Tarea 1

Esteban Leiva

Enero 2024

1. a) Muestre que la inversa de una matriz triangular inferior también es triangular inferior.

   Primero vamos a probar los siguientes resultados:

   - Si $L, T \in \mathbb{R}^{n\times n}$ son matrices triangulares inferiores entonces $(LT)_{ii} = L_{ii}T_{ii}$ para $i = 1, ..., n$.

$$\begin{aligned}(LT)_{ii} &= \sum_{k=1}^{n} l_{ik}t_{ki} \\ &= \sum_{k=1}^{i} l_{ik}t_{ki} + \sum_{k=i+1}^{n} l_{ik}t_{ki} \\ &= \sum_{k=1}^{i} l_{ik}t_{ki} \\ &= \sum_{k=1}^{i-1} l_{ik}t_{ki} + l_{ii}t_{ii} \\ &= l_{ii}t_{ii} = L_{ii}T_{ii}\end{aligned}$$

   - Si una matriz triangular inferior $L \in \mathbb{R}^{n\times n}$ es invertible entonces sus elementos diagonales no son nulos.

     Sabemos que si la invertibilidad es equivalente a que para todo $b \in \mathbb{R}^n$ existe exactamente un $x \in \mathbb{R}^n$ tal que $Lx = b$. Entonces, tome un $b \in \mathbb{R}^n$ arbitrario. Vamos a construir el x tal que $Lx = b$ y mostraremos que si la diagonal es nula, no es posible que para todo $b \in \mathbb{R}^n$ existe exactamente un $x \in \mathbb{R}^n$ tal que $Lx = b$.

$$\begin{aligned} b_i &= \sum_{j=1}^{n} l_{ij}x_j \\ &= \sum_{j=1}^{i} l_{ij}x_j + \sum_{j=i+1}^{n} l_{ij}x_j \\ &= \sum_{j=1}^{i} l_{ij}x_j \end{aligned}$$

Entonces, $b_1 = \sum_{j=1}^{1} l_{1j}x_j = l_{11}x_1$. Si $l_{11} = 0$ entonces $b_1 = 0$ y no se podría encontrar un b con $b_1 \neq 0$ como $Lx = b$; por lo tanto, $l_{11} \neq 0$. Además, x_1 está únicamente determinado pues $x_1 = \frac{b_1}{l_{11}}$.

Ahora, considere $b_2 = \sum_{j=1}^{2} l_{2j}x_j = l_{21}x_1 + l_{22}x_2 = l_{21}\frac{b_1}{l_{11}} + l_{22}x_2$. Si $l_{22} = 0$ entonces $b_2 = l_{21}\frac{b_1}{l_{11}}$ y no se podría obtener para un valor diferente de b_2 contrario a lo que necesitamos; luego, $l_{22} \neq 0$. Además, x_2 está únicamente determinado pues $x_2 = \frac{b_2}{l_{22}} - \frac{l_{21}b_1}{l_{22}l_{11}}$.

Esta construcción continua hasta n y llegamos a que para $i = 1, ..., n$ tenemos $l_{ii} \neq 0$ como queríamos.

- Sean $\{e_k, k = 1, ..., n\}$ la base canónica de $\mathbb{R}^n$, L una matriz triangular inferior invertible y x un vector de $\mathbb{R}^n$ tal que $Lx = e_k$. entonces $x_j = 0$ para $j < k$.

  Si $y = e_k = Lx$ y $1 < k < n$ (los casos en donde $1 = k$ y $n = k$ son complementamente análogos al procedimiento siguiente)

$$\begin{aligned} y_i &= \sum_{j=1}^{n} l_{ij}x_j \\ &= \sum_{j=1}^{i} l_{ij}x_j + \sum_{j=i+1}^{n} l_{ij}x_j \\ &= \sum_{j=1}^{i} l_{ij}x_j \end{aligned}$$

