecto, dados¹³ $\mathbf{A}, \mathbf{B} \in \mathbb{K}^{n \times n}$, consideremos el elemento i, j de la matriz producto

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

donde hemos usado la Proposición 3.1.2 en la última desigualdad. Sumando en i y en j se tiene el resultado deseado

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

esto es

$$\|\mathbf{AB}\|_F \leq \|\mathbf{A}\|_F \|\mathbf{B}\|_F.$$

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.

¹³El siguiente argumento vale también para matrices rectangulares.

PROPOSICIÓN 3.3.7. Sea $\mathbf{A} \in \mathbb{K}^{n \times m}$, $\mathbf{Q} \in \mathbb{K}^{n \times n}$, \mathbf{Q} unitaria/ortogonal, resulta

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

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

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

$$\|\mathbf{QA}\|_2 = \sup_{\mathbf{v}, \|\mathbf{v}\|_2=1} \|\mathbf{QA}\mathbf{v}\|_2 = \sup_{\mathbf{v}, \|\mathbf{v}\|=1} \|\mathbf{Av}\|_2 = \|\mathbf{A}\|_2.$$

Finalmente, para (4) escribimos

$$\|\mathbf{QA}\|_F^2 = \text{tr}((\mathbf{QA})^*(\mathbf{QA})) = \text{tr}(\mathbf{A}^*\mathbf{A}) = \|\mathbf{A}\|_F^2.$$

□

3.4. Condición de matrices

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

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

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

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

El número que está a la derecha es independiente de \mathbf{h} y podemos usarlo como representante de la condición de evaluar \mathbf{A} en \mathbf{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 \mathbf{v} , ¿qué podríamos hacer? En el caso en que $n = m$ y \mathbf{A} sea invertible, podemos llamar $\mathbf{w} = \mathbf{Av}$ y el cociente $\frac{\|\mathbf{v}\|}{\|\mathbf{Av}\|}$ puede acotarse independiente de \mathbf{v}

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

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

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

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

DEFINICIÓN 3.4.1. (Condición de una matriz) Dada $\mathbf{A} \in \mathbb{K}^{n \times n}$ invertible definimos la condición de \mathbf{A} , $\kappa(\mathbf{A})$, asociada a una norma subordinada $\|\cdot\|$, como

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

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

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

OBSERVACIÓN 3.4.2. 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 $\mathbf{0} \neq \alpha \in \mathbb{K}$, $\kappa(\alpha\mathbf{A}) = \kappa(\mathbf{A})$.
3. $\kappa(\mathbf{I}) = 1$ para toda norma.
4. Como consecuencia del ítem previo, para toda norma, $1 \leq \kappa(\mathbf{A})$.
5. Para toda \mathbf{Q} unitaria/ortogonal $\kappa(\mathbf{Q}) = 1$. (ver ítem (2) en Proposición 3.3.7)
6. $\kappa(\mathbf{A}) = \kappa(\mathbf{A}^{-1})$

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

$$\mathbf{Ax} = \mathbf{b}$$

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

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

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

Tenemos, usando que $\mathbf{Ax} = \mathbf{b}$

$$\Delta\mathbf{Ax} + \mathbf{A}\Delta\mathbf{x} + \Delta\mathbf{A}\Delta\mathbf{x} = \Delta\mathbf{b}.$$

Procediendo *informalmente*, despreciamos el término de “segundo orden” $\Delta\mathbf{A}\Delta\mathbf{x}$ y resulta

$$\Delta\mathbf{Ax} + \mathbf{A}\Delta\mathbf{x} \sim \Delta\mathbf{b}.$$

de donde

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

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

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

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

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

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

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

$$\frac{\|\Delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \kappa(\mathbf{A}) \frac{\|\Delta\mathbf{b}\|}{\|\mathbf{b}\|}.$$

De hecho, escribiendo $\mathbf{A}^{-1}\mathbf{b} = \mathbf{x}$, y $\mathbf{A}^{-1}(\mathbf{b} + \Delta\mathbf{b}) = \mathbf{x} + \Delta\mathbf{x}$ se puede argumentar con \mathbf{A}^{-1} y por (6) de la Observación 3.4.2, se ve que

$$\frac{\|\Delta\mathbf{b}\|}{\|\mathbf{b}\|} \leq \kappa(\mathbf{A}) \frac{\|\Delta\mathbf{x}\|}{\|\mathbf{x}\|}.$$

Resumiendo

$$\frac{1}{\kappa(\mathbf{A})} \frac{\|\Delta\mathbf{b}\|}{\|\mathbf{b}\|} \leq \frac{\|\Delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \kappa(\mathbf{A}) \frac{\|\Delta\mathbf{b}\|}{\|\mathbf{b}\|}.$$

EJERCICIO 3.4.3. 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 $\log_{10}(\kappa(\mathbf{A}))$ dígitos significativos al resolver el sistema¹⁴.

El siguiente resultado da una idea cualitativa de $\kappa(\mathbf{A})$ como el recíproco de la distancia relativa de \mathbf{A} a las matrices singulares.

TEOREMA 3.4.4. $\mathbf{A} \in \mathbb{R}^{n \times n}$, se tiene,

$$(3.11) \quad \frac{1}{\kappa(\mathbf{A})} = \inf_{\mathbf{B} \text{ singular}} \frac{\|\mathbf{A} - \mathbf{B}\|}{\|\mathbf{A}\|},$$

y en donde asumimos $\frac{1}{\kappa(\mathbf{A})} = 0$ en caso de que \mathbf{A} no sea invertible.

DEMOSTRACIÓN. Si \mathbf{A} es no invertible el resultado es evidente. Supongamos entonces que \mathbf{A} es invertible. Probaremos únicamente que vale el \leq en (3.11).

Sea $\mathbf{B} \in \mathbb{R}^{n \times n}$ una matriz singular cualquiera. Tomemos $\mathbf{0} \neq \mathbf{v} \in \text{Ker}(\mathbf{B})$, y definamos $\mathbf{0} \neq \mathbf{w} = \mathbf{A}\mathbf{v}$. Se tiene

$$0 \neq \|\mathbf{w}\| = \|(\mathbf{A} - \mathbf{B})\mathbf{v}\| \leq \|\mathbf{A} - \mathbf{B}\| \|\mathbf{A}^{-1}\mathbf{w}\| \leq \|\mathbf{A} - \mathbf{B}\| \|\mathbf{A}^{-1}\| \|\mathbf{w}\|,$$

dividiendo por $\|\mathbf{A}\| \|\mathbf{A}^{-1}\| \|\mathbf{w}\| \neq 0$ se obtiene

$$\frac{1}{\kappa(\mathbf{A})} \leq \frac{\|\mathbf{A} - \mathbf{B}\|}{\|\mathbf{A}\|},$$

y por ser \mathbf{B} arbitrario

$$\frac{1}{\kappa(\mathbf{A})} \leq \inf_{\mathbf{B} \text{ singular}} \frac{\|\mathbf{A} - \mathbf{B}\|}{\|\mathbf{A}\|},$$

□

¹⁴Por ejemplo, si $\kappa(\mathbf{A}) \sim 10^{16}$ 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 3.4.2 dice que el número de condición no cambia por múltiplos de una matriz sin embargo $\det(\alpha \mathbf{A}) = \alpha^n \det(\mathbf{A})$ que tiende a cero si $\alpha \rightarrow 0$.

3.5. Proyecciones y proyectores

Observemos otra forma equivalente de escribir la proyección ortogonal de \mathbf{w} sobre el subespacio generado por \mathbf{v} (ver (3.4)):

$$(3.12) \quad \mathbf{P}_v(\mathbf{w}) = \frac{\mathbf{v}}{\|\mathbf{v}\|_2} \frac{\mathbf{v}^*}{\|\mathbf{v}\|_2} \mathbf{w},$$

que puede verse como una aplicación lineal de matriz $\frac{\mathbf{v}\mathbf{v}^*}{\|\mathbf{v}\|_2\|\mathbf{v}\|_2}$.

Notar que la matriz $\frac{\mathbf{v}\mathbf{v}^*}{\|\mathbf{v}\|_2\|\mathbf{v}\|_2}$, asociada a la proyección \mathbf{P}_v , tiene rango 1, de hecho las columnas son todas múltiplos de \mathbf{v} .

Un caso notable es cuando \mathbf{v} es de norma uno. En ese caso

$$(3.13) \quad \mathbf{P}_v(\mathbf{w}) = \mathbf{v}\mathbf{v}^*\mathbf{w},$$

teniendo esto en mente volvamos un segundo a la ecuación (3.2). Si en vez de una base completa solo contamos con un conjunto de vectores ortonormales $\mathcal{C} = \{\mathbf{v}_1, \dots, \mathbf{v}_k\} \subset \mathbb{K}^n$, $k < n$, y llamamos $\mathcal{S} = \langle \mathbf{v}_1, \dots, \mathbf{v}_k \rangle$ al subespacio generado por esos vectores, vemos que la expresión

$$(3.14) \quad \mathbf{P}_{\mathcal{S}}(\mathbf{w}) = \sum_{1 \leq i \leq k} \mathbf{P}_{\mathbf{v}_i}(\mathbf{w}).$$

verifica

$$(\mathbf{w} - \mathbf{P}_{\mathcal{S}}(\mathbf{w})) \perp \mathbf{v}_j, \quad \forall 1 \leq j \leq k,$$

pues

$$\mathbf{v}_j^*(\mathbf{w} - \mathbf{P}_{\mathcal{S}}(\mathbf{w})) = \mathbf{v}_j^*\mathbf{w} - \sum_{1 \leq i \leq k} \mathbf{v}_j^* \mathbf{P}_{\mathbf{v}_i}(\mathbf{w}) = \mathbf{v}_j^*\mathbf{w} - \mathbf{v}_j^*\mathbf{w} = 0.$$

Luego, por ser ortogonal a todos los generadores de \mathcal{S} lo será a todos los elementos de \mathcal{S} . En otras palabras, teniendo una base ortonormal de \mathcal{S} es muy fácil construir la proyección ortogonal de un elemento \mathbf{w} sobre \mathcal{S} usando la expresión (3.14). Como además cada $\mathbf{P}_{\mathbf{v}_i}$ admite una representación matricial, lo mismo ocurre con $\mathbf{P}_{\mathcal{S}}$,

$$\mathbf{P}_{\mathcal{S}} = \left(\begin{array}{c|c|c|c} \mathbf{v}_1 & | & \mathbf{v}_2 & | \\ \hline & | & | & | \\ & \cdots & & \mathbf{v}_k \end{array} \right) \left(\begin{array}{cccccccc} & & & & \mathbf{v}_1^* & & & \\ & - & - & - & - & - & - & - \\ & & & & \mathbf{v}_2^* & & & \\ & - & - & - & - & - & - & - \\ & & & & \vdots & & & \\ & - & - & - & - & - & - & - \end{array} \right).$$

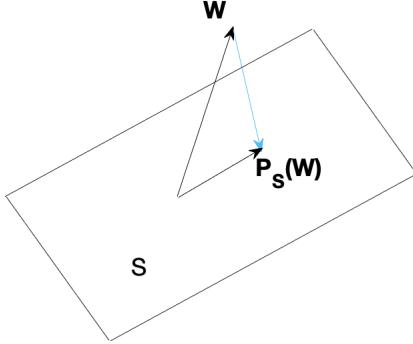


FIGURA 4. Proyección ortogonal de \mathbf{w} sobre el subespacio \mathcal{S} .

En efecto, tomando $w \in \mathbb{K}^n$ tenemos, llamando M a la matriz anterior que

$$M\mathbf{w} = \begin{pmatrix} \mathbf{v}_1 & | & \mathbf{v}_2 & | & \cdots & | & \mathbf{v}_k \end{pmatrix} \begin{pmatrix} \mathbf{v}_1^* \mathbf{w} \\ \mathbf{v}_2^* \mathbf{w} \\ \vdots \\ \mathbf{v}_k^* \mathbf{w} \end{pmatrix} = \sum_{1 \leq i \leq k} \mathbf{v}_i \mathbf{v}_i^* \mathbf{w} = \sum_{1 \leq i \leq k} P_{\mathbf{v}_i}(\mathbf{w}).$$

es decir que coincide con (3.14).

Por lo tanto, si queremos obtener la proyección ortogonal de un vector sobre un subespacio \mathcal{S} podemos hacerlo a través de una base ortonormal de \mathcal{S} . Sabemos que obtener una familia de generadores de \mathcal{S} , *i.e.* es relativamente sencillo. Sería ideal tener un método para obtener, a partir de esa familia, un sistema de generadores ortonormales. Veamos como hacerlo.

Dada una familia de vectores $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n\}$ linealmente independientes. La ortonormalización de Gram-Schmidt es un algoritmo que produce una sucesión de vectores ortonormales $\{\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_n\}$ del siguiente modo. El proceso consiste en restarle a cada vector \mathbf{v}_i sus componentes en los vectores anteriores.

Podemos construir primero una familia ortogonal y luego normalizarla (o normalizarla durante la construcción). Llamemos \mathbf{u}_i a los ortogonalizados, el método es el siguiente

- $\mathbf{u}_1 = \mathbf{v}_1, \mathbf{u}_1 = \frac{\mathbf{u}_1}{\|\mathbf{u}_1\|}$
- $\mathbf{u}_2 = \mathbf{v}_2 - P_{\mathbf{u}_1}(\mathbf{v}_2), \mathbf{u}_2 = \frac{\mathbf{u}_2}{\|\mathbf{u}_2\|}$
- $\mathbf{u}_3 = \mathbf{v}_3 - P_{\mathbf{u}_1}(\mathbf{v}_3) - P_{\mathbf{u}_2}(\mathbf{v}_3), \mathbf{u}_3 = \frac{\mathbf{u}_3}{\|\mathbf{u}_3\|}$
- \dots
- $\mathbf{u}_k = \mathbf{v}_k - \sum_{i=1}^{k-1} P_{\mathbf{u}_i}(\mathbf{v}_k), \mathbf{u}_k = \frac{\mathbf{u}_k}{\|\mathbf{u}_k\|}$.

Notemos que el algoritmo solo puede detenerse (es decir alcanzar una operación inválida) si y solo si algún $u_j = \mathbf{0}$ para algún j . Pero si esto ocurre, significa que \mathbf{v}_j es combinación lineal de

$\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{j-1}\}$ ¹⁵ lo cual contradice el presupuesto de ser $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n\}$ linealmente independientes.

Escribimos debajo el algoritmo de forma compacta llamando \mathbf{q}_i a los vectores ortonormales obtenidos¹⁶.

```

Gramm-Schmidt
for j=1 to n
     $\mathbf{v}_j = \mathbf{a}_j$ 
    for i=1 to j-1
         $r_{i,j} = \mathbf{q}_i^* \mathbf{a}_j$ 
         $\mathbf{v}_j = \mathbf{v}_j - r_{i,j} \mathbf{q}_i$ 
     $r_{j,j} = \|\mathbf{v}_j\|_2$ 
     $\mathbf{q}_j = \mathbf{v}_j / r_{j,j}$ 
```

Esta familia de vectores verifica,

- $\{\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_k\}$ es ortonormal.
- $\langle \mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_k \rangle = \langle \mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k \rangle$ para todo $1 \leq k \leq n$.

Ya hemos notado que las proyecciones ortogonales de vectores sobre subespacios resultaban aplicaciones lineales. Vamos a describir las matrices asociadas a esas aplicaciones.

DEFINICIÓN 3.5.1. Una matriz $\mathbf{0} \neq \mathbf{P} \in \mathbb{K}^{n \times n}$ se dice un proyector si $\mathbf{P}^2 = \mathbf{P}$.

Notar que dados dos subespacios S_1, S_2 tales que $S_1 \oplus S_2 = \mathbb{K}^n$, siempre es posible construir un proyector \mathbf{P} tal que $Im(\mathbf{P}) = S_1$ y $Nu(\mathbf{P}) = S_2$. En efecto, como los subespacios están en suma directa, para todo $\mathbf{v} \in \mathbb{C}^n$ existen únicos $\mathbf{v}_i \in S_i$ tales que $\mathbf{v} = \mathbf{v}_1 + \mathbf{v}_2$. Definiendo $\mathbf{Pv} = \mathbf{v}_1$ obtenemos el proyector buscado.

Si \mathbf{P} es un proyector y $\mathbf{P} \neq \mathbf{I}$ entonces $\mathbf{I} - \mathbf{P}$ también es un proyector denominado *proyector complementario*.

Un proyector \mathbf{P} deja invariante su rango, ya que si $\mathbf{v} \in Rg(\mathbf{P})$ entonces $\exists \mathbf{w}$ tal que $\mathbf{Pw} = \mathbf{v}$, por lo que

$$\mathbf{Pv} = \mathbf{PPw} = \mathbf{Pw} = \mathbf{v},$$

decimos que \mathbf{P} proyecta sobre su rango $Rg(\mathbf{P})$.

Notemos que

$$Rg(\mathbf{I} - \mathbf{P}) = Ker(\mathbf{P}),$$

ya que si $\mathbf{w} \in Rg(\mathbf{I} - \mathbf{P})$, es $\mathbf{w} = \mathbf{v} - \mathbf{Pv}$ de donde $\mathbf{Pw} = \mathbf{0}$, i.e. $\mathbf{w} \in Ker(\mathbf{P})$. Por otro lado, si $\mathbf{w} \in Ker(\mathbf{P})$, $Rg(\mathbf{I} - \mathbf{P}) \ni (\mathbf{I} - \mathbf{P})\mathbf{w} = \mathbf{w}$. El argumento anterior también implica

$$Rg(\mathbf{P}) = Ker(\mathbf{I} - \mathbf{P}).$$

Por otro lado, se observa trivialmente que $Ker(\mathbf{I} - \mathbf{P}) \cap Ker(\mathbf{P}) = \{\mathbf{0}\}$ y entonces

$$Rg(\mathbf{P}) \cap Ker(\mathbf{P}) = \{\mathbf{0}\}.$$

¹⁵En efecto, observar que mientras no se detenga el algoritmo $\langle \mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_j \rangle = \langle \mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_j \rangle$ para todo $1 \leq j$.

¹⁶Notación compatible con \mathbf{Q} para las matrices ortogonales/unitarias.

Esto dice que

$$\mathbb{K}^n = \text{Ker}(\mathbf{P}) \oplus \text{Rg}(\mathbf{P}).$$

DEFINICIÓN 3.5.2. Un proyector \mathbf{P} se dice *ortogonal* si $\text{Ker}(\mathbf{P}) \perp \text{Rg}(\mathbf{P})$. En otro caso, se dice *oblícuo*.

PROPOSICIÓN 3.5.3. Un proyector \mathbf{P} es ortogonal si y solo si $\mathbf{P} = \mathbf{P}^*$.

DEMOSTRACIÓN. Si \mathbf{P} es ortogonal, $\forall \mathbf{v}, \mathbf{w}$

$$0 = ((\mathbf{I} - \mathbf{P})\mathbf{v})^* \mathbf{P}\mathbf{w} = \mathbf{v}^* (\mathbf{I} - \mathbf{P})^* \mathbf{P}\mathbf{w},$$

lo que indica que

$$\mathbf{0} = (\mathbf{I} - \mathbf{P})^* \mathbf{P} = \mathbf{P} - \mathbf{P}^* \mathbf{P},$$

es decir $\mathbf{P} = \mathbf{P}^* \mathbf{P}$ y en particular resulta $\mathbf{P} = \mathbf{P}^*$.

Recíprocamente, si $\mathbf{P} = \mathbf{P}^*$,

$$((\mathbf{I} - \mathbf{P})\mathbf{v})^* \mathbf{P}\mathbf{w} = \mathbf{v}^* (\mathbf{I} - \mathbf{P})^* \mathbf{P}\mathbf{w} = \mathbf{v}^* (\mathbf{P} - \mathbf{P}^2)\mathbf{w} = 0,$$

$\forall \mathbf{v}, \mathbf{w}$, lo que dice que \mathbf{P} es ortogonal. \square

TEOREMA 3.5.4. Sea \mathbf{P} un proyector. Son equivalentes:

1. \mathbf{P} es ortogonal
2. $\|\mathbf{v}\|_2 = \|(\mathbf{I} - \mathbf{P})\mathbf{v}\|_2 + \|\mathbf{P}\mathbf{v}\|_2$ para todo \mathbf{v}
3. $\|\mathbf{P}\|_2 = 1$

DEMOSTRACIÓN. (1) \rightarrow (2): para todo \mathbf{v} escribimos

$$\mathbf{v} = (\mathbf{I} - \mathbf{P})\mathbf{v} + \mathbf{P}\mathbf{v},$$

y como por ser proyector $\text{Im}(\mathbf{I} - \mathbf{P}) = \text{Nu}(\mathbf{P}) \perp \text{Im}(\mathbf{P})$, se tiene inmediatamente

$$\|\mathbf{v}\|_2 = \|(\mathbf{I} - \mathbf{P})\mathbf{v}\|_2 + \|\mathbf{P}\mathbf{v}\|_2.$$

(2) \rightarrow (3): la identidad de (2) dice que $\|\mathbf{P}\mathbf{v}\|_2 \leq \|\mathbf{v}\|_2$, de donde $\|\mathbf{P}\|_2 \leq 1$. Como \mathbf{P} es proyector, $\mathbf{P} \neq \mathbf{0}$ y existe $\mathbf{0} \neq \mathbf{w}$ tal que $\mathbf{v} = \mathbf{P}\mathbf{w} \neq \mathbf{0}$, para ese \mathbf{v} , $\mathbf{P}\mathbf{v} = \mathbf{v} \neq \mathbf{0}$ y luego $\|\mathbf{P}\|_2 \geq 1$, lo dice $\|\mathbf{P}\|_2 = 1$.

(3) \rightarrow (1): asumamos que existen $\mathbf{v} \in \text{Im}(\mathbf{P})$ y $\mathbf{w} \in \text{Nu}(\mathbf{P})$ tales que $\mathbf{v}^* \mathbf{w} \neq 0$ y $\|\mathbf{v}\| = 1 = \|\mathbf{w}\|$. Definamos $\mathbf{u} = \mathbf{v} - (\mathbf{w}^* \mathbf{v})\mathbf{w}$, se tiene $\|\mathbf{P}\mathbf{u}\| = \|\mathbf{P}\mathbf{v}\| = \|\mathbf{v}\| = 1$ como $\|\mathbf{P}\| = 1$ resulta

$$1 = \|\mathbf{P}\mathbf{u}\| \leq \|\mathbf{u}\|^2 = \|\mathbf{v}\|^2 - (\mathbf{w}^* \mathbf{v})(\mathbf{v}^* \mathbf{w}) - \overline{(\mathbf{w}^* \mathbf{v})} \mathbf{w}^* \mathbf{v} + |\mathbf{w}^* \mathbf{v}|^2 \|\mathbf{w}\|^2 =$$

$$1 - \overline{(\mathbf{v} \mathbf{w}^*)} (\mathbf{v}^* \mathbf{w}) - \overline{(\mathbf{w}^* \mathbf{v})} \mathbf{w}^* \mathbf{v} + |\mathbf{w}^* \mathbf{v}|^2 = 1 - (\mathbf{w}^* \mathbf{v})^2 < 1,$$

absurdos. Luego $\text{Im}(\mathbf{P}) \perp \text{Nu}(\mathbf{P})$ y \mathbf{P} es ortogonal. \square

OBSERVACIÓN 3.5.5. Dada una matriz $\mathbf{A} \in \mathbb{C}^{n \times k}$ con columnas *ortonormales*, se tiene que $\mathbf{P} = \mathbf{A}\mathbf{A}^* \in \mathbb{C}^{n \times n}$ es un proyector ortogonal, ya que al ser $\mathbf{A}^*\mathbf{A} = \mathbf{I} \in \mathbb{C}^{k \times k}$, resulta

$$\mathbf{P}^2 = \mathbf{A}(\mathbf{A}^*\mathbf{A})\mathbf{A}^* = \mathbf{A}\mathbf{A}^* = \mathbf{P},$$

i.e. un proyector, además ortogonal por ser $\mathbf{P}^* = \mathbf{P}$ (por Proposición 3.5.3).

Llamando \mathbf{q}_i , $1 \leq i \leq k$, a la columna i de \mathbf{A} , resulta para cada $\mathbf{v} \in \mathbb{C}^n$, la expresión

$$\mathbf{P}\mathbf{v} = \mathbf{A}\mathbf{A}^*\mathbf{v} = \sum_{1 \leq i \leq k} (\mathbf{q}_i^*\mathbf{v})\mathbf{q}_i$$

que clarifica el efecto de multiplicar por \mathbf{P} .

Vamos a construir una herramienta de mucha utilidad. Notemos que para todo \mathbf{v} de norma uno (de no ser de norma uno dedemos dividir por $\mathbf{v}^*\mathbf{v}$), podemos definir el proyector ortogonal de rango uno

$$\mathbf{P}_{\mathbf{v}} = \mathbf{v}\mathbf{v}^*,$$

que proyecta sobre el subespacio generado por \mathbf{v} , y con este su complementario

$$\mathbf{P}_{\mathbf{v}^\perp} = \mathbf{I} - \mathbf{P}_{\mathbf{v}},$$

de rango $n - 1$ que proyecta sobre el ortogonal a \mathbf{v} .

Como aplicación de estos proyectores podemos reformular Gramm-Schmidt (recordemos que las columnas de \mathbf{A} se donotan con \mathbf{a}_i).

```

Gram-Schmidt Modificado
for j=1 to n
     $\mathbf{v}_j = \mathbf{a}_j$ 
    for j=1 to n
         $r_{j,j} = \|\mathbf{v}_j\|$ 
         $\mathbf{q}_j = \mathbf{v}_j / r_{j,j}$ 
        for i=j+1 to n
             $r_{j,i} = \mathbf{q}_j^* \mathbf{v}_i$ 
             $\mathbf{v}_j = \mathbf{v}_j - r_{j,i} \mathbf{q}_j$ 
```

esta versión modificada, más estable que la original como veremos con un ejemplo, puede ponerse en términos de proyectores de rango $n - 1$. En efecto, habiendo calculado $\{\mathbf{v}_1, \dots, \mathbf{v}_{j-1}\}$ y $\{\mathbf{q}_1, \dots, \mathbf{q}_{j-1}\}$, el vector \mathbf{v}_j

$$\mathbf{v}_j = \mathbf{P}_{\mathbf{q}_{j-1}^\perp} \mathbf{P}_{\mathbf{q}_{j-2}^\perp} \cdots \mathbf{P}_{\mathbf{q}_1^\perp} \mathbf{a}_j.$$

se obtiene de la aplicación sucesiva de proyectores de este tipo.

OBSERVACIÓN 3.5.6. Gram-Schmidt es ampliamente utilizado en la forma descripta, su costo es $\sim 2n^3$.

Capítulo 4

Métodos Directos Para Sistemas Lineales

4.1. Sistemas y Factorización de Matrices

Comenzaremos trabajando con matrices cuadradas, i.e. $\mathbf{A} \in \mathbb{R}^{n \times n}$ (o más en general $\mathbb{K}^{n \times n}$ ya que la mayoría de las consideraciones se aplican a los complejos).

Recordemos que los vectores $\mathbf{v} \in \mathbb{K}^n$ los identificamos con matrices columna de $n \times 1$ de donde tiene sentido $\mathbf{A}\mathbf{v}$.

Factorizar significa descomponer una expresión en factores de algún modo más simples o que nos otorguen algún beneficio teórico o práctico. La factorización LU consiste en escribir una matriz invertible \mathbf{A} como producto de dos factores: uno triangular inferior \mathbf{L} y otro triangular superior \mathbf{U} , es decir

$$\mathbf{L} = \begin{pmatrix} * & 0 & \cdots & 0 \\ * & * & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ * & * & \cdots & * \end{pmatrix}, \mathbf{U} = \begin{pmatrix} * & * & \cdots & * \\ 0 & * & \cdots & * \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & * \end{pmatrix}$$

y

$$\mathbf{A} = \mathbf{LU}.$$

Un beneficio inmediato de esta factorización es que para resolver un sistema

$$\mathbf{Ax} = \mathbf{b},$$

basta resolver

$$\mathbf{Ly} = \mathbf{b},$$

$$\mathbf{Ux} = \mathbf{y},$$

y cada sistema triangular puede resolverse por sustitución (regresiva o progresiva) en $\sim n^2$ operaciones¹.

EJERCICIO 4.1.1. Escriba un algoritmo para resolver sistemas triangulares con orden n^2 .

4.2. Descomposición $\mathbf{LU} = \mathbf{A}$

La factorización LU es un subproducto del método de eliminación de Gauss. Dada la matriz

$$\mathbf{A} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix}$$

¹De 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 \mathbf{A} para generar ceros debajo del elemento $a_{1,1}$. Esto se hace fila a fila multiplicando la fila 1 de \mathbf{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 \mathbf{A} por el multiplicador \mathbf{M}_1

$$\mathbf{M}_1 = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ -\frac{a_{2,1}}{a_{1,1}} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -\frac{a_{n,1}}{a_{1,1}} & 0 & \cdots & 1 \end{pmatrix}$$

El efecto neto del producto resulta

$$\mathbf{M}_1 \mathbf{A} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ 0 & a_{2,2}^{(1)} & a_{2,3}^{(1)} & \cdots & a_{2,n}^{(1)} \\ 0 & a_{3,2}^{(1)} & a_{3,3}^{(1)} & \cdots & a_{3,n}^{(1)} \\ \vdots & \vdots & \ddots & & \vdots \\ 0 & a_{n,2}^{(1)} & a_{n,3}^{(1)} & \cdots & a_{n,n}^{(1)} \end{pmatrix}$$

donde se destaca con un superíndice (1) una nueva matriz $\mathbf{A}^{(1)} \in \mathbb{R}^{(n-1) \times (n-1)}$

$$\mathbf{A}^{(1)} = \begin{pmatrix} a_{2,2}^{(1)} & \cdots & a_{2,n}^{(1)} \\ \vdots & \vdots & \ddots \\ a_{n,2}^{(1)} & \cdots & a_{n,n}^{(1)} \end{pmatrix}$$

que en caso de tener el pivot $a_{2,2}^{(1)} \neq 0$ admite un nuevo paso en la iteración tomando

$$\mathbf{M}_2 = \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ 0 & -\frac{a_{3,2}^{(1)}}{a_{2,2}^{(1)}} & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & -\frac{a_{n,2}^{(1)}}{a_{2,2}^{(1)}} & 0 & \cdots & 1 \end{pmatrix}$$

de donde

$$\mathbf{M}_2 \mathbf{M}_1 \mathbf{A} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ 0 & a_{2,2}^{(1)} & a_{2,3}^{(1)} & \cdots & a_{2,n}^{(1)} \\ 0 & 0 & a_{3,3}^{(2)} & \cdots & a_{3,n}^{(2)} \\ \vdots & \vdots & \ddots & & \vdots \\ 0 & 0 & a_{n,3}^{(2)} & \cdots & a_{n,n}^{(2)} \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ 0 & a_{2,2}^{(1)} & a_{2,3}^{(1)} & \cdots & a_{2,n}^{(1)} \\ 0 & 0 & 0 & & \mathbf{A}^{(2)} \\ \vdots & \vdots & \vdots & & \\ 0 & 0 & 0 & & \end{pmatrix},$$

con $\mathbf{A}^{(2)} \in \mathbb{R}^{(n-2) \times (n-2)}$. Este procedimiento, dado que cada \mathbf{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 \mathbf{U}

invertible

$$\mathbf{M}_{n-1}\mathbf{M}_{n-2}\cdots\mathbf{M}_1\mathbf{A} = \mathbf{U} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} & \cdots & a_{1,n} \\ 0 & a_{2,2}^{(1)} & a_{2,3}^{(1)} & a_{2,4}^{(1)} & \cdots & a_{2,n}^{(1)} \\ 0 & 0 & a_{3,3}^{(2)} & a_{3,4}^{(2)} & \cdots & a_{3,n}^{(2)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & a_{n,n}^{(n-1)} \end{pmatrix}$$

Observemos que la productoria

$$\mathbf{M}_{n-1}\cdots\mathbf{M}_2\mathbf{M}_1 = \mathbf{M},$$

es una matriz triangular inferior, dado que es producto de matrices triangulares inferiores. La inversa de \mathbf{M} , es por lo tanto triangular inferior. La llamaremos $\mathbf{L} = \mathbf{M}^{-1}$ y por ende

$$\mathbf{A} = \mathbf{LU}.$$

El problema que aparece inmediatamente es que no hemos obtenido \mathbf{L} constructivamente (en el sentido de que aún falta invertir \mathbf{M}). Sin embargo, un hecho notable facilita ese trabajo.

Observemos que cada \mathbf{M}_i tiene la forma

$$(4.1) \quad \mathbf{M}_i = \mathbf{I} - \mathbf{z}_i \mathbf{e}_i^T,$$

donde \mathbf{e}_i representa el i -ésimo canónico y \mathbf{z}_i es de la forma

$$\mathbf{z}_i = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ z_{i+1}^{(i)} \\ \vdots \\ z_n^{(i)} \end{pmatrix},$$

con $z_j^{(i)} = \frac{a_{j,i}^{(i-1)}}{a_{i,i}^{(i-1)}}, i+1 \leq j \leq n$.

LEMA 4.2.1. Sea $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$, tales que $\mathbf{v}^T \mathbf{u} \neq 1$ entonces $\mathbf{I} - \mathbf{u} \mathbf{v}^T$ es invertible y además

$$(\mathbf{I} - \mathbf{u} \mathbf{v}^T)^{-1} = \mathbf{I} + \frac{\mathbf{u} \mathbf{v}^T}{1 - \mathbf{v}^T \mathbf{u}}.$$

DEMOSTRACIÓN.

$$(\mathbf{I} - \mathbf{u} \mathbf{v}^T)(\mathbf{I} + \frac{\mathbf{u} \mathbf{v}^T}{1 - \mathbf{v}^T \mathbf{u}}) = \mathbf{I} + \mathbf{u} \mathbf{v}^T (\frac{1}{1 - \mathbf{v}^T \mathbf{u}} - 1) - \frac{\mathbf{u} \mathbf{v}^T \mathbf{u} \mathbf{v}^T}{1 - \mathbf{v}^T \mathbf{u}} = \mathbf{I}.$$

□

Considerando el lema anterior y observando que en (4.1) se tiene $\mathbf{e}_i^T \mathbf{z}_i = 0$, resulta ²

$$\mathbf{M}_i^{-1} = \mathbf{I} + \mathbf{z}_i \mathbf{e}_i^T,$$

²Eso decir que el costo de invertir \mathbf{M}_i es casi nulo (apenas un cambio de signos debajo de la diagonal).

de donde

$$\mathbf{L} = \mathbf{M}_1^{-1} \mathbf{M}_2^{-1} \cdots \mathbf{M}_{(n-1)}^{-1} = (\mathbf{I} + \mathbf{z}_1 \mathbf{e}_1^T) (\mathbf{I} + \mathbf{z}_2 \mathbf{e}_2^T) \cdots (\mathbf{I} + \mathbf{z}_{n-1} \mathbf{e}_{n-1}^T).$$

Ademas de la sencillez en el cómputo de \mathbf{M}_i^{-1} hay otro hecho “afortunado” que permite expresar \mathbf{L} sin cálculos adicionales. Como $\mathbf{e}_i^T \mathbf{z}_k = 0$ no solo para $k = i$ sino para todo $i \leq k \leq n$, se puede desarrollar la expresión para \mathbf{L} y obtener

$$\mathbf{L} = \mathbf{I} + \sum_{i=1}^{n-1} \mathbf{z}_i \mathbf{e}_i^T,$$

o dicho de otra forma,

$$\mathbf{L} = \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ \frac{a_{2,1}}{a_{1,1}} & 1 & \cdots & 0 & 0 \\ \frac{a_{3,1}}{a_{1,1}} & \frac{a_{3,2}^{(1)}}{a_{2,2}^{(1)}} & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & 1 & \vdots \\ \frac{a_{n,1}}{a_{1,1}} & \frac{a_{n,2}^{(1)}}{a_{2,2}^{(1)}} & \cdots & \frac{a_{n,n-1}^{(n-1)}}{a_{n-1,n-1}^{(n-1)}} & 1 \end{pmatrix},$$

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 $\det(\mathbf{A}) \neq 0$ como se ve en tomando por ejemplo

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Sin embargo puede garantizarse si se requiere que todos los menores de la matriz \mathbf{A} , sean invertibles.

