



**DEPARTAMENTO  
DE COMPUTACION**

Facultad de Ciencias Exactas y Naturales - UBA

# Trabajo Práctico 1

## Con 15 $\theta$ s discretizo alto horno...

Jueves 3 de septiembre de 2015

Métodos Numéricos

Integrante	LU	Correo electrónico
Lascano, Nahuel	476/11	laski.nahuel@gmail.com
Vileriño, Silvio	106/12	svilerino@gmail.com

En este trabajo aplicamos dos métodos de resolución de sistemas de ecuaciones lineales (Factorización LU y Eliminación Gaussiana) para el cálculo de isotermas de una corona circular, dadas las temperaturas de las circunferencias interior y exterior.

Pudimos verificar experimentalmente que el método de factorización LU resulta más eficiente si se tienen varias posibles soluciones para una misma matriz, pero que para una única instancia es conveniente usar la eliminación gaussiana.

Palabras clave: factorización LU, eliminación gaussiana, sistemas de ecuaciones lineales, matriz banda



**Facultad de Ciencias Exactas y Naturales**  
Universidad de Buenos Aires  
Ciudad Universitaria - (Pabellon I/Planta Baja)  
Intendente Güiraldes 2160 - C1428EGA  
Ciudad Autónoma de Buenos Aires - Rep. Argentina  
Tel/Fax: (54 11) 4576-3359  
<http://www.fcen.uba.ar>

# Índice

<b>1. Introducción teórica</b>	<b>2</b>
<b>2. Desarrollo</b>	<b>3</b>
2.1. Detalle de los algoritmos utilizados . . . . .	3
2.1.1. Eliminación Gaussiana . . . . .	3
2.1.2. Backward substitution . . . . .	3
2.1.3. Forward substitution . . . . .	4
2.1.4. Factorización LU . . . . .	4
2.2. Discretización . . . . .	5
2.3. Armado del sistema de ecuaciones . . . . .	5
2.4. Resolución del sistema de ecuaciones . . . . .	7
2.4.1. Caracterización de la matriz . . . . .	7
2.4.2. Resolución sin pivoteo . . . . .	8
2.5. M étodología de experimentaci ón . . . . .	9
2.5.1. Consideraciones iniciales . . . . .	9
2.5.2. Generaci ón de casos de prueba . . . . .	9
2.5.3. M étricas de performance . . . . .	10
2.5.4. M étodo de posicionamiento estimado de la isoterma . . . . .	10
2.5.5. M étricas de seguridad de la isoterma . . . . .	10
2.6. Esquema de experimentaci ón . . . . .	11
2.6.1. Experimentos focalizados en la performance . . . . .	11
2.6.2. Evoluci ón de la temperatura y posici ón de la isoterma con distintas discretizaciones	12
2.6.3. Evoluci ón de la posici ón de la isoterma con distintas discretizaciones y temperaturas externas . . . . .	14
<b>3. Resultados</b>	<b>15</b>
3.1. Performance . . . . .	15
3.1.1. Performance de los m etodos en funci ón de la discretizaci ón para una instancia	15
3.1.2. Performance de EG vs LU variando la cantidad de instancias y la granularidad de la discretizaci ón . . . . .	15
3.1.3. Evoluci ón estimaci ón de la isoterma y temperatura . . . . .	17
3.1.4. Estimaci ón de estabilidad de la pared del horno . . . . .	22
<b>4. Conclusiones</b>	<b>25</b>
<b>5. Trabajo Futuro</b>	<b>25</b>
<b>6. Apéndice</b>	<b>26</b>
6.1. Apéndice A: Enunciado . . . . .	26
6.2. Apéndice B: Código relevante . . . . .	30

## 1. Introducción teórica

En el presente trabajo se intenta evaluar computacionalmente la seguridad térmica de un horno circular. El problema presentado consiste en estimar el riesgo que corre el mismo de fracturarse por efecto de la elevada temperatura. Dicho de otro modo, dadas las temperaturas de las paredes internas y externas del horno (obtenidas a través de sensores) se quiere estimar la ubicación de la isotermia de 500°C, cuya cercanía a la pared externa es un indicador de la peligrosidad de la estructura.

