

MÉTODOS DIRECTOS PARA SISTEMAS LINEALES

ALAN REYES-FIGUEROA
MÉTODOS NUMÉRICOS II

(AULA 06) 21.JULIO.2022

Sistemas Lineales

Recordemos los métodos para resolver un sistema de ecuaciones lineales $A\mathbf{x} = \mathbf{b}$, con $A \in \mathbb{R}^{n \times n}$, $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$.

Pensemos en la eliminación Gaussiana, por ejemplo. Si $A \in \mathbb{R}^{m \times n}$ lo que se quiere es (el algoritmo también puede aplicarse a matrices rectangulares, pero es más comúnmente usado para matrices cuadradas). La idea es transformar A en una matriz a su forma escalonada

$$\begin{bmatrix} 0 & * & \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ 0 & 0 & 0 & 0 & * & \bullet & \bullet & \bullet \\ \vdots & 0 & 0 & 0 & 0 & 0 & * & \bullet \\ \vdots & \vdots & \vdots & & \dots & & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Sistemas Lineales

En el caso de una matriz cuadrada A , transformamos A en una matriz triangular superior U introduciendo ceros debajo de la diagonal, primero en la columna 1, luego en la columna 2, y así sucesivamente.

$$\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \rightarrow \begin{bmatrix} \times & \times & \times & \times \\ \mathbf{0} & \times & \times & \times \\ \mathbf{0} & \times & \times & \times \\ \mathbf{0} & \times & \times & \times \end{bmatrix} \rightarrow \begin{bmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ \mathbf{0} & \times & \times & \\ \mathbf{0} & \times & \times & \end{bmatrix} \cdots \rightarrow \begin{bmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & & \times & \times \\ & & \mathbf{0} & \times \end{bmatrix}.$$

Esto se hace restando múltiplos de cada fila de las filas subsiguientes, en la forma

$$F_j = F_j - \ell_{jk} F_k, \quad k = j + 1, \dots, m.$$

Este proceso de “eliminación” es equivalente a multiplicar A por una secuencia de matrices L_k a la izquierda

Reducción Gaussiana

$$\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \xrightarrow{L_1} \begin{bmatrix} \times & \times & \times & \times \\ \mathbf{0} & \times & \times & \times \\ \mathbf{0} & \times & \times & \times \\ \mathbf{0} & \times & \times & \times \end{bmatrix} \xrightarrow{L_2} \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \mathbf{0} & \times & \times & \times \\ \mathbf{0} & \times & \times & \times \end{bmatrix} \xrightarrow{L_3} \begin{bmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & & \times & \times \\ & & \mathbf{0} & \times \end{bmatrix}.$$

$A \qquad L_1 A \qquad L_2 L_1 A \qquad L_3 L_2 L_1 A$

$$x_k = \begin{bmatrix} x_{1k} \\ \vdots \\ x_{kk} \\ x_{k+1,k} \\ \vdots \\ x_{mk} \end{bmatrix} \xrightarrow{L_k} L_k x_k = \begin{bmatrix} x_{1k} \\ \vdots \\ x_{kk} \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Para hacer esto, es necesario efectuar $F_j = F_j - \ell_{jk} F_k$, donde

$$\ell_{jk} = \frac{x_{jk}}{x_{kk}}, \quad j = k + 1, \dots, m.$$

Reducción Gaussiana

$$L_k = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -\ell_{k+1,k} & 1 & \\ & & \vdots & & \ddots \\ & & -\ell_{m,k} & & 1 \end{bmatrix}, \quad \ell_k = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \ell_{k+1,k} \\ \vdots \\ \ell_{m,k} \end{bmatrix}.$$

- Si $\ell_k = (0, \dots, 0, \ell_{k+1,k}, \dots, \ell_{m,k})^T$, entonces $L_k = I - \ell_k \mathbf{e}_k^T$. Observe que

$$(I - \ell_k \mathbf{e}_k^T)(I + \ell_k \mathbf{e}_k^T) = I - \ell_k (\mathbf{e}_k^T \ell_k) \mathbf{e}_k^T = I - \ell(\mathbf{o}) \mathbf{e}_k^T = I,$$

de modo que $I + \ell_k \mathbf{e}_k^T$ es la inversa de L_k .