PROPOSICIÓN 4.2.2. Sea $\mathbf{A} \in \mathbb{K}^{n \times n}$, entonces existe factorización LU de \mathbf{A} con \mathbf{L}, \mathbf{U} invertibles sí y solo sí para todo $1 \leq k \leq n$, $\det(\mathbf{A}(1:k, 1:k)) \neq 0$.

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

$$\mathbf{A} = \mathbf{M}_1^{-1} \mathbf{U}_1$$

donde

$$\mathbf{M}_1^{-1} \mathbf{U}_1 = \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ \frac{a_{2,1}}{a_{1,1}} & 1 & \cdots & 0 & 0 \\ \frac{a_{3,1}}{a_{1,1}} & 0 & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & 1 & \vdots \\ \frac{a_{n,1}}{a_{1,1}} & 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ 0 & a_{2,2}^{(1)} & a_{2,3}^{(1)} & \cdots & a_{2,n}^{(1)} \\ 0 & 0 & & & \\ \vdots & \vdots & & & \mathbf{A}^{(2)} \\ 0 & 0 & & & \end{pmatrix}$$

Por lo tanto

$$\mathbf{A}(1 : 2, 1 : 2) = \begin{pmatrix} 1 & 0 \\ \frac{a_{2,1}}{a_{1,1}} & 1 \end{pmatrix} \begin{pmatrix} a_{1,1} & a_{1,2} \\ 0 & a_{2,2}^{(1)} \end{pmatrix}$$

y como por hipótesis $0 \neq \det(\mathbf{A}(1 : 2, 1 : 2))$ tenemos $\det(\mathbf{A}(1 : 2, 1 : 2)) = a_{1,1}a_{2,2}^{(1)} \neq 0$, por lo tanto el segundo pivot no se anula. La demostración se sigue por inducción.

Para la implicación \Rightarrow : Notamos que si $\mathbf{A} = \mathbf{LU}$ con \mathbf{L} y \mathbf{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 $\det(A(1 : k, 1 : k)) \neq 0$. \square

Para ver otro criterio necesitamos una definición.

DEFINICIÓN 4.2.3. Una matriz $\mathbf{A} \in \mathbb{R}^{n \times n}, \mathbb{C}^{n \times n}$ se dice *diagonal dominante* (*estrictamente diagonal dominante*) y se denota EDD (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 4.2.4. Sea $\mathbf{A} \in \mathbb{R}^{n \times n}, \mathbb{C}^{n \times n}$ EDD entonces \mathbf{A} es invertible.

DEMOSTRACIÓN. Supongamos que no. Existe entonces $\mathbf{0} \neq \mathbf{v} \in \mathbb{C}^n$ tal que $\mathbf{Av} = \mathbf{0}$. Sea i , $1 \leq i \leq n$ tal que $0 \neq |v_i| = \|\mathbf{v}\|_\infty$, 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. \square

PROPOSICIÓN 4.2.5. Si \mathbf{A} es EDD entonces, trabajando con aritmética exacta, el proceso de eliminación de Gauss no produce pivots nulos.

DEMOSTRACIÓN. Ejercicio. \square

OBSERVACIÓN 4.2.6. 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 devolverá en general los verdaderos factores \mathbf{L} y \mathbf{U} , sino una aproximación de ellos. Mas explícitamente podemos enunciar el siguiente teorema (ver [5]).

TEOREMA 4.2.7. Sea $\mathbf{A} \in \mathbb{K}^{n \times n}$, si el algoritmo LU no produce pivots nulos, la factorización LU producida por la máquina verifica

$$\tilde{\mathbf{L}}\tilde{\mathbf{U}} = \mathbf{A} + \Delta\mathbf{A}$$

donde $|\Delta\mathbf{A}| \lesssim 2(n-1)\varepsilon(|\mathbf{A}| + |\tilde{\mathbf{L}}||\tilde{\mathbf{U}}|)$.

OBSERVACIÓN 4.2.8. El Teorema 4.2.7, sugiere controlar el tamaño de los factores $\tilde{\mathbf{L}}, \tilde{\mathbf{U}}$. En efecto, si el producto $|\tilde{\mathbf{L}}||\tilde{\mathbf{U}}|$ es muy grande no tendremos control (al menos teórico) sobre el error. El *pivoteo parcial* permite garantizar al menos el control $|\mathbf{L}| \leq 1$ a un costo razonable. Si bien, esto no necesariamente implica que $|\tilde{\mathbf{L}}||\tilde{\mathbf{U}}|$ sea pequeño, resulta suficiente en la mayoría de los casos prácticos (ver sin embargo la Observación 4.3.2).

EJERCICIO 4.2.9. 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 \mathbf{A} sea *rala* -cosa que afortunadamente ocurre en muchas aplicaciones, como en ecuaciones diferenciales- los factores \mathbf{L}, \mathbf{U} pueden no serlo.

EJERCICIO 4.2.10. Cuanta memoria ocuparía una matriz de $10^6 \times 10^6$ llena?.

4.3. Descomposición $LU = PA$ (pivoteo parcial)

Una matriz de permutaciones $\mathbf{P} \in \mathbb{K}^{n \times n}$ 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,

$$\mathbf{P} = \begin{pmatrix} & e_{i_1}^T \\ \cdots & \cdots \\ & e_{i_2}^T \\ & \vdots \\ & \cdots & \cdots \\ & e_{i_n}^T \end{pmatrix}$$

donde $i_1, i_2, \dots, i_n \in \{1, 2, \dots, n\}$ son todos diferentes. Alternativamente podemos pensar una matriz de permutaciones como la identidad con las columnas permutadas

$$\mathbf{P} = (e_{i_1} \mid e_{i_2} \mid \dots \mid e_{i_n}).$$

Notemos que si

$$\mathbf{v} \in \mathbb{K}^n,$$

$$\mathbf{P}\mathbf{v} = \begin{pmatrix} v_{i_1} \\ v_{i_2} \\ \vdots \\ v_{i_n} \end{pmatrix}$$

$$\mathbf{v}^T \mathbf{P} = (v_{i_1}, v_{i_2}, \dots, v_{i_n}).$$

Resumamos las siguientes propiedades de las permutaciones.

- Dada $\mathbf{A} \in \mathbb{K}^{n \times n}$, \mathbf{PA} coincide con \mathbf{A} pero con las filas permutadas y \mathbf{AP} coincide con \mathbf{A} pero con las columnas permutadas.
- Dada $\mathbf{x} \in \mathbb{K}^n$, \mathbf{Px} coincide con \mathbf{x} pero con los elementos permutados.
- $\det(\mathbf{P}) = \pm 1$
- $\mathbf{P}^{-1} = \mathbf{P}^T$
- Si \mathbf{P}_1 y \mathbf{P}_2 son permutaciones entonces $\mathbf{P} = \mathbf{P}_1 \mathbf{P}_2$ 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 \mathbf{A} . Digamos que este es $a_{k,1}$ (si $\det(\mathbf{A}) \neq 0$ se tiene que $a_{k,1} \neq 0$). Permutemos la fila k con la fila 1, lo cual se hace con una matriz \mathbf{P}_1 . Es decir

$$\mathbf{P}_1 \mathbf{A} = \begin{pmatrix} a_{k,1} & a_{k,2} & \cdots & a_{k,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix},$$

a partir de aquí construimos el multiplicador, \mathbf{M}_1 de modo tal que

$$\mathbf{M}_1 \mathbf{P}_1 \mathbf{A} = \begin{pmatrix} a_{k,1} & a_{k,2} & \cdots & a_{k,n} \\ 0 & & & \\ \vdots & & \mathbf{A}^{(1)} & \\ 0 & & & \end{pmatrix}.$$

Llegado a este punto, repetimos el procedimiento con la primer columna de $\mathbf{A}^{(1)}$ que eventualmente requerirá otra matriz \mathbf{P}_2 antes de la construcción del nuevo multiplicador \mathbf{M}_2 . En definitiva,

$$(4.2) \quad \mathbf{M}_{n-1} \mathbf{P}_{n-1} \cdots \mathbf{M}_1 \mathbf{P}_1 \mathbf{A} = \mathbf{U},$$

se obtiene una nueva matriz triangular superior \mathbf{U} , y donde $\mathbf{M}_i = \mathbf{I} - \mathbf{z}_i \mathbf{e}_i^T$ con

$$\mathbf{z}_i = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ z_{i+1}^{(i)} \\ \vdots \\ z_n^{(i)} \end{pmatrix},$$

$|\mathbf{z}_i| \leq 1$ (i.e. los multiplicadores son menores o iguales a 1 en valor absoluto). El problema ahora es que en la forma (4.2) no parece verse el modo de reconstruir los factores \mathbf{L} y \mathbf{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 $\mathbf{e}_j^T \mathbf{P}_k = \mathbf{e}_j^T$ para todo $j < k$. En particular, si bien *no es cierto* que \mathbf{P}_k commuta con \mathbf{M}_j sí es cierto, *para todo* $j < k$, que

$$\mathbf{P}_k \mathbf{M}_j = \mathbf{P}_k (\mathbf{I} - \mathbf{z}_j \mathbf{e}_j^T) = (\mathbf{I} - \mathbf{P}_k \mathbf{z}_j \mathbf{e}_j^T) \mathbf{P}_k = \tilde{\mathbf{M}}_j \mathbf{P}_k,$$

es decir que podemos pasar la permutación a la derecha permutando adecuadamente los multiplicadores de la matriz \mathbf{M}_j . Lo importante es que esta nueva matriz, denotada con $\tilde{\mathbf{M}}_j$ tiene la misma estructura que \mathbf{M}_j . Así que volviendo a (4.2) tenemos

$$\mathbf{M}_{n-1} \mathbf{P}_{n-1} \cdots \mathbf{M}_1 \mathbf{P}_1 \mathbf{A} = \tilde{\mathbf{M}}_{n-1} \cdots \tilde{\mathbf{M}}_1 \mathbf{P}_{n-1} \cdots \mathbf{P}_1 \mathbf{A} = \mathbf{U}.$$

Argumentando como en el caso son pivoteo (usando que la estructura de las $\tilde{\mathbf{M}}_i$ es igual a la de las \mathbf{M}_i) y llamando $\mathbf{P} = \mathbf{P}_{n-1} \cdots \mathbf{P}_1$ se llega a

$$\mathbf{PA} = \mathbf{LU}.$$

PROPOSICIÓN 4.3.1. Trabajando con aritmética exacta se tiene que si $\det(\mathbf{A}) \neq 0$ el algoritmo de pivoteo parcial no tiene pivots nulos.

OBSERVACIÓN 4.3.2. El proceso de pivoteo parcial garantiza que $|\mathbf{L}| \leq 1$ pero no se tiene un control demasiado benigno sobre el tamaño de $|\mathbf{U}|$, 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 $\|\mathbf{U}\|_\infty \geq 2^n \|\mathbf{A}\|_\infty$. Este ejemplo sin embargo parece ser singular en el sentido de que en los casos genéricos el crecimiento de $|\mathbf{U}|$ estaría controlado por \sqrt{n} [5]^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.

^aAl presente no parece haber una demostración rigurosa de esto.

Controlar el tamaño del ambos factores \mathbf{L} y \mathbf{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 $\mathbf{P}_1, \tilde{\mathbf{P}}_1$ antes de aplicar el paso de eliminación. Es decir

$$\mathbf{M}_1 \mathbf{P}_1 \mathbf{A} \tilde{\mathbf{P}}_1 = \begin{pmatrix} * & * & \cdots & * \\ 0 & & & \\ \vdots & & \mathbf{A}^{(1)} & \\ 0 & & & \end{pmatrix}.$$

Repitiendo el procedimiento se obtiene \mathbf{U}

$$\mathbf{M}_{n-1} \mathbf{P}_{n-1} \cdots \mathbf{M}_1 \mathbf{P}_1 \mathbf{A} \tilde{\mathbf{P}}_1 \cdots \tilde{\mathbf{P}}_{n-1} = \mathbf{U},$$

y argumentando como antes se obtienen ahora dos matrices de permutaciones $\mathbf{P}, \tilde{\mathbf{P}}$ tales que

$$\mathbf{P} \mathbf{A} \tilde{\mathbf{P}} = \mathbf{L} \mathbf{U}.$$

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 4.3.3. En aritmética exacta, para el algoritmo de pivoteo completo se tiene

$$\|\mathbf{U}\|_\infty \leq \sqrt{n}(2.3^{1/2}.4^{1/3} \cdots n^{1/(n-1)})^{1/2} \|\mathbf{A}\|_\infty$$

EJERCICIO 4.3.4. Estimar numéricamente el crecimiento del factor $\sqrt{n}(2.3^{1/2}.4^{1/3} \cdots n^{1/(n-1)})^{1/2}$.

4.4. Descomposición 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 4.4.1. Si \mathbf{A} es definida positiva entonces es invertible y también lo son todos sus menores.

DEMOSTRACIÓN. En efecto, de no ser invertible, existiría $\mathbf{0} \neq \mathbf{v}$ tal que $\mathbf{A}\mathbf{v} = \mathbf{0}$ lo que contradice la definida positividad ya que para ese \mathbf{v} se tendría $\mathbf{v}^* \mathbf{A} \mathbf{v} = 0$.

Respecto de los menores el resultado se sigue del anterior. En efecto, sea k fijo, $1 \leq k \leq n$ y tomemos vectores $\mathbf{v} \in \mathbb{K}^n$ de la forma $(\tilde{\mathbf{v}}, 0, \dots, 0)$ con $\tilde{\mathbf{v}} \in \mathbb{K}^k$ arbitrario. Resulta

$$0 < \mathbf{v}^* \mathbf{A} \mathbf{v} = \tilde{\mathbf{v}}^* \mathbf{A}(1:k, 1:k) \tilde{\mathbf{v}},$$

y de ahí la definida positividad de $\mathbf{A}(1:k, 1:k)$ y por ende su invertibilidad. \square

Si \mathbf{A} es definida positiva admite una descomposición $\mathbf{A} = \mathbf{L}\mathbf{U}$ (gracias a las Proposiciones 4.4.1 y 4.2.2) y podemos escribir $\mathbf{U} = \mathbf{D}\tilde{\mathbf{L}}^*$, con $\tilde{\mathbf{L}}$ triangular inferior con 1 en la diagonal y \mathbf{D} una matriz diagonal. De ahí resulta

$$\mathbf{A} = \mathbf{L}\mathbf{D}\tilde{\mathbf{L}}^* = \tilde{\mathbf{L}}\mathbf{D}^*\mathbf{L}^*,$$

entonces

$$\mathbf{D}(\mathbf{L}^{-1}\tilde{\mathbf{L}})^* = \mathbf{L}^{-1}\tilde{\mathbf{L}}\mathbf{D}^*,$$

y como el lado izquierdo de esta ecuación es triangular superior y el derecho triangular inferior deberán ser ambos diagonales. Luego, debe serlo también $\mathbf{L}^{-1}\tilde{\mathbf{L}}$. Por construcción, $\tilde{\mathbf{L}}_{i,i} = 1 = \mathbf{L}_{i,i}$, para todo $1 \leq i \leq n$. Debido a la primera igualdad se tiene en particular que $\mathbf{L}_{i,i}^{-1} = 1$. De aquí resulta $\mathbf{L}^{-1}\tilde{\mathbf{L}} = \mathbf{I}$, es decir $\mathbf{L} = \tilde{\mathbf{L}}$ y además $\mathbf{D} = \mathbf{D}^*$. En particular se obtiene la factorización

$$\mathbf{A} = \mathbf{LDL}^*.$$

Como además \mathbf{A} es definida positiva, entonces $\mathbf{D} > 0$, pues para todo $\mathbf{v} \neq 0$

$$0 < \mathbf{v}^* \mathbf{Av} = \mathbf{v}^* \mathbf{LDL}^* \mathbf{v} = (\mathbf{L}^* \mathbf{v})^* \mathbf{DL}^* \mathbf{v},$$

de donde, llamando $\mathbf{w} = \mathbf{L}^* \mathbf{v}$ y usando que \mathbf{L} (y por ende \mathbf{L}^*) es invertible, resulta la definida positividad de \mathbf{D} , lo que en este caso, por ser diagonal, equivale a $\mathbf{D} > 0$. Esto nos garantiza la existencia de la factorización de Cholesky:

$$\mathbf{A} = \mathbf{LD}^{\frac{1}{2}} \mathbf{D}^{\frac{1}{2}} \mathbf{L}^* = \mathbf{CC}^*,$$

donde \mathbf{C} es triangular inferior con elementos diagonales positivos. La construcción de \mathbf{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 $chol(\mathbf{A})$ devuelve el factor \mathbf{C} .

Ya que $\mathbf{A} = \mathbf{CC}^*$, con \mathbf{C} triangular inferior $c_{i,i} > 0$, (de hecho, ya probamos la existencia de esta factorización). Intentaremos “despejar” \mathbf{C} .

Escribimos

$$\mathbf{C} = \begin{pmatrix} c_{1,1} & | & \mathbf{0} \\ \hline \mathbf{C}(2:n, 1) & | & \mathbf{C}(2:n, 2:n) \end{pmatrix}$$

donde $\mathbf{C}(2:n, 2:n)$ tiene las mismas características que \mathbf{C} . Sabiendo que

$$(4.3) \quad \begin{pmatrix} a_{1,1} & | & \mathbf{A}(2:n, 1)^* \\ \hline \mathbf{A}(2:n, 1) & | & \mathbf{A}(2:n, 2:n) \end{pmatrix} = \begin{pmatrix} c_{1,1} & | & \mathbf{C}(2:n, 1)^* \\ \hline \mathbf{0} & | & \mathbf{C}(2:n, 2:n)^* \end{pmatrix}$$

se tiene, respectivamente, que

$$\textbf{Paso 1} \quad c_{1,1}^2 = a_{1,1} \quad \rightarrow \quad c_{1,1} := \sqrt{a_{1,1}},$$

$$\textbf{Paso 2} \quad \mathbf{A}(2:n, 1) = c_{1,1} \mathbf{C}(2:n, 1) \quad \rightarrow \quad \mathbf{C}(2:n, 1) := \mathbf{A}(2:n, 1)/c_{1,1},$$

lo cual nos permitió “despejar” las incógnitas $\mathbf{C}(1:n, 1)$, puesto que $a_{1,1} > 0$ por hipótesis, lo que permite elegir $c_{1,1} > 0$. Ahora consideramos

$$\textbf{Paso 3} \quad \mathbf{A}(2:n, 2:n) = \mathbf{C}(1:n, 1)\mathbf{C}(1:n, 1)^* + \mathbf{C}(2:n, 2:n)\mathbf{C}(2:n, 2:n)^* \rightarrow$$

³En Python `np.linalg.cholesky()` en Matlab `chol()`.

$$\mathbf{C}(2:n, 2:n) := \text{chol}(\mathbf{A}(2:n, 2:n) - \mathbf{C}(1:n, 1)\mathbf{C}(1:n, 1)^*),$$

de donde el problema de resolver $\text{chol}(\mathbf{A}(1:n, 1:n))$ para una matriz de $n \times n$ se reduce a resolver $\text{chol}(\mathbf{A}(2:n, 2:n) - \mathbf{C}(2:n, 1)\mathbf{C}(2:n, 1)^*)$ 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 4.4.2. Si \mathbf{A} es SDP entonces $\mathbf{A}(2:n, 2:n) - \mathbf{C}(2:n, 1)\mathbf{C}(2:n, 1)^*$, donde $\mathbf{C} = (2:n, 1) = \mathbf{A}(2:n, 1)/\sqrt{a_{1,1}}$, es DP.

DEMOSTRACIÓN. Claramente $\mathbf{A}(2:n, 2:n) - \mathbf{C}(2:n, 1)\mathbf{C}(2:n, 1)^*$ es Hermitiana (o simétrica, en el caso real). Por otro lado, y por hipótesis, para todo $\mathbf{0} \neq \mathbf{u} = (u_1, \mathbf{v}) \in \mathbb{C}^n$ se tiene

$$0 < \mathbf{u}^* \mathbf{A} \mathbf{u} = \begin{pmatrix} u_1 \\ - \\ \mathbf{v} \end{pmatrix}^* \begin{pmatrix} a_{1,1} & | & \mathbf{A}(2:n, 1)^* \\ - & | & - \\ \mathbf{A}(2:n, 1) & | & \mathbf{A}(2:n, 2:n) \end{pmatrix} \begin{pmatrix} u_1 \\ - \\ \mathbf{v} \end{pmatrix} =$$

$$|u_1|^2 a_{1,1} + u_1^* \mathbf{A}(2:n, 1)^* \mathbf{v} + u_1 \mathbf{v}^* \mathbf{A}(2:n, 1) + \mathbf{v}^* \mathbf{A}(2:n, 2:n) \mathbf{v}.$$

Elijamos $u_1 = -\frac{\mathbf{A}(2:n, 1)^* \mathbf{v}}{a_{1,1}}$ y resulta para todo $\mathbf{v} \neq \mathbf{0}$,

$$0 < \mathbf{v}^* (\mathbf{A}(2:n, 2:n) - \mathbf{C}(2:n, 1)\mathbf{C}(2:n, 1)^*) \mathbf{v},$$

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 utiliza solo la información de la parte triangular inferior de \mathbf{A} (la parte superior es redundante) y en donde, además, se almacenan los datos de la matriz \mathbf{C} .

4.5. Ortogonalidad en Métodos 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 $\mathbf{A} \in \mathbb{K}^{m \times n}$, con columnas $\{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n\}$ linealmente independientes, podemos considerar los subespacios anidados generados por sus columnas:

$$\langle \mathbf{a}_1 \rangle \subset \langle \mathbf{a}_1, \mathbf{a}_2 \rangle \subset \dots \subset \langle \mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_k \rangle.^4$$

En varias aplicaciones es de interés encontrar generadores *ortonormales* de esos subespacios. Esto es, una colección $\{\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_l\}$, tales que $\mathbf{q}_i^* \mathbf{q}_j = \delta_i^j$ y

$$\langle \mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_l \rangle = \langle \mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_l \rangle$$

para todo l .

Matricialmente, y observando los subespacios anidados, es obvio que esto equivale a hallar una matriz $\mathbf{R} \in \mathbb{C}^{n \times n}$ triangular superior

$$\mathbf{R} = \begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} & \cdots & r_{1,n} \\ 0 & r_{2,2} & r_{2,3} & \cdots & r_{2,n} \\ 0 & 0 & r_{3,3} & \ddots & r_{3,n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & r_{n,n} \end{pmatrix}$$

tal que

$$\left(\begin{array}{c|c|c|c} \mathbf{a}_1 & \mathbf{a}_2 & \cdots & \mathbf{a}_n \end{array} \right) = \left(\begin{array}{c|c|c|c} \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_n \end{array} \right) \left(\begin{array}{cccccc} r_{1,1} & r_{1,2} & r_{1,3} & \cdots & r_{1,n} \\ 0 & r_{2,2} & r_{2,3} & \cdots & r_{2,n} \\ 0 & 0 & r_{3,3} & \ddots & r_{3,n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & r_{n,n} \end{array} \right),$$

de donde se observa que tanto los \mathbf{q}_i (elegidos ortonormales) como los $r_{i,j}$ pueden obtenerse desde las ecuaciones

$$\begin{aligned} \mathbf{a}_1 &= r_{1,1} \mathbf{q}_1 \\ \mathbf{a}_2 &= r_{1,2} \mathbf{q}_1 + r_{2,2} \mathbf{q}_2 \\ &\vdots \\ \mathbf{a}_n &= r_{1,n} \mathbf{q}_1 + r_{2,n} \mathbf{q}_2 + \cdots + r_{n,n} \mathbf{q}_n \end{aligned}$$

por ejemplo, a través de Gramm-Schmidt. Notemos que en caso de que \mathbf{A} resulte cuadrada ($\mathbf{A} \in \mathbb{K}^{n \times n}$), tendremos que \mathbf{Q} será una matriz unitaria (ortogonal en caso de ser una matriz real). En particular, a través de Gramm-Schmidt hemos concluido que toda matriz $\mathbf{A} \in \mathbb{K}^{n \times n}$ invertible (en particular con todas sus columnas l.i.) puede factorizarse como una ortogonal \mathbf{Q} por una triangular superior \mathbf{R}

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

factorización que se denomina QR.

⁴Si las columnas son l.i. las inclusiones son obviamente estrictas, pero no en el caso general.

Capítulo 5

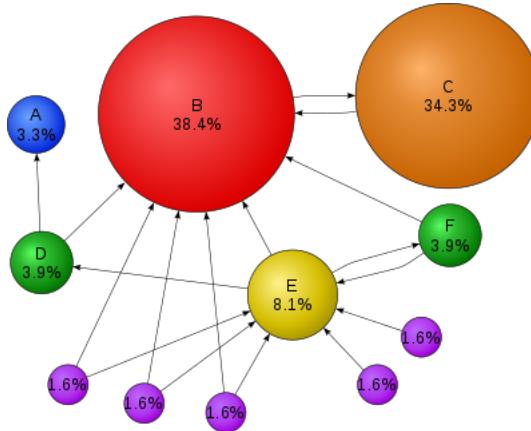
Diagonalización

5.1. Motivación: Google Page Rank

El éxito inicial del motor de búsquedas de Google se debe en gran parte al algoritmo desarrollado para rankear las páginas, es decir, ordenarlas por orden de importancia, y de esa forma poder mostrar a los usuarios resultados relevantes.

¿Cómo determinar si una página es relevante? Una primer medida de relevancia es considerar la cantidad de otras páginas que tienen un enlace a esa página. Sin embargo, con este criterio tan simple, no distinguimos si las páginas que enlazan a cada página son importantes o no. Queremos darle mayor relevancia a enlaces provenientes de páginas relevantes, y caemos en una especie de círculo vicioso.

Podemos resumir la solución propuesta por Google de la siguiente forma. Consideraremos que tenemos 11 páginas, que contienen enlaces entre las distintas páginas. Una flecha en el gráfico indica que la página de partida tiene un enlace a la página de llegada.



Como podemos ver, la página B es relevante porque recibe enlaces desde muchas páginas. La página C en cambio es relevante porque recibe un enlace de B, que es una página relevante. ¿Cómo podemos definir la relevancia? Pensamos que realizamos el siguiente experimento: ponemos a 1.000.000 de personas a navegar por internet. Inicialmente eligen una página cualquiera al azar y luego cada segundo cambian de página siguiendo algún link al azar de la página en la que están. Después de varios segundos, ¿cuántos visitantes habrá en cada página?

Podemos modelar esto llamando $\mathbf{v}^{(0)} \in \mathbb{R}^{11}$ a la cantidad inicial de visitantes en cada sitio, y $\mathbf{v}^{(k)}$ a la cantidad de visitantes luego de k segundos. Vamos a ver que podemos construir una

matriz $\mathbf{A} \in \mathbb{R}^{11 \times 11}$, llamada matriz de transición, tal que

$$\mathbf{v}^{(k)} = \mathbf{A}\mathbf{v}^{(k-1)}$$

y recursivamente,

$$\mathbf{v}^{(k)} = \mathbf{A}^k \mathbf{v}^{(0)}.$$

Por lo tanto, para estudiar este proceso, nos interesa poder calcular potencias altas de matrices, o predecir su comportamiento.

Podemos preguntarnos también si existe un límite del proceso, es decir, si luego de varios segundos, la cantidad de visitantes en cada página se mantiene estable. En ese caso, el vector límite debe cumplir la relación

$$\mathbf{A}\mathbf{v}^\infty = \mathbf{v}^\infty.$$

Los vectores con esta propiedad se llaman autovectores, y es el tema principal de este capítulo.

5.2. Autovalores y autovectores

Dada una matriz cuadrada $\mathbf{A} \in \mathbb{K}^{n \times n}$, un vector $\mathbf{v} \in \mathbb{K}^n$ no nulo tal que

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}, \text{ para algún } \lambda \in \mathbb{K}$$

se llama *autovector* de \mathbf{A} con *autovalor* λ .

Ejemplo. Buscamos los autovalores y autovectores de la matriz $\mathbf{A} = \begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix}$. Tenemos que encontrar \mathbf{v} y λ tales que $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$. El término de la derecha podemos reescribirlo como $\lambda\mathbf{I}\mathbf{v}$, y esto nos permite agrupar:

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{v} = 0,$$

que en nuestro caso queda

$$\begin{pmatrix} 2 - \lambda & 3 \\ 2 & 1 - \lambda \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

Si el determinante de $\mathbf{A} - \lambda\mathbf{I}$ es no nulo, el sistema tiene solución única $(v_1, v_2) = (0, 0)$. Como estamos buscando vectores no nulos, debemos encontrar λ tal que $\det(\mathbf{A} - \lambda\mathbf{I}) = 0$.

Calculamos

$$\det(\mathbf{A} - \lambda\mathbf{I}) = (2 - \lambda)(1 - \lambda) - 6 = \lambda^2 - 3\lambda - 4.$$

Esta ecuación es un polinomio de grado 2 en λ y tiene raíces

$$\lambda_1 = -1 \quad \text{y} \quad \lambda_2 = 4.$$

Estos son los autovalores de \mathbf{A} .

Para calcular los autovectores correspondientes a esos autovalores, calculamos los núcleos de las matrices

$$\mathbf{A} - (-1)\mathbf{I}_2 = \begin{pmatrix} 3 & 3 \\ 2 & 2 \end{pmatrix} \quad \text{y} \quad \mathbf{A} - (4)\mathbf{I}_2 = \begin{pmatrix} -2 & 3 \\ 2 & -3 \end{pmatrix}.$$

```
import numpy as np
import scipy.linalg

A = np.array([[2, 3], [2, 1]])
A1 = A - (-1) * np.eye(2)
print(scipy.linalg.null_space(A1))
```

```
%% [-0.70710678]
%% [ 0.70710678]
```

```
import numpy as np
import scipy.linalg

A = np.array([[2, 3], [2, 1]])
A1 = A - (-1) * np.eye(2)
print(scipy.linalg.null_space(A1))
```

```
%% [0.83205029]
%% [0.5547002 ]
```

El comando `null_space` normaliza los vectores a norma-2 igual a 1. Podemos verificar que el espacio de autovectores correspondiente a $\lambda_1 = -1$ es

$$E_{\lambda_1} = \langle (1, -1) \rangle$$

y el espacio de autovectores correspondiente a $\lambda_2 = 4$ es

$$E_{\lambda_2} = \langle (3, 2) \rangle.$$

Es decir, por ejemplo, $(1, -1)$ es autovector de \mathbf{A} de autovalor -1 , como también lo es cualquier múltiplo no nulo de ese vector. En este caso, obtuvimos que el espacio de autovectores de cada autovalor tiene dimensión 1, o abusando un poco la notación, que a cada autovalor le corresponde un único autovector.

Repasando,

- Los *autovalores* de una matriz $\mathbf{A} \in \mathbb{K}^{n \times n}$ ($\mathbb{K} = \mathbb{R}$ o \mathbb{C}) son los valores $\lambda \in \mathbb{C}$ para los cuales $\det(\mathbf{A} - \lambda \mathbf{I}_n) = 0$.
- El espacio de autovalores de un autovalor λ es el núcleo de la matriz $\mathbf{A} - \lambda \mathbf{I}_n$.
- Si λ es autovalor, el núcleo siempre tiene dimensión mayor o igual que 1.
- Si el núcleo tiene dimensión exactamente 1, decimos informalmente que el autovalor λ tiene un único autovector.

DEFINICIÓN 5.2.1. Dada $\mathbf{A} \in \mathbb{K}^{n \times n}$, llamamos *polinomio característico* de \mathbf{A} al determinante

$$\chi_{\mathbf{A}}(\lambda) = \det(\mathbf{A} - \lambda \mathbf{I}).$$

Los autovalores de \mathbf{A} son las raíces de $\chi_{\mathbf{A}}(\lambda)$. El polinomio $\det(\lambda \mathbf{I} - \mathbf{A})$ es un polinomio mónico con las mismas raíces que $\det(\lambda \mathbf{I} - \mathbf{A})$, y llamamos indistintamente polinomio característico a cualquiera de estos dos polinomios.

EJERCICIO 5.2.2. Calcular el polinomio característico, los autovalores y autovectores de la matriz

$$B = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 3 & 4 \\ 0 & 4 & 9 \end{pmatrix}.$$

EJEMPLO 5.2.3. Verificamos en Python los cálculos para las dos matrices \mathbf{A} y \mathbf{B} utilizando el comando `linalg.eig` de numpy.

```
A = np.array([[2, 3], [2, 1]])
e = np.linalg.eig(A)    # e es una lista con dos elementos
print("Autovalores: ", e[0])    # El primer elemento es un array
                                # de autovalores
print("Autovectores:\n", e[1]) # El segundo elemento es una matriz
                                # con los autovectores como columnas.
```

```
%% Autovalores:  [ 4. -1.]
%% Autovectores:
%%  [[ 0.83205029 -0.70710678]
%%   [ 0.5547002   0.70710678]]
```

EJERCICIO 5.2.4. Calcular en Python los autovalores y los autovectores asociados a cada autovalor para las siguientes matrices.

$$\begin{aligned} 1. \quad \mathbf{A} &= \begin{pmatrix} -1 & 4 & -2 \\ -3 & 4 & 0 \\ -3 & 1 & 3 \end{pmatrix} \\ 2. \quad \mathbf{A} &= \begin{pmatrix} 0 & 0 & 2 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & -2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{aligned}$$

Vamos a utilizar la siguiente propiedad más adelante.

PROPOSICIÓN 5.2.5. Para $\mathbf{A} \in \mathbb{R}^{n \times n}$, los autovalores de \mathbf{A} y \mathbf{A}^T son los mismos.

DEMOSTRACIÓN. Recordemos que para cualquier $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\det(\mathbf{A}) = \det(\mathbf{A}^T)$.

Los autovalores de \mathbf{A} son las raíces del polinomio característico $p_{\mathbf{A}}(\lambda) = \det(\mathbf{A} - \lambda\mathbf{I})$ y los autovalores de \mathbf{A}^T son las raíces del polinomio característico $p_{\mathbf{A}^T}(\lambda) = \det(\mathbf{A}^T - \lambda\mathbf{I})$. Como $(\mathbf{A} - \lambda\mathbf{I})^T = \mathbf{A}^T - \lambda\mathbf{I}$, ambos determinantes son iguales y por lo tanto las raíces son las mismas. \square

5.3. Diagonalización

Como vimos, podemos pensar una matriz $\mathbf{A} \in \mathbb{K}^{n \times n}$ como una transformación lineal de $T_{\mathbf{A}} : \mathbb{K}^n \rightarrow \mathbb{K}^n$ dada por $T_{\mathbf{A}}(\mathbf{v}) = \mathbf{A} \cdot \mathbf{v}$ para $\mathbf{v} \in \mathbb{K}^n$. En este caso estamos pensando que tanto el vector \mathbf{v} como $T_{\mathbf{A}}(\mathbf{v})$ están expresados en coordenadas de la base canónica. Veamos cómo cambia la matriz de una transformación cuando expresamos los vectores en otra base.