Dado un corte transversal del horno podemos definir  $r_i$  y  $r_e$  como los radios de la pared interna y externa, ambos circulares. Nos interesa analizar la temperatura de los puntos entre ellas. Para referirnos a los puntos de dicha corona circular utilizaremos coordenadas polares, por lo que cada punto  $P$  quedará definido por un radio  $r$  y un ángulo  $\theta$ . Si llamamos  $T(r, \theta)$  a la temperatura del punto  $P_{r,\theta}$ , podemos utilizar la ecuación del calor de Laplace para encontrar el estado de equilibrio del sistema:

$$\frac{\partial^2 T(r, \theta)}{\partial r^2} + \frac{1}{r} \frac{\partial T(r, \theta)}{\partial r} + \frac{1}{r^2} \frac{\partial^2 T(r, \theta)}{\partial \theta^2} = 0 \quad (1)$$

la cual debe cumplirse para todos los puntos internos del horno.

Para aproximar dicha ecuación usando sistemas lineales, usamos una discretización de los puntos que nos interesa analizar (todos los pertenecientes a la pared del horno, que forman una corona circular) y discretizamos asimismo la ecuación 1. De esa manera arribamos a un sistema de ecuaciones lineales que podemos representar en su versión matricial como  $Ax = b$  y que intentaremos resolver usando dos métodos: la Eliminación Gaussiana y Factorización LU.

Nos interesa comparar estos dos métodos por el tiempo que les lleva resolver un único sistema  $Ax = b$  en función de la granularidad de la discretización. Posteriormente podemos complejizar el problema suponiendo que tenemos múltiples mediciones para las temperaturas de las paredes (a lo largo del tiempo), por lo que debemos comparar su performance a la hora de resolver múltiples sistemas  $Ax = b_i$  con diferentes vectores  $b_i$ .

La solución del sistema de ecuaciones nos permitirá conocer el valor de la función  $T$  en los puntos de la discretización elegida, pero es posible que ninguno de ellos coincida con el valor de la isotermia buscada. Otro objetivo del trabajo será evaluar diferentes formas de estimar esa isotermia y compararlas variando la granularidad de la discretización.

## 2. Desarrollo

### 2.1. Detalle de los algoritmos utilizados

#### 2.1.1. Eliminación Gaussiana

La Eliminación Gaussiana es un método que consiste en resolver el sistema triangulando la matriz (aplicando operaciones entre filas) y aplicando las mismas operaciones al vector solución, para obtener un sistema equivalente pero con una matriz triangular superior que se puede resolver de manera sencilla mediante *backward substitution*.

El algoritmo de trabaja con la matriz ampliada del sistema:

$$\left( \begin{array}{ccc|c} a_{11} & \dots & a_{1n} & b_1 \\ \vdots & \ddots & \vdots & \vdots \\ a_{n1} & \dots & a_{nn} & b_n \end{array} \right)$$

En cada paso  $k$  el algoritmo produce, mediante operaciones de filas, la aparición de ceros debajo de la diagonal en la columna  $k$ :

$$\begin{aligned} F_i &\leftarrow F_i - \frac{a_{i1}^{(0)}}{a_{11}^{(0)}} F_1 & \forall i = 2, \dots, n \\ F_i &\leftarrow F_i - \frac{a_{i2}^{(1)}}{a_{22}^{(1)}} F_2 & \forall i = 3, \dots, n \\ &\dots \\ F_i &\leftarrow F_i - \frac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}} F_k & \forall i = k+1, \dots, n \\ &\dots \\ F_n &\leftarrow F_n - \frac{a_{n(n-1)}^{(n-2)}}{a_{(n-1)(n-1)}^{(n-2)}} F_{n-1} \end{aligned}$$

Notemos que es necesario pedir que ningún valor de la diagonal se anule en ningún paso del algoritmo. Más adelante veremos que esto se puede garantizar gracias a la estructura de nuestra matriz.