- $L_k^{-1} L_{k+1}^{-1} = (I + \ell_k \mathbf{e}_k^T)(I + \ell_{k+1} \mathbf{e}_{k+1}^T) = I + \ell_k \mathbf{e}_k^T + \ell_{k+1} \mathbf{e}_{k+1}^T$,
y podemos combinar varias de estas matrices en una sola.

Reducción Gaussiana

$$L = L_1^{-1} L_2^{-1} \cdots L_{m-1}^{-1} = \begin{bmatrix} 1 & & & & \\ \ell_{21} & 1 & & & \\ \ell_{31} & \ell_{32} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ \ell_{m1} & \ell_{m2} & \cdots & \ell_{m,m-1} & 1 \end{bmatrix}.$$

Repitiendo este proceso a todas las columnas de A obtenemos

$$\underbrace{L_{n-1} \cdots L_3 L_2 L_1}_{L^{-1}} A = U.$$

Haciendo $L = L_1^{-1} L_2^{-1} \cdots L_{n-1}^{-1}$, da $A = LU$, y obtenemos una factoración LU de A .

Reducción Gaussiana

Definición

Una **factoración LU** de A , es una factoración en la forma $A = LU$, donde

- U es triangular superior,
- L es triangular inferior, con entradas diagonales iguales a 1.

Hay una relación directa entre la eliminación gaussiana la factoración LU .

Algoritmo: (Eliminación Gaussiana).

Inputs: $A \in \mathbb{R}^{m \times n}$, Outputs: $L \in \mathbb{R}^{m \times m}$, $U \in \mathbb{R}^{m \times n}$.

Initialize $U = A$, $L = I$

for $k = 1$ to $n - 1$:

 for $j = k + 1$ to m :

$$L_{j,k} = 1/U_{j,k}$$

$$U_{j,k:n} = U_{j,k:n} - L_{j,k} U_{j,k:n}.$$

Reducción Gaussiana

Número de operaciones:

Paso	Operaciones
Initialize $U = A, L = I$ for $k = 1$ to $m - 1$: for $j = k + 1$ to m : $L_{j,k} = 1/U_{j,k}$ $U_{j,k:n} = U_{j,k:n} - L_{j,k:n} U_{j,k:n}$	$2n^2$ $m - 1$ ciclos $m - k$ ciclos 2 $3(n - k + 1)$

$$\begin{aligned}\#Operaciones &= 2n^2 + \sum_{k=1}^{n-1} \sum_{j=k+1}^n (2 + 3(n - k + 1)) \\ &= 2n^2 + \sum_{k=1}^{n-1} \left[(n - k)(2 + 3(n - k + 1)) \right] = O\left(\frac{1}{3}n^3\right).\end{aligned}$$

Reducción Gaussiana

Ejemplo: Para la matriz $A = \begin{pmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{pmatrix}$

$$L_1 A = \begin{pmatrix} 1 & & & \\ -2 & 1 & & \\ -4 & & 1 & \\ -3 & & & 1 \end{pmatrix} \begin{pmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{pmatrix} = \begin{pmatrix} 2 & 1 & 1 & 0 \\ \mathbf{0} & 1 & 1 & 1 \\ \mathbf{0} & 3 & 5 & 5 \\ \mathbf{0} & 4 & 6 & 8 \end{pmatrix} = U,$$

$$L_2 L_1 A = \begin{pmatrix} 1 & & & \\ & 1 & & \\ -3 & & 1 & \\ -4 & & & 1 \end{pmatrix} \begin{pmatrix} 1 & & & \\ -2 & 1 & & \\ -4 & & 1 & \\ -3 & & & 1 \end{pmatrix} \begin{pmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{pmatrix} = \begin{pmatrix} 2 & 1 & 1 & 0 \\ \mathbf{0} & 1 & 1 & 1 \\ \mathbf{0} & \mathbf{0} & 2 & 2 \\ \mathbf{0} & \mathbf{0} & 2 & 4 \end{pmatrix} = U,$$