EJEMPLO 5.3.1. Continuando el ejemplo anterior, consideramos $\mathbf{A} = \begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix}$ y la transformación asociada

$$f(\mathbf{v}) = T_{\mathbf{A}}(\mathbf{v}) = \mathbf{A}\mathbf{v} = \begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}.$$

Para esta transformación, vale

$$T((1, 0)) = (2, 2)$$

$$T((0, 1)) = (3, 1)$$

También vimos que $\mathbf{w}_1 = (1, -1)$ y $\mathbf{w}_2 = (3, 2)$ son autovectores de \mathbf{A} con autovalores -1 y 4 respectivamente. Si tomamos la base $\mathcal{B} = \{\mathbf{w}_1, \mathbf{w}_2\}$, la transformación en esta base toma una forma más simple:

$$T(\mathbf{w}_1) = -\mathbf{w}_1$$

$$T(\mathbf{w}_2) = 4\mathbf{w}_2$$

o si escribimos a estos vectores en las coordenadas de la base \mathcal{B} , obtenemos

$$T((1, 0)_{\mathcal{B}}) = (-1, 0)_{\mathcal{B}}$$

$$T((0, 1)_{\mathcal{B}}) = (0, 4)_{\mathcal{B}}$$

Podemos entonces considerar la matriz de la transformación T expresada en coordenadas de la base \mathcal{B} y llamamos $[f]_{\mathcal{B}}$ a esta matriz. Obtuvimos:

$$[f]_{\mathcal{B}} = \begin{pmatrix} -1 & 0 \\ 0 & 4 \end{pmatrix}.$$

En general, tenemos el siguiente resultado.

PROPOSICIÓN 5.3.2. Sea $\mathbf{A} \in \mathbb{K}^{n \times n}$ tal que existe una base $\mathcal{B} = \{\mathbf{w}_1, \dots, \mathbf{w}_n\}$ de \mathbb{K}^n formada por autovectores de \mathbf{A} , con autovalores $\lambda_1, \dots, \lambda_n \in \mathbb{C}$ (no necesariamente distintos). Sea $f : \mathbb{K}^n \rightarrow \mathbb{K}^n$ la transformación asociada, $f(\mathbf{v}) = \mathbf{A}\mathbf{v}$. La matriz de la f en la base \mathcal{B} es diagonal,

$$[f]_{\mathcal{B}} = \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix}$$

Estudiamos ahora la relación entre la matriz original \mathbf{A} y la matriz $[f]_{\mathcal{B}}$ en la base de autovectores.

Recordemos que dada una base $\mathcal{B} = \{\mathbf{w}_1, \dots, \mathbf{w}_n\}$ de \mathbb{K}^n y un vector $\mathbf{v} \in \mathbb{K}^n$ con coordenadas en la base \mathcal{B}

$$\mathbf{v} = (u_1, \dots, u_n)_{\mathcal{B}},$$

podemos encontrar las coordenadas de \mathbf{v} en la base canónica mediante la multiplicación

$$\mathbf{v}_{\mathcal{E}} = \mathbf{C}_{\mathcal{B}\mathcal{E}} \mathbf{v}_{\mathcal{B}},$$

donde $\mathbf{C}_{\mathcal{B}\mathcal{E}}$ es la matriz de cambio de base de \mathcal{B} a \mathcal{E} ,

$$\mathbf{C}_{\mathcal{B}\mathcal{E}} = \begin{pmatrix} | & | & | \\ \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_n \\ | & | & | \end{pmatrix}.$$

Si en cambio, dadas las coordenadas canónicas de un vector $\mathbf{v} = (v_1, \dots, v_n)_{\mathcal{E}}$, queremos encontrar las coordenadas en la base \mathcal{B} , realizamos la multiplicación

$$\mathbf{v}_{\mathcal{B}} = \mathbf{C}_{\mathcal{EB}} \mathbf{v}_{\mathcal{E}},$$

donde $\mathbf{C}_{\mathcal{EB}}$ es la matriz de cambio de base de \mathcal{E} a \mathcal{B} , que podemos calcular invirtiendo la matriz $\mathbf{C}_{\mathcal{BE}}$,

$$\mathbf{C}_{\mathcal{EB}} = \mathbf{C}_{\mathcal{BE}}^{-1}.$$

Por lo tanto, si f es una transformación lineal dada en coordenadas de la base canónica por una matriz \mathbf{A} , podemos obtener la matriz de la transformación en una base \mathcal{B} (es decir que tanto \mathbf{v} como $T_{\mathbf{A}}(\mathbf{v})$ estén expresados en coordenadas de la base \mathcal{B}) aplicando el siguiente cambio de base a la matriz \mathbf{A} :

$$\mathbf{A}_{\mathcal{B}} = \mathbf{C}_{\mathcal{EB}} \mathbf{A} \mathbf{C}_{\mathcal{BE}}.$$

Vemos que tanto \mathbf{A} como $\mathbf{A}_{\mathcal{B}}$ representan la misma transformación lineal pero expresada en distintas bases. En este caso, decimos que las matrices son *semejantes*.

Más generalmente, dos matrices \mathbf{A}, \mathbf{B} son *semejantes* si existe una matriz inversible \mathbf{C} tal que

$$\mathbf{B} = \mathbf{C}^{-1} \mathbf{A} \mathbf{C}.$$

Ejemplo. Continuamos el ejemplo $\mathbf{A} = \begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix}$. Vimos que \mathbf{A} tiene autovalores $\lambda_1 = -1$ y $\lambda_2 = 4$ con autovectores $v_1 = (1, -1)$ y $v_2 = (3, 2)$ respectivamente. Estos autovectores forman una base de \mathbb{R}^2 . Si expresamos la matriz \mathbf{A} en la base $\mathcal{B} = \{v_1, v_2\}$ obtenemos

$$\mathbf{A}_{\mathcal{B}} = \begin{pmatrix} -1 & 0 \\ 0 & 4 \end{pmatrix}$$

(en las columnas de $\mathbf{A}_{\mathcal{B}}$ ponemos las imágenes de los vectores de la base \mathcal{B} expresados también en coordenadas de la base \mathcal{B}).

Verificamos en Python.

```
A = np.array([[2, 3], [2, 1]])
e = np.linalg.eig(A)
C_BE = e[1]
C_EB = np.linalg.inv(C_BE)
print("A_B = \n", C_EB @ A @ C_BE)
```

```
%% A_B =
%% [[ 4.00000000e+00  0.00000000e+00]
%% [-5.55111512e-17 -1.00000000e+00]]
```

La siguiente propiedad resume lo que acabamos de ver.

PROPOSICIÓN 5.3.3. Si $\mathbf{A} \in \mathbb{K}^{n \times n}$ y $\mathcal{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ es una base de \mathbb{K}^n formada por autovectores de \mathbf{A} con autovalores $\{\lambda_1, \dots, \lambda_n\}$, entonces tomando

- \mathbf{D} la matriz diagonal de autovalores, $\mathbf{D} = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix}$,
- \mathbf{C} la matriz de autovectores de \mathbf{A} , $\mathbf{C} = \begin{pmatrix} | & & | \\ \mathbf{b}_1 & \dots & \mathbf{b}_n \\ | & & | \end{pmatrix}$ obtenemos la diagonalización de la matriz \mathbf{A}

$$\mathbf{A} = \mathbf{CDC}^{-1} \quad \text{y} \quad \mathbf{D} = \mathbf{C}^{-1}\mathbf{AC}.$$

EJERCICIO 5.3.4. Calcular en Python los autovalores y autovectores de las siguientes matrices. Según los resultados obtenidos, ¿alguna de estas matrices es diagonalizable?

$$\begin{aligned} 1. \quad \mathbf{A} &= \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \\ 2. \quad \mathbf{A} &= \begin{pmatrix} -1 & 3 & -1 \\ -3 & 5 & -1 \\ -3 & 3 & 1 \end{pmatrix} \end{aligned}$$

Para la matriz que resulte diagonalizable, ¿cuál es la base \mathcal{B} de autovectores? ¿Quién es \mathbf{D} ? Hallar \mathbf{C} tal que $\mathbf{A} = \mathbf{CDC}^{-1}$.

Como vimos en el ejercicio, no cualquier matriz es diagonalizable.

Dada una matriz \mathbf{A} , sean $\lambda_1, \dots, \lambda_k$ los autovalores distintos de \mathbf{A} . Definimos E_{λ_i} el espacio de autovectores asociados al autovalor λ_i . \mathbf{A} es diagonalizable si

$$\dim(E_{\lambda_1}) + \dim(E_{\lambda_2}) + \cdots + \dim(E_{\lambda_k}) = n.$$

En particular, como $\dim(E_\lambda) \geq 1$ para λ autovector (¿por qué?), si \mathbf{A} tiene n autovalores distintos, \mathbf{A} es diagonalizable.

EJERCICIO 5.3.5. Para cada una de las siguientes matrices, calcular en Python los autovalores, las dimensiones de los espacios asociados a cada autovalor y determinar si las matrices son diagonalizables.

$$\begin{aligned} 1. \quad \mathbf{A} &= \begin{pmatrix} 101 & 2 & 3 & 4 \\ 1 & 102 & 3 & 4 \\ 1 & 2 & 103 & 4 \\ 1 & 2 & 3 & 104 \end{pmatrix} \\ 2. \quad \mathbf{A} &= \begin{pmatrix} 5 & 4 & 2 & 1 \\ 0 & 1 & -1 & -1 \\ -1 & -1 & 3 & 0 \\ 1 & 1 & -1 & 2 \end{pmatrix} \end{aligned}$$

5.3.1. Multiplicidad algebraica y geométrica. Dada una matriz $\mathbf{A} \in \mathbb{K}^{n \times n}$ y un autovalor λ de \mathbf{A} , definimos:

- La *multiplicidad aritmética* de λ es la multiplicidad de λ como raíz del polinomio característico $\chi_{\mathbf{A}}$.

- La *multiplicidad geométrica* de λ es la dimensión del espacio de autovectores asociado a λ , o equivalentemente, la dimensión de $\text{Nu}(\mathbf{A} - \lambda\mathbf{I})$.

PROPOSICIÓN 5.3.6. Dada una matriz $\mathbf{A} \in \mathbb{K}^{n \times n}$ y un autovalor λ de \mathbf{A} , la multiplicidad geométrica es siempre menor o igual que la multiplicidad aritmética.

DEMOSTRACIÓN. Sea s la multiplicidad geométrica de λ y $\{\mathbf{v}_1, \dots, \mathbf{v}_s\}$ una base de autovectores de E_λ , el espacio de autovectores asociados a λ . Completamos la base de E_λ a una base de \mathbb{K}^n :

$$\mathcal{B} = \{\mathbf{v}_1, \dots, \mathbf{v}_s, \mathbf{w}_1, \dots, \mathbf{w}_{n-s}\}.$$

Consideramos la matriz de la transformación f definida por \mathbf{A} en la base \mathcal{B} :

$$[f]_{\mathcal{B}} = \left(\begin{array}{cc|c} \lambda & & \\ & \ddots & \\ & & \lambda \\ \hline & & \\ 0 & & D \end{array} \right),$$

donde el primer bloque es una matriz de $s \times s$.

Utilizando la fórmula para el cálculo de determinantes desarrollando una columna, es fácil ver que

$$\chi_{[f]_{\mathcal{B}}}(x) = \det(x\mathbf{I}_n - [f]_{\mathcal{B}}) = (x - \lambda)^s \det(x\mathbf{I}_n - [f]_{\mathcal{B}}).$$

Ya vimos que $[f]_{\mathcal{B}}$ y $[f]_{\mathcal{E}}$ tienen el mismo polinomio característico. Por lo tanto, la multiplicidad algebraica de λ como autovalor de \mathbf{A} es al menos s , como queríamos ver. \square

Para una matriz $\mathbf{A} \in \mathbb{K}^{n \times n}$, el polinomio característico tiene grado n y por lo tanto n raíces contando sus multiplicidades. Es decir, que la suma de las multiplicidades aritméticas de todos los autovalores es siempre exactamente n . Más aún, autovectores correspondientes a autovalores distintos son linealmente independientes.

PROPOSICIÓN 5.3.7. Dada $\mathbf{A} \in \mathbb{K}^{n \times n}$, sean $\mathbf{v}_1, \dots, \mathbf{v}_s$ autovectores de \mathbf{A} correspondientes a autovalores distintos $\lambda_1, \dots, \lambda_s$. El conjunto $\{\mathbf{v}_1, \dots, \mathbf{v}_s\}$ es linealmente independiente.

DEMOSTRACIÓN. Consideramos una relación de dependencia lineal:

$$a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \cdots + a_s\mathbf{v}_s = 0$$

y supongamos que es la relación que involucra la menor cantidad posible de vectores, en particular todos los a_i son no nulos.

Si multiplicamos la igualdad por $(\mathbf{A} - \lambda_1\mathbf{I}_n)$ obtenemos:

$$a_1(\mathbf{A} - \lambda_1\mathbf{I}_n)\mathbf{v}_1 + a_2(\mathbf{A} - \lambda_1\mathbf{I}_n)\mathbf{v}_2 + \cdots + a_s(\mathbf{A} - \lambda_1\mathbf{I}_n)\mathbf{v}_s = 0,$$

y simplificando,

$$0 + a_2(\lambda_2 - \lambda_1)\mathbf{v}_2 + \cdots + a_s(\lambda_s - \lambda_1)\mathbf{v}_s = 0,$$

que es una relación con menos términos no nulos que la anterior, lo cual es un absurdo. \square

Combinando este resultado con la propiedad anterior, obtenemos el siguiente resultado.

PROPOSICIÓN 5.3.8. Una matriz \mathbf{A} es diagonalizable si para todo autovalor λ de \mathbf{A} , la multiplicidad artimética y la multiplicidad geométrica coinciden.

5.3.2. Propiedades de las matrices diagonalizables.

PROPOSICIÓN 5.3.9. Si $\mathbf{A} = \mathbf{C}^{-1}\mathbf{D}\mathbf{C}$,

- $\mathbf{A}^2 = \mathbf{C}^{-1}\mathbf{D}\mathbf{C}\mathbf{C}^{-1}\mathbf{D}\mathbf{C} = \mathbf{C}^{-1}\mathbf{D}\mathbf{D}\mathbf{C} = \mathbf{C}^{-1}\mathbf{D}^2\mathbf{C}$
- Inductivamente, $\mathbf{A}^k = \mathbf{C}^{-1}\mathbf{D}\mathbf{C} \dots \mathbf{C}^{-1}\mathbf{D}\mathbf{C} = \mathbf{C}^{-1}\mathbf{D}^k\mathbf{C}$
- Si \mathbf{A} inversible, $\mathbf{A}^{-1} = (\mathbf{C}^{-1}\mathbf{D}\mathbf{C})^{-1} = \mathbf{C}^{-1}\mathbf{D}^{-1}\mathbf{C}$.
- Si $p(x) \in \mathbb{R}[x]$, $p(\mathbf{A}) = \mathbf{C}^{-1}p(\mathbf{D})\mathbf{C}$.

En todos estos casos, como vimos anteriormente, la función aplicada a una matriz diagonal se puede calcular fácilmente aplicando la función a cada elemento de la diagonal.

5.3.3. Aplicación.

Consideramos el siguiente problema como ejemplo de aplicación.

Para guardar cadenas de caracteres en la computadora se pueden utilizar distintas codificaciones. Una codificación consiste en asignarla a cada carácter un número y ese número se guarda en binario en la memoria de la computadora. Dependiendo la codificación utilizada pueden utilizarse entre 1 y 4 bytes por carácter. Supongamos ahora que recibimos un mensaje en el que se han mezclado caracteres guardados utilizando 1 byte y caracteres utilizando 2 bytes, y no sabemos cuáles caracteres usan 1 byte y cuáles usan 2 bytes. ¿De cuántas formas distintas se puede interpretar el mensaje?

Planteando el problema en forma más abstracta, tenemos un tablero de $n \times 1$, y queremos dividir el tablero en bloques, que pueden ser de 1×1 o de 2×1 . ¿De cuántas formas podemos hacerlo?

Llamamos a_n a esta cantidad. Es posible demostrar la siguiente fórmula recursiva para los valores de la sucesión:

$$\begin{aligned} a_1 &= 1 \\ a_2 &= 2 \\ a_n &= a_{n-1} + a_{n-2} \end{aligned}$$

Es decir, los valores a_n forman una sucesión de Fibonacci. Para utilizar la teoría de transformaciones lineales, nos gustaría encontrar una relación lineal entre cada término y el siguiente. Sin embargo, como está planteada la sucesión, cada término depende de los dos anteriores. Aplicamos un truco simple para plantear la relación de forma que cada término dependa solo del anterior. Plantear la ecuación de recurrencia en forma matricial de la siguiente forma:

$$\begin{pmatrix} a_n \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} a_{n-1} \\ a_{n-2} \end{pmatrix}.$$

Si llamamos $\mathbf{A} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$, obtenemos

$$\begin{pmatrix} a_n \\ a_{n-1} \end{pmatrix} = \mathbf{A} \begin{pmatrix} a_{n-1} \\ a_{n-2} \end{pmatrix} = \mathbf{A} \mathbf{A} \begin{pmatrix} a_{n-2} \\ a_{n-3} \end{pmatrix} = \dots = (\mathbf{A})^{n-2} \begin{pmatrix} a_2 \\ a_1 \end{pmatrix}$$

Por lo tanto, podemos dar una fórmula cerrada para la sucesión si encontramos una expresión cerrada para la matriz $(\mathbf{A})^{n-2}$. Esto podemos hacerlo diagonalizando la matriz.

Calculamos autovalores y autovectores de \mathbf{A} en Python.

```
A = np.array([[1, 1], [1, 0]])
e = np.linalg.eig(A)
print("Autovalores: ", e[0])
print("Autovectores: \n", e[1])
```

```
%% Autovalores: [ 1.61803399 -0.61803399]
%% Autovectores:
%% [[ 0.85065081 -0.52573111]
%% [ 0.52573111  0.85065081]]
```

Como hay dos autovalores distintos, podemos diagonalizar \mathbf{A} . Obtenemos

$$\mathbf{A} = \mathbf{C}_{\mathcal{B}\mathcal{E}} \begin{pmatrix} 1.618 & 0 \\ 0 & -0.618 \end{pmatrix} \mathbf{C}_{\mathcal{E}\mathcal{B}},$$

con $\mathbf{C}_{\mathcal{B}\mathcal{E}} = \begin{pmatrix} 0.85 & -0.52 \\ 0.52 & 0.85 \end{pmatrix}$ y $\mathbf{C}_{\mathcal{E}\mathcal{B}} = \mathbf{C}_{\mathcal{B}\mathcal{E}}^{-1}$.

Esto nos permite dar la siguiente expresión para los términos de la sucesión:

$$\begin{pmatrix} a_n \\ a_{n-1} \end{pmatrix} = \mathbf{C}_{\mathcal{B}\mathcal{E}} \begin{pmatrix} (1.618)^{n-2} & 0 \\ 0 & (-0.618)^{n-2} \end{pmatrix} \mathbf{C}_{\mathcal{E}\mathcal{B}} \begin{pmatrix} a_2 \\ a_1 \end{pmatrix}.$$

Vemos en particular que los términos de la sucesión crecen con velocidad exponencial. Haciendo las cuentas en forma simbólica, podemos dar una expresión exacta para los términos de la sucesión (los autovalores de la matriz son $\frac{1+\sqrt{5}}{2}$ y $\frac{1-\sqrt{5}}{2}$).

5.4. Descomposición de Schur

Si bien vimos que no toda matriz es diagonalizable, es decir, semejante a una matriz diagonal, vamos a ver que toda la matriz es semejante a una matriz triangular superior con los autovalores en la diagonal. Más aun, la matriz de cambio de base se puede tomar unitaria. En este caso, decimos que las matrices son unitariamente equivalentes. Concretamente, tenemos el siguiente resultado.

TEOREMA 5.4.1. Dada una matriz $\mathbf{A} \in \mathbb{K}^{n \times n}$ con autovalores $\lambda_1, \dots, \lambda_n$ (no necesariamente distintos), existen matrices \mathbf{U} unitaria y \mathbf{T} triangular superior tales que

$$\mathbf{A} = \mathbf{U}\mathbf{T}\mathbf{U}^*$$

y $t_{ii} = \lambda_i$. Es decir, \mathbf{A} es unitariamente equivalente a una matriz \mathbf{T} triangular superior con los autovalores de \mathbf{A} en la diagonal en cualquier orden arbitrario. Si $\mathbf{A} \in \mathbb{R}^{n \times n}$ y los autovalores de \mathbf{A} son reales, la matriz \mathbf{U} se puede tomar real y ortogonal.

Recordemos que para una matriz \mathbf{U} unitaria, vale $\mathbf{U}^{-1} = \mathbf{U}^*$, por lo tanto las matrices \mathbf{A} y \mathbf{T} son semejantes.

DEMOSTRACIÓN. Construimos \mathbf{U} y \mathbf{T} paso a paso.

Tomamos λ_1 autovalor de \mathbf{A} y \mathbf{w}_1 autovector de \mathbf{A} con autovalor λ_1 y $\|\mathbf{w}_1\|_2 = 1$.

Completando $\{\mathbf{w}_1\}$ a una base de \mathbb{K}^n y aplicando ortonormalización de Gram-Schmidt, podemos obtener una base ortonormal de \mathbb{C}^n ,

$$\{\mathbf{w}^{(1)}, \mathbf{z}^{(2)}, \dots, \mathbf{z}^{(n)}\}.$$

Construimos la matriz \mathbf{U}_1 tomando estos vectores como columnas de la matriz. Como la primera columna de \mathbf{AU}_1 es $\lambda_1 \mathbf{w}^{(1)}$, obtenemos

$$\mathbf{U}_1^* \mathbf{AU}_1 = \left[\begin{array}{c|c} \lambda_1 & * \\ 0 & \mathbf{A}_1 \end{array} \right],$$

Como $\mathbf{U}_1^* \mathbf{AU}$ y \mathbf{A} son semejantes, tienen el mismo polinomio característico. Luego $\mathbf{A}_1 \in \mathbb{C}^{(n-1) \times (n-1)}$ tiene autovalores $\lambda_2, \dots, \lambda_n$. Tomamos ahora $\mathbf{w}^{(2)} \in \mathbb{C}^{n-1}$ autovector normalizado de \mathbf{A}_1 correspondiente al autovalor λ_2 y repetimos el procedimiento. Construimos $\mathbf{U}_2 \in \mathbb{C}^{(n-1) \times (n-1)}$ unitaria tal que

$$\mathbf{U}_2^* \mathbf{A}_1 \mathbf{U}_2 = \left[\begin{array}{c|c} \lambda_2 & * \\ 0 & \mathbf{A}_2 \end{array} \right],$$

y definimos

$$\mathbf{V}_2 = \left(\begin{array}{c|c} 1 & 0 \\ 0 & \mathbf{U}_2 \end{array} \right).$$

Las matrices \mathbf{V}_2 y $\mathbf{U}_1 \mathbf{V}_2$ son unitarias, y $\mathbf{V}_2^* \mathbf{U}_1^* \mathbf{AU}_1 \mathbf{V}_2$ tiene la forma

$$\mathbf{V}_2^* \mathbf{U}_1^* \mathbf{AU}_1 \mathbf{V}_2 = \left(\begin{array}{cc|c} \lambda_1 & * & * \\ 0 & \lambda_2 & \\ \hline 0 & & \mathbf{A}_2 \end{array} \right).$$

Repitiendo este procedimiento, obtenemos matrices unitarias $\mathbf{U}_i \in \mathbb{C}^{(n-i+1) \times (n-i+1)}$, $i = 1, \dots, n-1$, y matrices unitarias $\mathbf{V}_i \in \mathbb{C}^{n \times n}$, $i = 2, \dots, n-1$. La matriz

$$\mathbf{U} = \mathbf{U}_1 \mathbf{V}_2 \mathbf{V}_3 \dots \mathbf{V}_{n-1}$$

es unitaria y $\mathbf{U}^* \mathbf{AU}$ es la factorización que buscamos.

Si todos los autovalores de \mathbf{A} son reales, los autovectores pueden elegirse también reales y todas las operaciones se pueden hacer sobre los reales, lo que prueba la última información. \square

5.5. Autovalores y autovectores de matrices especiales

5.5.1. Matrices ortogonales. Si $\mathbf{A} \in \mathbb{K}^{n \times n}$ es unitaria, todos sus autovalores cumplen $|\lambda| = 1$.

5.5.2. Matrices normales. Decimos que una matriz \mathbf{A} es normal si commuta con la matriz adjunta, es decir, si $\mathbf{AA}^* = \mathbf{A}^* \mathbf{A}$.

Recordemos que dos matrices \mathbf{A} y \mathbf{B} son unitariamente equivalentes si existe \mathbf{U} unitaria tal que

$$\mathbf{U}^* \mathbf{AU} = \mathbf{B}.$$

PROPOSICIÓN 5.5.1. Si \mathbf{A} es normal y \mathbf{B} es unitariamente equivalente a \mathbf{A} , entonces \mathbf{B} es normal.

DEMOSTRACIÓN. Tenemos que existe \mathbf{U} unitaria tal que $\mathbf{U}^* \mathbf{A} \mathbf{U} = \mathbf{B}$. Por lo tanto,

$$\mathbf{B} \mathbf{B}^* = \mathbf{U}^* \mathbf{A} \mathbf{U} \mathbf{U}^* \mathbf{A}^* \mathbf{U} = \mathbf{U}^* \mathbf{A} \mathbf{A}^* \mathbf{U} = \mathbf{U}^* \mathbf{A}^* \mathbf{A} \mathbf{U} = \mathbf{U}^* \mathbf{A}^* \mathbf{U} \mathbf{U}^* \mathbf{A} \mathbf{U} = \mathbf{B}^* \mathbf{B}.$$

□

Decimos que una matriz \mathbf{A} es *unitariamente diagonalizable* si existe una matriz \mathbf{U} unitaria tal que $\mathbf{U} \mathbf{A} \mathbf{U}^*$ es diagonal. Esto es equivalente a decir que \mathbf{A} tiene una base ortonormal de autovectores.

PROPOSICIÓN 5.5.2. Si \mathbf{A} es normal, entonces \mathbf{A} es unitariamente diagonalizable. En particular, autovectores de \mathbf{A} correspondientes a autovalores distintos son ortogonales.

DEMOSTRACIÓN. Por la descomposición de Shur, sabemos que existe \mathbf{T} triangular superior unitariamente equivalente a \mathbf{A} . Por lo tanto, podemos probar el resultado para \mathbf{T} .

Hacemos la demostración probando mediante cálculos explícitos que una matriz \mathbf{T} triangular y normal debe ser diagonal. Tenemos

$$\begin{pmatrix} \bar{t}_{11} & 0 & \cdots & 0 \\ \bar{t}_{21} & \bar{t}_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \bar{t}_{n1} & \bar{t}_{n2} & \cdots & \bar{t}_{nn} \end{pmatrix} \begin{pmatrix} t_{11} & t_{12} & \cdots & t_{1n} \\ 0 & t_{22} & \cdots & t_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & t_{nn} \end{pmatrix} = \begin{pmatrix} t_{11} & t_{12} & \cdots & t_{1n} \\ 0 & t_{22} & \cdots & t_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & t_{nn} \end{pmatrix} \begin{pmatrix} \bar{t}_{11} & 0 & \cdots & 0 \\ \bar{t}_{21} & \bar{t}_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \bar{t}_{n1} & \bar{t}_{n2} & \cdots & \bar{t}_{nn} \end{pmatrix}.$$

Comparando la casilla (1, 1) del producto a ambos lados,

$$\bar{t}_{11} t_{11} = t_{11} \bar{t}_{11} + \sum_{j=2}^n t_{1j} \bar{t}_{j1}.$$

Recordando que para cualquier $a \in \mathbb{C}$, $a\bar{a} = \bar{a}a = |a|^2$, obtenemos

$$|t_{11}|^2 = |t_{11}|^2 + \sum_{j=2}^n |t_{1j}|^2.$$

y por lo tanto

$$0 = \sum_{j=2}^n |t_{1j}|^2.$$

Una suma de términos no-negativos solo puede dar 0 si todos sus términos son 0, luego

$$t_{1j} = 0, j = 2, \dots, n.$$

Ahora usamos que la casillas (2, 2) de $\mathbf{T}^* \mathbf{T}$ y $\mathbf{T} \mathbf{T}^*$ son iguales y obtenemos

$$\bar{t}_{22} t_{22} = t_{22} \bar{t}_{22} + \sum_{j=3}^n t_{2j} \bar{t}_{j2},$$

de donde concluimos

$$t_{2j} = 0, j = 3, \dots, n.$$

Repetiendo este mismo argumento para todas las casillas de la diagonal, obtenemos

$$t_{ij} = 0, \text{ para todo } j > i$$

y como

$$t_{ij} = 0, \text{ para todo } j < i$$

porque \mathbf{T} es triangular superior, concluimos que \mathbf{T} es diagonal. \square

5.5.3. Matrices simétricas y hermitianas. Una matriz $\mathbf{A} \in \mathbb{R}^{n \times n}$ es simétrica si $\mathbf{A} = \mathbf{A}^T$. Más generalmente, $\mathbf{A} \in \mathbb{C}^{n \times n}$ es Hermitiana si $\mathbf{A} = \mathbf{A}^*$.

PROPOSICIÓN 5.5.3. Si \mathbf{A} es Hermitiana, entonces

1. $\mathbf{x}^* \mathbf{A} \mathbf{x}$ es real para todo $\mathbf{x} \in \mathbb{C}^n$,
2. todos los autovalores de \mathbf{A} son reales,
3. $\mathbf{S}^* \mathbf{A} \mathbf{S}$ es Hermitiana para toda $\mathbf{S} \in \mathbb{C}^{n \times n}$.

DEMOSTRACIÓN. Para 1, calculamos

$$\overline{\mathbf{x}^* \mathbf{A} \mathbf{x}} = (\mathbf{x}^* \mathbf{A} \mathbf{x})^* = \mathbf{x}^* \mathbf{A}^* \mathbf{x} = \mathbf{x}^* \mathbf{A} \mathbf{x}.$$

Como el conjugado de $\mathbf{x}^* \mathbf{A} \mathbf{x}$ es igual a si mismo, es un número real.

Para 2, si $\mathbf{A} \mathbf{x} = \lambda \mathbf{x}$ y suponemos \mathbf{x} normalizado $\|\mathbf{x}\|_2 = \mathbf{x}^* \mathbf{x} = 1$, entonces

$$\lambda = \lambda \mathbf{x}^* \mathbf{x} = \mathbf{x}^* \lambda \mathbf{x} = \mathbf{x}^* \mathbf{A} \mathbf{x},$$

que es un número real por 1.

Por último, $(\mathbf{S}^* \mathbf{A} \mathbf{S})^* = \mathbf{S}^* \mathbf{A}^* \mathbf{S} = \mathbf{S}^* \mathbf{A} \mathbf{S}$, y por lo tanto $\mathbf{S}^* \mathbf{A} \mathbf{S}$ es Hermitiana. \square

EJEMPLO 5.5.4. Buscamos una base ortogonal de autovectores de

$$\mathbf{A} = \begin{pmatrix} -4 & 2 & -2 \\ 2 & -7 & 4 \\ -2 & 4 & -7 \end{pmatrix}.$$

Calculamos $\det(\mathbf{A} - \lambda \mathbf{I}_3) = -(\lambda^3 + 18\lambda^2 + 81\lambda + 108) = -(\lambda + 12)(\lambda + 3)^2$, por lo tanto hay dos autovalores $\lambda_1 = -12$ y $\lambda_2 = -3$.

Calculando los núcleos de $\mathbf{A} - \lambda_i \mathbf{I}_3$, $i = 1, 2$, obtenemos los espacios de autovectores

$$E_{\lambda_1} = \langle (1, -2, 2) \rangle$$

$$E_{\lambda_2} = \langle (2, 1, 0), (2, 0, -1) \rangle$$

Por la Proposición 5.5.1, los espacios de autovalores correspondientes a distintos autovectores son ortogonales. Por lo tanto, para obtener una base de autovectores ortogonales, solo debemos ortogonalizar cada uno de los autoespacios por separado.

El autovector $(1, -2, 2)$ lo normalizamos a $\frac{\sqrt{5}}{5}(1, -2, 2)$ y para encontrar una base ortonormal de E_{λ_2} aplicamos Gram-Schmidt y obtenemos $E_{\lambda_2} = \left\langle \frac{\sqrt{3}}{3}(2, 1, 0), \frac{\sqrt{5}}{15}(2, -4, -5) \right\rangle$.

La base ortogonal de autovectores es $\mathcal{B} = \left\{ \frac{\sqrt{5}}{5}(1, -2, 2), \frac{\sqrt{3}}{3}(2, 1, 0), \frac{\sqrt{5}}{15}(2, -4, -5) \right\}$.

EJERCICIO 5.5.5. Calcular los autovalores y autovectores de las siguientes matrices.

- $\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{pmatrix}$, para $\alpha = \pi/2$ y $\alpha = \pi/4$.
- $\mathbf{A} = \begin{pmatrix} 0 & 0 & -2 \\ 0 & -2 & 0 \\ -2 & 0 & -3 \end{pmatrix}$

En el segundo punto, encontrar una matriz \mathbf{Q} ortogonal y una matriz \mathbf{D} diagonal tales que $\mathbf{A} = \mathbf{Q}^T \mathbf{D} \mathbf{Q}$.

5.5.4. Matrices definidas positivas. Decimos que una matriz $\mathbf{A} \in \mathbb{C}^{n \times n}$ es *semidefinida positiva* si es hermitiana y $\mathbf{v}^T \mathbf{A} \mathbf{v} \geq 0$ para todo $\mathbf{v} \in \mathbb{R}^n$. Si $\mathbf{v}^T \mathbf{A} \mathbf{v} > 0$ para todo $\mathbf{v} \in \mathbb{R}^n$ decimos que \mathbf{A} es *definida positiva*.

Como \mathbf{A} es hermitiana, todos los autovalores de \mathbf{A} son reales.

PROPOSICIÓN 5.5.6. Para \mathbf{A} hermitiana con autovalores $\lambda_1, \dots, \lambda_n$,

- \mathbf{A} es *definida positiva* si y solo si $\lambda_i(\mathbf{A}) > 0$ para todo $1 \leq i \leq n$.
- \mathbf{A} es *semidefinida positiva* si y solo si $\lambda_i(\mathbf{A}) \geq 0$ para todo $1 \leq i \leq n$.

DEMOSTRACIÓN. Lo vemos en el caso $\mathbf{A} \in \mathbb{R}^{2 \times 2}$. Si $\mathbf{A} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$, y $\mathbf{v} = (v_1, v_2)$,

$$\begin{aligned} \mathbf{v}^T \mathbf{A} \mathbf{v} &= (v_1 \ v_2) \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \\ &= (v_1 \ v_2) \begin{pmatrix} \lambda_1 v_1 \\ \lambda_2 v_2 \end{pmatrix} \\ &= (\lambda_1 v_1^2 + \lambda_2 v_2^2), \end{aligned}$$

y $\lambda_1 v_1^2 + \lambda_2 v_2^2 \geq 0$ si $\lambda_1, \lambda_2 \geq 0$.

□

PROPOSICIÓN 5.5.7. Si $\mathbf{A} \in \mathbb{R}^{n \times m}$, $\mathbf{A}^T \mathbf{A}$ y $\mathbf{A} \mathbf{A}^T$ son matrices simétricas y semidefinidas positivas.

DEMOSTRACIÓN. Ejercicio.

□

5.6. El método de la potencia

Veamos cómo podemos calcular mediante aproximaciones numéricas los autovalores de una matriz.

Comencemos con un ejemplo. Consideremos la matriz

$$\mathbf{A} = \begin{pmatrix} 0.9 & 0.075 & 0.025 \\ 0.15 & 0.8 & 0.05 \\ 0.25 & 0.25 & 0.5 \end{pmatrix}$$

Tomamos un vector cualquiera no nulo, por ejemplo $\mathbf{v} = (1, 2, 3)$ y calculamos $\mathbf{A}^k \mathbf{v}$ para distintos valores de k . O, equivalentemente, calculamos distintos términos de la sucesión definida

recursivamente por

$$\begin{aligned}\mathbf{v}^{(0)} &= (1, 2, 3) \\ \mathbf{v}^{(k)} &= \mathbf{A}\mathbf{v}^{(k-1)}\end{aligned}$$

Lo programamos en Python.

```
def estado(A, v, k):
    for i in range(k):
        v = A @ v
    return (v)

A = np.array([[0.9, 0.075, 0.025], [0.15, 0.8, 0.05], [0.25, 0.25,
0.5]])
v = np.array([1, 2, 3])
print(estado(A, v, 1))
print(estado(A, v, 10))
print(estado(A, v, 100))
print(estado(A, v, 1000))

%% [1.125 1.9 2.25 ]
%% [1.41762971 1.47373372 1.45503428]
%% [1.4375 1.4375 1.4375]
%% [1.4375 1.4375 1.4375]
```

Observamos en el experimento que $\mathbf{v}^{(k)}$ tiende a $\mathbf{w} = (1.4375, 1.4375, 1.4375)$. Si esto es cierto, entonces $\mathbf{Aw} = \mathbf{w}$. Luego encontramos un autovector de autovalor 1. ¿Cómo podemos demostrar lo que observamos?

