Álgebra Lineal Computacional

2do Cuatrimestre 2025

Laboratorio N° 4: Descomposición LU.

En este laboratorio nos enfocaremos en construir una que sea capaz de calcular la descomposición LU de una matriz, así como distintas funciones asociadas a este proceso. Revisen las diapositivas que acompañan para un ejemplo del cálculo de la descomposición LU paso a paso.

Ejercicio 1. Factorización LU

(a) Completar la función incluida en el script `elim_gaussiana.py` de `Python` de tal forma que en la iteración k-ésima del algoritmo se calcule la matriz $\widetilde{A}^{(k)}$ según se muestra a continuación para una matriz A de ejemplo.

$$A = \widetilde{A}^{(0)} = \begin{pmatrix} 2 & 1 & 2 & 3 \\ 4 & 3 & 3 & 4 \\ -2 & 2 & -4 & -12 \\ 4 & 1 & 8 & -3 \end{pmatrix} \rightsquigarrow \widetilde{A}^{(1)} = \begin{pmatrix} 2 & 1 & 2 & 3 \\ 2 & 1 & -1 & -2 \\ -1 & 3 & -2 & -9 \\ 2 & -1 & 4 & -9 \end{pmatrix} \rightsquigarrow$$

$$\rightsquigarrow \widetilde{A}^{(2)} = \begin{pmatrix} 2 & 1 & 2 & 3 \\ 2 & 1 & -1 & -2 \\ -1 & 3 & 1 & -3 \\ 2 & -1 & 3 & -11 \end{pmatrix} \rightsquigarrow \widetilde{A}^{(3)} = \begin{pmatrix} 2 & 1 & 2 & 3 \\ 2 & 1 & -1 & -2 \\ -1 & 3 & 1 & -3 \\ 2 & -1 & 3 & -2 \end{pmatrix}$$

De esta forma logramos optimizar espacio de almacenamiento ya que los coeficientes que formarán parte de la matriz L se almacenan en la parte triangular inferior de $\widetilde{A}^{(k)}$ (se muestran en rojo en el ejemplo). Notar que estos valores en rojo se corresponden a los valores en 0 que fuimos poniendo en cada paso de la triangulación de las $A^{(k)}$. Para $k = n-1$, se obtiene en la parte triangular superior de $\widetilde{A}^{(k)}$ a los coeficientes de U.

Luego,

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ -1 & 3 & 1 & 0 \\ 2 & -1 & 3 & 1 \end{pmatrix}, \qquad U = \begin{pmatrix} 2 & 1 & 2 & 3 \\ 0 & 1 & -1 & -2 \\ 0 & 0 & 1 & -3 \\ 0 & 0 & 0 & -2 \end{pmatrix}$$

Además la función de `Python` debe retornar la cantidad de operaciones aritméticas realizadas. Se deberá llevar un conteo de la cantidad de sumas, restas, multiplicaciones y divisiones que se realizan durante la triangulación.

(b) Pruebe la descomposición LU para matrices generadas al azar de $n \times n$ y grafique el error cometido al aproximar $A = LU$ conforme crece el número de operaciones. Para esto, realice un gráfico de $||A - LU||$ para una norma de su preferencia, en función de n. Realicen este gráfico en escala $\log - \log$ y estimen visualmente la pendiente de la misma. ¿Cómo crece el error con el número de operaciones?

Ejercicio 2. Estudiar la relación entre la cantidad de operaciones para encontrar la factorización LU y el tamaño de la matriz. Para ello, considerar como ejemplo a la siguiente matriz $\boldsymbol{B}_n = (b_{ij}) \in \mathbb{R}^{n\times n}, n \geq 2$, definida como

$$b_{ij} = \begin{cases} 1 & \text{si } i = j \text{ o } j = n, \\ -1 & \text{si } i > j, \\ 0 & \text{en otro caso.} \end{cases}$$

Ejemplo:

$$\boldsymbol{B}_2 = \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}, \boldsymbol{B}_3 = \begin{pmatrix} 1 & 0 & 1 \\ -1 & 1 & 1 \\ -1 & -1 & 1 \end{pmatrix}, \boldsymbol{B}_4 = \begin{pmatrix} 1 & 0 & 0 & 1 \\ -1 & 1 & 0 & 1 \\ -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & 1 \end{pmatrix}.$$

Siendo $\boldsymbol{B}_n = L_n U_n$ la descomposición LU de $\boldsymbol{B}_n$, verificar que $\|U_n\|_\infty = 2^{n-1}$. Elijan un rango de valores de n que les permita identificar una tendencia, y apóyense en el uso de escalas logarítmicas para ver cambios en órdenes de magnitud.

Ejercicio 3. Escribir funciones de `Python` que calculen la solución de un sistema:

(a) $\boldsymbol{Ly} = \boldsymbol{b}$, siendo $\boldsymbol{L}$ triangular inferior.

(b) $\boldsymbol{Ux} = \boldsymbol{y}$, siendo $\boldsymbol{U}$ triangular superior.