Entonces,

$$y_1 = \sum_{j=1}^{1} l_{1j}x_j = l_{11}x_1$$

$$\vdots$$

$$y_k = \sum_{j=1}^{k} l_{kj}x_j = l_{k1}x_1 + \cdots + l_{kk}x_k$$

$$\vdots$$

$$y_n = \sum_{j=1}^{n} l_{nj}x_j = l_{n1}x_1 + \cdots + l_{nk}x_n$$

Como L es invertible, su diagonal no es nula. Como $y_1 = 0$, $l_{11}x_1 = 0$ entonces $x_1 = 0$. Ahora considere $y_2 = l_{21}x_1 + l_{21}x_2 = 0 = l_{21}x_2$ entonces $x_2 = 0$. Hacemos este procedimiento hasta $k-1$ y obtenemos que para $j < k$ $x_j = 0$ como queríamos.

Ahora, estamos listos para probar que si L es triangular inferior invertible entonces L^{-1} es triangular inferior.

$$LL^{-1} = I = [e_1 \ \cdots \ e_n]$$
$$LL^{-1} = L[y_1 \ \cdots \ y_n] = [Ly_1 \ \cdots \ Ly_n]$$

Entonces, $Ly_k = e_k$ y por el tercer ítem, la k-ésima columna y_k tiene entradas igual a 0 en las filas antes de la k-ésima fila. Por lo tanto, L^{-1} es triangular inferior.

b) Escriba un programa que calcule la inversa de una matriz triangular inferior. Compruebe la eficiencia de su programa con la matriz $a_{ij} = (i+j)2$ con $i \geq j$ y 0 de lo contrario. Para tamaños $n = 2^k$ con $k = 2, ..., 15$ haga el producto de la matriz con su inversa y reporte la diferencia con la matriz identidad. Haga una gráfica `loglog` del tiempo requerido contra el tamaño de la matriz.

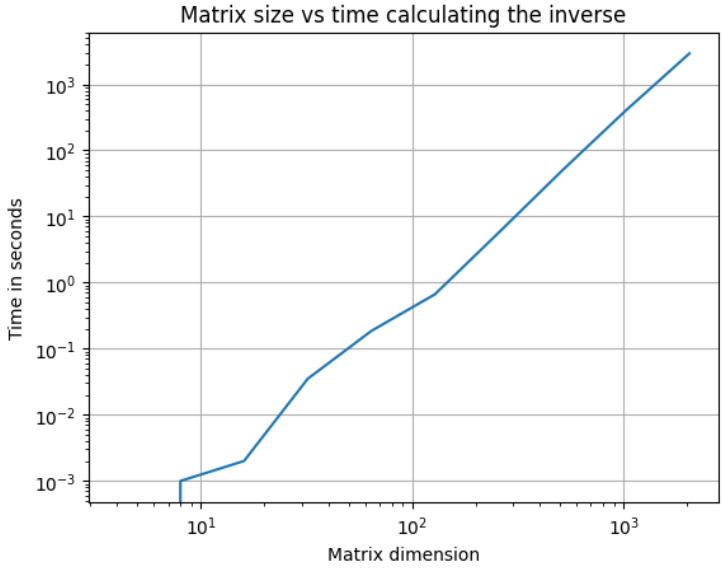

Figure 1: Gráfica loglog.

```
Size: 4
Time: 0.0
Difference: 2.7755575615628914e-16
Size: 8
Time: 0.0009932518005371094
Difference: 7.068998164605489e-16
Size: 16
Time: 0.001993894577026367
Difference: 1.5629858518551032e-15
Size: 32
Time: 0.034906625747680664
Difference: 6.38378239159465e-15
Size: 64
Time: 0.18502092361450195
Difference: 2.1835127022445944e-14
Size: 128
Time: 0.6642682552337646
Difference: 6.386559304408929e-14
Size: 256
Time: 5.662587642669678
Difference: 3.187470801896148e-13
Size: 512
Time: 49.54169464111328
Difference: 8.170745305339551e-13
Size: 1024
Time: 400.9723160266876
Difference: 2.533213169476541e-12
Size: 2048
Time: 2927.5631663799286
Difference: 8.196455411001706e-12
```

Figure 2: Diferencia y tiempos

2. ¿Cuántas multiplicaciones y divisiones son necesarias para resolver un sistema de ecuaciones $Ax = b$ haciendo primero la factorización LU.