Calculamos autovalores y autovectores de \mathbf{A} y verificamos si \mathbf{A} es diagonalizable.

```
e = np.linalg.eig(A)
print(e[0])
print(e[1])

%% [1. 0.74142136 0.45857864]
%% [[-0.57735027 -0.44371857 -0.03400257]
%% [-0.57735027 0.81130706 -0.13017638]
%% [-0.57735027 0.38065035 0.99090763]]
```

Obtenemos que \mathbf{A} es diagonalizable con autovalores $\lambda_1 = 1$, $\lambda_2 = 0.74$, $\lambda_3 = 0.45$. Llamamos $\mathcal{B} = \{\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3\}$ a la base de autovectores.

Luego podemos escribir a cualquier vector $\mathbf{v}^{(0)} \in \mathbb{R}^3$ como combinación lineal de vectores de la base \mathcal{B} :

$$\mathbf{v}^{(0)} = a_1 \mathbf{w}_1 + a_2 \mathbf{w}_2 + a_3 \mathbf{w}_3.$$

Usando esta representación, ¿quién es $\mathbf{A}\mathbf{v}^{(0)}$? Obtenemos

$$\begin{aligned}\mathbf{v}^{(1)} &= \mathbf{A}\mathbf{v}^{(0)} = a_1\mathbf{A}\mathbf{w}_1 + a_2\mathbf{A}\mathbf{w}_2 + a_3\mathbf{A}\mathbf{w}_3 \\ &= a_1\lambda_1\mathbf{w}_1 + a_2\lambda_2\mathbf{w}_2 + a_3\lambda_3\mathbf{w}_3\end{aligned}$$

y si multiplicamos nuevamente por \mathbf{A} ,

$$\begin{aligned}\mathbf{v}^{(2)} &= \mathbf{A}^2\mathbf{v} = \mathbf{A}(\mathbf{A}\mathbf{v}) = a_1\mathbf{A}\lambda_1\mathbf{w}_1 + a_2\mathbf{A}\lambda_2\mathbf{w}_2 + a_3\mathbf{A}\lambda_3\mathbf{w}_3 \\ &= a_1\lambda_1^2\mathbf{w}_1 + a_2\lambda_2^2\mathbf{w}_2 + a_3\lambda_3^2\mathbf{w}_3\end{aligned}$$

Repitiendo el mismo razonamiento obtenemos

$$\mathbf{v}^{(k)} = \mathbf{A}^k\mathbf{v} = a_1(\lambda_1)^k\mathbf{w}_1 + a_2(\lambda_2)^k\mathbf{w}_2 + a_3(\lambda_3)^k\mathbf{w}_3.$$

Como $|\lambda_2| < 1$ y $|\lambda_3| < 1$, tenemos que $(\lambda_2)^k \xrightarrow[k \rightarrow \infty]{} 0$ y $(\lambda_3)^k \xrightarrow[k \rightarrow \infty]{} 0$. Por otro lado, $\lambda_1^k = 1$ para todo $k \in \mathbb{N}$.

Tomando límite, concluimos entonces

$$\mathbf{v}^{(k)} \xrightarrow[k \rightarrow \infty]{} a_1\mathbf{w}_1,$$

es decir $\mathbf{v}^{(k)}$ tiende a un vector de autovalor 1, que es el autovalor de mayor módulo de la matriz.

En general, las matrices pueden tener autovalores de cualquier módulo y la sucesión anterior podría tender a $\mathbf{0}$ o diverger o no tener límite. Para el caso general, realizamos la siguiente modificación.

Tomamos un vector cualquiera no nulo $\mathbf{v}^{(0)}$ y en cada iteración de la recursión anterior normalizamos el resultado a norma-2 = 1:

$$\mathbf{v}^{(k)} = \frac{\mathbf{A}\mathbf{v}^{(k-1)}}{\|\mathbf{A}\mathbf{v}^{(k-1)}\|_2}, \text{ para } k \geq 1.$$

Calculamos en Python los vectores de esta sucesión tomando $\mathbf{v}^{(0)} = (1, 2, 3)$ y

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

```
def estado(A, v, k):
    for i in range(k):
        Av = A @ v
        v = Av / np.linalg.norm(Av, 2)
    return (v)

A = np.array([[3, 1, 2], [0, 1, -2], [1, 2, 4]])
v = np.array([1, 2, 3])
print(estado(A, v, 1))
print(estado(A, v, 10))
print(estado(A, v, 100))
print(estado(A, v, 1000))
```

```
%% [ 0.53295174 -0.19380063  0.82365269]
%% [ 0.74404261 -0.37049752  0.55599656]
%% [ 0.74278135 -0.37139068  0.55708601]
%% [ 0.74278135 -0.37139068  0.55708601]
```

Observamos que en este caso también existe límite $\mathbf{v}^{(k)} \xrightarrow{k \rightarrow \infty} \mathbf{w} = (2.97, -1.49, 2.23)$. ¿Será cierto en este caso también que \mathbf{w} es un autovector? ¿Cuál es el autovalor correspondiente? Para responder estas preguntas, utilizamos nuevamente que si \mathbf{w} es el límite de la sucesión, al aplicar la fórmula recursiva a \mathbf{w} obtenemos nuevamente \mathbf{w} . Es decir,

$$\mathbf{w} = \frac{\mathbf{Aw}}{\|\mathbf{Aw}\|_2},$$

y despejando, obtenemos

$$\mathbf{Aw} = \|\mathbf{Aw}\|_2 \mathbf{w}.$$

Concluimos que \mathbf{w} es autovector de \mathbf{A} con autovalor $\lambda = \|\mathbf{Aw}\|_2 = 4$ (verificar esta última cuenta en Python).

Para demostrar el resultado en este caso, calculamos nuevamente autovalores y autovectores de \mathbf{A} :

```
e = np.linalg.eig(A)
print(e[0])
print(e[1])
```

```
%% [4.+0.j 2.+1.j 2.-1.j]
%% [[ 0.74278135+0.j           0.35355339-0.35355339j  0.35355339+0.35355339j]
%% [-0.37139068+0.j          -0.70710678+0.j           -0.70710678-0.j      ]
%% [ 0.55708601+0.j           0.35355339+0.35355339j  0.35355339-0.35355339j]]
```

Nuevamente la matriz es diagonalizable con autovalores $\lambda_1 = 4$, $\lambda_2 = 2 + i$, $\lambda_3 = 2 - i$. Aplicando el razonamiento anterior,

$$\begin{aligned} \mathbf{v}^{(1)} &= \frac{\mathbf{Av}^{(0)}}{\|\mathbf{Av}^{(0)}\|_2}, \\ \mathbf{v}^{(2)} &= \frac{\mathbf{Av}^{(1)}}{\|\mathbf{Av}^{(1)}\|_2} = \frac{\mathbf{A} \frac{\mathbf{Av}^{(0)}}{\|\mathbf{Av}^{(0)}\|_2}}{\left\| \mathbf{A} \frac{\mathbf{Av}^{(0)}}{\|\mathbf{Av}^{(0)}\|_2} \right\|_2} = \frac{\frac{\mathbf{A}^2 \mathbf{v}^{(0)}}{\|\mathbf{Av}^{(0)}\|_2}}{\frac{\|\mathbf{A}^2 \mathbf{v}^{(0)}\|_2}{\|\mathbf{Av}^{(0)}\|_2}} = \frac{\mathbf{A}^2 \mathbf{v}^{(0)}}{\|\mathbf{A}^2 \mathbf{v}^{(0)}\|_2} \\ &\dots \\ \mathbf{v}^{(k)} &= \frac{\mathbf{Av}^{(k-1)}}{\|\mathbf{Av}^{(k-1)}\|_2} = \frac{\mathbf{A}^k \mathbf{v}^{(0)}}{\|\mathbf{A}^k \mathbf{v}^{(0)}\|_2}. \end{aligned}$$

Obtenemos en este caso,

$$\begin{aligned}
\mathbf{v}^{(k)} &= \frac{\mathbf{A}^k \mathbf{v}^{(0)}}{\|\mathbf{A}^k \mathbf{v}^{(0)}\|_2} \\
&= \frac{a_1(\lambda_1)^k \mathbf{w}_1 + a_2(\lambda_2)^k \mathbf{w}_2 + a_3(\lambda_3)^k \mathbf{w}_3}{\|a_1(\lambda_1)^k \mathbf{w}_1 + a_2(\lambda_2)^k \mathbf{w}_2 + a_3(\lambda_3)^k \mathbf{w}_3\|_2} \\
&= \frac{\lambda_1^k \left(a_1 \mathbf{w}_1 + a_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \mathbf{w}_2 + a_3 \left(\frac{\lambda_3}{\lambda_1} \right)^k \mathbf{w}_3 \right)}{\|\lambda_1|^k \|a_1 \mathbf{w}_1 + a_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \mathbf{w}_2 + a_3 \left(\frac{\lambda_3}{\lambda_1} \right)^k \mathbf{w}_3\|_2} \\
&= \left(\frac{\lambda_1}{|\lambda_1|} \right)^k \frac{a_1 \mathbf{w}_1 + a_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \mathbf{w}_2 + a_3 \left(\frac{\lambda_3}{\lambda_1} \right)^k \mathbf{w}_3}{\left\| a_1 \mathbf{w}_1 + a_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \mathbf{w}_2 + a_3 \left(\frac{\lambda_3}{\lambda_1} \right)^k \mathbf{w}_3 \right\|_2}
\end{aligned}$$

En nuestro ejemplo, $\lambda_1 = 4$ y $\left(\frac{\lambda_1}{|\lambda_1|} \right)^k = 1$. Los cocientes λ_i/λ_1 , $i = 2, 3$ tienden a 0 y por lo tanto $\mathbf{v}^{(k)}$ tiende a $\mathbf{w}_1/\|\mathbf{w}_1\|_2$.

Observamos que si λ_1 no es un número real positivo, $\lambda_1/|\lambda_1|$ es un número (complejo) de módulo 1, y $(\lambda_1/|\lambda_1|)^k$ tiene módulo 1 para todo k pero no tiene límite. Sin embargo, a partir de esta sucesión podemos aproximar el valor de λ_1 y conociendo λ_1 podemos obtener fácilmente un autovector correspondiente a este autovalor. Esta estrategia para calcular el autovalor de módulo máximo se conoce como el *método de la potencia*. Vamos a presentar el método realizando algunas suposiciones que facilitan la demostración. El método se puede aplicar en condiciones más generales, incluso cuando \mathbf{A} no es diagonalizable, pero en ese caso necesitamos utilizar la forma de Jordan de la matriz u otras herramientas que escapan el contenido de este apunte para la demostración.

LEMA 5.6.1. Si \mathbf{v} es un autovector de $\mathbf{A} \in \mathbb{C}^{n \times n}$ de autovalor λ , el cociente de Rayleigh

$$r(\mathbf{v}) = \frac{\mathbf{v}^T \mathbf{A} \mathbf{v}}{\mathbf{v}^T \mathbf{v}}$$

es igual al autovalor λ .

La demostración es inmediata usando que si \mathbf{v} es autovector de autovalor λ , $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$. A partir de la definición tenemos muchas formas de calcular el autovalor correspondiente a un autovector. Se puede demostrar que cuando tenemos una aproximación \mathbf{v} de un autovector, $r(\mathbf{v})$ es el valor que más se parece a un autovalor, en el sentido de que $\alpha = r(\mathbf{v})$ minimiza la norma de $\|\mathbf{A}\mathbf{v} - \alpha\mathbf{v}\|_2$.

Algorithm 1: Método de la potencia

```

 $\mathbf{A} \in \mathbb{C}^{n \times n}$ ,  $\mathbf{v}^{(0)} \in \mathbb{C}^n$ ;
for  $k = 1, 2, \dots$  do
     $\mathbf{v}^{(k)} = \frac{\mathbf{A}\mathbf{v}^{(k-1)}}{\|\mathbf{A}\mathbf{v}^{(k-1)}\|_2};$ 
     $r_k = \frac{(\mathbf{v}^{(k)})^T \mathbf{A} \mathbf{v}^{(k)}}{(\mathbf{v}^{(k)})^T \mathbf{v}^{(k)}}$ 
end

```

TEOREMA 5.6.2. Sea $\mathbf{A} \in \mathbb{C}^{n \times n}$ una matriz diagonalizable con base de autovectores $\mathcal{B} = \{\mathbf{w}_1, \dots, \mathbf{w}_n\}$ y autovalores correspondientes $\{\lambda_1, \dots, \lambda_n\}$ con $|\lambda_1| > |\lambda_2| \geq \dots \geq |\lambda_n| \geq 0$ (existe un único autovalor de módulo máximo). Dado un vector $\mathbf{v}^{(0)} \in \mathbb{C}^n$ tal que en su escritura en la base \mathcal{B} el coeficiente de \mathbf{w}_1 es no nulo, el algoritmo 1 satisface $|r_k - \lambda_1| \rightarrow 0$ cuando $k \rightarrow \infty$.

DEMOSTRACIÓN. Por hipótesis, podemos escribir

$$\mathbf{v}^{(0)} = a_1 \mathbf{w}_1 + \dots + a_n \mathbf{w}_n,$$

con $a_1 \neq 0$. Siguiendo los pasos del ejemplo, obtenemos

$$\mathbf{v}^{(k)} = \left(\frac{\lambda_1}{|\lambda_1|} \right)^k \frac{a_1 \mathbf{w}_1 + a_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \mathbf{w}_2 + \dots + a_n \left(\frac{\lambda_n}{\lambda_1} \right)^k \mathbf{w}_n}{\left\| a_1 \mathbf{w}_1 + a_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \mathbf{w}_2 + \dots + a_n \left(\frac{\lambda_n}{\lambda_1} \right)^k \mathbf{w}_n \right\|_2}.$$

Despejando,

$$\left(\frac{|\lambda_1|}{\lambda_1} \right)^k \mathbf{v}^{(k)} = \frac{a_1 \mathbf{w}_1 + a_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \mathbf{w}_2 + \dots + a_n \left(\frac{\lambda_n}{\lambda_1} \right)^k \mathbf{w}_n}{\left\| a_1 \mathbf{w}_1 + a_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \mathbf{w}_2 + \dots + a_n \left(\frac{\lambda_n}{\lambda_1} \right)^k \mathbf{w}_n \right\|_2}$$

y por lo tanto, llamando $\tilde{\mathbf{v}}^{(k)} = \left(\frac{|\lambda_1|}{\lambda_1} \right)^k \mathbf{v}^{(k)}$,

$$\tilde{\mathbf{v}}^{(k)} \xrightarrow{k \rightarrow \infty} \frac{a_1 \mathbf{w}_1}{\|a_1 \mathbf{w}_1\|_2}.$$

Como $\tilde{\mathbf{v}}^{(k)}$ es igual a $\mathbf{v}^{(k)}$ multiplicado por un escalar, podemos fácilmente ver que

$$r(\tilde{\mathbf{v}}^{(k)}) = \frac{(\tilde{\mathbf{v}}^{(k)})^T \mathbf{A} \tilde{\mathbf{v}}^{(k)}}{(\tilde{\mathbf{v}}^{(k)})^T \tilde{\mathbf{v}}^{(k)}} = \frac{(\mathbf{v}^{(k)})^T \mathbf{A} \mathbf{v}^{(k)}}{(\mathbf{v}^{(k)})^T \mathbf{v}^{(k)}} = r(\mathbf{v}^{(k)}).$$

Como $\tilde{\mathbf{v}}^{(k)}$ tiende a un autovector de \mathbf{A} , concluimos

$$\lim_{k \rightarrow \infty} r(\mathbf{v}^{(k)}) = \lim_{k \rightarrow \infty} r(\tilde{\mathbf{v}}^{(k)}) = \lambda.$$

□

5.7. Norma-2 de matrices

El estudio de autovalores y autovectores que realizamos en este capítulo nos permite dar una mejor interpretación a la norma-2 de matrices. Recordemos que dada una matriz $\mathbf{A} \in \mathbb{K}^{n \times n}$ y una norma vectorial $\|\cdot\|_n$ en \mathbb{K}^n , definimos la norma inducida

$$\|\mathbf{A}\| = \max_{\mathbf{0} \neq \mathbf{x} \in \mathbb{K}^n} \frac{\|\mathbf{Ax}\|}{\|\mathbf{x}\|} = \max_{\mathbf{x} \in \mathbb{K}^n, \|\mathbf{x}\|=1} \|\mathbf{Ax}\|.$$

En el caso de matrices diagonales, el cálculo de normas matriciales es muy simple.

PROPOSICIÓN 5.7.1. Si \mathbf{A} diagonal,

$$\mathbf{A} = \begin{pmatrix} \lambda_1 & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & \cdots & 0 \\ 0 & 0 & \lambda_3 & \cdots & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & & \lambda_n \end{pmatrix},$$

$$\|\mathbf{A}\|_2 = \max\{|\lambda_i|\}.$$

DEMOSTRACIÓN. Ejercicio. □

Esa simple observación motiva la siguiente definición.

DEFINICIÓN 5.7.2. Dada $\mathbf{A} \in \mathbb{K}^{n \times n}$ definimos el radio espectral de \mathbf{A} como

$$\rho(\mathbf{A}) = \max\{|\lambda|, \text{ con } \lambda \text{ autovalor de } \mathbf{A}\}$$

Supongamos que $\mathbf{A} \in \mathbb{R}^{n \times n}$ es simétrica, sea $\mathcal{B} = \{\mathbf{v}_1, \dots, \mathbf{v}_n\}$ una base ortonormal de autovectores y $\{\lambda_1, \lambda_2, \dots, \lambda_n\} \subset \mathbb{R}$ los respectivos autovalores. Sea $\mathbf{x} \in \mathbb{R}^n$ tal que $\|\mathbf{x}\|_2 = 1$, escribiendo \mathbf{x} en la base \mathcal{B} se tiene

$$\mathbf{x} = \sum_{i=1}^n \alpha_i \mathbf{v}_i,$$

y por ser \mathcal{B} un conjunto ortonormal tenemos por la Observación 3.2.8

$$1 = \|\mathbf{x}\|_2^2 = \sum_{i=1}^n \alpha_i^2.$$

Por otro lado, como $\mathbf{Ax} = \sum_{i=1}^n \alpha_i \lambda_i \mathbf{v}_i$, usando nuevamente la Observación 3.2.8

$$\|\mathbf{Ax}\|_2^2 = \sum_{i=1}^n \alpha_i^2 \lambda_i^2.$$

Por lo tanto

$$\frac{\|\mathbf{Ax}\|^2}{\|\mathbf{x}\|^2} = \sum_{i=1}^n \alpha_i^2 \lambda_i^2 \leq \rho(\mathbf{A})^2, \quad , \forall \mathbf{x}, \|\mathbf{x}\| \leq 1$$

ya que cada $|\lambda_i| \leq \rho(\mathbf{A})$. Por otro lado, llamando j a un índice tal que $|\lambda_j| = \rho(\mathbf{A})$ y considerando el vector de norma 1 $\mathbf{x} = \mathbf{v}_j$, se tiene

$$\|\mathbf{Ax}\|^2 = |\lambda_j|^2 \|\mathbf{x}\|_2^2 = \rho(\mathbf{A})^2.$$

En definitiva

$$\sup_{\|\mathbf{x}\|=1} \|\mathbf{Ax}\|^2 = \rho(\mathbf{A})^2.$$

Resultado que recuadramos debajo.

Si $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{A}^T = \mathbf{A}$ entonces

$$\|\mathbf{A}\|_2 = \rho(\mathbf{A}).$$

Podemos hacer una cuenta similar para el caso general. En efecto, consideremos ahora $\mathbf{A} \in \mathbb{R}^{n \times n}$ no necesariamente simétrica. Queremos calcular

$$\sup_{\|\mathbf{x}\|_2=1} \|\mathbf{Ax}\|_2^2 = (\mathbf{Ax})^T \mathbf{Ax} = \mathbf{x}^T (\mathbf{A}^T \mathbf{A}) \mathbf{x}$$

Si llamamos $\mathbf{M} = \mathbf{A}^T \mathbf{A}$, resulta simétrica. Consideremos una base $\mathcal{B} = \{\mathbf{v}_1, \dots, \mathbf{v}_n\}$ ortonormal de autovectores de \mathbf{M} y $\{\lambda_1, \lambda_2, \dots, \lambda_n\} \subset \mathbb{R}$ los respectivos autovalores. Notemos que \mathbf{M} es semidefinida positiva (ver Definición 3.2.10), pues de la ecuación previa se ve que para todo $\mathbf{x} \in \mathbb{R}^n$

$$\mathbf{x}^T \mathbf{M} \mathbf{x} \geq 0,$$

y en particular se sigue que $\lambda_i \geq 0$ pues si $\mathbf{v}_i \in \ker(\mathbf{A})$ debe ser $\lambda_i = 0$ y en caso contrario

$$0 < \mathbf{v}_i^T \mathbf{M} \mathbf{v}_i = \mathbf{v}_i^T \lambda_i \mathbf{v}_i = \lambda_i.$$

Con esta información escribimos como antes

$$\mathbf{x} = \sum_{i=1}^n \alpha_i \mathbf{v}_i, \quad 1 = \|\mathbf{x}\|_2^2 = \sum_{i=1}^n \alpha_i^2, \quad \mathbf{M} \mathbf{x} = \sum_{i=1}^n \alpha_i \mathbf{M} \mathbf{v}_i = \sum_{i=1}^n \lambda_i \alpha_i \mathbf{v}_i$$

de donde por la ortonormalidad de los \mathbf{v}_i

$$\mathbf{x}^T \mathbf{M} \mathbf{x} = \sum_{i=1}^n \lambda_i \alpha_i^2.$$

Luego,

$$\sup_{\|\mathbf{x}\|_2=1} \|\mathbf{Ax}\|_2^2 = \sup_{\sum_{i=1}^n \alpha_i^2=1} \sum_{i=1}^n \lambda_i \alpha_i^2 = \lambda_{max} = \rho(\mathbf{M}).$$

En definitiva tenemos

Sea $\mathbf{A} \in \mathbb{R}^{n \times n}$, entonces^a

$$\|\mathbf{A}\|_2 = \sqrt{\rho(\mathbf{A}^T \mathbf{A})}.$$

^aTambién vale la demostración escrita más arriba para el caso de matrices rectangulares.

EJERCICIO 5.7.3. Verifique que de la última expresión se obtiene como caso particular la de las matrices simétricas.

Otra propiedad importante es la siguiente

PROPOSICIÓN 5.7.4. Sea $\mathbf{A} \in \mathbb{K}^{n \times n}$, entonces

$$\rho(\mathbf{A}) = \inf\{\|\mathbf{A}\| : \|\cdot\| \text{ es una norma subordinada}\}.$$

DEMOSTRACIÓN. Solo vamos a probar que $\rho(\mathbf{A}) \leq \|\mathbf{A}\|$ para cualquier norma subordinada. La otra desigualdad es mas complicada y no la probaremos aquí.

Si $\mathbb{K} = \mathbb{C}$ la demostración es mas sencilla. Sea λ autovalor de \mathbf{A} y sea $\|\cdot\|$ una norma matricial en $\mathbb{C}^{n \times n}$ subordinada a la norma vectorial¹ $\|\cdot\|$. Luego existe $\mathbf{0} \neq \mathbf{v}$ autovector de autovalor λ .

¹Usamos la misma notación para ambas normas ya que el contexto indica cuando estamos usando una o la otra.

Podemos suponer que $\|\mathbf{v}\| = 1$ y entonces

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v},$$

de donde $\|\mathbf{A}\mathbf{v}\| \leq |\lambda|\|\mathbf{v}\| = |\lambda|$. Por lo tanto, para todo λ autovalor

$$\|\mathbf{A}\| = \sup_{\|\mathbf{x}\|=1} \|\mathbf{A}\mathbf{x}\| \geq \|\mathbf{A}\mathbf{v}\| \geq |\lambda|.$$

de donde

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

Si $\mathbb{K} = \mathbb{R}$ hay un problema si intentamos repetir el argumento previo vemos que funciona perfectamente si $\lambda \in \mathbb{R}$, sin embargo el autovalor y el autovector podrían ser complejos y no tener sentido la expresión $\|\mathbf{v}\|$ si se trata de una norma en \mathbb{R}^n . Supongamos entonces que $\lambda = a+ib \in \mathbb{C}$ y escribamos entonces,²

$$\mathbf{v} = \operatorname{Re}(\mathbf{v}) + i \operatorname{Im}(\mathbf{v})$$

y

$$\mathbf{A}\mathbf{v} = (a+ib)(\operatorname{Re}(\mathbf{v}) + i \operatorname{Im}(\mathbf{v})),$$

luego

$$\mathbf{A}\operatorname{Re}(\mathbf{v}) = a\operatorname{Re}(\mathbf{v}) - b\operatorname{Im}(\mathbf{v}) \quad \mathbf{A}\operatorname{Im}(\mathbf{v}) = b\operatorname{Re}(\mathbf{v}) + a\operatorname{Im}(\mathbf{v}),$$

y entonces

$$\begin{aligned} \|\mathbf{A}\operatorname{Re}(\mathbf{v})\|_2^2 &= a^2\|\operatorname{Re}(\mathbf{v})\|_2^2 - 2ab\langle \operatorname{Im}(\mathbf{v}), \operatorname{Re}(\mathbf{v}) \rangle + b^2\|\operatorname{Im}(\mathbf{v})\|_2^2 \\ \|\mathbf{A}\operatorname{Im}(\mathbf{v})\|_2^2 &= b^2\|\operatorname{Re}(\mathbf{v})\|_2^2 - 2ab\langle \operatorname{Im}(\mathbf{v}), \operatorname{Re}(\mathbf{v}) \rangle + a^2\|\operatorname{Im}(\mathbf{v})\|_2^2 \end{aligned}$$

sumando miembro a miembro

$$\|\mathbf{A}\operatorname{Re}(\mathbf{v})\|_2^2 + \|\mathbf{A}\operatorname{Im}(\mathbf{v})\|_2^2 = (a^2 + b^2) (\|\operatorname{Re}(\mathbf{v})\|_2^2 + \|\operatorname{Im}(\mathbf{v})\|_2^2)$$

de donde

$$(\|\operatorname{Re}(\mathbf{v})\|_2^2 + \|\operatorname{Im}(\mathbf{v})\|_2^2) \|\mathbf{A}\|_2^2 \geq (a^2 + b^2) (\|\operatorname{Re}(\mathbf{v})\|_2^2 + \|\operatorname{Im}(\mathbf{v})\|_2^2)$$

y entonces

$$\|\mathbf{A}\|_2^2 \geq (a^2 + b^2) = |\lambda|^2.$$

Es decir -ya que λ es arbitrario-

$$\|\mathbf{A}\|_2 \geq \rho(\mathbf{A}).$$

Sea ahora $\|\cdot\|$ una norma arbitraria en \mathbb{R}^n y $k \in \mathbb{N}$: por un lado sabemos que³ $\rho(\mathbf{A})^k \leq \rho(\mathbf{A}^k)$ y que además (por Proposición 3.1.8) existe una constante C -independiente de k - tal que $\|\mathbf{A}^k\|_2 \leq C\|\mathbf{A}^k\|$. Por lo tanto,

$$\rho(\mathbf{A})^k \leq \rho(\mathbf{A}^k) \leq \|\mathbf{A}^k\|_2 \leq C\|\mathbf{A}^k\| \leq C\|\mathbf{A}\|^k,$$

donde hemos usado (3.8) en la última desigualdad. Tomando raíz k -ésima

$$\rho(\mathbf{A}) \leq C^{1/k} \|\mathbf{A}\|,$$

tomando $k \rightarrow \infty$ se obtiene $\rho(\mathbf{A}) \leq \|\mathbf{A}\|$, como queríamos probar. \square

² Debemos este argumento a Ariel Lombardi, colega de la UNR.

³ En realidad vale el igual. Para ver la desigualdad basta notar que si λ es autovalor de \mathbf{A} entonces λ^k lo es de \mathbf{A}^k .

Como aplicación, obtenemos un criterio para determinar cuándo las potencias de una matriz tienden a la matriz nula.

PROPOSICIÓN 5.7.5. Sea $\mathbf{M} \in \mathbb{K}^{n \times n}$, se tiene que

$$\lim_{k \rightarrow \infty} \|\mathbf{M}^k\| = 0$$

para cierta norma $\|\cdot\|$ (y por ende para cualquier otra ya que son todas equivalentes) sí y solo si $\rho(\mathbf{M}) < 1$.

DEMOSTRACIÓN. Si $\rho(\mathbf{M}) < 1$, entonces existe $\|\cdot\|$ tal que $\|\mathbf{M}\| = \mu < 1$ y listo.

Por otro lado,

$$\rho(\mathbf{M})^k = \rho(\mathbf{M}^k) \leq \|\mathbf{M}^k\| \rightarrow 0,$$

de donde necesariamente $\rho(\mathbf{M}) < 1$. \square

5.8. El algoritmo QR

Vimos anteriormente el método de la potencia para calcular el autovalor de mayor módulo de una matriz. En este sección, veremos otro algoritmo que nos permite aproximar todos los autovalores y autovectores de una matriz de forma muy eficiente.

Una demostración completa de este algoritmo excede las intenciones de este apunte, por lo tanto nos limitaremos a ver las ideas principales.

La primera idea es extender el método de la potencia a subespacios.

El método de la potencia nos permite calcular el autovalor de mayor módulo mediante una sucesión $\{\mathbf{v}^{(k)}\}$ con

$$\mathbf{v}^{(k)} = \left(\frac{\lambda_1}{|\lambda_1|} \right)^k \frac{a_1 \mathbf{w}_1 + a_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \mathbf{w}_2 + \cdots + a_n \left(\frac{\lambda_n}{\lambda_1} \right)^k \mathbf{w}_n}{\left\| a_1 \mathbf{w}_1 + a_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k \mathbf{w}_2 + \cdots + a_n \left(\frac{\lambda_n}{\lambda_1} \right)^k \mathbf{w}_n \right\|_2}$$

Si bien, en general, $\{\mathbf{v}^{(k)}\} \not\rightarrow \mathbf{w}_1$ (ni a un múltiplo fijo), podemos intuitivamente pensar en convergencia de subespacios:

$$\langle \mathbf{v}^{(k)} \rangle \rightarrow \langle \mathbf{w}_1 \rangle.$$

¿Cómo calculamos los demás autovalores y autovectores?

Si aplicamos el método (sin normalizar) a dos vectores en vez de uno, podemos escribir

$$\begin{aligned} \mathbf{v}_1^{(k)} &= (\lambda_2)^k \left(a_1 \left(\frac{\lambda_1}{\lambda_2} \right)^k \mathbf{w}_1 + a_2 \mathbf{w}_2 + a_3 \left(\frac{\lambda_3}{\lambda_2} \right)^k \mathbf{w}_3 + \cdots + a_n \left(\frac{\lambda_n}{\lambda_2} \right)^k \mathbf{w}_n \right), \\ \mathbf{v}_2^{(k)} &= (\lambda_2)^k \left(b_1 \left(\frac{\lambda_1}{\lambda_2} \right)^k \mathbf{w}_1 + b_2 \mathbf{w}_2 + b_3 \left(\frac{\lambda_3}{\lambda_2} \right)^k \mathbf{w}_3 + \cdots + b_n \left(\frac{\lambda_n}{\lambda_2} \right)^k \mathbf{w}_n \right). \end{aligned}$$

Intuitivamente vemos que $\langle \mathbf{v}_1^{(k)}, \mathbf{v}_2^{(k)} \rangle \rightarrow \langle \mathbf{w}_1, \mathbf{w}_2 \rangle$ cuando $k \rightarrow \infty$.

1. Si bien $\langle \mathbf{v}_1^{(k)}, \mathbf{v}_2^{(k)} \rangle \rightarrow \langle \mathbf{w}_1, \mathbf{w}_2 \rangle$, individualmente $\langle \mathbf{v}_1^{(k)} \rangle \rightarrow \langle \mathbf{w}_1 \rangle$ y $\langle \mathbf{v}_2^{(k)} \rangle \rightarrow \langle \mathbf{w}_2 \rangle$.

2. Si logramos “sacarle” a $\mathbf{v}_2^{(k)}$ la componente en la dirección de \mathbf{w}_1 , vamos a obtener $\langle \tilde{\mathbf{v}}_2^{(k)} \rangle \rightarrow \langle \mathbf{w}_2 \rangle$.
3. Si $\{\mathbf{w}_1, \dots, \mathbf{w}_n\}$ base ortonormal y $\mathbf{v} = a_1\mathbf{w}_1 + \dots + a_n\mathbf{w}_n$, tenemos $(\mathbf{v}, \mathbf{w}_1) = a_1$.
4. Por lo tanto, si conocemos \mathbf{w}_1 , podemos hacer

$$\mathbf{v} - (\mathbf{v}, \mathbf{w}_1)\mathbf{w}_1 = a_2\mathbf{w}_2 + \dots + a_n\mathbf{w}_n.$$

5. En nuestro caso,

$$\langle \mathbf{v}_2^{(k)} - (\mathbf{v}_2^{(k)}, \mathbf{w}_1)\mathbf{w}_1 \rangle \rightarrow \langle \mathbf{w}_2 \rangle.$$

En general, no conocemos \mathbf{w}_1 pero podemos usar $\mathbf{v}_1^{(k)}$.

Si $\langle \mathbf{v}_1^{(k)} \rangle \rightarrow \langle \mathbf{w}_1 \rangle$ y definimos $\mathbf{u}_1^{(k)} = \mathbf{v}_1^{(k)}$, $\tilde{\mathbf{u}}_1^{(k)} = \frac{\mathbf{u}_1^{(k)}}{\|\mathbf{u}_1^{(k)}\|}$,

$$\mathbf{u}_2^{(k)} = \mathbf{v}_2^{(k)} - \langle \tilde{\mathbf{u}}_1^{(k)}, \mathbf{v}_2^{(k)} \rangle \tilde{\mathbf{u}}_1^{(k)} \quad \text{y} \quad \tilde{\mathbf{u}}_2^{(k)} = \frac{\mathbf{u}_2^{(k)}}{\|\mathbf{u}_2^{(k)}\|},$$

obtenemos $\langle \tilde{\mathbf{u}}_2^{(k)} \rangle \rightarrow \langle \mathbf{w}_2 \rangle$.

OBSERVACIÓN 5.8.1. $\{\tilde{\mathbf{u}}_1^{(k)}, \tilde{\mathbf{u}}_2^{(k)}\}$ es el resultado de aplicar Gram-Schmidt a $\{\mathbf{v}_1^{(k)}, \mathbf{v}_2^{(k)}\}$.

Analizamos ahora el caso de matrices simétricas.

Si extendemos el razonamiento a un conjunto inicial $\{\mathbf{v}_1^{(0)}, \dots, \mathbf{v}_n^{(0)}\}$ de vectores linealmente independientes podemos generar sucesiones de vectores tales que $\langle \tilde{\mathbf{u}}_i^{(k)} \rangle \rightarrow \langle \mathbf{w}_i \rangle$.

Trabajando con matrices, podemos aplicar el método simultáneamente a un conjunto de vectores. Obtenemos el siguiente método para matrices simétricas.