$$L_3 L_2 L_1 A = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ -1 & & & 1 \end{pmatrix} \begin{pmatrix} 1 & & & \\ & 1 & & \\ -3 & & 1 & \\ -4 & & & 1 \end{pmatrix} \begin{pmatrix} 1 & & & \\ -2 & 1 & & \\ -4 & & 1 & \\ -3 & & & 1 \end{pmatrix} \begin{pmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{pmatrix} = \begin{pmatrix} 2 & 1 & 1 & 0 \\ \mathbf{0} & 1 & 1 & 1 \\ \mathbf{0} & \mathbf{0} & 2 & 2 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & 2 \end{pmatrix} = U.$$

$$\text{Luego, } L = \begin{pmatrix} 1 & & & \\ 2 & 1 & & \\ 4 & 3 & 1 & \\ 3 & 4 & 1 & 1 \end{pmatrix}, \quad U = \begin{pmatrix} 2 & 1 & 1 & 0 \\ \mathbf{0} & 1 & 1 & 1 \\ \mathbf{0} & \mathbf{0} & 2 & 2 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & 2 \end{pmatrix}.$$

Observaciones:

- En su forma más general, una factoración LU de $A \in \mathbb{R}^{m \times n}$ satisface que $U \in \mathbb{R}^{m \times n}$ está en forma escalonada, mientras que $L \in \mathbb{R}^{m \times m}$ es triangular inferior, con entradas 0 ó 1 en la diagonal.
- En el caso que $A \in \mathbb{R}^{n \times n}$ es cuadrada, obtenemos U triangular superior.
- No todas las matrices admiten una factoración LU (ya que el algoritmo de eliminación gaussiana puede fallar), el siguiente resultado ilustra esto.

Reducción Gaussiana

Teorema (Factoración LU)

- i) Una matriz $A \in \mathbb{R}^{m \times n}$ admite una factoración LU si, y sólo si, el algoritmo de eliminación gaussiana sin cambios de fila, conduce a una reducción de A en forma escalonada. En ese caso, L será triangular inferior con 0 ó 1 en la diagonal principal, y U tendrá forma escalonada. En el caso $m = n$, U será triangular superior.
- ii) La matriz L en dicha factoración LU tendrá entradas 1 en la diagonal principal, si y sólo si, durante el algoritmo de reducción gaussiana sin cambio de renglones, en cada paso del algoritmo todas las entradas $U_{j,j}$ son no nulas, para $k = 1, 2, \dots, m - 1$. \square

Limitantes del algoritmo de eliminación gaussiana:

- Si $U_{j,j} = 0$, habrá que efectuar un cambio de filas.
- Si todas las entradas de la columna j , $U_{j,j:m}$ son nulas, entonces hay que saltarse el paso en esa columna (forma escalonada con variables inactivas).

Reducción Gaussiana

Eliminación gaussiana con intercambio de renglones:

En el caso general, es posible que en algún momento del algoritmo gaussiano tengamos $U_{kk} = 0$, y no podamos efectuar la división en el pivote actual. En ese caso es necesario efectuar un cambio de renglones.

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & \mathbf{x_{ik}} & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ & \times & \times & \times & \times \end{bmatrix} \xrightarrow{P_1} \begin{bmatrix} \times & \times & \times & \times & \times \\ & \mathbf{x_{ik}} & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ & \times & \times & \times & \times \\ & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ & \times & \times & \times & \times \end{bmatrix} \xrightarrow{L_1} \begin{bmatrix} \times & \times & \times & \times & \times \\ & \mathbf{x_{ik}} & \times & \times & \times \\ & 0 & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ & 0 & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ & 0 & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \end{bmatrix}.$$

Row interchange Elimination

Este también se expresa como el producto por una matriz. El pivote parcial implica aplicar una matriz de permutación P_k a la izquierda de la matriz de trabajo, previo a cada eliminación.