El algoritmo tiene complejidad  $\mathcal{O}(n^3)$  para una matriz  $A \in \mathbb{R}^{n \times n}$  ya que realiza  $\mathcal{O}(n)$  operaciones escalares por fila por paso; hay  $n$  filas y realiza  $n - 1$  pasos.

#### 2.1.2. Backward substitution

Dada una matriz triangular superior  $A$  sin elementos nulos en la diagonal, entonces el vector solución  $x$  es único, y se obtiene haciendo

$$\begin{aligned} x_n &= \frac{b_n}{a_{nn}} \\ x_{n-1} &= \frac{b_{n-1} - a_{(n-1)n}x_n}{a_{(n-1)(n-1)}} \\ &\vdots \\ x_1 &= \frac{b_1 - a_{12}x_2 - a_{13}x_3 - \dots - a_{1n}x_n}{a_{11}} \end{aligned}$$

Notemos nuevamente la importancia de que los elementos de la diagonal sean no nulos para que las operaciones estén bien definidas.

Este algoritmo tiene complejidad  $\mathcal{O}(n^2)$  pues para despejar cada incógnita realiza  $\mathcal{O}(n)$  operaciones y hay  $n$  incógnitas.

### 2.1.3. Forward substitution

La idea es idéntica a backward substitution para matrices triangulares inferiores.

### 2.1.4. Factorización LU

El método de Factorización LU consiste en aplicar el método de Eliminación Gaussiana pero “almacenando” las operaciones realizadas para obtener una factorización  $A = LU$  con  $L$  triangular inferior (con unos en la diagonal) y  $U$  triangular superior. Más formalmente, cada paso  $k$  de la Eliminación Gaussiana es equivalente a multiplicar a la matriz  $A^{(k-1)}$  a izquierda por la matriz

$$M_k = \begin{pmatrix} 1 & \dots & 0 & \dots & 0 \\ 0 & \ddots & 0 & \dots & 0 \\ \vdots & & 1 & & \vdots \\ 0 & \dots & -\frac{a_{(k+1)k}^{(k-1)}}{a_{kk}^{(k-1)}} & \dots & 0 \\ \vdots & & \vdots & \ddots & \vdots \\ 0 & \dots & -\frac{a_{(n)k}^{(k-1)}}{a_{kk}^{(k-1)}} & \dots & 1 \end{pmatrix}$$

por lo que el resultado de la Eliminación Gaussiana es una matriz triangular superior  $U$  que coincide con  $M_{n-1} \dots M_1 A$ . Pero como todas las matrices  $M_k$  son invertibles por ser triangulares inferiores con unos en la diagonal, vale  $A = M_1^{-1} \dots M_{n-1}^{-1} U = LU$ .

Si bien ya hemos llegado a la factorización LU de  $A$ , podemos simplificar la expresión de  $L$ .

Notemos que  $M_k = I_n - m_k e_k^t$  con  $m_k = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ m_{(k+1)k} \\ \vdots \\ m_{nk} \end{pmatrix}$  y  $e_k$  el  $k$ -ésimo vector canónico de  $\mathbb{R}^n$ . Usando esta escritura es fácil ver que  $M_k^{-1} = I_n + m_k e_k^t$ , ya que

$$\begin{aligned} (I_n - m_k e_k^t)(I_n + m_k e_k^t) &= I_n^2 + m_k e_k^t - m_k e_k^t - m_k e_k^t m_k e_k^t \\ &= I_n^2 - m_k e_k^t m_k e_k^t \\ &= I_n \end{aligned} \quad (\text{pues } e_k^t m_k = 0)$$