Algorithm 2: Iteración simultánea

A matriz (simétrica) con autovalores $|\lambda_1| > |\lambda_2| > \dots > |\lambda_n|$;
B⁽⁰⁾ = \mathbf{I}_n (o cualquier matriz de rango máximo);
for $k = 1, 2, \dots$ **do**
 $\tilde{\mathbf{B}}^{(k)} = \mathbf{AB}^{(k-1)}$;
 $\tilde{\mathbf{B}}^{(k)} = \mathbf{Q}^{(k)} \mathbf{R}^{(k)}$, descomposición QR de $\tilde{\mathbf{B}}^{(k)}$;
 $\mathbf{B}^{(k)} = \mathbf{Q}^{(k)}$, el resultado de aplicar G-S a las columnas de $\tilde{\mathbf{B}}^{(k)}$
end

Si $\mathbf{c}_i^{(k)}$ es la i -ésima columna de $\mathbf{B}^{(k)}$,

$$\langle \mathbf{c}_i^{(k)} \rangle \rightarrow \langle \mathbf{w}_i \rangle.$$

¿Cómo aproximamos los autovalores? La matriz \mathbf{A} en la base $\mathcal{B}^{(k)}$ dada por $\mathbf{B}^{(k)}$,

$$[\mathbf{A}]_{\mathcal{B}^{(k)}} = (\mathbf{B}^{(k)})^T \mathbf{AB}^{(k)},$$

es una matriz diagonal con los autovalores en la diagonal.

EJERCICIO 5.8.2. Aplicar el algoritmo a una matriz simétrica \mathbf{A} arbitraria y comparar los autovalores y autovectores obtenidos con los que se obtienen por el comando `eig` de Python.

¿Qué sucede si aplicamos el método anterior para matrices generales con autovalores $|\lambda_1| > |\lambda_2| > \dots > |\lambda_n|$?

Si bien no se cumple $\langle \tilde{\mathbf{v}}_i^{(k)} \rangle \rightarrow \langle \mathbf{w}_i \rangle$ para todo i , obtenemos en cambio

$$\left\{ \begin{array}{l} \langle \tilde{\mathbf{v}}_1^{(k)} \rangle \rightarrow \langle \mathbf{w}_1 \rangle \\ \langle \tilde{\mathbf{v}}_1^{(k)}, \tilde{\mathbf{v}}_2^{(k)} \rangle \rightarrow \langle \mathbf{w}_1, \mathbf{w}_2 \rangle \\ \vdots \\ \langle \tilde{\mathbf{v}}_1^{(k)}, \dots, \tilde{\mathbf{v}}_n^{(k)} \rangle \rightarrow \langle \mathbf{w}_1, \dots, \mathbf{w}_n \rangle \end{array} \right.$$

Por lo tanto,

$$[\mathbf{A}]_{\mathcal{B}^{(k)}} = (\mathbf{B}^{(k)})^T \mathbf{A} \mathbf{B}^{(k)},$$

tiende a una matriz triangular superior con los autovalores en la diagonal.

Finalmente, acomodando las operaciones convenientemente, obtenemos el siguiente método.

Algorithm 3: Método QR

```

 $\mathbf{A}^{(0)} = \mathbf{A} \in \mathbb{C}^{n \times n}$  con autovalores  $|\lambda_1| > |\lambda_2| > \dots > |\lambda_n|$ ;
for  $k = 0, 1, 2, \dots$  do
    | Calcular la descomposición QR de  $\mathbf{A}^{(k)}$ ,  $\mathbf{A}^{(k)} = \mathbf{Q}^{(k)} \mathbf{R}^{(k)}$  ;
    |  $\mathbf{A}^{(k+1)} = \mathbf{R}^{(k)} \mathbf{Q}^{(k)}$ 
end

```

PROPOSICIÓN 5.8.3. La matriz $\mathbf{A}^{(k)}$ tiende a una matriz triangular superior con los autovalores de \mathbf{A} en la diagonal.

OBSERVACIÓN 5.8.4. La matriz que se obtiene en cada paso es semejante a la matriz del paso anterior:

$$(\mathbf{Q}^{(k)})^T \mathbf{A}^{(k)} \mathbf{Q}^{(k)} = (\mathbf{Q}^{(k)})^T (\mathbf{Q}^{(k)} \mathbf{R}^{(k)}) \mathbf{Q}^{(k)} = \mathbf{R}^{(k)} \mathbf{Q}^{(k)} = \mathbf{A}^{(k+1)},$$

por lo tanto las matrices $\mathbf{A}^{(k)}$ tienen los mismos autovalores para todo $k = 0, 1, 2, \dots$

EJERCICIO 5.8.5. Aplicar el algoritmo a una matriz \mathbf{A} arbitraria y comparar los autovalores obtenidos con los que se obtienen por el comando `eig` de Python. ¿Cómo podemos calcular los autovectores?

EJERCICIO 5.8.6. ¿Qué sucede si hay autovalores del mismo módulo? Analizar el caso $\mathbf{A} \in \mathbb{R}^{n \times n}$ con autovalores complejos conjugados.

5.9. Teorema de los círculos de Gershgorin

Esta es otra herramienta en la caja de los buscadores de autovalores.

Dada $\mathbf{A} \in \mathbb{C}^{n \times n}$, definimos para $1 \leq i \leq n$, R_i la suma de los módulos de los elementos de la i -ésima fila de \mathbf{A} fuera de la diagonal,

$$R_i = \sum_{j \neq i} |a_{ij}|.$$

Llamamos $D(a_{ii}, R_i) \subset \mathbb{C}$ al disco cerrado centrado en a_{ii} de radio R_i .

TEOREMA 5.9.1. Todos los autovalores de \mathbf{A} están en alguno de los discos de Gershgorin $D(a_{ii}, R_i)$.

Capítulo 6

Procesos de Markov

Estudiamos ahora los procesos de Markov como aplicación de diagonalización de matrices. Comenzamos por un ejemplo.

EJEMPLO 6.0.1. En una ciudad, todos los adultos tienen teléfono celular de una de las tres compañías A, B o C. Cada mes, algunos clientes se cambian de compañía, según la siguiente fórmula:

$$\begin{pmatrix} a_{k+1} \\ b_{k+1} \\ c_{k+1} \end{pmatrix} = \begin{pmatrix} 0.9 & 0.15 & 0.25 \\ 0.075 & 0.8 & 0.25 \\ 0.025 & 0.05 & 0.5 \end{pmatrix} \begin{pmatrix} a_k \\ b_k \\ c_k \end{pmatrix}.$$

La casillas de la matriz indican que porcentaje de los clientes de cada compañía se queda en la misma compañía o se cambian a otra compañía. Por ejemplo $a_{k+1} = 0.9a_k + 0.15b_k + 0.25c_k$, lo que significa que el 90 % de los clientes de la compañía A en el mes k siguen en la compañía A en el mes $k+1$, el 15 % de los clientes de la compañía B se pasan a A y el 25 por ciento de los clientes de la compañía C se pasan a A.

Queremos ver cuántos clientes hay en cada compañía con el paso del tiempo, y si esas cantidades convergen a algún valor límite. Para esto, contestamos las siguientes preguntas utilizando Python.

Si inicialmente hay 1000 clientes en cada compañía, es decir $v_0 = (a_0, b_0, c_0) = (1000, 1000, 1000)$,

1. hallar $v_1 = (a_1, b_1, c_1)$ y $v_2 = (a_2, b_2, c_2)$,
2. hallar v_{10} y v_{100} ,
3. hallar (si existe) el límite de la cantidad de clientes en cada compañía, es decir $\lim_{k \rightarrow \infty} (a_k, b_k, c_k)$.

Para contestar 1, calculamos $v_1 = Av_0$ y $v_2 = Av_1$.

```
import numpy as np
A = np.array([[0.9, 0.15, 0.25], [0.075, 0.8, 0.25], [0.025, 0.05, 0.5]])

v0 = np.array([1000, 1000, 1000])
v1 = A @ v0
print("v1 = ", v1)
v2 = A @ v1
print("v2 = ", v2)

%% v1 =  [1300. 1125. 575.]
```

```
%% v2 = [1482.5 1141.25 376.25]
```

En general, para calcular $v^{(k)}$, tenemos

$$v^{(k)} = Av^{(k-1)} = A(Av^{(k-2)}) = \dots = A(A(\dots(Av^{(0)})\dots)),$$

y podemos calcular estos vectores mediante una iteración en Python:

```
v = np.array([1000, 1000, 1000])
k = 10
for i in range(k):
    v = A @ v
print("v(", k, ") = ", v)

v = np.array([1000, 1000, 1000])
k = 100
for i in range(k):
    v = A @ v
print("v(", k, ") = ", v)
```

```
%% v( 10 ) = [1844.0598284 965.482631 190.4575406]
%% v( 100 ) = [1875. 937.5 187.5]
```

Finalmente, para calcular el límite mediante cálculos en Python, tomamos valores mayores de k y vemos si las coordenadas de $v^{(k)}$ se estabilizan.

```
v = np.array([1000, 1000, 1000])
k = 1000
for i in range(k):
    v = A @ v
print("v(", k, ") = ", v)

v = np.array([1000, 1000, 1000])
k = 10000
for i in range(k):
    v = A @ v
print("v(", k, ") = ", v)
```

```
%% v( 1000 ) = [1875. 937.5 187.5]
%% v( 10000 ) = [1875. 937.5 187.5]
```

Observamos que $\lim_{k \rightarrow \infty} v^{(k)} = (1875, 937.5, 187.5)$.

A partir de este ejemplo, nos planteamos las siguientes preguntas:

- ¿Cómo podemos analizar el comportamiento de $v^{(k)}$?
- ¿Será cierto que para cualquier vector inicial, existe $v^{(\infty)} = \lim_{k \rightarrow \infty} v^{(k)}$.

Para contestar estas preguntas vamos a estudiar los llamados procesos de Markov.

6.1. Matrices de Markov

DEFINICIÓN 6.1.1. Decimos que una matriz $\mathbf{A} \in \mathbb{R}^{n \times n}$ es *de Markov* (o estocástica) si

- Todas las casillas de \mathbf{A} son números reales no negativos
- La suma de las casillas en cualquier columna de \mathbf{A} es siempre 1.

En particular, estas dos condiciones implican que todas las coordenadas de la matriz deben ser números entre 0 y 1. Observamos que la matriz del ejemplo cumple estas condiciones.

6.1.1. Espectro de las matrices de Markov. Observamos que si \mathbf{v}^∞ es el límite de un proceso de Markov, debe cumplir $\mathbf{A}\mathbf{v}^\infty = \mathbf{v}^\infty$, es decir \mathbf{v}^∞ es un autovector de \mathbf{A} de autovalor 1.

Estudiamos ahora entonces los autovalores y autovectores de las matrices estocásticas.

PROPOSICIÓN 6.1.2. Si \mathbf{A} es una matriz de Markov, $\lambda = 1$ es un autovalor de \mathbf{A} .

DEMOSTRACIÓN. Como las columnas de \mathbf{A} suman 1, si $\mathbf{v} = (1, 1, 1, \dots, 1)$,

$$\mathbf{A}^T \mathbf{v} = \mathbf{v},$$

luego \mathbf{v} es autovector de \mathbf{A}^T con autovalor 1. Como los autovalores de \mathbf{A} y \mathbf{A}^T son los mismos, concluimos que $\lambda = 1$ es autovalor de \mathbf{A} (aunque no necesariamente el autovector correspondiente es $\mathbf{v} = (1, 1, \dots, 1)$). \square

PROPOSICIÓN 6.1.3. Si \mathbf{A} es una matriz de Markov y λ es un autovalor de \mathbf{A} , entonces $|\lambda| \leq 1$.

DEMOSTRACIÓN. Si $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$, para $1 \leq i \leq n$,

$$|\lambda||v_i| = |\lambda v_i| = |A_{i1}v_1 + \dots + A_{in}v_n| \leq |A_{i1}v_1| + \dots + |A_{in}v_n| = A_{i1}|v_1| + \dots + A_{in}|v_n|$$

(para la última igualdad utilizamos que todas las coordenadas de \mathbf{A} son no-negativas).

Ahora sumamos sobre todos los i :

$$\begin{aligned} |\lambda|(|v_1| + \dots + |v_n|) &\leq (A_{11}|v_1| + \dots + A_{1n}|v_n|) + \dots + (A_{n1}|v_1| + \dots + A_{nn}|v_n|) \\ &= (A_{11} + \dots + A_{n1})|v_1| + \dots + (A_{1n} + \dots + A_{nn})|v_n| \\ &= |v_1| + \dots + |v_n| \end{aligned}$$

Como $|v_1| + \dots + |v_n| \geq 0$, obtenemos $|\lambda| < 1$. \square

PROPOSICIÓN 6.1.4. Si \mathbf{A} es una matriz de Markov, $\lambda \neq 1$ es un autovalor de \mathbf{A} y \mathbf{v} es un autovector asociado a λ , entonces $v_1 + \dots + v_n = 0$.

DEMOSTRACIÓN. Si $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$, para $1 \leq i \leq n$,

$$\lambda v_i = A_{i1}v_1 + \dots + A_{in}v_n$$

y

$$\begin{aligned} \lambda(v_1 + \dots + v_n) &= (A_{11}v_1 + \dots + A_{1n}v_n) + \dots + (A_{n1}v_1 + \dots + A_{nn}v_n) \\ &= (A_{11} + \dots + A_{n1})v_1 + \dots + (A_{1n} + \dots + A_{nn})v_n \\ &= v_1 + \dots + v_n \end{aligned}$$

Como $\lambda \neq 1$, debe ser $v_1 + \dots + v_n = 0$. \square

6.1.2. Probabilidades y proporciones. Si consideramos un sistema con n estados distintos, podemos considerar a la casillas (i, j) de una matriz de Markov como la probabilidad de pasar del estado j al estado i .

EJEMPLO 6.1.5. Para modelar el estado del tiempo en una ciudad, consideramos tres estados: soleado, nublado y lluvioso, y la matriz de Markov que indica la probabilidad de pasar de un estado a otro de un día a otro:

$$\begin{pmatrix} 0.6 & 0.2 & 0.1 \\ 0.3 & 0.6 & 0.4 \\ 0.1 & 0.2 & 0.5 \end{pmatrix}.$$

Si consideramos que inicialmente tenemos la misma probabilidad de que el día sea soleado, nublado o lluvioso, es decir, comenzamos con un vector inicial de probabilidades:

$$\mathbf{v}^{(0)} = (1/3, 1/3, 1/3)$$

calculando $\mathbf{v}^{(k)}$ podremos estimar la probabilidad de que llueva en un día específico, o calculando $\mathbf{v}^{(\infty)}$ podemos estimar la probabilidad de que llueva a largo plazo.

6.1.3. Conjunto de estados. Definimos el conjunto \mathcal{P}_n de estados

$$\mathcal{P}_n = \{\boldsymbol{\pi} \in \mathbb{R}_+^n : \pi_1 + \dots + \pi_n = 1\}$$

que podemos pensar como vectores de probabilidades.

PROPOSICIÓN 6.1.6. Si $\boldsymbol{\pi} \in \mathcal{P}_n$ y $\mathbf{M} \in \mathbb{R}^{n \times n}$ es una matriz de Markov, $\mathbf{M}\boldsymbol{\pi} \in \mathcal{P}_n$.

PROPOSICIÓN 6.1.7. Si $\mathbf{M} \in \mathbb{R}^{n \times n}$, existe $\boldsymbol{\pi} \in \mathcal{P}_n$ tal que $\mathbf{M}\boldsymbol{\pi} = \boldsymbol{\pi}$.

DEMOSTRACIÓN. Ver que si $\boldsymbol{\pi} = (\pi_1, \dots, \pi_n)$ satisface $\mathbf{M}\boldsymbol{\pi} = \boldsymbol{\pi}$, entonces también $|\boldsymbol{\pi}| = (|\pi_1|, \dots, |\pi_n|)$ satisface $\mathbf{M}|\boldsymbol{\pi}| = |\boldsymbol{\pi}|$ (ejercicio). \square

6.1.4. Estados de equilibrio. Un estado $\boldsymbol{\pi} \in \mathcal{P}_n$ se dice *estado de equilibrio* de una matriz de Markov \mathbf{M} si cumple

$$\mathbf{M}\boldsymbol{\pi} = \boldsymbol{\pi},$$

o equivalentemente, si es un autovector de \mathbf{M} de autovalor 1.

- Toda matriz de Markov tiene al menos un estado de equilibrio.
- Una matriz de Markov puede tener varios estados de equilibrio.
- Para el autovalor $\lambda = 1$, la multiplicidad algebraica coincide con la multiplicad geométrica, y podemos hallar una base de autovectores de E_1 formada por vectores de estado (sin demostración).

EJERCICIO 6.1.8. Hallar dos estados de equilibrio distintos para la matriz

$$\mathbf{M} = \begin{pmatrix} 0.5 & 0.3 & 0 & 0 \\ 0.5 & 0.7 & 0 & 0 \\ 0 & 0 & 0.1 & 0.5 \\ 0 & 0 & 0.9 & 0.5 \end{pmatrix}.$$

¿Cuántos estados de equilibrio tiene \mathbf{M} ?

6.1.5. Estados límite. Nos preguntamos si dado un estado inicial $\boldsymbol{\pi}^{(0)}$, existe

$$\boldsymbol{\pi}^* = \lim_{k \rightarrow \infty} \boldsymbol{\pi}^{(k)}.$$

En tal caso,

$$\boldsymbol{\pi}^* = \lim_{k \rightarrow \infty} \boldsymbol{\pi}^{(k+1)} = \lim_{k \rightarrow \infty} \mathbf{M}\boldsymbol{\pi}^{(k)} = \mathbf{M}\boldsymbol{\pi}^*,$$

es decir, $\boldsymbol{\pi}^*$ es un estado de equilibrio.

Preguntas:

- ¿Dado un estado inicial $\boldsymbol{\pi}^{(0)}$, el sistema evoluciona siempre hacia un estado límite?
- Si el estado de equilibrio no es único, ¿cómo determinamos a qué estado converge?

6.1.6. Autovalores y autovectores. Supongamos \mathbf{M} diagonalizable, con autovalores $|\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_n| \geq 0$, y autovectores correspondientes $\mathbf{w}_1, \dots, \mathbf{w}_n$.

Usando las propiedades vistas, si s es la multiplicidad del autovalor 1, podemos suponer $\lambda_1 = \lambda_2 = \dots = \lambda_s = 1$ y $\mathbf{w}_1, \dots, \mathbf{w}_s$ vectores de estado.

Dado $\boldsymbol{\pi}^{(0)} \in \mathcal{P}$, escribimos

$$\boldsymbol{\pi}^{(0)} = a_1 \mathbf{w}_1 + \dots + a_n \mathbf{w}_n$$

y

$$\boldsymbol{\pi}^{(k)} = a_1(\lambda_1)^k \mathbf{w}_1 + \dots + a_n(\lambda_n)^k \mathbf{w}_n$$

EJERCICIO 6.1.9. Si $\boldsymbol{\pi}^{(0)} \in \mathcal{P}$, existe $1 \leq i \leq s$ tal que $a_i \neq 0$. ¿Por qué?

6.1.7. Cálculo analítico de estados límite. A partir de la expresión

$$\begin{aligned} \boldsymbol{\pi}^{(k)} &= a_1(\lambda_1)^k \mathbf{w}_1 + \dots + a_n(\lambda_n)^k \mathbf{w}_n \\ &= a_1 \mathbf{w}_1 + \dots + a_s \mathbf{w}_s + a_{s+1}(\lambda_{s+1})^k \mathbf{w}_{s+1} + \dots + a_n(\lambda_n)^k \mathbf{w}_n \end{aligned}$$

utilizando que $\lambda_i^k \rightarrow 0$ si $|\lambda_i| < 0$ deducimos:

EJEMPLO 6.1.10.

- Si 1 es el único autovalor de módulo 1,

$$\lim_{k \rightarrow \infty} \boldsymbol{\pi}^{(k)} = a_1 \mathbf{w}_1 + \dots + a_s \mathbf{w}_s.$$

- Si además $s = 1$, $\boldsymbol{\pi}^{(k)} \rightarrow \mathbf{w}_1$.
- Si existe $\lambda_i \neq 1$, $|\lambda_i| = 1$, el proceso de Markov puede no tener límite (depende del vector inicial de estados).

6.1.8. Diagonalización de matrices de Markov. Si \mathbf{M} diagonalizable, con autovalores $1 = |\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_n| \geq 0$, y autovectores correspondientes $\mathbf{w}_1, \dots, \mathbf{w}_n$, tenemos

$$\mathbf{M} = \mathbf{C}_{\mathcal{B}\mathcal{E}} \mathbf{D} \mathbf{C}_{\mathcal{E}\mathcal{B}} = \mathbf{C}_{\mathcal{B}\mathcal{E}} \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix} \mathbf{C}_{\mathcal{E}\mathcal{B}} \quad \text{y}$$

$$\mathbf{M}^k = \mathbf{C}_{\mathcal{B}\mathcal{E}} \mathbf{D}^k \mathbf{C}_{\mathcal{E}\mathcal{B}} = \mathbf{C}_{\mathcal{B}\mathcal{E}} \begin{pmatrix} \lambda_1^k & & \\ & \ddots & \\ & & \lambda_n^k \end{pmatrix} \mathbf{C}_{\mathcal{E}\mathcal{B}},$$

$$\text{con } \mathbf{C}_{\mathcal{BE}} = \begin{pmatrix} | & & | \\ \mathbf{w}_1 & \dots & \mathbf{w}_n \\ | & & | \end{pmatrix}.$$

6.1.9. Matriz \mathbf{M}^∞ . Notamos $\mathbf{M}^\infty = \lim_{k \rightarrow \infty} \mathbf{M}^k$, cuando ese límite existe.

PROPOSICIÓN 6.1.11. Se cumplen las siguientes propiedades.

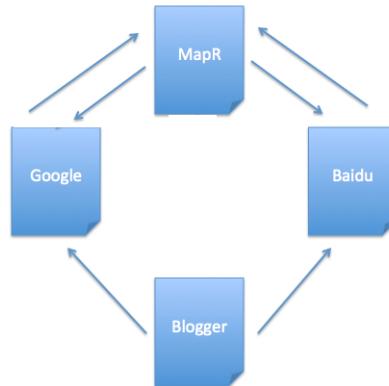
- Si existe \mathbf{M}^∞ , $\boldsymbol{\pi}^\infty = \mathbf{M}^\infty \boldsymbol{\pi}^{(0)}$ para cualquier vector inicial de estados.
- Existe \mathbf{M}^∞ si y solo si $\lambda = 1$ es el único autovalor de módulo 1.
- En este caso, $\mathbf{M}^\infty = \mathbf{C}_{\mathcal{BE}} \mathbf{D}^\infty \mathbf{C}_{\mathcal{EB}}$, y tomando $1^\infty = 1$ y $\lambda^\infty = 0$ para $|\lambda| < 1$ obtenemos

$$\mathbf{D}^\infty = \begin{pmatrix} \lambda_1^\infty & & & \\ & \ddots & & \\ & & \lambda_n^\infty & \\ & & & \end{pmatrix} = \begin{pmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & & 0 \end{pmatrix}$$

- Si el autovalor $\lambda = 1$ tiene multiplicidad 1, con autovector de estado \mathbf{w} , entonces

$$\mathbf{M}^\infty = \begin{pmatrix} | & & | \\ \mathbf{w} & \dots & \mathbf{w} \\ | & & | \end{pmatrix}.$$

EJERCICIO 6.1.12. Para estudiar la relevancia de 4 páginas, se utiliza el modelo de Google Page Rank.



Las flechas indican los enlaces que hay en cada página. Se supone que inicialmente hay $1/4$ del total de personas en cada una de las 4 páginas.

Hallar la matriz de Markov y determinar, si existe, $\boldsymbol{\pi}^{(\infty)}$ en cada caso.

1. Considerando que cada minuto, los navegantes clickean alguno de los links al azar disponibles en la página que se encuentran.
2. Considerando que cada minuto, la mitad de los navegantes que están en una página se quedan en esa página y la otra mitad clickean alguno de los links al azar disponibles en la página que se encuentran.

Capítulo 7

Métodos Iterativos

Como ya hemos visto, los métodos directos poseen en general una complejidad de orden cúbico en el número de incógnitas $\mathcal{O}(N^3)$. Esto dice que si en un problema determinado se duplica el número de incógnitas entonces el número de operaciones se multiplica por 8. Eso ha llevado a intentar desarrollar métodos de orden menor¹ y los métodos iterativos son una posible alternativa.

Hay, además de la complejidad, otros problemas con los métodos directos. Por ejemplo, en muchas aplicaciones aparecen problemas con un número muy grande de incógnitas pero con matrices *rajas*, es decir, matrices con un elevado porcentaje de coeficiente nulos. Esto permite en principio reducir notablemente el número de operaciones del algoritmo pero sin embargo como desventaja aparece un “rellenado” innecesario en la factorización resultante. En la siguiente sección vemos un ejemplo sencillo.

7.1. Un ejemplo: distribución de temperatura

Ciertas ecuaciones diferenciales lineales pueden resolverse de modo aproximado a través de *discretizaciones*. En dimensión uno es fácil imaginar como puede hacerse. Por ejemplo, consideremos en el intervalo $[a, b]$ la ecuación diferencial

$$(7.4) \quad \begin{cases} u''(x) &= f(x), \\ u(a) &= 0, \\ u(b) &= 0. \end{cases}$$

Aquí, $f(x)$ representa un dato conocido. Podemos aproximar el problema en dimensión finita del siguiente modo: consideramos puntos (en este ejemplo uniformemente distribuidos) $a = x_0 < x_1 < \dots < x_{n+1} = b$ tales que $x_i = a + ih$ donde $h = \frac{b-a}{n+1}$. Cuando $n \rightarrow \infty$ (aumentamos el número de puntos) resulta que $h \rightarrow 0$ y como

$$\frac{u(x_{i+1}) - 2u(x_i) + u(x_{i-1})}{h^2} = \frac{u(x_i + h) - 2u(x_i) + u(x_i - h)}{h^2}$$

tenemos para u suficientemente derivable

$$\begin{aligned} u(x_i + h) &= u(x_i) + u'(x_i)h + u''(x_i)h^2/2 + u'''(x_i)h^3/3! + \mathcal{O}(h^4) \\ u(x_i - h) &= u(x_i) - u'(x_i)h + u''(x_i)h^2/2 - u'''(x_i)h^3/3! + \mathcal{O}(h^4) \end{aligned}$$

y así

$$\frac{u(x_i + h) - 2u(x_i) + u(x_i - h)}{h^2} = u''(x_i) + \mathcal{O}(h^2),$$

¹Observe que el límite inferior para un método de resolución es $\mathcal{O}(N^2)$ ya que ese es el número de coeficientes de la matriz.

por lo que usando que $u''(x_i) = f(x_i)$, vemos que podemos aproximar la temperatura en el punto x_i resolviendo

$$u(x_{i+1}) - 2u(x_i) + u(x_{i-1}) = f(x_i)h^2,$$

lo que conduce a un sistema *tridiagonal y simétrico*, con una ecuación por cada x_i , que se escribe²

$$(7.5) \quad \begin{pmatrix} -2 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & -2 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & -2 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & -2 \end{pmatrix} \begin{pmatrix} u(x_1) \\ u(x_2) \\ \vdots \\ u(x_n) \end{pmatrix} = h^2 \begin{pmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{pmatrix}.$$

La ecuación (7.4) modeliza (en un caso muy elemental y sin contemplar constantes físicas) el problema de la distribución de temperatura en una barra unidimensional. En dimensión mas alta (por ejemplo en dimensión 3) el problema se puede modelizar con la ecuación³

$$\Delta u(x) = f(x).$$

Para ver un ejemplo concreto en la Figura 7.1 se muestra un cuarto de pistón (pieza que se desplaza dentro de uno de los cilindros de un motor). La distribución de temperatura en esa pieza puede aproximarse como antes través de una discretización. No vamos a describir aquí

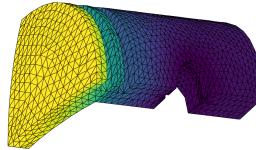


FIGURA 1. Distribución de temperatura en un pistón. Se visualiza un cuarto de la pieza.

como hacer esto, pero una vez discretizado, el problema puede escribirse, al igual que el problema unidimensional, como un sistema lineal de ecuaciones. En este ejemplo particular, se obtiene un sistema

$$\mathbf{Ax} = \mathbf{b},$$

$$\mathbf{A} \in \mathbb{R}^{3319 \times 3319}$$

donde cada incógnita es el valor de la temperatura en un punto interior del pistón. Este ejemplo (que llamaremos (E)) nos servirá para ejemplificar algunas particularidades que aparecen en este capítulo. Si bien esta matriz \mathbf{A} no es mas tridiagonal *sí es simétrica*, además de rala⁴. En la Figura 7.1 se observan ambos hechos (con el comando np.spy(\mathbf{A})). En este caso, solamente 39753 coeficientes son diferentes de cero, en vez $3319^2 = 11015761$ que representan el total de elementos de la matriz. Desde el punto de vista de la memoria utilizada en el almacenamiento

² $u(x_0) = u(x_{n+1}) = 0$ no son incógnitas del sistema, por ello no están en la ecuación.

³Recordar que Δ representa al Laplaciano de u . En dimensión 3 por ejemplo se escribe $\Delta u = u_{xx} + u_{yy} + u_{zz}$.

⁴En inglés sparse: un porcentaje significativo de elementos de la matriz es nulo.

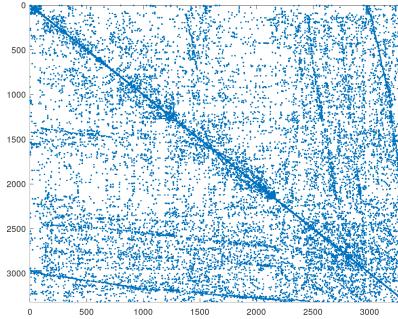


FIGURA 2. Estructura de la matriz A : se observa que es rala y simétrica.

de \mathbf{A} esto es significativo. Teniendo en cuenta que en doble precisión cada número ocupa 8 bytes tendríamos en cada caso:

- $39753 \frac{8}{1024} \sim 300K$
- $11015761 \frac{8}{(1024)^2} \sim 84M$.

Esta ventaja se puede perder al hacer LU . En efecto, en este ejemplo, a pesar de que A tiene solo 39753 elementos no nulos, el número de elementos no nulos de L es 2440542 ($\sim 18M$). Este fenómeno se conoce como *rellenado*. La apariencia de L y su comparación con la matriz original puede verse en la Figura 7.1

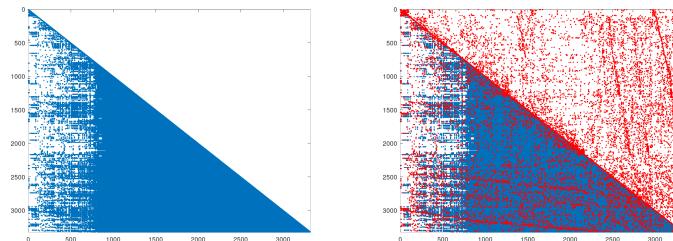


FIGURA 3. Estructura de las matrices A y L en el ejemplo (E). Izquierda, matriz L de la factorización $A = LU$ donde se nota el relleno. Derecha, superposición de A y L .

Si bien hay heurísticas para evitar el relleno, buscando permutar las variables y las ecuaciones no las vamos a comentar en este texto. Vamos a concentrarnos en los métodos iterativos. Estos no buscan ninguna factorización y se basan en iterar un procedimiento de orden $\mathcal{O}(N^2)$ (básicamente el producto de una matriz por vector) con la esperanza de conseguir una buena aproximación de la solución buscada en unas pocas⁵ iteraciones.

Ya hemos visto que $\rho(\mathbf{A})$ no es una norma, sin embargo verifica:

$$\rho(\mathbf{A}) = \inf_{\|\cdot\|} \|\mathbf{A}\|.$$

⁵Pocas al menos relativo al tamaño N de la matriz.

Lo que implica que para todo $0 < \varepsilon$ existe una norma $\|\cdot\|$ tal que

$$\|\mathbf{A}\| \leq \rho(\mathbf{A}) + \varepsilon$$

Usaremos esta propiedad en la demostración del siguiente resultado que da la clave para predecir el orden de convergencia de los métodos iterativos.

PROPOSICIÓN 7.1.1. Para toda norma subordinada $\|\cdot\|$ vale que

$$\rho(\mathbf{M}) = \lim_{k \rightarrow \infty} \|\mathbf{M}^k\|^{1/k}.$$

DEMOSTRACIÓN. Dado $\epsilon > 0$ existe una norma subordinada $\|\cdot\|_*$ tal que

$$\|\mathbf{M}\|_* \leq \rho(\mathbf{M}) + \epsilon.$$

$$\rho(\mathbf{M})^k = \rho(\mathbf{M}^k) \leq \|\mathbf{M}^k\| \leq C\|\mathbf{M}^k\|_* \leq C\|\mathbf{M}\|_*^k \leq C(\rho(\mathbf{M}) + \epsilon)^k,$$

$$\rho(\mathbf{M}) \leq \|\mathbf{M}^k\|^{1/k} \leq C^{1/k}(\rho(\mathbf{M}) + \epsilon)$$

tomando límite y notando que ϵ es arbitrario se demuestra el resultado. \square

7.2. Métodos Iterativos

Supongamos que queremos resolver

$$\mathbf{Ax} = \mathbf{b},$$

pero la matriz \mathbf{A} es difícil de invertir. Entonces proponemos una descomposición

$$\mathbf{A} = \mathbf{B} + \mathbf{C},$$

donde \mathbf{B} es elegida de modo que sea *fácil* de invertir y escribimos

$$\mathbf{Bx} = -\mathbf{Cx} + \mathbf{b},$$

o equivalentemente

$$\mathbf{x} = -\mathbf{B}^{-1}\mathbf{Cx} + \mathbf{B}^{-1}\mathbf{b}.$$

Notemos que llamando $\mathbf{M}_I = -\mathbf{B}^{-1}\mathbf{C}$ (denominada *matriz de iteraciones*) y $\mathbf{B}^{-1}\mathbf{b} = \tilde{\mathbf{b}}$, resolver el sistema equivale a hallar \mathbf{x} tal que

$$\mathbf{x} = \mathbf{M}_I\mathbf{x} + \tilde{\mathbf{b}}.$$

La idea es tomar un vector inicial \mathbf{x}_0 e iterar

$$\mathbf{x}_{n+1} = \mathbf{M}_I\mathbf{x}_n + \tilde{\mathbf{b}},$$

con la esperanza de que $\mathbf{x}_k \rightarrow \mathbf{x}$. Para ver si nuestra idea puede funcionar estudiamos el error $\mathbf{e}_k = \mathbf{x} - \mathbf{x}_k$ restando miembro a miembro las dos ecuaciones de arriba

$$\mathbf{e}_{k+1} = \mathbf{M}_I\mathbf{e}_k = \mathbf{M}_I\mathbf{M}_I\mathbf{e}_{k-1} = \cdots = \mathbf{M}_I^{k+1}\mathbf{e}_0$$

de donde vemos que $\mathbf{e}_k \rightarrow \mathbf{0}$ para *todo* error inicial \mathbf{e}_0 sí y solo sí $\mathbf{M}_I^k \rightarrow \mathbf{0}$ lo cual ocurre sí y solo sí $\rho(\mathbf{M}_I) < 1$.

Más aún

$$\|\mathbf{e}_k\| \leq C\rho(\mathbf{M}_I)^k$$

lo que nos da la velocidad de convergencia del algoritmo.

Comencemos estudiando un caso particular llamado el método de Richardson. Elegimos \mathbf{B} como un múltiplo de la identidad

$$\mathbf{B} = \omega^{-1} \mathbf{I}, \quad \omega > 0.$$

Luego, se tiene

$$\mathbf{C} = \mathbf{A} - \omega^{-1} \mathbf{I} = \omega^{-1}(\omega \mathbf{A} - \mathbf{I})$$

y la matriz de iteraciones resulta

$$\mathbf{M}_I = -\mathbf{B}^{-1}\mathbf{C} = \mathbf{I} - \omega\mathbf{A}$$

PROPOSICIÓN 7.2.1. Si \mathbf{A} es SDP, el método de Richardson converge -para todo dato inicial- sí y solo sí $\omega < \frac{2}{\rho(\mathbf{A})}$.