Reducción Gaussiana

Al final, obtenemo una descomposición de la forma

$$L_{m-1}P_{m-1} \cdots L_3P_3L_2P_2L_1P_1A = U.$$

Esta factoración puede reescribirse reordenando los términos de forma que las matrices triangulares L_j aparecen juntas, y las matrices de permutación aparecen juntas

$$L'_{m-1} \cdot L'_3L'_2L'_1P_{m-1} \cdots P_3P_2P_1A = U,$$

donde $L'_{m-1} = L_{m-1}$, pero reordenando sus filas, según la relación

$$L'_j = P_{m-1} \cdots P_{j+1}L_jP_{j+1}^{-1} \cdots P_{m-1}^{-1}, \quad j = 1, \dots, m-1.$$

Escribiendo $L = (L'_{m-1} \cdots L'_2L'_1)^{-1}$ y $P = P_{m-1} \cdots P_2P_1$ obtenemos que $L^{-1}PA = U$, o equivalentemente $PA = LU$.

Teorema (Factoración $PA = LU$)

- i) *Toda matriz $A \in \mathbb{R}^{m \times n}$ admite una factoración $PA = LU$ si, y sólo si, el algoritmo de eliminación gaussiana con cambios de fila, conduce a una reducción de A en forma escalonada. En ese caso, L será triangular inferior con 0 ó 1 en la diagonal principal, y U tendrá forma escalonada. En el caso $m = n$, U será triangular superior.*
- ii) *En el caso $m = n$, la matriz L en dicha factoración LU tendrá entradas 1 en la diagonal principal, si y sólo si, todas las variables son activas. En ese caso, U es triangular elementos diagonales no nulos. \square*

Obs: Debido a la búsqueda e intercambio de renglones, la complejidad se incrementa de $O(\frac{n^3}{3})$ a $O(n^3)$.

Reducción Gaussiana

Algoritmo: (Eliminación Gaussiana con intercambio de renglones).

Inputs: $A \in \mathbb{R}^{m \times n}$, *Outputs:* $L, P \in \mathbb{R}^{m \times m}$, $U \in \mathbb{R}^{m \times n}$.

Initialize $U = A$, $L = I$, $P = I$.

for $k = 1$ to $n - 1$:

 Choose $k < i \leq m$ such that $u_{ik} \neq 0$:

 Interchange $u_{k,k:n} \leftrightarrow u_{i,k:n}$

 Interchange $\ell_{k,1:k-1} \leftrightarrow \ell_{i,1:k-1}$

 Interchange $p_{k,:} \leftrightarrow p_{i,:}$

 for $j = k + 1$ to m :

$$L_{j,k} = 1/U_{j,k}$$

$$U_{j,k:n} = U_{j,k:n} - L_{j,k} U_{j,k:n}.$$

1. Cálculo del Determinante:

Cuando $m = n$, la reducción gaussiana proporciona un algoritmo eficiente para el cálculo del determinante de una matriz.

Comparación con otros métodos para el determinante:

- Fórmula de Leibniz (desarrollo por permutaciones): $O(n!)$.
- Fórmula de Lagrange (desarrollo por cofactores): $O(n!)$.
- Eliminación gaussiana: $O(n^3)$.

Observaciones:

- En el caso $A = LU$, $\det A$ se calcula como $D = \prod_{j=1}^n u_{jj}$.
- En el caso $PA = LU$, al D anterior hay que multiplicarle el signo de la matriz de permutación P .
- Para matrices grandes ($n > 5$), es más eficiente.

2. Solución de sistemas lineales:

Para resolver un sistema $A\mathbf{x} = \mathbf{b}$, con $A \in \mathbb{R}^{n \times n}$, $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$.

En este caso, es necesario aplicar el algoritmo de eliminación gaussiana a la matriz extendida $[A \mid \mathbf{b}]$ para llevarla a la forma $[U \mid \mathbf{b}']$. Además, luego del algoritmo de eliminación gaussiana, es necesario aplicar el método de *sustitución hacia atrás*.

Algoritmo: (Sustitución hacia atrás).

Inputs: $U \in \mathbb{R}^{n \times n}$, $\mathbf{b}' \in \mathbb{R}^n$ (U la forma escalonada de A , \mathbf{b}' la reducción gaussiana de \mathbf{b})

Outputs: $\mathbf{x} \in \mathbb{R}^n$.

for $j = n$ to 1:

$$x_j = (b_j - \sum_{k=j+1}^n u_{jk}x_k) / u_{jj}$$

Aplicaciones

Otras alternativas para resolver el sistema $A\mathbf{x} = \mathbf{b}$, $A \in \mathbb{R}^{n \times n}$, $\mathbf{b} \in \mathbb{R}^n$, son:

Descomposición $A = LU$ ó $PA = LU$

Ejemplificamos el caso $A = LU$. Como $A = LU$, el sistema se escribe $LU\mathbf{x} = L\mathbf{y} = \mathbf{b}$, donde $U\mathbf{x} = \mathbf{y}$.