Algorithm 1 Factorización LU

0: **for** $k = 1, 2, \ldots, n$ **do**
0: $l_{kk}u_{kk} = a_{kk} - \sum_{s=1}^{k-1} l_{ks}u_{sk}$
0: **for** $j = k + 1 \ldots, n$ **do**
0: $u_{kj} \leftarrow \left(a_{kj} - \sum_{s=1}^{k-1} l_{ks}u_{sj}\right) \div l_{kk}$
0: **end for**
0: **for** $i = k + 1 \ldots, n$ **do**
0: $l_{ik} \leftarrow \left(a_{ik} - \sum_{s=1}^{k-1} l_{is}u_{sk}\right) \div u_{kk}$
0: **end for**
0: **end for**=0

(a) En $l_{kk}u_{kk} = a_{kk} - \sum_{s=1}^{k-1} l_{ks}u_{sk}$ realizamos $(k-1)$ multiplicaciones.

(b) En $u_{kj} \leftarrow \left(a_{kj} - \sum_{s=1}^{k-1} l_{ks}u_{sj}\right) \div l_{kk}$ realizamos $(k-1)(n-k)$ multiplicaciones y $(n-k)$ divisiones.

(c) En $l_{ik} \leftarrow \left(a_{ik} - \sum_{s=1}^{k-1} l_{is}u_{sk}\right) \div u_{kk}$ realizamos $(k-1)(n-k)$ multiplicaciones y $(n-k)$ divisiones.

Entonces en la factorización realizamos

$$\sum_{k=1}^{n}(k-1) + 2(k-1)(n-k) + 2(n-k) = \sum_{k=1}^{n}(k-1) + 2(n-k)[(k-1)+1]$$

$$= \sum_{k=1}^{n}(k-1) + 2(n-k)[k] = \frac{n^2}{2} - \frac{n}{2} + \sum_{k=1}^{n} 2(n-k)[k]$$

$$= \frac{n^2}{2} - \frac{n}{2} + 2n\sum_{k=1}^{n} k - 2\sum_{k=1}^{n} k^2$$

$$= \frac{n^2}{2} - \frac{n}{2} + \frac{2n^2(n+1)}{2} - \frac{n(n+1)(2n+1)}{6}$$

multiplicaciones y divisiones.

Si tenemos la factorización LU tenemos que primero hacer $y = Ux$ y después $Ly = b$. Entonces, primero debemos hacer una multiplicación matricial en la que se deben realizar n^2 multiplicaciones. Luego, tenemos que resolver $Ly = b$ con el algoritmo forward.

Algorithm 2 Forward

0: **for** $i = 1, 2, \ldots, n$ **do**
0: $y_i \leftarrow b_i$
0: **for** $j = 1, 2, \ldots, i-1$ **do**
0: $y_i \leftarrow y_i - l_{ij} \cdot y_j$
0: **end for**
0: $y_i \leftarrow y_i \div l_{ii}$
0: **end for**=0

Para cada i el algoritmo realiza $(i-1)$ multiplicaciones. Entonces, el numero de multiplicaciones es

$$\sum_{i=1}^{n}(i-1) = \sum_{i=1}^{n} i - \sum_{i=1}^{n} 1 = \frac{n(n+1)}{2} - n = \frac{n^2}{2} - \frac{n}{2}$$

y el número de divisiones es n porque está dentro de un solo for loop de 1 a n.

Por lo tanto, el total de multiplicaciones y divisiones después de hacer la factorización LU es $\frac{3n^2}{2} + \frac{n}{2}$. Si sumamos estos dos pasos llegamos a que el total es $\frac{n^2}{2} - \frac{n}{2} + n^2(n+1) - \frac{n(n+1)(2n+1)}{6} + \frac{3n^2}{2} + \frac{n}{2} = 2n^2 + n^2(n+1) - \frac{n(n+1)(2n+1)}{6}$. Entonces es $O(n^3)$.