DEMOSTRACIÓN. Basta ver bajo que condiciones $\rho(\mathbf{M}_I) < 1$, con $\mathbf{M}_I = \mathbf{I} - \omega\mathbf{A}$. Como \mathbf{A} es SDP sus autovalores cumplen $0 < \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$. Los autovalores de \mathbf{M}_I son entonces $1 - \omega\lambda_i$. Queremos ver bajo que condiciones $|1 - \omega\lambda_i| < 1$, para todo $1 \leq i \leq n$. Equivalentemente,

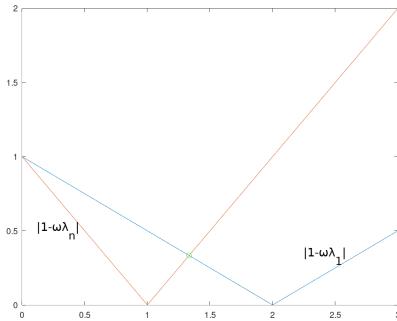
$$-1 < 1 - \omega\lambda_i < 1$$

i.e. es necesario y suficiente que $\omega < \frac{2}{\lambda_i}$ para todo $1 \leq i \leq n$ lo que equivale a $\omega < \frac{2}{\rho(\mathbf{A})}$. \square

Claramente, al variar ω en el rango $(0, \frac{2}{\rho(\mathbf{A})})$ obtendremos distintas velocidades de convergencia para nuestro método. Podemos intentar localizar el óptimo ω_{opt} , es decir el que garantiza orden máximo de convergencia (equivalentemente, el ω_{opt} para el cual $\rho(\mathbf{M}_I)$ es mínimo). Para ello observemos que (ver Figura 7.2) como

$$\rho(\mathbf{M}_I) = \max\{|1 - \omega\lambda_n|, |1 - \omega\lambda_1|\}$$

ω_{opt} ocurre en el punto de cruce de los gráficos. Ese punto puede despejarse



$$\omega_{opt} = \frac{2}{\lambda_n + \lambda_1}$$

de donde el respectivo radiopectral óptimo es

$$\rho(\mathbf{M}_I) = \frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1} = \frac{\kappa(\mathbf{A}) - 1}{\kappa(\mathbf{A}) + 1}.$$

Notemos que la velocidad de convergencia del método se deteriora si $\kappa(\mathbf{A})$ es grande.

Para el ejemplo (E) de la distribución de temperatura, dado al principio del capítulo, tenemos (aproximados por el método de la potencia)

$$\lambda_1 \sim 0.0009 \quad \lambda_n \sim 1.93$$

$$\kappa(\mathbf{A}) \sim 2144$$

$$\omega_{opt} \sim 1.04$$

lo que da

$$\rho(\mathbf{M}_I) \sim 0.99907$$

si queremos reducir el error inicial en un orden de 10^{-3} precisamos iterar aproximadamente 60000 veces, número mucho mayor que el tamaño $n \sim 3000$ de la matriz. Como vemos no resulta muy competitivo en este caso.

7.3. Algunos métodos clásicos

Si tomamos $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$ (triangular inferior+diagonal+triangular superior), podemos ensayar varias descomposiciones $\mathbf{A} = \mathbf{B} + \mathbf{C}$:

1. $\mathbf{B} = \mathbf{D}$ método de Jacobi
2. $\mathbf{B} = \mathbf{L} + \mathbf{D}$ método de Gauss-Seidel

Las matrices de iteración son

1. Jacobi $\mathbf{M}_J = -\mathbf{D}^{-1}(\mathbf{L} + \mathbf{U})$
2. Gauss-Seidel $\mathbf{M}_{GS} = -(\mathbf{L} + \mathbf{D})^{-1}\mathbf{U}$

Si bien en el caso GS, contrariamente al caso J, la matriz no parece obviamente invertible, originalmente fueron concebidos del modo siguiente que no requiere inversiones explícitas: de

$$\mathbf{Ax} = \mathbf{b},$$

se tiene, para todo $1 \leq i \leq n$

$$\sum_{j=1}^n a_{i,j}x_j = b_i,$$

en Jacobi se escribe (asumiendo $a_{i,i} \neq 0$)

$$x_i = \left(- \sum_{j=1, j \neq i}^n a_{i,j}x_j + b_i \right) / a_{i,i} \quad \forall 1 \leq i \leq n$$

que da lugar al método

$$x_i^{(k+1)} = \left(- \sum_{j=1, j \neq i}^n a_{i,j}x_j^{(k)} + b_i \right) / a_{i,i} \quad \forall 1 \leq i \leq n$$

para pasar del iterado k al $k + 1$.

Para Gauss-Seidel se escribe

$$x_i^{(k+1)} = \left(- \sum_{j=i+1}^n a_{ij} x_j^{(k)} - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} + b_i \right) / a_{ii} \quad \forall 1 \leq i \leq n$$

para pasar del iterado k al $k+1$, con la idea de utilizar $x_1^{(k+1)}, \dots, x_{i-1}^{(k+1)}$ (considerados *mejores* aproximaciones) en el cálculo de $x_i^{(k+1)}$.

Notar que:

- No es necesario construir explícitamente M_J ni M_{GS} .
- Jacobi puede “paralelizarse” pero no así Gauss-Seidel.

PROPOSICIÓN 7.3.1. Sea $\mathbf{A} \in \mathbb{K}^{n \times n}$ estrictamente diagonal dominante, entonces Jacobi y Gauss-Seidel convergen.

DEMOSTRACIÓN. 1. Jacobi: sea $\mathbf{v} \in \mathbb{K}^n$ autovector de \mathbf{M}_J de autovalor λ q.v.q. $|\lambda| < 1$.

$$-D^{-1}(L+U)\mathbf{v} = \lambda\mathbf{v},$$

sea $1 \leq i \leq n$ tal que $0 \neq |v_i| \geq |v_k|$ para todo $1 \leq k \leq n$. Tenemos

$$-\lambda v_i a_{ii} = \sum_{i \neq j=1}^n a_{ij} v_j$$

, como $v_i \neq 0 \neq a_{ii}$

$$-\lambda = \sum_{i \neq j=1}^n \frac{a_{ij} v_j}{a_{ii} v_i}$$

tomando modulo

$$|\lambda| \leq \sum_{i \neq j=1}^n \left| \frac{a_{ij}}{a_{ii}} \right| < 1$$

pues \mathbf{A} es EDD.

2. Gauss-Seidel: como antes sea \mathbf{v} autovector de autovalor λ de \mathbf{M}_{GS} , y $0 \neq |v_i| \geq |v_k|$ con $1 \leq k \leq n$:

$$\begin{aligned} \mathbf{U}\mathbf{v} &= -\lambda(\mathbf{D} + \mathbf{L})\mathbf{v} \\ \sum_{j=i+1}^n a_{ij} \mathbf{v}_j &= -\lambda \left(\sum_{j=1}^i a_{ij} \mathbf{v}_j \right) \end{aligned}$$

dividiendo por v_i y tomando módulos

$$\sum_{j=i+1}^n |a_{ij}| \geq \left| \sum_{j=i+1}^n a_{ij} \frac{v_j}{v_i} \right| = |\lambda| \left| \left(\sum_{j=1}^i a_{ij} \frac{v_j}{v_i} \right) \right| \geq |\lambda| \left(|a_{11}| - \sum_{j=1}^{i-1} |a_{ij}| \right)$$

de donde

$$|\lambda| \leq \frac{\sum_{j=i+1}^n |a_{ij}|}{|a_{11}| - \sum_{j=1}^{i-1} |a_{ij}|} < 1$$

por ser estrictamente diagonal dominante. \square

Si trabajamos una matriz *tridiagonal*, como por ejemplo el de la ecuación (7.5), podemos probar fácilmente la propiedad siguiente.

PROPOSICIÓN 7.3.2. Sea $\mathbf{A} \in \mathbb{K}^{n \times n}$ tridiagonal (i.e. $|a_{i,j}| = 0$ si $|j - i| > 1$) con $a_{ii} \neq 0$ para todo $1 \leq i \leq n$. Entonces $\rho(\mathbf{M}_{GS}) = \rho(\mathbf{M}_J)^2$.

DEMOSTRACIÓN. Escribimos $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$. Sea $0 \neq \lambda$ autovalor de la matriz $\mathbf{M}_J = -\mathbf{D}^{-1}(\mathbf{L} + \mathbf{U})$. Queremos ver que λ^2 es autovalor de $\mathbf{M}_{GS} = -(\mathbf{D} + \mathbf{L})^{-1}\mathbf{U}$. En particular

$$\det(-\mathbf{D}^{-1}(\mathbf{L} + \mathbf{U}) - \lambda\mathbf{I}) = 0$$

por lo tanto (sacando $-\mathbf{D}^{-1}$ de factor común y usando propiedades del determinante)

$$\det(\mathbf{L} + \mathbf{U} + \lambda\mathbf{D}) = 0$$

donde la matriz $\mathbf{M} = \mathbf{L} + \mathbf{U} + \lambda\mathbf{D}$ es *tridiagonal*. En particular, tomando un $0 \neq \alpha \in \mathbb{K}$ cualquiera tenemos que

$$\det(\mathbf{M}) = \det \left(\begin{pmatrix} \alpha & 0 & 0 & \cdots & 0 \\ 0 & \alpha^2 & 0 & \cdots & 0 \\ 0 & 0 & \alpha^3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & \cdots & \alpha^n \end{pmatrix} \mathbf{M} \begin{pmatrix} \alpha^{-1} & 0 & 0 & \cdots & 0 \\ 0 & \alpha^{-2} & 0 & \cdots & 0 \\ 0 & 0 & \alpha^{-3} & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & \cdots & \alpha^{-n} \end{pmatrix} \right),$$

propiedad que vale para cualquier matriz \mathbf{M} pero en este caso particular, *por ser tridiagonal*, un cálculo directo muestra que

$$\begin{pmatrix} \alpha & 0 & 0 & \cdots & 0 \\ 0 & \alpha^2 & 0 & \cdots & 0 \\ 0 & 0 & \alpha^3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & \cdots & \alpha^n \end{pmatrix} \mathbf{M} \begin{pmatrix} \alpha^{-1} & 0 & 0 & \cdots & 0 \\ 0 & \alpha^{-2} & 0 & \cdots & 0 \\ 0 & 0 & \alpha^{-3} & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & \cdots & \alpha^{-n} \end{pmatrix} = \alpha^{-1}\mathbf{L} + \lambda\mathbf{D} + \alpha\mathbf{U}$$

por lo que tomando $\alpha^{-1} = \lambda$ tenemos que

$$0 = \det(\mathbf{L} + \mathbf{U} + \lambda\mathbf{D}) = (-\lambda)^n \det(-\mathbf{L} - \mathbf{D} - \alpha^2\mathbf{U}) = (-\lambda)^n \det(-(\mathbf{L} + \mathbf{D})^{-1}\mathbf{U} - \lambda^2\mathbf{I}),$$

es decir λ^2 es autovalor de \mathbf{M}_{GS} . Queda como ejercicio seguir los pasos en sentido inverso y ver que si $\lambda^2 \neq 0$ es autovalor de \mathbf{M}_{GS} entonces λ lo es de \mathbf{M}_J . Los autovalores nulos no son de interés, porque no modifican el radio espectral. \square

PROPOSICIÓN 7.3.3. Si $\mathbf{A} \in \mathbb{K}^{n \times n}$ Hermitiana y definida positiva, entonces Gauss-Seidel converge.

DEMOSTRACIÓN. Sea \mathbf{v} autovector de autovalor λ de la matriz de iteraciones:

$$\begin{aligned} -(\mathbf{D} + \mathbf{L})^{-1}\mathbf{U}\mathbf{v} &= \lambda\mathbf{v}, \\ -\mathbf{U}\mathbf{v} &= \lambda(\mathbf{D} + \mathbf{L})\mathbf{v} = \lambda\mathbf{A}\mathbf{v} - \lambda\mathbf{U} \\ (\lambda - 1)\mathbf{U}\mathbf{v} &= \lambda\mathbf{A}\mathbf{v} \\ (\lambda - 1)\mathbf{v}^*\mathbf{U}\mathbf{v} &= \lambda\mathbf{v}^*\mathbf{A}\mathbf{v} \end{aligned}$$

$$\frac{\lambda}{\lambda - 1} = \frac{\mathbf{v}^* \mathbf{U} \mathbf{v}}{\mathbf{v}^* \mathbf{A} \mathbf{v}}$$

como $\lambda \neq 1$ y \mathbf{A} es Hermitiana (simétrica, en el caso real) $\mathbf{U} = \mathbf{L}^*$,

$$2Re\left(\frac{\lambda}{\lambda - 1}\right) = \frac{\mathbf{v}^* (\mathbf{U} + \mathbf{L}) \mathbf{v}}{\mathbf{v}^* \mathbf{A} \mathbf{v}} = \frac{\mathbf{v}^* \mathbf{U} \mathbf{v}}{\mathbf{v}^* \mathbf{A} \mathbf{v}} = 1 - \frac{\mathbf{v}^* \mathbf{D} \mathbf{v}}{\mathbf{v}^* \mathbf{A} \mathbf{v}} < 1.$$

Llamando $\lambda = a + ib$ resulta

$$2 \frac{(a(a-1) + b^2)}{(a-1)^2 + b^2} < 1,$$

y así $|\lambda|^2 = a^2 + b^2 < 1$. □

Volviendo a nuestro ejemplo (E) de distribución de temperatura, la matriz \mathbf{A} es simétrica y definida positiva (no estamos probando esta última afirmación). Por la proposición anterior sabemos que GS converge. Por otro lado no es EDD sino solamente DD, tampoco es tridiagonal, sin embargo cumple con otras hipótesis (que no veremos en este curso) que garantizan que

$$\rho(\mathbf{M}_{GS}) = \rho(\mathbf{M}_J)^2,$$

por lo cual Jacobi y Gauss-Seidel convergen. En este ejemplo (E), usando el método de la potencia, obtenemos $\rho(M_J) \sim 0.998$ y $\rho(\mathbf{M}_{GS}) \sim 0.996$ que satisface la relación predicha. Este resultado no vale siempre pero ocurre en muchas matrices asociadas a estas ecuaciones diferenciales. Como vemos, precisaríamos unas 3500 iteraciones para que en J se reduzca el error en un orden 10^{-3} y unas 1750 en GS para el mismo nivel de reducción del error.

Observemos que todos los métodos estudiados se originaron a partir de la identidad

$$\mathbf{x} = -\mathbf{B}^{-1} \mathbf{C} \mathbf{x} + \mathbf{B}^{-1} \mathbf{b},$$

que podemos reescribir

$$\mathbf{x} = \mathbf{x} - \mathbf{B}^{-1} (\mathbf{A} \mathbf{x} - \mathbf{b}),$$

y entonces pensar en las iteraciones del siguiente modo

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{B}^{-1} (\mathbf{A} \mathbf{x}_k - \mathbf{b}),$$

que pueden verse como una corrección que involucra al *residuo* $\mathbf{r}_k = \mathbf{A} \mathbf{x}_k - \mathbf{b}$. Con esta idea podemos modificar el método del modo siguiente: tomamos un $0 < \omega$ y escribimos una nueva variante

$$\mathbf{x}_{j+1} = (1 - \omega) \mathbf{x}_j + \omega (\mathbf{x}_j - \mathbf{B}^{-1} (\mathbf{A} \mathbf{x}_j - \mathbf{b})),$$

dando mas o menos importancia a la iteración original. Si $0 < \omega < 1$ estamos *atenuado* y si $\omega > 1$ estamos *amplificado* la corrección. Si aplicamos estas ideas a Gauss-Seidel obtenemos el denominado método SOR⁶. Para estudiarlo un poco reescribamos la iteración adecuadamente: para Gauss-Seidel (GS) tenemos que $B = L + D$ y $C = U$, en particular podemos escribir en la coordenada i – *esima* como:

$$\mathbf{x}_i^{k+1} = \frac{1}{a_{i,i}} \left(- \sum_{j=1}^{i-1} a_{i,j} \mathbf{x}_j^{k+1} - \sum_{j=i+1}^n a_{i,j} \mathbf{x}_j^k + \mathbf{b}_i \right),$$

⁶Sobre Relajación Sucesiva.

por lo tanto, el método SOR en la coordenada $i - esima$, toma la forma

$$\mathbf{x}_i^{k+1} = (1 - \omega)\mathbf{x}_i^k + \frac{\omega}{a_{i,i}} \left(-\sum_{j=1}^{i-1} a_{i,j}\mathbf{x}_j^{k+1} - \sum_{j=i+1}^n a_{i,j}\mathbf{x}_j^k + \mathbf{b}_i \right),$$

o sea

$$a_{i,i}\mathbf{x}_i^{k+1} + \omega \sum_{j=1}^{i-1} a_{i,j}\mathbf{x}_j^{k+1} = (1 - \omega)a_{i,i}\mathbf{x}_i^k - \omega \sum_{j=i+1}^n a_{i,j}\mathbf{x}_j^k - \omega\mathbf{b}_i.$$

En términos matriciales

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

en particular, y como es de esperar, $\omega = 1$ recupera a GS. El método SOR tiene una matriz de iteraciones que puede escribirse

$$\mathbf{M}_\omega = (\mathbf{D} + \omega\mathbf{L})^{-1}((1 - \omega)\mathbf{D} - \omega\mathbf{U}).$$

No vamos a estudiar con detalle la convergencia para este método, pero observemos que como \mathbf{M}_ω es producto de matrices triangulares, es fácil ver que

$$\det(\mathbf{M}_\omega) = (1 - \omega)^n,$$

lo que dice que $\rho(\mathbf{M}_\omega) \geq |1 - \omega|$ en particular y vemos que una condición necesaria para la convergencia del método es que $0 < \omega < 2$.

El hecho de que la condición $0 < \omega < 2$ sea *necesaria* para la convergencia de SOR es resultado general (vale para matrices arbitrarias). Para las matrices que son como las del ejemplo (E) de distribución de temperatura se puede probar que el método converge sí y solo sí $0 < \omega < 2$. Además, trabajando numéricamente^a con la matriz podemos ver como se comporta $\rho(\mathbf{M}_\omega)$ en términos del ω (ver Figura 7.3). En nuestro ejemplo: tenemos $\omega_{opt} \sim 1.904$,

$$\rho(\mathbf{M}_{\omega_{opt}}) \sim 0.9$$

con lo cual podemos lograr una reducción del error en 10^{-4} con solo 150 iteraciones!.

^aEn realidad hay una teoría completa que predice el comportamiento del método SOR para estas matrices pero no veremos esos detalles aquí.

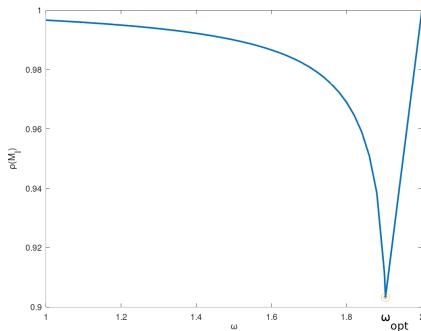


FIGURA 4. Comportamiento de $\rho(\mathbf{M}_\omega)$. Se ve que tiene un mínimo cerca de $\omega = 1.904$ que se corresponde con un radio espectral cercano a 0.9.

7.4. Método de gradiente (o del descenso más rápido)

Cuando trabajamos con matrices simétricas definidas positivas es posible reemplazar el problema:

$$(P1) \text{ Hallar } \mathbf{x} \in \mathbb{R}^n \text{ tal que } \mathbf{Ax} = \mathbf{b}$$

por un problema de minimización asociado a la función $J : \mathbb{R}^n \rightarrow \mathbb{R}$, $J(\mathbf{z}) = \frac{1}{2}\mathbf{z}^T \mathbf{Az} - \mathbf{z}^T \mathbf{b}$.

$$(P2) \text{ Hallar } \mathbf{x} \in \mathbb{R}^n \text{ tal que } J(\mathbf{x}) = \min_{\mathbf{y} \in \mathbb{R}^n} J(\mathbf{y}).$$

En efecto, calculemos para $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{z} \in \mathbb{R}^n$, $\mathbf{z} \neq \mathbf{0}$ y $t \in \mathbb{R}$ arbitrarios, la expresión

$$h(t) = J(\mathbf{x} + t\mathbf{z}) = \frac{1}{2}(\mathbf{x} + t\mathbf{z})^T \mathbf{A}(\mathbf{x} + t\mathbf{z}) - (\mathbf{x} + t\mathbf{z})^T \mathbf{b},$$

desarrollando los términos resulta

$$(7.6) \quad h(t) = \frac{1}{2}t^2 \mathbf{z}^T \mathbf{Az} + t(\mathbf{z}^T \mathbf{Ax} - \mathbf{z}^T \mathbf{b}) + \frac{1}{2}\mathbf{x}^T \mathbf{Ax} - \mathbf{x}^T \mathbf{b}$$

una cuadrática en la variable t , con mínimo en

$$(7.7) \quad t_m = \frac{\mathbf{z}^T \mathbf{Ax} - \mathbf{z}^T \mathbf{b}}{\mathbf{z}^T \mathbf{Az}} = \frac{\mathbf{z}^T(\mathbf{Ax} - \mathbf{b})}{\mathbf{z}^T \mathbf{Az}}.$$

Supongamos que \mathbf{x} resuelve (P1). En ese caso debe ser $t_m = 0$ y entonces

$$J(\mathbf{x}) = h(0) \leq h(t) = J(\mathbf{x} + t\mathbf{z})$$

para todo t y vemos que

$$J(\mathbf{x}) = \min_{\mathbf{y} \in \mathbb{R}^n} J(\mathbf{y}),$$

lo que dice que \mathbf{x} resuelve (P2). Recíprocamente: si \mathbf{x} es tal que

$$J(\mathbf{x}) = \min_{\mathbf{y} \in \mathbb{R}^n} J(\mathbf{y}),$$

repetimos la cuenta anterior y vemos que el mínimo de $h(t)$ está en $t = 0$ lo que dice que

$$0 = \frac{\mathbf{z}^T \mathbf{Ax} - \mathbf{z}^T \mathbf{b}}{\mathbf{z}^T \mathbf{Az}}$$

es decir

$$\mathbf{z}^T(\mathbf{Ax} - \mathbf{b}) = 0,$$

para todo $\mathbf{z} \neq \mathbf{0}$, lo que indica que ⁷

$$\mathbf{Ax} - \mathbf{b} = \mathbf{0},$$

es decir

$$\mathbf{Ax} = \mathbf{b}.$$

y entonces \mathbf{x} resuelve (P1).

⁷Tomando, por ejemplo, $\mathbf{z} = \mathbf{Ax} - \mathbf{b}$.

Ejemplo en \mathbb{R}^2 . Consideremos la matriz $M = \begin{pmatrix} 5 & 2 \\ 2 & 4 \end{pmatrix}$ y el vector $b = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$. El sistema asociado

$$Mx = b,$$

tiene por solución $x = \begin{pmatrix} 1/4 \\ -1/8 \end{pmatrix}$. Por otro lado,

$$J(x_1, x_2) = 5/2x_1^2 + 2x_2^2 + 2x_1x_2 - x_1,$$

y en la Figura 7.4 vemos las curvas de nivel de J y el punto x (en rojo) solución del sistema.

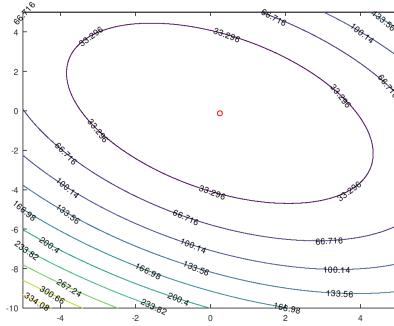


FIGURA 5. Curvas de nivel de $J(x_1, x_2) = 5/2x_1^2 + 2x_2^2 + 2x_1x_2 - x_1$.

Antes de continuar observemos que tomando en (7.6), $z = e_i$ (el canónico i) se tiene que

$$\frac{\partial J(\mathbf{x})}{\partial x_i} = \lim_{t \rightarrow 0} \frac{J(\mathbf{x} + t\mathbf{e}_i) - J(\mathbf{x})}{t} = [\mathbf{A}\mathbf{x} - \mathbf{b}]_i,$$

donde el corchete representa la coordenada i del vector $\mathbf{A}\mathbf{x} - \mathbf{b}$. En particular tenemos

$$\nabla J(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}.$$

Lo que hemos visto indica que para resolver un sistema con una matriz SDP podemos alternativamente atacar un problema de minimización en \mathbb{R}^n . Para comenzar simplificando este nuevo problema vamos a intentar llevarlo a uno que sea iterativo en \mathbb{R} . Este es un método de búsqueda por líneas: dado \mathbf{x}_i elegimos una dirección de búsqueda \mathbf{z}_i y un escalar λ_i tal que

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \lambda_i \mathbf{z}_i.$$

En un caso general quisieramos λ_i tal que

$$J(\mathbf{x}_i + \lambda_i \mathbf{z}_i) = \min_{t \in \mathbb{R}} J(\mathbf{x}_i + t\mathbf{z}_i),$$

lo que nos llevaría a resolver

$$0 = \frac{dJ(\mathbf{x}_i + t\mathbf{z}_i)}{dt} \Big|_{t=\lambda_i} = \nabla J(\mathbf{x}_i + \lambda_i \mathbf{z}_i)^T \mathbf{z}_i.$$

En el caso que nos interesa ya sabemos despejar λ_i (ver (7.7)):

$$\lambda_i = -\frac{\mathbf{z}_i^T(\mathbf{A}\mathbf{x}_i - \mathbf{b})}{\mathbf{z}_i^T \mathbf{A} \mathbf{z}_i} = \frac{\mathbf{z}_i^T \mathbf{r}_i}{\mathbf{z}_i^T \mathbf{A} \mathbf{z}_i}$$

donde $\mathbf{r}_i = \mathbf{b} - \mathbf{A}\mathbf{x}_i$ es el residuo.

Este método no nos sugiere como elegir el vector de búsqueda \mathbf{z}_i en cada paso. Una forma natural es utilizar la dirección de mayor decrecimiento de la función J en el punto $\mathbf{x}_i : -\nabla J(\mathbf{x}_i) = \mathbf{b} - \mathbf{A}\mathbf{x}_i = \mathbf{r}_i$. Esto da lugar al método del gradiente o del descenso más rápido:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \frac{\mathbf{r}_i^T \mathbf{r}_i}{\mathbf{r}_i^T \mathbf{A} \mathbf{r}_i} \mathbf{r}_i.$$

El proceso anterior puede converger muy lentamente. En cada paso minimizamos (de forma exacta en aritmética exacta) a lo largo de la recta $\mathbf{x}_i + \lambda \mathbf{z}_i$.

Una mejora sustancial sería comenzar en un x_0 , minimizar a lo largo de una recta, luego a lo largo de un plano, e ir aumentando la dimensión. Por ejemplo podríamos intentar minimizar el paso i en

$$J(\mathbf{x}_{i+1}) = \min_{\mathbf{y} \in \langle \mathbf{z}_0, \dots, \mathbf{z}_i \rangle + \mathbf{x}_0} J(\mathbf{y})$$

que claramente daría un algoritmo que finaliza⁸ en a lo sumo n pasos. Obviamente, vemos que el problema que aparece es la dificultad de minimizar en dimensiones mayores a uno. Hay, sin embargo, un modo de lograr nuestro propósito minimizando sobre líneas, si es que elegimos las direcciones de búsqueda de modo adecuado. Este algoritmo lo estudiamos en la siguiente sección.

7.5. Gradiente conjugado

Observemos, antes de continuar, que una matriz simétrica (o Hermitiana) definida positiva define un producto interno a través de la operación⁹

$$\mathbf{x}^T \mathbf{A} \mathbf{y}.$$

En particular este producto posee la siguiente norma asociada

$$\|\mathbf{x}\|_A = \sqrt{\mathbf{x}^T \mathbf{A} \mathbf{x}}.$$

Teniendo esta norma definida probamos lo siguiente.

LEMA 7.5.1. Supongamos que las direcciones de búsqueda son \mathbf{A} ortogonales (i.e. $\mathbf{z}_i^T \mathbf{A} \mathbf{z}_j = 0$) y en cada paso λ_i se elige para minimizar sobre la línea correspondiente. Entonces

$$\min_{\mathbf{x}_i + \langle \mathbf{z}_i \rangle} J(\mathbf{y}) = J(\mathbf{x}_{i+1}) = \min_{\mathbf{y} \in \mathbf{x}_0 + \langle \mathbf{z}_0, \mathbf{z}_1, \dots, \mathbf{z}_i \rangle} J(\mathbf{y})$$

DEMOSTRACIÓN. Sea $W_{i+1} = \langle \mathbf{z}_0, \mathbf{z}_1, \dots, \mathbf{z}_i \rangle$ y tomo $\mathbf{z} \in \mathbf{x}_0 + W_{i+1}$ arbitrario. Por construcción, se puede escribir

$$\mathbf{z} = \mathbf{y} + t \mathbf{z}_i$$

con $\mathbf{y} \in \mathbf{x}_0 + W_i = \mathbf{x}_0 + \langle \mathbf{z}_0, \mathbf{z}_1, \dots, \mathbf{z}_{i-1} \rangle$.

⁸Al menos en aritmética exacta.

⁹Dejamos como ejercicio esta afirmación.

Observamos que

$$\min_{\mathbf{z} \in \mathbf{x}_0 + W_{i+1}} J(\mathbf{z}) = \min_{\mathbf{y} \in \mathbf{x}_0 + W_i, t \in \mathbb{R}} J(\mathbf{y} + t\mathbf{z}_i)$$

Escribimos

$$J(z) = J(\mathbf{y} + t\mathbf{z}_i) = \frac{1}{2}t^2\mathbf{z}_i^T \mathbf{A}\mathbf{z}_i + t(\mathbf{z}_i^T \mathbf{A}\mathbf{y} - \mathbf{z}_i^T \mathbf{b}) + \frac{1}{2}\mathbf{y}^T \mathbf{A}\mathbf{y} - \mathbf{y}^T \mathbf{b}$$

pero, usando que $\mathbf{z}_i^T \mathbf{A}\mathbf{z}_j = 0$

$$\mathbf{z}_i^T \mathbf{A}\mathbf{y} = \mathbf{z}_i^T \mathbf{A}\mathbf{x}_0 = \mathbf{z}_i^T \mathbf{A}\mathbf{x}_i$$

$$J(\mathbf{y} + t\mathbf{z}_i) = \left(\frac{1}{2}t^2\mathbf{z}_i^T \mathbf{A}\mathbf{z}_i + t(\mathbf{z}_i^T \mathbf{A}\mathbf{x}_i - \mathbf{z}_i^T \mathbf{b}) \right) + \frac{1}{2}\mathbf{y}^T \mathbf{A}\mathbf{y} - \mathbf{y}^T \mathbf{b}$$

de donde resulta que para minimizar en \mathbf{z} basta minimizar ambos términos: el primero en $t \in \mathbb{R}$ el segundo en $\mathbf{y} \in W_i$. Pero el t óptimo es justamente $\lambda = \frac{\mathbf{z}_i^T \mathbf{A}\mathbf{r}_i}{\mathbf{z}_i^T \mathbf{A}\mathbf{z}_i}$ provisto por nuestro método. En definitiva el resultado sale por inducción. \square

Las direcciones \mathbf{A} ortogonales suelen llamarse conjugadas lo que da nombre al algoritmo que veremos mas adelante.

Queremos un método que minimice sobre direcciones conjugadas (y no sea costoso)

Idea: A-ortogonalizar los residuos

- $\mathbf{z}_0 = \mathbf{r}_0$
- $\mathbf{z}_i = \mathbf{r}_i - \sum_{j=0}^{i-1} \frac{\mathbf{z}_j^T \mathbf{A}\mathbf{r}_i}{\mathbf{z}_j^T \mathbf{A}\mathbf{z}_j} \mathbf{z}_j$

Vale lo siguiente

- $\langle \mathbf{z}_0, \mathbf{z}_1 \dots, \mathbf{z}_i \rangle = \langle \mathbf{r}_0, \mathbf{r}_1 \dots, \mathbf{r}_i \rangle$
- $\mathbf{r}_i^T \mathbf{r}_j = 0$, pues $J(\mathbf{x}_i + t\mathbf{r}_j)$ tiene un mínimo en $t = 0$, si $0 \leq j < i$. Luego derivando y evaluando en $t = 0$

$$0 = \mathbf{r}_j^T (\mathbf{A}\mathbf{x}_i - \mathbf{b}) = \mathbf{r}_j^T \mathbf{r}_i.$$

- Existe $m \leq n$ tal que

$$\mathbf{x}_0 \neq \mathbf{x}_1 \neq \dots \neq \mathbf{x}_m = \mathbf{x}$$

$$W_0 \subsetneq W_1 \subsetneq \dots \subsetneq W_m$$

- Para todo $1 \leq i \leq m$, $\{\mathbf{z}_0, \mathbf{z}_1 \dots, \mathbf{z}_{i-1}\}$ (resp. $\{\mathbf{r}_0, \mathbf{r}_1 \dots, \mathbf{r}_{i-1}\}$) es una base A ortogonal (resp. ortogonal) de W_i .

- $\mathbf{z}_i^T \mathbf{r}_i = \mathbf{r}_i^T \mathbf{r}_i$. Pues $\mathbf{z}_i = \mathbf{r}_i - \sum_{j=0}^{i-1} \frac{\mathbf{z}_j^T \mathbf{A}\mathbf{r}_i}{\mathbf{z}_j^T \mathbf{A}\mathbf{z}_j} \mathbf{z}_j$ y entonces

$$\mathbf{r}_i^T \mathbf{z}_i = \mathbf{r}_i^T \mathbf{r}_i - \sum_{j=0}^{i-1} \frac{\mathbf{z}_j^T \mathbf{A}\mathbf{r}_i}{\mathbf{z}_j^T \mathbf{A}\mathbf{z}_j} \mathbf{r}_i^T \mathbf{z}_j = \mathbf{r}_i^T \mathbf{r}_i$$

pues para cada $j \leq i-1$, $\mathbf{z}_j \in W_i = \langle \mathbf{r}_0, \dots, \mathbf{r}_{i-1} \rangle$, $\mathbf{r}_i^T \mathbf{z}_j = 0$.

Queremos hallar las direcciones de búsqueda sin hacer Gram-Schmidt (o sea mas rápido)

Escribamos \mathbf{z}_i en la base de los residuos (ortogonales),

$$\mathbf{z}_i = \sum_{j=0}^i \frac{\mathbf{z}_i^T \mathbf{r}_j}{\mathbf{r}_j^T \mathbf{r}_j} \mathbf{r}_j,$$

ya vimos que $\mathbf{z}_i^T \mathbf{r}_i = \mathbf{r}_i^T \mathbf{r}_i$ pero además $\mathbf{z}_i^T (\mathbf{r}_i - \mathbf{r}_l) = \mathbf{z}_i^T \mathbf{A}(\mathbf{x}_i - \mathbf{x}_l) = 0$ porque $\mathbf{x}_i - \mathbf{x}_l \in W_i$ y \mathbf{z}_i es \mathbf{A} ortogonal a W_i .

$$\mathbf{z}_i = \sum_{j=0}^i \frac{\mathbf{z}_i^T \mathbf{r}_j}{\mathbf{r}_j^T \mathbf{r}_j} \mathbf{r}_j = \sum_{j=0}^i \frac{\mathbf{r}_i^T \mathbf{r}_i}{\mathbf{r}_j^T \mathbf{r}_j} \mathbf{r}_j = \mathbf{r}_i + \mathbf{r}_i^T \mathbf{r}_i \sum_{j=0}^{i-1} \frac{\mathbf{r}_j}{\mathbf{r}_j^T \mathbf{r}_j},$$

como la ultima sumatoria permite escribir recursivamente las direcciones de búsqueda

$$\mathbf{r}_i + \mathbf{r}_i^T \mathbf{r}_i \sum_{j=0}^{i-1} \frac{\mathbf{r}_j}{\mathbf{r}_j^T \mathbf{r}_j} = \mathbf{r}_i + \frac{\mathbf{r}_i^T \mathbf{r}_i}{\mathbf{r}_{i-1}^T \mathbf{r}_{i-1}} \left(\mathbf{r}_{i-1} + \mathbf{r}_{i-1}^T \mathbf{r}_{i-1} \sum_{j=0}^{i-2} \frac{\mathbf{r}_j}{\mathbf{r}_j^T \mathbf{r}_j} \right)$$

entonces

$$\mathbf{z}_i = \mathbf{r}_i + \frac{\mathbf{r}_i^T \mathbf{r}_i}{\mathbf{r}_{i-1}^T \mathbf{r}_{i-1}} \mathbf{z}_{i-1}$$

los residuos pueden obtenerse de

$$\begin{aligned} \mathbf{x}_{i+1} &= \mathbf{x}_i + \lambda \mathbf{z}_i \\ \mathbf{r}_{i+1} &= \mathbf{r}_i + \lambda_i \mathbf{A} \mathbf{z}_i \end{aligned}$$

Tomo \mathbf{x}_0 , calculo \mathbf{r}_0 asigno $\mathbf{z}_0 = \mathbf{r}_0$

for i=0...