Resolvemos el sistema $A\mathbf{x} = \mathbf{b}$ en dos pasos:

1. Resolver $L\mathbf{y} = \mathbf{b}$ por sustitución hacia atrás.
2. Resolver $U\mathbf{x} = \mathbf{y}$ por sustitución hacia atrás.

En el caso de $PA = LU$, tenemos $A = P^T LU$. Resolvemos el sistema $A\mathbf{x} = \mathbf{b}$ en tres pasos:

1. Resolver $\mathbf{z} = P\mathbf{b}$ directamente (permutamos las entradas de \mathbf{b}).
2. Resolver $L\mathbf{y} = \mathbf{z}$ por sustitución hacia atrás.
3. Resolver $U\mathbf{x} = \mathbf{y}$ por sustitución hacia atrás.

Método de Gauss-Jordan

El método de Gauss-Jordan es una extensión de la eliminación gaussiana. Aquí el objetivo es continuar aplicando la reducción gaussiana, hasta llevar la matriz A a una forma escalonada en la que todos los elementos pivote son 1, y el resto de las entradas son 0.

$$\begin{bmatrix} 0 & 1 & & & & & & \\ 0 & 0 & 0 & 0 & 1 & & & \\ \vdots & 0 & 0 & 0 & 0 & 0 & 1 & \\ \vdots & \vdots & \vdots & & \dots & & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Algoritmo: (Gauss-Jordan).

Inputs: $A \in \mathbb{R}^{m \times n}$, *Outputs:* $L \in \mathbb{R}^{m \times m}$, $U \in \mathbb{R}^{m \times n}$.

Initialize $U = A$, $L = I$

for $k = 1$ to $n - 1$:

 for $j = k$ to m :

$$L_{j,k} = 1/U_{j,k}$$

$$U_{j,k:n} = U_{j,k:n} - L_{j,k} U_{j,k:n}.$$

for $k = n$ to 2 :

 for $j = 1$ to $k - 1$:

$$L_{j,k} = 1/U_{j,k}$$

$$U_{j,1:k} = U_{j,1:k} - L_{j,k} U_{j,1:k}.$$

Obs! En este caso la complejidad se duplica a $O(2n^3)$.

3. Solución de sistemas lineales:

Para resolver un sistema $A\mathbf{x} = \mathbf{b}$, $A \in \mathbb{R}^{n \times n}$, $\mathbf{b} \in \mathbb{R}^n$, aplicamos el método de Gauss-Jordan a la matriz extendida $[A \mid \mathbf{b}]$ para obtener $[I \mid \mathbf{x}]$.

En este caso, la solución del sistema es inmediata, y ocurre en la última columna de la respuesta.

4. Inversa de una matriz:

Podemos utilizar Gauss-Jordan para calcular la inversa de una matriz no singular $A \in \mathbb{R}^{n \times n}$.

Para ello, aplicamos el algoritmo a la matriz extendida $[A \mid I] \in \mathbb{R}^{n \times 2n}$. Al final del algoritmo, obtenemos la salida $[I \mid A^{-1}]$.

En este caso, la inversa ocurre en la segunda parte de la respuesta.

Obs: Para n grande ($n > 5$), este método es más eficiente.

Estrategias de Pivoteo

Las técnicas de pivoteo son un método que se aplica a la reducción gaussiana desde la década de 1950. Sirven para reducir el error relativo de los cálculos.

La técnica consiste en elegir “adecuadamente” el pivote x_{kk} en cada paso $k = 1, 2, \dots, n - 1$ de la reducción gaussiana.