(c) Resolver un sistema $\boldsymbol{Ax} = \boldsymbol{b}$, utilizando las funciones de los ítems anteriores.

(d) Utilizar diferentes vectores aleatorios $\boldsymbol{b} \in \mathbb{R}^n$ para resolver el sistema $\boldsymbol{B}_n\boldsymbol{x} = \boldsymbol{b}$ como se indica en el Ejercicio 2.

Ejercicio 4 (Descomposición LDV)**.** Con restricciones similares a las necesarias para aplicar la factorización LU, es posible realizar una descomposición denominada LDV. En la descomposición LDV la matriz L es la misma que en la factorización LU, y la matrices D y V resultan de aplicar la descomposición LU de la matriz U^t. Gracias a que la matriz U es triangular superior e inversible, aplicar LU a U^t resulta en una matriz diagonal D cuyos elementos son iguales a la diagonal de U y una matriz V triangular superior tal que $V^tD = U^t$ o $DV = U$.

Más aún, cuando A es simétrica, resulta que $V = L^t$ y entonces $A = LDL^t$. Esta representación permite caracterizar a A en términos de su positividad. Decimos que A es Simétrica Definida Positiva (SDP) si $x^tAx > 0,\ \forall x \in \mathbb{R}^n, x \neq 0$ (de forma similar, SDN si $x^tAx < 0,\ \forall x \in \mathbb{R}^n, x \neq 0$).

a) Programe una función `calculaLDV(A)` que calcule la descomposición LDV de A usando como punto de partida la función que calcula LU.

b) Programe una función `esSDP(A,atol=1e-10)` que decida si A es simétrica definida positiva mediante la descomposición LDV. La función debe chequear que la matriz sea simétrica primero, y luego revisar si los elementos de la diagonal de D son todos positivos, garantizando que A sea definida positiva (*¿Por qué alcanza con verificar eso?*).

Ejercicios Extra:

Ejercicio 5 (Cholesky)**.** En los casos en los que A es SDP, se puede factorizar en la forma RR^t, donde R es una matriz triangular inferior tal que $R = LD^{\frac{1}{2}}$, donde $D^{\frac{1}{2}}$ es una matriz diagonal con las raices de los elementos de la diagonal de la matriz D, y L y D resultan de la factorización LDV de A.

a) Programe una función `calculaCholesky(A,atol=1e-10)` que verifique si la matriz es SDP y en caso afirmativo devuelva la matriz R asociada a A.

b) Una de las aplicaciones de la factorización de Cholesky es generar puntos con una dada correlación entre ellos. Para explorar esto, genere $Np = 1000$ vectores aleatorios de $\mathbb{R}^2$, que notamos (x_i, y_i) usando la función `numpy.random.norm`. Con los vectores generados, realice el siguiente análisis para las matrices A que se listan debajo:

   a) Grafique los vectores (x_i, y_i) generados.

   b) Grafique el resultado de aplicar la matriz R a los vectores previamente generados

   c) Calcule la varianza entre los vectores generados como $\sigma_x^2 = \sum_{i=1}^{Np} x_i^2/Np - \left(\sum_{i=1}^{Np} x_i/Np\right)^2$ y la covarianza : $\sigma_{xy} = \sum_{i=1}^{Np} x_i y_i/Np - \sum_{i=1}^{Np} x_i/Np \cdot \sum_{i=1}^{Np} y_i/Np$. Compare los elementos de las matrices A empleadas con los valores de σ_x^2, σ_y^2 y σ_{xy}.

   Matrices:

   - $A = \begin{pmatrix} 1 & .5 \\ .5 & 1 \end{pmatrix}$
   - $A = \begin{pmatrix} 1.05 & 1 \\ 1 & 1.05 \end{pmatrix}$
   - $A = \begin{pmatrix} 1 & -0.9 \\ -0.9 & 1 \end{pmatrix}$

   ¿Qué efecto producen los elementos positivos y negativos en la matriz? ¿Y aumentar la traza de la misma?

Módulo ALC

Para el módulo ALC, deben programar:

```
def calculaLU(A):
"""
Calcula la factorizacion LU de la matriz A y retorna las matrices L y U, junto con el numero de operaciones realizadas. En caso de que la matriz no pueda factorizarse retorna None.
"""

def res_tri(L,b,inferior=True):
"""
Resuelve el sistema Lx = b, donde L es triangular. Se puede indicar si es triangular inferior o superior usando el argumento inferior (por default asumir que es triangular inferior).
"""

def inversa(A):
"""
Calcula la inversa de A empleando la factorizacion LU
y las funciones que resuelven sistemas triangulares.
"""

def calculaLDV(A):
"""
Calcula la factorizacion LDV de la matriz A, de forma tal que A = LDV, con L triangular inferior, D diagonal y V triangular superior. En caso de que la matriz no pueda factorizarse retorna None.
"""

def esSDP(A, atol=1e-8):
"""
Checkea si la matriz A es simetrica definida positiva (SDP) usando la factorizacion LDV.
"""
```

Nota: no está permitodo el uso de la multiplicación matricial de numpy (`@`, `np.matmul`, etc)