- $\lambda_i = \frac{\mathbf{r}_i^T \mathbf{r}_i}{\mathbf{z}_i^T \mathbf{A} \mathbf{z}_i}$
- $\mathbf{x}_{i+1} = \mathbf{x}_i + \lambda_i \mathbf{z}_i$
- $\mathbf{r}_{i+1} = \mathbf{r}_i + \lambda_i \mathbf{A} \mathbf{z}_i$
- $\mathbf{z}_{i+1} = \mathbf{r}_{i+1} + \frac{\mathbf{r}_{i+1}^T \mathbf{r}_{i+1}}{\mathbf{r}_i^T \mathbf{r}_i} \mathbf{z}_i$

end

Aplicado a nuestro caso de ejemplo: para un error de orden 10^{-4} tarda 0.057 segundos...

7.6. GC y aproximación polinomial

Para estudiar el error del GC notemos

$$W_i = \langle \mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \dots, \mathbf{A}^{i-1}\mathbf{r}_0 \rangle$$

lo cual se ve por inducción. Si $i = 1$ es cierto, y asumiendo que vale para i queremos ver que vale para $i + 1$. Para eso vemos que basta con ver que

$$W_{i+1} \subset \langle \mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \dots, \mathbf{A}^i\mathbf{r}_0 \rangle$$

puesto que $\dim(W_{i+1}) = i + 1$

Asumiendo entonces que

$$W_i \ni \mathbf{r}_{i-1}, \mathbf{z}_{i-1} \in \langle \mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \dots, \mathbf{A}^{i-1}\mathbf{r}_0 \rangle$$

se tiene que $\mathbf{r}_i = \mathbf{r}_{i-1} - \lambda_{i-1} \mathbf{A} \mathbf{z}_{i-1} \in \langle \mathbf{r}_0, \mathbf{A} \mathbf{r}_0, \dots, \mathbf{A}^i \mathbf{r}_0 \rangle$ que es lo que queríamos ver.

En general, dada una matriz \mathbf{A} , subespacios de la forma $\langle \mathbf{b}, \mathbf{Ab}, \dots, \mathbf{A}^i \mathbf{b} \rangle$, se llaman subespacios de Krylov. La identidad de abajo permite relacionar esos subespacios con polinomios.

$$\begin{aligned}\langle \mathbf{r}_0, \mathbf{Ar}_0, \dots, \mathbf{A}^{i-1} \mathbf{r}_0 \rangle &= \{p(\mathbf{A}) \mathbf{r}_0 : p \in P_{i-1}[x]\} = \\ &= \{q(\mathbf{A})(\mathbf{x} - \mathbf{x}_0) : q \in P_i[x], q(0) = 0\}\end{aligned}$$

Va a resultar cómodo estudiar el error en la norma $\|\cdot\|_A$: es decir, acotar

$$\|\mathbf{x} - \mathbf{x}_i\|_A$$

Como \mathbf{r}_i es ortogonal a W_i , $\mathbf{x} - \mathbf{x}_i$ es A ortogonal a W_i

$$\|\mathbf{x} - \mathbf{x}_i\|_A = \inf_{\mathbf{y} \in W_i} \|\mathbf{x} - \mathbf{x}_i - \mathbf{y}\|_A$$

como $\mathbf{x}_0 - \mathbf{x}_i \in W_i$

$$\|\mathbf{x} - \mathbf{x}_i\|_A = \inf_{\mathbf{y} \in W_i} \|\mathbf{x} - \mathbf{x}_0 - \mathbf{y}\|_A =$$

$$\begin{aligned}&= \inf_{q \in P_i[x], q(0)=0} \|\mathbf{x} - \mathbf{x}_0 - q(\mathbf{A})(\mathbf{x} - \mathbf{x}_0)\|_A = \\ &= \inf_{p \in P_i[x], p(0)=1} \|p(\mathbf{A})(\mathbf{x} - \mathbf{x}_0)\|_A \leq \\ &\quad \inf_{p \in P_i[x], p(0)=1} \|p(\mathbf{A})\|_A \|(\mathbf{x} - \mathbf{x}_0)\|_A\end{aligned}$$

pero

$$\begin{aligned}\|p(\mathbf{A})\|_A^2 &= \sup_{\mathbf{x} \neq \mathbf{0}} \frac{\|p(\mathbf{A})\mathbf{x}\|_A^2}{\|\mathbf{x}\|_A^2} = \\ &= \sup_{\mathbf{x} \neq \mathbf{0}} \frac{(p(\mathbf{A})\mathbf{x})^T \mathbf{A} (p(\mathbf{A})\mathbf{x})}{\mathbf{x}^T \mathbf{A} \mathbf{x}} = \sup_{\mathbf{x} \neq \mathbf{0}} \frac{\mathbf{x}^T p(\mathbf{A})^T \mathbf{A} p(\mathbf{A}) \mathbf{x}}{\mathbf{x}^T \mathbf{A} \mathbf{x}} = \\ &= \sup_{\mathbf{x} \neq \mathbf{0}} \frac{\mathbf{x}^T \mathbf{A}^{1/2} p(\mathbf{A})^T p(\mathbf{A}) \mathbf{A}^{1/2} \mathbf{x}}{\mathbf{x}^T \mathbf{A}^{1/2} \mathbf{A}^{1/2} \mathbf{x}} = \sup_{\mathbf{y} \neq \mathbf{0}} \frac{\mathbf{y}^T p(\mathbf{A})^T p(\mathbf{A}) \mathbf{y}}{\mathbf{y}^T \mathbf{y}} = \\ &= \sup_{\mathbf{y} \neq \mathbf{0}} \frac{\|p(\mathbf{A})\mathbf{y}\|_2^2}{\|\mathbf{y}\|_2^2} = \|p(\mathbf{A})\|_2^2 = \rho(p(\mathbf{A}))^2 \leq \max_{\lambda \in [\lambda_1, \lambda_n]} p(\lambda)^2\end{aligned}$$

En definitiva

$$\|p(\mathbf{A})\|_A \leq \max_{x \in [\lambda_1, \lambda_n]} |p(x)|$$

y hay que acotar

$$\inf_{p \in P_i[x], p(0)=1} \max_{x \in [\lambda_1, \lambda_n]} |p(x)|$$

esto, por suerte, lo estudió Chebyshev.

En el intervalo $[-1, 1]$ los polinomios de Chebyshev son

$$T_0(x) = 1, T_1(x) = x, T_2(x) = 2x^2 - 1, \dots$$

en general

$$T_{n+1}(x) = 2T_n(x) - T_{n-1}(x)$$

Para $x \in [-1, 1]$, $|T_n(x)| \leq 1$

(de hecho en ese rango $T_n(x) = \cos(n \cos^{-1}(x))$)

para $|x| \geq 1$

$$T_n(x) = \frac{1}{2} \left((x + \sqrt{x^2 - 1})^n + (x - \sqrt{x^2 - 1})^n \right)$$

El polinomio $p(x) \in P_n[x]$ que minimiza $\max_{x \in [\lambda_1, \lambda_n]} |p(x)|$ con la condición $p(0) = 1$ es

$$\frac{1}{T_n(\frac{\lambda_n + \lambda_1}{\lambda_1 - \lambda_n})} T_n \left(\frac{2x - \lambda_n - \lambda_1}{\lambda_n - \lambda_1} \right)$$

$$\inf_{p \in P_i[x], p(0)=1} \max_{x \in [\lambda_1, \lambda_n]} |p(x)| \leq \frac{2}{\left(\frac{\sqrt{\frac{\lambda_n}{\lambda_1}} + 1}{\sqrt{\frac{\lambda_n}{\lambda_1}} - 1} \right)^i + \left(\frac{\sqrt{\frac{\lambda_n}{\lambda_1}} - 1}{\sqrt{\frac{\lambda_n}{\lambda_1}} + 1} \right)^i}$$

es decir

$$\inf_{p \in P_i[x], p(0)=1} \max_{x \in [\lambda_1, \lambda_n]} |p(x)| \leq 2 \left(\frac{\sqrt{\kappa(\mathbf{A})} - 1}{\sqrt{\kappa(\mathbf{A})} + 1} \right)^i$$

$$\|\mathbf{x} - \mathbf{x}_i\|_A \leq 2 \left(\frac{\sqrt{\kappa(\mathbf{A})} - 1}{\sqrt{\kappa(\mathbf{A})} + 1} \right)^i \|\mathbf{x} - \mathbf{x}_0\|_A$$

Capítulo 8

Descomposición en valores singulares

8.1. Motivación: número de condición y autovalores

Ya vimos que una matriz cuadrada está mal condicionada si está cerca de una matriz singular. O equivalentemente, está cerca de una matriz que no tiene el máximo rango posible. Podemos extender esta última idea a matrices rectangulares.

La matriz

$$\mathbf{A} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 + \epsilon \end{pmatrix}$$

tiene rango 2 si $\epsilon \neq 0$ pero está muy cerca de una matriz de rango 1 si ϵ es chico. Podemos pensar intuitivamente que la matriz está “mal condicionada”, aunque la definición que vimos no se aplica para matrices rectangulares.

OBSERVACIÓN 8.1.1. Para saber si una matriz cuadrada está mal condicionada no alcanza ver que hay un autovalor cercano a 0 o el determinante es chico.

EJEMPLO 8.1.2. La matriz $\mathbf{A} = \begin{pmatrix} \epsilon & 0 & 0 \\ 0 & \epsilon & 0 \\ 0 & 0 & \epsilon \end{pmatrix}$ está bien condicionada para cualquier $\epsilon \neq 0$.

Relacionamos ahora el número de condición con los autovalores de una matriz.

Para matrices simétricas,

$$\kappa_2(\mathbf{A}) = \|\mathbf{A}\|_2 \|\mathbf{A}^{-1}\|_2 = |\lambda_{\max}(\mathbf{A})| \left| \frac{1}{\lambda_{\min}(\mathbf{A})} \right| = \frac{|\lambda_{\max}(\mathbf{A})|}{|\lambda_{\min}(\mathbf{A})|}.$$

Vemos que lo que determina que una matriz está mal condicionada es la relación entre el autovalor de mayor módulo y el autovalor de menor módulo.

8.2. Introducción

Comencemos la sección con un experimento numérico. Generaremos matrices simétricas al azar y grafiquemos la imagen del círculo unitario por la transformación lineal resultante. Para conseguir figuras uniformes, reescalamos los gráficos de modo que el círculo unitario (en azul) contenga a su imagen (en rojo). El resultado obtenido es esperable. Las imágenes son elipses debido a que cada una de estas matrices posee una base $\{\mathbf{v}_1, \mathbf{v}_2\}$ de autovectores ortonormales. Es decir $\mathbf{A}\mathbf{v}_i = \lambda_i \mathbf{v}_i$, por lo que cualquier punto \mathbf{v} del círculo unitario puede escribirse como $\mathbf{v} = x_1 \mathbf{v}_1 + x_2 \mathbf{v}_2$ con $1 = x_1^2 + x_2^2$, por lo que $A\mathbf{v} = x_1 \lambda_1 \mathbf{v}_1 + x_2 \lambda_2 \mathbf{v}_2$ es un punto de una elipse de semiejes a lo largo de las rectas generadas por \mathbf{v}_1 y \mathbf{v}_2 de longitudes $|\lambda_1|$ y $|\lambda_2|$.

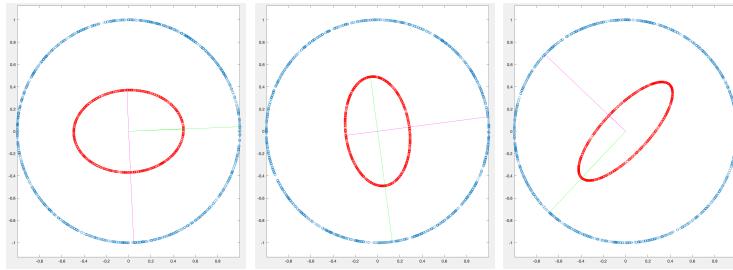


FIGURA 1. Elipses obtenidas por el mapeo del círculo unitario a través de matrices simétricas.

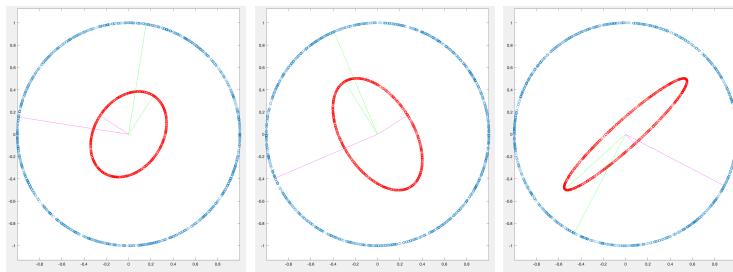


FIGURA 2. Elipses obtenidas por el mapeo del círculo unitario a través de matrices generales.

Lo extraordinario es que algo similar ocurre con matrices aleatorias cualesquiera, como podemos ver en la Figura 8.2. En este caso no podemos garantizar una base de autovectores ortonormales, sin embargo podemos estudiar las preimágenes de los semiejes de las elipses (en verde y rosa respectivamente) y notamos que provienen de vectores unitarios ortogonales (con el mismo color en las figuras). Es decir que nuestro experimento sugiere la existencia de dos pares de vectores ortonormales $\{\mathbf{v}_1, \mathbf{v}_2\}$ y $\{\mathbf{u}_1, \mathbf{u}_2\}$ y dos escalares σ_1, σ_2 , que podemos suponer ordenados (i.e. $\sigma_1 \geq \sigma_2 \geq 0$) y no negativos, cambiando si hiciera falta \mathbf{u}_i por $-\mathbf{u}_i$, tales que $\mathbf{A}\mathbf{v}_i = \sigma_i \mathbf{w}_i$.

Esta observación nos permite escribir

$$\mathbf{AV} = \mathbf{U}\Sigma$$

donde

$$\mathbf{V} = \begin{pmatrix} | & | \\ \mathbf{v}_1 & \mathbf{v}_2 \\ | & | \end{pmatrix} \quad \Sigma = \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix} \quad \mathbf{U} = \begin{pmatrix} | & | \\ \mathbf{u}_1 & \mathbf{u}_2 \\ | & | \end{pmatrix}$$

con matrices \mathbf{V}, \mathbf{U} ortogonales. Dicho otro modo puede factorizarse \mathbf{A} como

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^*$$

Esta factorización generaliza en algún sentido a la obtenida al diagonalizar matrices simétricas. Si bien, no toda matriz es diagonalizable en una base ortonormal $\{\mathbf{v}_1, \mathbf{v}_2\}$, nuestro experimento dice que algo similar puede obtenerse si nos permiten trabajar con *bases diferentes*: una $\{\mathbf{v}_1, \mathbf{v}_2\}$ para el espacio de partida y otra $\{\mathbf{u}_1, \mathbf{u}_2\}$ para el de llegada.

Lo realmente notable, como veremos en breve, es que esta factorización siempre existe y no solo para matrices cuadradas.

DEFINICIÓN 8.2.1. Dada $\mathbf{A} \in \mathbb{K}^{m \times n}$, la descomposición en valores singulares SVD es una factorización de \mathbf{A}

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^*,$$

con $\mathbf{U} \in \mathbb{K}^{m \times m}$ y $\mathbf{V} \in \mathbb{K}^{n \times n}$ matrices unitarias y $\Sigma \in \mathbb{R}^{m \times n}$. La matriz Σ tiene la forma

$$\Sigma = \begin{pmatrix} \mathbf{D} \\ \mathbf{0}_{(m-n) \times n} \end{pmatrix},$$

si $m \geq n$, o la forma

$$\Sigma = (\mathbf{D} \quad \mathbf{0}_{m \times (n-m)})$$

si $n > m$. En ambos casos $\mathbf{D} \in \mathbb{R}^{p \times p}$, donde $p = \min\{m, n\}$, es una matriz diagonal con elementos en la diagonal $d_{11} \geq d_{22} \geq \dots \geq d_{pp} \geq 0$. Los elementos no nulos de la diagonal de \mathbf{D} se denominan *valores singulares* de \mathbf{A} , y se notan $\sigma_i(\mathbf{A}) = d_{ii}$.

EJEMPLO 8.2.2.

$$\begin{pmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{pmatrix} \begin{pmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \end{pmatrix} \begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 \\ \frac{\sqrt{2}}{6} & -\frac{\sqrt{2}}{6} & \frac{2\sqrt{2}}{3} \\ \frac{2}{3} & -\frac{2}{3} & -\frac{1}{3} \end{pmatrix}$$

8.3. Construcción

Para constuir la descomposición SVD observamos las siguientes relaciones.

Dada una matriz $\mathbf{A} \in \mathbb{K}^{m \times n}$, si $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^*$:

- $\mathbf{AA}^* = (\mathbf{U}\Sigma\mathbf{V}^*)(\mathbf{V}\Sigma^*\mathbf{U}^*) = \mathbf{U}(\Sigma\Sigma^*)\mathbf{U}^*$, con $\Sigma\Sigma^* \in \mathbb{R}^{m \times m}$ diagonal.
- $\mathbf{A}^*\mathbf{A} = (\mathbf{V}\Sigma^*\mathbf{U}^*)(\mathbf{U}\Sigma\mathbf{V}^*) = \mathbf{V}(\Sigma^*\Sigma)\mathbf{V}^*$, con $\Sigma^*\Sigma \in \mathbb{R}^{n \times n}$ diagonal.

Recordar que $\mathbf{A}^*\mathbf{A}$ y \mathbf{AA}^* son matrices semidefinidas positivas y por lo tanto diagonalizables con autovalores no-negativos y base ortonormal de autovectores.

En base a estas relaciones, concluimos:

- Las columnas de \mathbf{U} son una base de autovectores de $\mathbf{AA}^* \in \mathbb{K}^{m \times m}$.
- Las columnas de \mathbf{V} son una base autovectores de $\mathbf{A}^*\mathbf{A} \in \mathbb{K}^{n \times n}$.
- Los valores singulares en Σ son la raíz cuadrada de los autovalores no nulos de $\mathbf{A}^*\mathbf{A}$ o \mathbf{AA}^* ordenados de mayor a menor.

Esto nos da un método para calcular la descomposición.

Primero construimos la matriz \mathbf{V} tomando una base de autovectores de $\mathbf{A}^*\mathbf{A}$.

Luego obtenemos las filas de \mathbf{U} a partir de la relación

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^*$$

multiplicando por \mathbf{V} :

$$\mathbf{AV} = \mathbf{U}\Sigma.$$

En el término de la izquierda tenemos

$$\mathbf{A}\mathbf{V} = \begin{pmatrix} | & & | \\ \mathbf{A}\mathbf{v}_1 & \dots & \mathbf{A}\mathbf{v}_n \\ | & & | \end{pmatrix}.$$

8.3.1. Si $m \leq n$. Si $m \leq n$,

$$\mathbf{A}\mathbf{V} = \mathbf{U}\Sigma = \begin{pmatrix} | & & | \\ \mathbf{u}_1 & \dots & \mathbf{u}_m \\ | & & | \end{pmatrix} \begin{pmatrix} \sigma_1 & & & \\ & \ddots & & \\ & & \sigma_m & \\ & & & 0 \end{pmatrix} = \begin{pmatrix} | & & | \\ \sigma_1\mathbf{u}_1 & \dots & \sigma_m\mathbf{u}_m \\ | & & | \end{pmatrix} \begin{pmatrix} & & \\ & & \\ & & 0 \end{pmatrix}$$

Por lo tanto, para $1 \leq i \leq n$, las columnas de \mathbf{U} cumplen $\sigma_i\mathbf{u}_i = \mathbf{A}\mathbf{v}_i$, donde \mathbf{v}_i son las columnas de \mathbf{V} . Obtenemos

$$\mathbf{u}_i = \frac{1}{\sigma_i} \mathbf{A}\mathbf{v}_i.$$

para $1 \leq i \leq m$ y $\sigma_i \neq 0$. Si existen valores singulares nulos, las columnas correspondientes de \mathbf{U} las elegimos completando una base ortonormal $\{\mathbf{u}_1, \dots, \mathbf{u}_m\}$ de $\mathbb{K}^{m \times m}$.

8.3.2. Si $m > n$. Si $m \geq n$,

$$\mathbf{A}\mathbf{V} = \mathbf{U}\Sigma = \begin{pmatrix} | & & | \\ \mathbf{u}_1 & \dots & \mathbf{u}_m \\ | & & | \end{pmatrix} \begin{pmatrix} \sigma_1 & & & \\ & \ddots & & \\ & & \sigma_n & \\ & & & 0 \end{pmatrix} = \begin{pmatrix} | & & | \\ \sigma_1\mathbf{u}_1 & \dots & \sigma_n\mathbf{u}_n \\ | & & | \end{pmatrix}$$

Por lo tanto, para $1 \leq i \leq n$, las columnas de \mathbf{U} cumplen $\sigma_i\mathbf{u}_i = \mathbf{A}\mathbf{v}_i$, donde \mathbf{v}_i son las columnas de \mathbf{V} . Obtenemos

$$\mathbf{u}_i = \frac{1}{\sigma_i} \mathbf{A}\mathbf{v}_i.$$

para $1 \leq i \leq n$ y $\sigma_i \neq 0$. El resto de las columnas de \mathbf{U} las elegimos completando una base ortonormal $\{\mathbf{u}_1, \dots, \mathbf{u}_m\}$ de $\mathbb{K}^{m \times m}$.

EJERCICIO 8.3.1. Calcular la descomposición SVD de la matriz

$$\mathbf{A} = \begin{pmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{pmatrix} \in \mathbb{R}^{2 \times 3}.$$

realizando los siguientes pasos:

1. Construir la matriz \mathbf{V} tomando como columnas una base ortonormal de autovectores de $\mathbf{A}^*\mathbf{A} \in \mathbb{R}^{3 \times 3}$.
2. Calcular los valores singulares $\sigma_i = \sqrt{\lambda_i}$, $i = 1, 2$, con λ_i los autovalores (no nulos) de $\mathbf{A}^*\mathbf{A}$.
3. Calcular las columnas de \mathbf{U} por la fórmula $\mathbf{u}_i = \frac{1}{\sigma_i} \mathbf{A}\mathbf{v}_i$, $i = 1, 2$.
4. Verificar.

Ya vimos cómo construir una descomposición SVD. Si repasamos la construcción que realizamos, vemos que utilizamos la existencia de tal descomposición para obtener propiedades de las matrices en la descomposición. Para probar la existencia de la descomposición nos falta completar algunos detalles de la construcción.

TEOREMA 8.3.2. Dada una matriz rectangular $\mathbf{A} \in \mathbb{K}^{m \times n}$, existen matrices

- $\mathbf{U} \in \mathbb{K}^{m \times m}$ unitaria,
- $\Sigma \in \mathbb{R}^{m \times n}$, con $\Sigma = (\mathbf{D}|0)$ o $\Sigma = \left(\begin{smallmatrix} \mathbf{D} & 0 \\ 0 & 0 \end{smallmatrix}\right)$, \mathbf{D} diagonal y $d_{11} \geq \dots \geq d_{ss} \geq 0$ ($s = \min\{m, n\}$),
- $\mathbf{V} \in \mathbb{K}^{n \times n}$ unitaria,

tales que $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^*$. Los valores no nulos $d_{11} \geq \dots \geq d_{ss} > 0$ son únicos y se denominan *valores singulares* de \mathbf{A} , $\sigma_i = d_{ii}$.

DEMOSTRACIÓN. En la construcción que hicimos, falta ver que los vectores $\mathbf{u}_i = \frac{1}{\sigma_i} \mathbf{A} \mathbf{v}_i$ construidos son ortogonales. Calculamos

$$\begin{aligned} \mathbf{u}_j^* \mathbf{u}_i &= \left(\frac{1}{\sigma_j} \mathbf{v}_j^* \mathbf{A}^* \right) \left(\mathbf{A} \mathbf{v}_i \frac{1}{\sigma_i} \right) \\ &= \frac{1}{\sigma_j} \mathbf{v}_j^* (\mathbf{A}^* \mathbf{A}) \mathbf{v}_i \frac{1}{\sigma_i} = \frac{1}{\sigma_j} (\mathbf{v}_j^*) (\mathbf{V} \Sigma^* \Sigma \mathbf{V}^*) \mathbf{v}_i \frac{1}{\sigma_i} \end{aligned}$$

y usando que $\mathbf{V}^* \mathbf{v}_i = \mathbf{e}_i$, obtenemos

$$\mathbf{u}_j^* \mathbf{u}_i = \frac{1}{\sigma_j} (\mathbf{e}_j^* \Sigma^*) (\Sigma \mathbf{e}_i) \frac{1}{\sigma_i} = \frac{1}{\sigma_j} (\sigma_j \mathbf{e}_j^*) (\sigma_i \mathbf{e}_i) \frac{1}{\sigma_i} = \mathbf{e}_j^* \mathbf{e}_i = \begin{cases} 1 & \text{si } i = j \\ 0 & \text{si } i \neq j \end{cases}$$

□

8.4. Valores singulares y norma-2

Ahora estamos en condiciones de retomar la discusión inicial de este capítulo.

TEOREMA 8.4.1. Dada una matriz rectangular $\mathbf{A} \in \mathbb{K}^{m \times n}$,

$$\|\mathbf{A}\|_2 = \sigma_1(\mathbf{A}).$$

DEMOSTRACIÓN. Recordando que multiplicar por una matriz unitaria es una transformación inversible que no cambia la norma de un vector,

$$\begin{aligned} \|\mathbf{A}\|_2 &= \sup_{\|\mathbf{x}\|_2=1} \|\mathbf{A}\mathbf{x}\|_2 \\ &= \sup_{\|\mathbf{x}\|_2=1} \|\mathbf{U}\Sigma\mathbf{V}^*\mathbf{x}\|_2 = \sup_{\|\mathbf{x}\|_2=1} \|\Sigma\mathbf{V}^*\mathbf{x}\|_2 \\ &= \sup_{\|\mathbf{y}\|_2=1} \|\Sigma\mathbf{y}\|_2 = \sup_{\|\mathbf{y}\|_2=1} \left(\sum_{i=1}^r \sigma_i^2 |y_i|^2 \right)^{\frac{1}{2}} \leq \sigma_1 \end{aligned}$$

□

PROPOSICIÓN 8.4.2. Si $\mathbf{A} \in \mathbb{K}^{m \times n}$ y $\text{rango}(\mathbf{A}) = n$, entonces

$$\min_{\|\mathbf{x}\|_2=1} \|\mathbf{A}\mathbf{x}\|_2 = \sigma_n(\mathbf{A})$$

COROLARIO 8.4.3. Si $\mathbf{A} \in \mathbb{K}^{n \times n}$ inversible,

$$\text{cond}_2(\mathbf{A}) = \frac{\sigma_1(\mathbf{A})}{\sigma_n(\mathbf{A})}.$$

En general, para cualquier matriz $\mathbf{A} \in \mathbb{K}^{m \times n}$ rectangular de rango máximo, definimos

$$\text{cond}_2(\mathbf{A}) = \frac{\sigma_1(\mathbf{A})}{\sigma_r(\mathbf{A})},$$

con $r = \min\{m, n\}$.

8.5. Distancia a matrices de menor rango

Por último, vemos cómo formalizar la noción de matrices rectangulares mal condicionadas o cercanas a matrices de menor rango.

Dada una matriz $\mathbf{A} \in \mathbb{K}^{n \times n}$, buscamos una matriz singular lo más cercana posible a \mathbf{A} . En otras palabras, buscamos $\mathbf{B} \in \mathbb{K}^{n \times n}$ de norma lo más chica posible tal que $\mathbf{A} + \mathbf{B}$ sea singular.

1. $\mathbf{A} + \mathbf{B} = \mathbf{A}(\mathbf{I} + \mathbf{A}^{-1}\mathbf{B})$
2. Si $\|\mathbf{A}^{-1}\mathbf{B}\| < 1$, entonces $\mathbf{I} + \mathbf{A}^{-1}\mathbf{B}$ (y por lo tanto $\mathbf{A} + \mathbf{B}$) es inversible.
3. Por lo tanto, si $\mathbf{A} + \mathbf{B}$ es singular y \mathbf{A} es inversible, debe ser $\|\mathbf{B}\| \geq 1/\|\mathbf{A}^{-1}\|$.
4. Tomando norma-2, y $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^*$ la SVD, $\|\mathbf{A}^{-1}\|_2 = 1/\sigma_n$.
5. Por lo tanto, si $\mathbf{A} + \mathbf{B}$ es singular, debe ser $\|\mathbf{B}\| \geq \sigma_n(\mathbf{A})$.
6. Podemos tomar $\mathbf{B} = \mathbf{U}\mathbf{E}\mathbf{V}^*$, con $\mathbf{E} = \text{diag}(0, 0, \dots, 0, -\sigma_n)$, que cumple $\|\mathbf{B}\|_2 = \|\mathbf{E}\|_2 = \sigma_n$.

TEOREMA 8.5.1. Dada $\mathbf{A} \in \mathbb{K}^{n \times n}$ inversible, con $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^*$ su descomposición en valores singulares. Definiendo

$$\Sigma_{n-1} = \begin{pmatrix} \sigma_1 & & & \\ & \ddots & & \\ & & \sigma_{n-1} & \\ & & & 0 \end{pmatrix},$$

la matriz $\tilde{\mathbf{A}} = \mathbf{U}\Sigma_{n-1}\mathbf{V}^*$ es la matriz singular más cercana a \mathbf{A} en norma-2.

EJERCICIO 8.5.2. Encontrar la matriz singular más cercana a $\mathbf{A} = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 5 \\ 1 & 5 & 8 \end{pmatrix}$.

Podemos ahora el resultado anterior para calcular las matrices más cercanas de distinto rango.

TEOREMA 8.5.3. Sea $\mathbf{A} \in \mathbb{K}^{m \times n}$ de rango $k > 0$, con $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^*$ su descomposición en valores singulares. La matriz $\mathbf{A}_1 \in \mathbb{K}^{m \times n}$ de rango $k_1 < k$ más cercana a \mathbf{A} en la norma 2 o norma Frobenius es $\mathbf{A}_1 = \mathbf{U}\Sigma_1\mathbf{V}^*$, con

$$\Sigma_1 = \begin{pmatrix} \sigma_1 & & & \\ & \ddots & & \\ & & \sigma_{k_1} & \\ & & & 0 \end{pmatrix},$$

la matriz igual a Σ excepto que solo se usan los valores singulares $\sigma_1, \dots, \sigma_{k_1}$ y los demás se reemplazan por 0.

8.6. Aplicaciones

8.6.1. Reducción de dimensionalidad. Para una población de m individuos analizamos n variables (features), con $m > n$, y queremos ver si hay variables redundantes o si podemos reducir la cantidad de variables sin perder mucha información.

Dada una matriz $\mathbf{A} \in \mathbb{R}^{m \times n}$, $m > n$, el k -ésimo valor singular nos dice la distancia de \mathbf{A} a una matriz de rango $k - 1$. Es decir, mide “cuánta información” vamos a perder si utilizamos $k - 1$ variables en lugar de las n variables originales.

EJERCICIO 8.6.1. Para la siguiente matriz, calcular los valores singulares. ¿Considera que hay variables redundantes? ¿Cuáles variables eliminaría?

	x_1	x_2	x_3	x_4
1	100	100	757	220
2	70	80	603	174
3	50	90	700	190
4	150	40	250	110
5	80	10	42	36

8.6.2. La pseudo-inversa. A partir de la descomposición en valores singulares $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^*$, construimos la pseudo inversa de \mathbf{A} ,

$$\mathbf{A}^\dagger = \mathbf{V}\Sigma^\dagger\mathbf{U}^*,$$

donde Σ^\dagger se obtiene a partir de Σ transponiéndola y reemplazando los elementos no nulos d_{ii} en la diagonal por $1/d_{ii}$.

EJERCICIO 8.6.2. Calcular la pseudo-inversa de

$$\begin{pmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{pmatrix} \begin{pmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \end{pmatrix} \begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & \frac{2\sqrt{2}}{3} \\ \frac{6}{3} & -\frac{2}{3} & -\frac{1}{3} \end{pmatrix}$$

Se cumplen las siguientes propiedades.

PROPOSICIÓN 8.6.3.

- $\mathbf{A}\mathbf{A}^\dagger\mathbf{A} = \mathbf{A}$ y $\mathbf{A}^\dagger\mathbf{A}\mathbf{A}^\dagger = \mathbf{A}^\dagger$ (pero en general no se cumple $\mathbf{A}\mathbf{A}^\dagger = I_m$ o $\mathbf{A}^\dagger\mathbf{A} = I_n$)
- Si $\mathbf{A} \in \mathbb{K}^{n \times n}$ es inversible, $\mathbf{A}^{-1} = \mathbf{A}^\dagger$.

DEMOSTRACIÓN. Ejercicio. □

EJERCICIO 8.6.4. Verificar $\mathbf{A}\mathbf{A}^\dagger\mathbf{A} = \mathbf{A}$ para $\mathbf{A} = \begin{pmatrix} 3 & 2 & 2 \\ 2 & 3 & -2 \end{pmatrix}$

8.6.3. Resolución de sistemas de ecuaciones. Enunciamos ahora el resultado principal, cuya demostración veremos en la Práctica 6 de Mínimos Cuadrados.

TEOREMA 8.6.5. Dada la ecuación $\mathbf{Ax} = \mathbf{b}$, con $\mathbf{A} \in \mathbb{K}^{m \times n}$, si el sistema tiene soluciones entonces

$$\mathbf{z} = \mathbf{A}^\dagger\mathbf{b}$$

es una solución y si hay infinitas soluciones, \mathbf{z} es la solución de mínima norma-2.

Utilizando la pseudo-inversa podemos obtener también un criterio para determinar si un sistema tiene solución.

TEOREMA 8.6.6. El sistema $\mathbf{Ax} = \mathbf{b}$ tiene solución si y solo si $\mathbf{AA}^\dagger \mathbf{b} = \mathbf{b}$.

DEMOSTRACIÓN.

- Si $\mathbf{AA}^\dagger \mathbf{b} = \mathbf{b}$ entonces $\mathbf{v} = \mathbf{A}^\dagger \mathbf{b}$ es solución del sistema.
- Recíprocamente, si existe \mathbf{v} tal que $\mathbf{Av} = \mathbf{b}$, entonces

$$\mathbf{b} = \mathbf{Av} = \mathbf{AA}^\dagger \mathbf{Av} = \mathbf{AA}^\dagger \mathbf{b}$$

(aunque usted no lo crea!).

□

Cuando el sistema solucón, la pseudo-inversa nos permite también hallar todas las soluciones.