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ & \mathbf{x_{kk}} & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \end{bmatrix} \rightarrow \begin{bmatrix} \times & \times & \times & \times & \times \\ & \mathbf{x_{kk}} & \times & \times & \times \\ & \mathbf{0} & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ & \mathbf{0} & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ & \mathbf{0} & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \end{bmatrix}.$$

Existen varias técnicas de pivoteo. Las más usadas son las siguientes:

- pivoteo parcial,
- pivoteo parcial con escalado de columna,
- pivoteo completo.

Estrategias de Pivoteo

Típicamente, en todas las estrategias de pivoteo se elige la entrada máxima (en módulo) sobre un cierto subconjunto de entradas posibles.

En el **pivoteo parcial**, en cada paso elegimos la fila i , $k \leq i \leq m$ tal que $|u_{ik}|$ es máximo.

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ \boxed{x_{kk}} & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \rightarrow \begin{bmatrix} \times & \times & \times & \times & \times \\ & x_{kk} & \times & \times & \times \\ & 0 & \times & \times & \times \\ & 0 & \times & \times & \times \\ & 0 & \times & \times & \times \end{bmatrix}.$$

- En cada paso hacemos una búsqueda sobre $O(m - k)$ valores posibles, lo que aumenta a $O(n^3)$ la complejidad de la eliminación gaussiana.
- Es similar al de reducción gaussiana con intercambio de filas.

Estrategias de Pivoteo

En el **pivoteo completo**, en cada paso elegimos la fila i y la columna j , con $k \leq i \leq m$, $k \leq j \leq n$ tales que $|u_{ij}|$ es máximo.

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ \mathbf{x_{kk}} & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \rightarrow \begin{bmatrix} \times & \times & \times & \times & \times \\ & x_{kk} & \times & \times & \times \\ 0 & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ 0 & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ 0 & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \end{bmatrix}.$$

- En cada paso hacemos una búsqueda sobre $O((m - k) \cdot (n - k))$ valores posibles, lo que aumenta todavía más la complejidad.
- En la práctica no es muy usado, ya el aumento en el costo computacional es mayor que el beneficio.

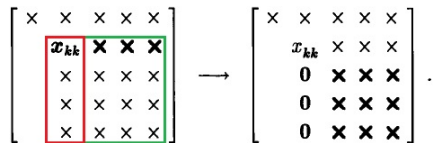
Estrategias de Pivoteo

En su lugar se usa el **pivoteo parcial con reescalado de columna**.

Primero, para cada $k \leq i \leq m$, definimos un factor de escala s_i como

$$s_i = \max_{k \leq j \leq n} |u_{ij}|, \quad k \leq i \leq m.$$

Luego, elegimos la fila i , con $k \leq i \leq m$, $k \leq j \leq n$ tal que $\frac{|u_{ik}|}{s_i}$ es máximo.


$$\begin{bmatrix} \times & \times & \times & \times & \times \\ \mathbf{x_{kk}} & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{bmatrix} \rightarrow \begin{bmatrix} \times & \times & \times & \times & \times \\ x_{kk} & \times & \times & \times & \times \\ 0 & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ 0 & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ 0 & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \end{bmatrix}.$$

- La idea es elegir sólo sobre la fila i , tomando en cuenta las escalas de los valores en todo el bloque derecho inferior de la matriz U .
- Si algún $|s_i| = 0$, el sistema no tiene solución única.

Estrategias de Pivoteo

En el **pivoteo completo**, en cada paso elegimos la fila i y la columna j , con $k \leq i \leq m$, $k \leq j \leq n$ tales que $|u_{ij}|$ es máximo.

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ \mathbf{x_{kk}} & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \rightarrow \begin{bmatrix} \times & \times & \times & \times & \times \\ & x_{kk} & \times & \times & \times \\ 0 & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ 0 & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \\ 0 & \mathbf{\times} & \mathbf{\times} & \mathbf{\times} \end{bmatrix}.$$

- En cada paso hacemos una búsqueda sobre $O((m - k) \cdot (n - k))$ valores posibles, lo que aumenta todavía más la complejidad.
- En la práctica no es muy usado, ya el aumento en el costo computacional es mayor que el beneficio.

Estrategias de Pivoteo

Ejemplo:

$$\begin{array}{rclcl} 0.00300X_1 & + & 59.14X_2 & = & 59.17 \\ 5.291X_1 & - & 6.130X_2 & = & 46.78 \end{array}$$