TEOREMA 8.6.7. Dadas $\mathbf{A} \in \mathbb{K}^{m \times n}$ y $\mathbf{b} \in \mathbb{K}^n$. Si $\mathbf{AA}^\dagger \mathbf{b} = \mathbf{b}$, todos los vectores de la forma

$$\mathbf{x} = \mathbf{A}^\dagger \mathbf{b} + (\mathbf{I}_n - \mathbf{A}^\dagger \mathbf{A})\mathbf{y}$$

con $\mathbf{y} \in \mathbb{K}^n$ son soluciones del sistema $\mathbf{Ax} = \mathbf{b}$.

Más aún, cualquier solución la puedo escribir de esa forma.

DEMOSTRACIÓN. Si \mathbf{x} es de esa forma,

$$\begin{aligned} \mathbf{Ax} &= \mathbf{A}(\mathbf{A}^\dagger \mathbf{b} + (\mathbf{I}_n - \mathbf{A}^\dagger \mathbf{A})\mathbf{y}) \\ &= (\mathbf{AA}^\dagger \mathbf{b}) + (\mathbf{A} - \mathbf{AA}^\dagger \mathbf{A})\mathbf{y} \\ &= \mathbf{b} + (\mathbf{A} - \mathbf{A})\mathbf{y} = \mathbf{b}. \end{aligned}$$

Ahora, dado un vector $\mathbf{z} \in \mathbb{K}^n$ tal que $\mathbf{Az} = \mathbf{b}$, escribimos

$$\mathbf{z} = (\mathbf{A}^\dagger \mathbf{A})\mathbf{z} + (\mathbf{I}_n - \mathbf{A}^\dagger \mathbf{A})\mathbf{z} = \mathbf{A}^\dagger \mathbf{b} + (\mathbf{I}_n - \mathbf{A}^\dagger \mathbf{A})\mathbf{z}$$

que es un vector de la forma buscada.

□

OBSERVACIÓN 8.6.8.

- Puede haber soluciones repetidas entre las soluciones obtenidas de esta forma.
- En particular, el sistema $\mathbf{Ax} = \mathbf{b}$ tiene solución única si y solo si $\mathbf{A}^\dagger \mathbf{A} = \mathbf{I}_n$.

EJERCICIO 8.6.9. Determinar cuántas soluciones tienen los siguientes sistemas de ecuaciones, y hallar una solución cuando sea posible.

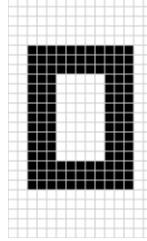
$$\left\{ \begin{array}{l} x - 2y = 2 \\ 2x + y = 1 \\ x + 3y = -1 \end{array} \right. \quad \left\{ \begin{array}{l} x - 2y + z = 2 \\ 2x + y + 5z = 1 \\ x + 3y + 4z = -1 \end{array} \right. \quad \left\{ \begin{array}{l} x - 2y + z = 2 \\ 2x + y + 5z = 1 \\ x + 3y + 4z = 7 \end{array} \right.$$

En Python puede utilizar `np.linalg.pinv` para calcular la pseudo-inversa.

8.6.4. Compresión de imágenes. Veamos como comprimir una imagen aplicando reducción de dimensionalidad por SVD.

Tomamos una matriz $\mathbf{A} \in \mathbb{R}^{m \times n}$, con $a_{ij} \in [0, 1]$, que representa una tonalidad de gris, siendo $0 = \text{negro}$ y $1 = \text{blanco}$.

EJEMPLO 8.6.10. Para representar la imagen de la figura, tomamos $\mathbf{A} \in \mathbb{R}^{25 \times 15}$.



¿Cuál es el rango de \mathbf{A} ? ¿Cuántos valores singulares no nulos tiene \mathbf{A} ?

Eliminamos columnas de \mathbf{U}

La descomposición en valores singulares de $\mathbf{A} \in \mathbb{R}^{25 \times 15}$,

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^*$$

tiene 3 valores singulares no nulos, $\mathbf{U} \in \mathbb{R}^{25 \times 25}$, $\Sigma \in \mathbb{R}^{25 \times 15}$ y $\mathbf{V} \in \mathbb{R}^{15 \times 15}$.

¿Podemos descartar filas de \mathbf{U} y \mathbf{V} sin perder información?

Recordando que

$$\mathbf{U}\Sigma = \begin{pmatrix} | & & | \\ \mathbf{u}_1 & \dots & \mathbf{u}_{25} \\ | & & | \end{pmatrix} \begin{pmatrix} \sigma_1 & & & \\ & \ddots & & \\ & & \sigma_{15} & \\ \hline & & & 0 \end{pmatrix} = \begin{pmatrix} | & & | \\ \sigma_1 \mathbf{u}_1 & \dots & \sigma_{15} \mathbf{u}_{15} \\ | & & | \end{pmatrix}$$

vemos que podemos tomar siempre $\mathbf{u}_i = \mathbf{0}$ para $i > 16$ sin perder información, y en nuestro caso también $\mathbf{u}_i = \mathbf{0}$ para $i > 3$.

Eliminamos columnas de \mathbf{V}

Análogamente, a partir de $(\Sigma\mathbf{V}^*)^* =$

$$\mathbf{V}\Sigma^* = \begin{pmatrix} | & & | \\ \mathbf{v}_1 & \dots & \mathbf{v}_{15} \\ | & & | \end{pmatrix} \left(\begin{array}{c|c} \sigma_1 & \\ \hline & \ddots \\ & & \sigma_{15} \end{array} \right) \left| \begin{array}{c} 0 \\ \hline 0 \end{array} \right. = \begin{pmatrix} | & & | \\ \sigma_1 \mathbf{v}_1 & \dots & \sigma_{15} \mathbf{v}_{15} \\ | & & | \end{pmatrix} \left| \begin{array}{c} 0 \\ \hline 0 \end{array} \right.$$

vemos que podemos tomar $\mathbf{v}_j = \mathbf{0}$ para $j > 3$ sin perder información.

En forma más compacta, para no guardar tantos 0's, podemos guardar la descomposición

$$\mathbf{A}_{25 \times 15} = \begin{pmatrix} | & & | \\ \mathbf{u}_1 & \dots & \mathbf{u}_3 \\ | & & | \end{pmatrix}_{25 \times 3} \begin{pmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{pmatrix}_{3 \times 3} \begin{pmatrix} -\mathbf{v}_1 - \\ -\mathbf{v}_2 - \\ -\mathbf{v}_3 - \end{pmatrix}_{3 \times 15}$$

Cantidad de datos

En total, necesitamos guardar:

- 3 columnas de $\mathbf{U} \rightarrow 75$ valores
- 3 columnas de $\mathbf{V} \rightarrow 45$ valores
- 3 valores singulares
- Total: $75 + 45 + 3 = 123$ valores

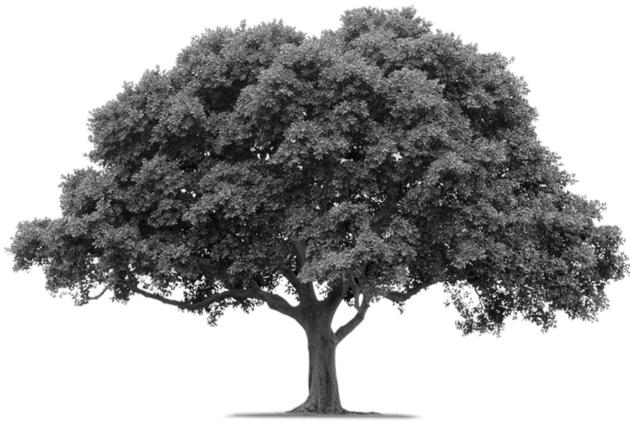
Para guardar la matriz original necesitamos $25 \times 15 = 375$ valores

Veamos otro ejemplo más realista.

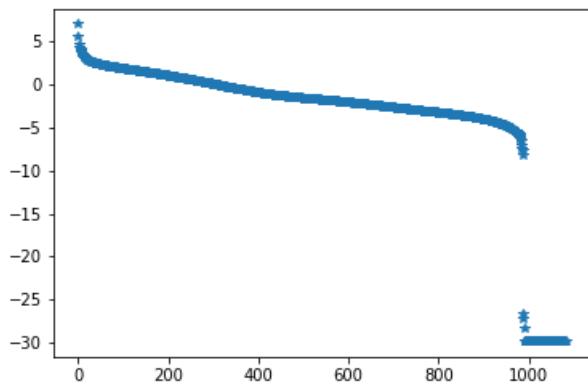
EJEMPLO 8.6.11. Tomamos ahora la imagen de un árbol $\mathbf{A} \in \mathbb{R}^{1080 \times 2000 \times 3}$.



Para imágenes en color se utilizan tres matrices con tonalidades RGB (red, green, blue), pero vamos a utilizar solo tonalidades de gris, tomamos $\mathbf{A} \in \mathbb{R}^{1080 \times 2000}$.



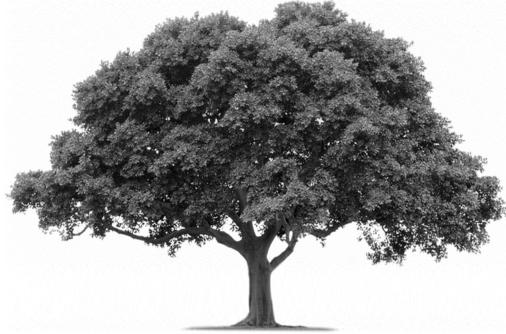
Calculamos la descomposición $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^*$ y graficamos los logartimos de los valores singulares. Podemos tomar los primeros 987 valores singulares sin casi perder información, pero no es suficiente compresión.



Probamos tomar los primeros 200 valores singulares. Es decir, tomamos

$$\tilde{\mathbf{A}} = \begin{matrix} \tilde{\mathbf{U}} & \tilde{\mathbf{\Sigma}} & \tilde{\mathbf{V}}^* \\ 1080 \times 200 & 200 \times 200 & 200 \times 2000 \end{matrix}$$

y obtenemos la imagen



EJERCICIO 8.6.12. Calcular el porcentaje de compresión (es decir, la relación entre la cantidad de números que guardamos para la imagen comprimida y la cantidad de números de la imagen original).

Capítulo 9

Cuadrados mínimos

9.1. Ajuste lineal

Comenzamos con el caso más simple. Dada un conjunto finito de N puntos $\{(x_i, y_i)\}_{1 \leq i \leq N}$ en el plano, queremos encontrar una recta $y = c_1x + c_0$ que mejor aproxime los datos.

Para una recta dada $f(x) = c_1x + c_0$, definimos los residuos

$$r_i = f(x_i) - y_i = (c_1x_i + c_0) - y_i, \quad 1 \leq i \leq N.$$

En el problema de cuadrados mínimos, buscamos minimizar la suma de los cuadrados de los residuos

$$S = \sum_{i=1}^N r_i^2 = \sum (y_i - f(x_i))^2.$$

Para resolver el problema, lo planteamos en forma matricial. Definimos una matriz $\mathbf{A} \in \mathbb{R}^{N \times 2}$ con unos en la primera columna y los valores x_i en la segunda,

$$\mathbf{A} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N \end{pmatrix}$$

y verificamos

$$\begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_N \end{pmatrix} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \end{pmatrix} - \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}.$$

Llamando $\mathbf{r} = \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_N \end{pmatrix}$, $\mathbf{c} = \begin{pmatrix} c_0 \\ c_1 \end{pmatrix}$ e $\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}$, obtenemos $\mathbf{r} = \mathbf{Ac} - \mathbf{y}$.

Por lo tanto, $S = \|\mathbf{Ac} - \mathbf{y}\|_2^2 = \|\mathbf{r}\|_2^2$. Es decir, buscamos el vector $\mathbf{c} = \begin{pmatrix} c_0 \\ c_1 \end{pmatrix}$ que minimice la norma-2 al cuadrado del vector de residuos.

Consideraremos el problema

$$\mathbf{Ax} = \mathbf{b}$$

con $\mathbf{A} \in \mathbb{R}^{m \times n}$. La existencia de soluciones de este problema equivale a que $\mathbf{b} \in \langle \mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n \rangle$ (\mathbf{a}_i son las columnas de \mathbf{A}) y en ese caso la unicidad depende de que el conjunto $\{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n\}$ sea *l.i.*.

En el caso en que $\mathbf{b} \notin \langle \mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n \rangle$ podemos intentar minimizar el residuo

$$\mathbf{r} = \mathbf{b} - \mathbf{Ax}$$

Para eso nos conviene usar una norma: la norma 2 es la mas adecuada como veremos en breve.

El problema es entonces, hallar $\mathbf{x} \in \mathbb{R}^n$ tal que

$$\|\mathbf{Ax} - \mathbf{b}\|_2 = \min_{\mathbf{z} \in \mathbb{R}^n} \|\mathbf{Az} - \mathbf{b}\|_2$$

o el equivalente hallar $\mathbf{x} \in \mathbb{R}^n$ tal que

$$\|\mathbf{Ax} - \mathbf{b}\|_2^2 = \min_{\mathbf{z} \in \mathbb{R}^n} \|\mathbf{Az} - \mathbf{b}\|_2^2.$$

9.1.1. Mínimos cuadrados como problema de minimización. Un enfoque posible (fuera del álgebra lineal numérica) es buscar una ecuación para el punto crítico \mathbf{x} como minimizante.

Eso equivale a anular el gradiente de la función $F : \mathbb{R}^n \rightarrow \mathbb{R}$, $F(\mathbf{z}) = \|\mathbf{Az} - \mathbf{b}\|_2^2$, es decir

$$\begin{aligned} 0 &= \frac{\partial F}{\partial z_k}|_{\mathbf{x}} = \sum_{i=1}^m \frac{\partial ([\mathbf{Az} - \mathbf{b}]_i)^2}{\partial z_k}|_{\mathbf{x}} = \\ &\sum_{i=1}^m \frac{\partial \left(\sum_{j=1}^n a_{ij} z_j - b_i \right)^2}{\partial z_k}|_{\mathbf{x}} = 2 \sum_{i=1}^m \left(\sum_{j=1}^n a_{ij} x_j - b_i \right) a_{ik} \end{aligned}$$

La validez de esta identidad para todo $1 \leq k \leq n$ dice que

$$\mathbf{A}^T (\mathbf{Ax} - \mathbf{b}) = \mathbf{0}$$

Dicho de otro modo

$$\mathbf{A}^T \mathbf{Ax} = \mathbf{A}^T \mathbf{b}$$

estas se llaman las ecuaciones normales.

A pesar de su popularidad esta no siempre es la mejor manera de calcular \mathbf{x} .

Desde esas ecuaciones podemos deducir cuando habrá solución única de nuestro problema: si las columnas de \mathbf{A} son *l.i.* entonces $\mathbf{A}^T \mathbf{Ax} = \mathbf{0}$ implica

$$0 = (\mathbf{A}^T \mathbf{Ax})^T \mathbf{x} = (\mathbf{Ax})^T \mathbf{Ax} = \|\mathbf{Ax}\|_2^2,$$

es decir $\mathbf{Ax} = \mathbf{0}$ y así $\mathbf{x} = \mathbf{0}$ por ser las columnas de \mathbf{A} *l.i.*, por lo que el núcleo de $\mathbf{A}^T \mathbf{A}$ es trivial.

Observemos que en este caso $\mathbf{A}^T \mathbf{A} \in \mathbb{R}^{n \times n}$ resulta *simétrica definida positiva*. En particular podemos resolver el sistema usando Cholesky a un costo $\sim mn^2 + \frac{1}{3}n^3$

En gran parte de las aplicaciones $m \gg n$ y las columnas de \mathbf{A} son *l.i.* lo cual muestra el éxito relativo de las ecuaciones normales.

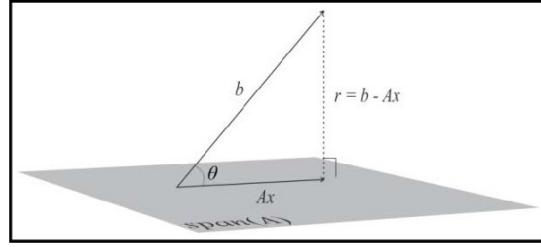
Intuitivamente si

$$\mathbf{A} = \mathbf{U} \boldsymbol{\Sigma} \mathbf{V}^*$$

y $\mathbf{A} \in \mathbb{R}^{m \times n}$, vimos que puede definirse la condición de \mathbf{A} como $\frac{\sigma_1}{\sigma_n}$. Si hacemos $\mathbf{A}^T \mathbf{A} = \mathbf{V} \boldsymbol{\Sigma}^2 \mathbf{V}^*$ vemos que en las ecuaciones normales la condición de la matriz final empeora $\kappa(\mathbf{A}^T \mathbf{A}) = \left(\frac{\sigma_1}{\sigma_n}\right)^2$.

9.1.2. Enfoque geométrico. Como vimos, para el sistema $\mathbf{Ax} = \mathbf{b}$, con $\mathbf{A} \in \mathbb{R}^{m \times n}$, la existencia de soluciones de este problema equivale a que $\mathbf{b} \in \langle \mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n \rangle$ (donde \mathbf{a}_i son las columnas de \mathbf{A}). Es decir, el sistema tiene solución cuando \mathbf{b} pertenece al espacio generado por las columnas de \mathbf{A} , o equivalentemente, a la imagen de \mathbf{A} . Si $\tilde{\mathbf{x}}$ es la solución, vale $\|\mathbf{A}\tilde{\mathbf{x}} - \mathbf{b}\| = 0$.

Cuando \mathbf{b} no pertenece a la imagen de \mathbf{A} , el problema de mínimos cuadrados consiste en encontrar el vector $\tilde{\mathbf{x}}$ que minimice $\|\mathbf{A}\tilde{\mathbf{x}} - \mathbf{b}\|_2$. Como \mathbf{Ax} es un punto en el subespacio Π generado por las columnas de \mathbf{A} , buscamos el punto de dicho subespacio más cercano a \mathbf{b} . Podemos entonces encontrar $\tilde{\mathbf{x}}$ calculando la proyección \mathbf{p} de \mathbf{b} sobre Π y luego buscando $\tilde{\mathbf{x}}$ tal que $\mathbf{A}\tilde{\mathbf{x}} = \mathbf{p}$.



Veamos cómo seguir este enfoque en forma práctica. Definimos el proyector P sobre la imagen de \mathbf{A} . Esto es, si $\{\mathbf{q}_1, \dots, \mathbf{q}_n\}$ es una base ortonormal de $\mathcal{S} = \text{im}(\mathbf{A})$, tenemos

$$P_{\mathcal{S}}(\mathbf{b}) = \mathbf{Q} \mathbf{Q}^* \mathbf{b}$$

donde

$$\mathbf{Q} = (\mathbf{q}_1 | \mathbf{q}_2 | \cdots | \mathbf{q}_n).$$

Podemos obtener la matriz $\mathbf{Q} \in \mathbb{R}^{m \times n}$ mediante la factorización QR de \mathbf{A} , es decir

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

con $\mathbf{R} \in \mathbb{R}^{n \times n}$ triangular superior.

Escribimos

$$\mathbf{Q} \mathbf{R} \mathbf{x} = \mathbf{A} \mathbf{x} = P_{\mathcal{S}}(\mathbf{b}) = \mathbf{Q} \mathbf{Q}^* \mathbf{b}$$

y como $\mathbf{Q}^* \mathbf{Q} = \mathbf{I}$ (porque las columnas de \mathbf{Q} son ortonormales), basta resolver

$$\mathbf{R} \mathbf{x} = \mathbf{Q}^* \mathbf{b}$$

Este algoritmo es más estable que el de las ecuaciones normales (ya que no elevamos al cuadrado la condición del problema) y su costo está gobernado por el costo de obtener la descomposición QR , que es $\sim 2mn^2 - \frac{1}{3}n^3$.

9.1.3. Mínimos cuadrados y pseudo-inversa. Volvemos a las ecuaciones normales: supongamos que $\mathbf{A} \in \mathbb{R}^{m \times n}$, $m \gg n$ tiene rango n (es decir, las columnas de \mathbf{A} son l.i.). Debemos resolver

$$\mathbf{A}^t \mathbf{A} \mathbf{x} = \mathbf{A}^t \mathbf{b}.$$

Si escribimos $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^*$, tenemos

$$\mathbf{V}\Sigma^t\Sigma\mathbf{V}^t\mathbf{x} = \mathbf{A}^t\mathbf{A}\mathbf{x} = \mathbf{A}^t\mathbf{b} = \mathbf{V}\Sigma^t\mathbf{U}^t\mathbf{b}.$$

Como

$$\Sigma = \left(\begin{array}{ccc|c} \sigma_1 & & & \\ & \ddots & & \\ & & \sigma_m & \\ \hline & & & 0 \end{array} \right)$$

y

$$\Sigma^t = \left(\begin{array}{cc|c} \sigma_1 & & \\ & \ddots & \\ \hline & & \sigma_n \\ \hline & 0 & \end{array} \right)$$

tenemos

$$\Sigma^t\Sigma = \left(\begin{array}{cccc} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_n^2 \end{array} \right)$$

Luego $\Sigma^t\Sigma$ es inversible (porque supusimos que \mathbf{A} tiene rango n y por lo tanto $\sigma_i \neq 0$ para todo $1 \leq i \leq n$).

Como \mathbf{V} es unitaria, de la igualdad anterior obtenemos $\Sigma^t\Sigma\mathbf{V}^t\mathbf{x} = \Sigma^t\mathbf{U}^t\mathbf{b}$ y por lo tanto, $\mathbf{x} = \mathbf{V}(\Sigma^t\Sigma)^{-1}\Sigma^t\mathbf{U}^t\mathbf{b}$. Obtenemos

$$\mathbf{x} = \mathbf{V} \left(\begin{array}{ccc|c} \sigma_1^{-1} & & & 0 \\ & \ddots & & \\ & & \sigma_m^{-1} & \\ \hline & & & 0 \end{array} \right) \mathbf{U}^t\mathbf{b}$$

o sea

$$\mathbf{x} = \mathbf{A}^+\mathbf{b}$$

y la pseudoinversa funciona como “inversa”.

En general en el caso $m \gg n$ podríamos tener rango menor a n (digamos $r < n$).

Recordemos que de $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^t$, tenemos que para $1 \leq i \leq r$

$$\mathbf{A}\mathbf{v}_i = \mathbf{U}\Sigma\mathbf{V}^t\mathbf{v}_i = \mathbf{U}\Sigma\mathbf{e}_i = \mathbf{U}\sigma_i\mathbf{e}_i = \sigma_i\mathbf{U}_i$$

y

$$\mathbf{A}\mathbf{v}_i = \mathbf{0}$$

si $r < i \leq n$.

Luego $\ker(\mathbf{A}) = \langle \mathbf{v}_{r+1}, \dots, \mathbf{v}_n \rangle$ y $\text{im}(\mathbf{A}) = \langle \mathbf{u}_1, \dots, \mathbf{u}_r \rangle$.

De hecho

$$\mathbf{A}^+ \mathbf{A} = \mathbf{v} \boldsymbol{\Sigma}^+ \boldsymbol{\Sigma} \mathbf{v}^t = \mathbf{v} \begin{pmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & 0 \end{pmatrix} \mathbf{v}^t$$

Queremos estudiar $\mathbf{A}^+ \mathbf{b}$ en este caso.

El producto

$$\mathbf{U}^t \mathbf{b} = \begin{pmatrix} \mathbf{u}_1^t \mathbf{b} \\ \mathbf{u}_2^t \mathbf{b} \\ \vdots \\ \mathbf{u}_m^t \mathbf{b} \end{pmatrix}$$

así

$$\boldsymbol{\Sigma}^+ \mathbf{U}^t \mathbf{b} = \begin{pmatrix} \sigma_1^{-1}(\mathbf{u}_1^t \mathbf{b}) \\ \sigma_2^{-1}(\mathbf{u}_2^t \mathbf{b}) \\ \vdots \\ \sigma_r^{-1}(\mathbf{u}_r^t \mathbf{b}) \\ 0 \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^{n \times 1}$$

entonces

$$\mathbf{x} = \mathbf{V} \boldsymbol{\Sigma}^+ \mathbf{U}^t \mathbf{b} = \sum_{j=1}^r \sigma_j^{-1}(\mathbf{u}_j^t \mathbf{b}) \mathbf{v}_j$$

Chequeamos

$$\mathbf{A} \mathbf{x} = \sum_{j=1}^r \sigma_j^{-1}(\mathbf{u}_j^t \mathbf{b}) \mathbf{A} \mathbf{v}_j = \sum_{j=1}^r (\mathbf{u}_j^t \mathbf{b}) \mathbf{u}_j = P_{\text{im}(A)}(\mathbf{b})$$

y vemos que es solución del problema de cuadrados mínimos.

Si hubiera otro $\tilde{\mathbf{x}}$ tal que $\mathbf{A} \tilde{\mathbf{x}} = P_{\text{im}(A)}(\mathbf{b})$ entonces

$$\mathbf{A}(\tilde{\mathbf{x}} - \mathbf{x}) = \mathbf{0}$$

luego

$$\begin{aligned} \tilde{\mathbf{x}} - \mathbf{x} &= \sum_{j=r+1}^n \alpha_j \mathbf{v}_j \\ \tilde{\mathbf{x}} &= \mathbf{x} + \sum_{j=r+1}^n \alpha_j \mathbf{v}_j \end{aligned}$$

y como $\mathbf{x} \perp \mathbf{v}_j$ para todo $r+1 \leq j \leq n$, obtenemos

$$\|\tilde{\mathbf{x}}\|_2^2 = \tilde{\mathbf{x}}^t \tilde{\mathbf{x}} = \mathbf{x}^t \mathbf{x} + \sum_{j=r+1}^n |\alpha_j|^2$$

Concluimos que

$$\|\mathbf{x}\|_2 < \|\tilde{\mathbf{x}}\|_2,$$

es decir, la pseudoinversa da la solución de menor norma 2.

9.2. Interpolación polinomial

Un caso tradicional de uso de cuadrados mínimos es en aproximación polinomial.

El problema de la interpolación consiste en lo siguiente: dados $n + 1$ pares de puntos

$$\{(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)\} \subset \mathbb{K}^n$$

hallar un polinomio $p_n(x)$ de grado a lo sumo n tal que $p_n(x_i) = y_i$.

Con los $x_i \neq x_j$ este problema siempre tiene solución única gracias a las siguientes observaciones.

Se propone $p_n(x) = a_0 + a_1x + \dots + a_{n-1}x^{n-1} + a_nx^n$ y se hace $p_n(x_i) = y_i$ para todo $0 \leq i \leq n$ que conduce a un sistema lineal con matriz de Vandermonde $V[x_0, x_1, \dots, x_n]$

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}.$$

Se tiene

$$\det V[x_0, x_1, \dots, x_n] = \prod_{0 \leq i < j \leq n} (x_j - x_i).$$

(La demo mas corta que conozco...)

Vale en 2×2 y sale por inducción: supongo que vale hasta $n - 1$.

LLamo $p_n(x) = \det V[x_0, x_1, \dots, x_{n-1}, x]$, claramente es polinomio y tiene grado n , su coeficiente principal es (desarrollo por la primera fila y uso hipótesis inductiva) $\det V[x_0, x_1, \dots, x_{n-1}] = \prod_{0 \leq i < j \leq n-1} (x_j - x_i)$.

Ademas las raíces de $p_n(x)$ son exactamente x_0, x_1, \dots, x_{n-1} , luego se escribe como

$$p_n(x) = \prod_{0 \leq i < j \leq n-1} (x_j - x_i) \prod_{k=0}^{n-1} (x - x_k),$$

evaluando en $x = x_n$ se obtiene el resultado mencionado.

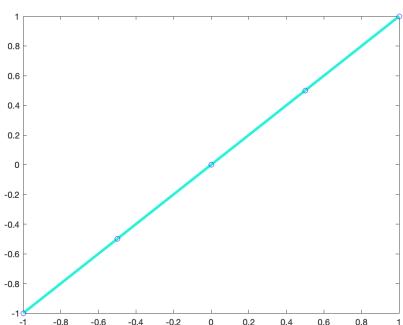
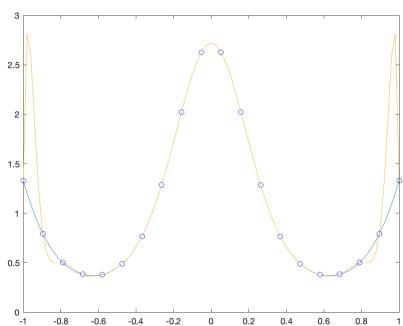
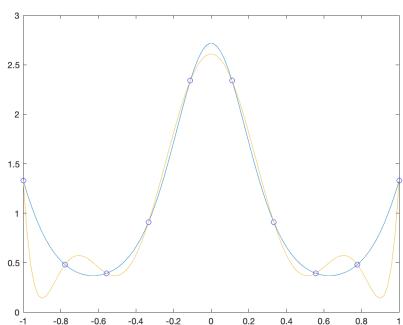
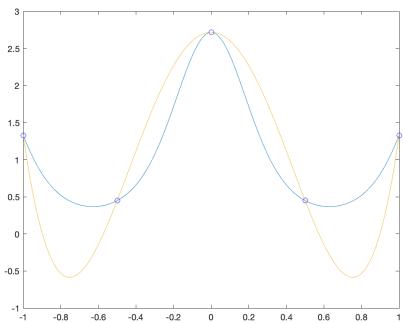
En particular el problema de interpolación tiene solución única...

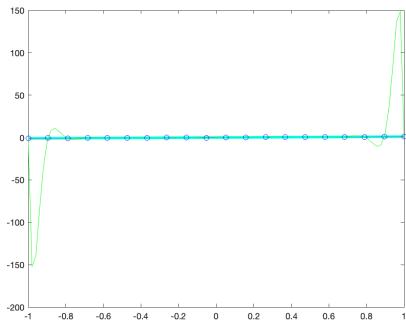
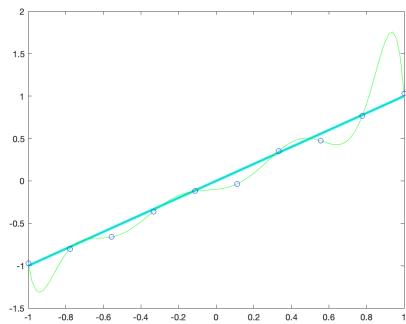
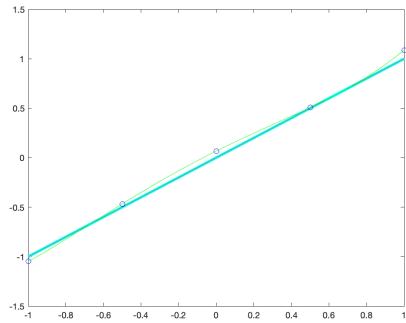
Hay otros enfoques... Lagrange

$$\begin{aligned} L_i(x) &= \frac{\prod_{0 \leq j \leq n, j \neq i} (x - x_j)}{\prod_{0 \leq j \leq n, j \neq i} (x_i - x_j)} \\ L_i(x_j) &= \delta_i^j \\ p_n(x) &= \sum_{i=0}^n y_i L_i(x). \end{aligned}$$

La unicidad sale aparte...

Problemas de la interpolación:





La teoría anterior se aplica igual las matrices de Vandermonde son ($m \gg n$)

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & x_m & x_m^2 & \cdots & x_m^n \end{pmatrix}$$

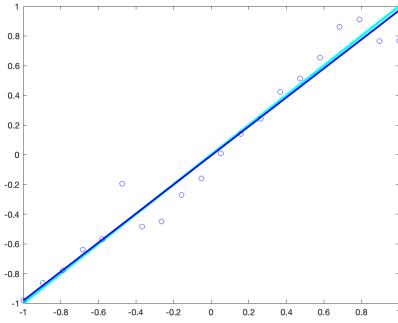
notar que las columnas son *l.i.* porque las primeras $n+1$ filas forman una matriz de Vandermonde cuadrada invertible.

Si hacemos $n = 1$

$$\begin{pmatrix} 1 & x_0 \\ 1 & x_1 \\ \vdots & \vdots \\ 1 & x_m \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_m \end{pmatrix}.$$

las ecuaciones normales conducen a la tradicional expresión

$$\left(\frac{1}{m+1} \sum_{i=0}^m x_i \quad \frac{\frac{1}{m+1} \sum_{i=0}^m x_i}{\frac{1}{m+1} \sum_{i=0}^m x_i^2} \right) \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} = \left(\frac{\frac{1}{m+1} \sum_{i=0}^m y_i}{\frac{1}{m+1} \sum_{i=0}^m x_i y_i} \right).$$



9.3. Regularización

En el caso de rango $r < n$ (o muy mal condicionado) también se puede pensar el problema del modo siguiente: minimizar

$$F(\mathbf{z}) = \|\mathbf{A}\mathbf{z} - \mathbf{b}\|^2 + \alpha^2 \|\mathbf{z}\|^2$$

y luego hacer $\alpha \rightarrow 0$.

- El término se elige α^2 para garantizar signo
- La expresión: $\alpha^2 \|\mathbf{z}\|^2$ se denomina penalización (se penalizan valores grandes de $\|\mathbf{z}\|$).
- Para cada α tenemos una solución: la denotamos \mathbf{x}_α .

Decimos que

$$\mathbf{x}_\alpha \rightarrow \mathbf{x}$$
 solución dada por \mathbf{A}^+

En efecto: para el mínimo de F se tiene

$$2(\mathbf{A}^t(\mathbf{A}\mathbf{x}_\alpha - \mathbf{b}) + \alpha^2 \mathbf{x}_\alpha) = \mathbf{0}$$

$$(\mathbf{A}^t \mathbf{A} + \alpha^2 \mathbf{I}) \mathbf{x}_\alpha = \mathbf{A}^t \mathbf{b}$$

en el caso mas interesante en que \mathbf{A} tiene rango $r < n$ la matriz simétrica $\mathbf{A}^t \mathbf{A}$ no es invertible pero para todo $\alpha \neq 0$, $\mathbf{A}^t \mathbf{A} + \alpha^2 \mathbf{I}$ sí lo es.

$$\mathbf{x}_\alpha = (\mathbf{A}^t \mathbf{A} + \alpha^2 \mathbf{I})^{-1} \mathbf{A}^t \mathbf{b}$$

usando SVD

$$\mathbf{A}^t \mathbf{A} + \alpha^2 \mathbf{I} = \mathbf{V} \begin{pmatrix} \sigma_1^2 + \alpha^2 & 0 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \sigma_2^2 + \alpha^2 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 0 & \ddots & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & \sigma_r^2 + \alpha^2 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & \alpha^2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 & \ddots & \vdots \\ 0 & 0 & \ddots & 0 & 0 & \cdots & \alpha^2 \end{pmatrix} \mathbf{V}^t$$

$$\mathbf{V} \begin{pmatrix} (\sigma_1^2 + \alpha^2)^{-1} & 0 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & (\sigma_2^2 + \alpha^2)^{-1} & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 0 & \ddots & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & (\sigma_r^2 + \alpha^2)^{-1} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & \alpha^{-2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 & \ddots & \vdots \\ 0 & 0 & \ddots & 0 & 0 & \cdots & \alpha^{-2} \end{pmatrix} \mathbf{V}^t$$

$$(\mathbf{A}^t \mathbf{A} + \alpha^2 \mathbf{I})^{-1} \mathbf{A}^t = (\mathbf{A}^t \mathbf{A} + \alpha^2 \mathbf{I})^{-1} \mathbf{V} \boldsymbol{\Sigma}^t \mathbf{U}^t =$$

$$\mathbf{V} \begin{pmatrix} \frac{\sigma_1}{\sigma_1^2 + \alpha^2} & 0 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \frac{\sigma_2}{\sigma_2^2 + \alpha^2} & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 0 & \ddots & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & \frac{\sigma_r}{\sigma_r^2 + \alpha^2} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 & \ddots & \vdots \\ 0 & 0 & \ddots & 0 & 0 & \cdots & 0 \end{pmatrix} \mathbf{U}^t$$

si hacemos $\alpha \rightarrow 0$

$$(\mathbf{A}^t \mathbf{A} + \alpha^2 \mathbf{I})^{-1} \mathbf{A}^t \rightarrow \mathbf{A}^+.$$

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 Technology, 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.