Luego,

$$\begin{aligned} L &= M_1^{-1} \dots M_{n-1}^{-1} \\ &= (I_n + m_1 e_1^t) \dots (I_n + m_{n-1} e_{n-1}^t) \\ &= I_n + m_1 e_1^t + \dots + m_{n-1} e_{n-1}^t \end{aligned} \quad (\text{pues } e_i^t m_j = 0 \text{ si } i \leq j)$$

$$\text{En definitiva } A = LU \text{ con } L = \begin{pmatrix} 1 & \cdots & 0 & \cdots & 0 \\ m_{21} & \ddots & 0 & \cdots & 0 \\ \vdots & & 1 & & \vdots \\ m_{(k+1)1} & \cdots & m_{(k+1)k} & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots \\ m_{n1} & \cdots & m_{nk} & \cdots & 1 \end{pmatrix}.$$

Notemos que es posible ir armando  $L$  a cada paso de la Eliminación Gaussiana por lo que la complejidad computacional se mantiene: computar la factorización  $LU$  toma también  $\mathcal{O}(n^3)$ .

Luego, dado cualquier vector  $b$ , para resolver el sistema  $Ax = b$  alcanza con calcular  $y$  tal que  $Ly = b$  y luego  $x$  tal que  $Ux = y$ . El primer sistema se puede resolver en  $\mathcal{O}(n^2)$  por ser  $L$  triangular inferior (usamos *forward substitution*, que es igual que *backward* pero a la inversa) y el segundo también, por ser  $U$  triangular superior (usamos *backward substitution* normalmente).

De este modo vemos que, si para una misma matriz  $A$  y múltiples vectores  $b_1, \dots, b_k$  queremos resolver todos los sistemas  $Ax = b_i$ , en lugar de aplicar  $k$  veces la Eliminación Gaussiana (que toma  $\mathcal{O}(n^3)$ ) parecería conveniente calcular una única vez la factorización  $LU$  en  $\mathcal{O}(n^3)$  y luego usarla para resolver los  $k$  sistemas en  $\mathcal{O}(n^2)$ . Este trabajo práctico tiene como objetivo principal contrastar dicha conjetura experimentalmente.

## 2.2. Discretización

Para discretizar el problema limitaremos la cantidad de radios y la cantidad de ángulos a evaluar, y una vez elegidas dichas cantidades los distribuiremos de manera uniforme por la corona circular. Así, los puntos de la discretización serán los puntos  $P_{j,k}$  de radio  $r_j$  y ángulo  $\theta_k$  con  $j = 0, \dots, m$ ;  $k = 0, \dots, n$  que cumplan

$$\begin{aligned} r_0 &= r_i \\ r_m &= r_e \\ \theta_0 &= 0 \\ \theta_n &= 2\pi \\ r_{j+1} - r_j &= \Delta r \text{ constante } \forall j = 0, \dots, m-1 \\ \theta_{k+1} - \theta_k &= \Delta\theta \text{ constante } \forall \theta = 0, \dots, n-1 \end{aligned}$$

Llamaremos  $t_{j,k}$  a la temperatura del punto  $P_{j,k}$  una vez alcanzado el equilibrio térmico.

De la discretización elegida y de la ecuación 1 resulta una nueva ecuación que nos permitirá armar nuestro sistema de ecuaciones<sup>1</sup>.

$$\frac{t_{j-1,k} - 2t_{jk} + t_{j+1,k}}{(\Delta r)^2} + \frac{1}{r_j} \frac{t_{j,k} - t_{j-1,k}}{\Delta r} + \frac{1}{r_j^2} \frac{t_{j,k-1} - 2t_{jk} + t_{j,k+1}}{(\Delta\theta)^2} = 0 \quad (2)$$

Esta ecuación vale para cada punto  $P_{j,k}$  del modelo salvo los límites, sobre los cuales hablaremos en breve.

## 2.3. Armado del sistema de ecuaciones

Para poder armar el sistema  $Ax = b$  es necesario:

---

<sup>1</sup>Los detalles sobre la discretización de la ecuación pueden encontrarse en el Apéndice 6.1

- Extraer los factores que multiplican a cada una de las cinco incógnitas en la ecuación 2:  $t_{j-1,k}$ ;  $t_{j,k}$ ;  $t_{j+1,k}$ ;  $t_{j,k-1}$  y  $t_{j,k+1}$

$$\begin{aligned} t_{j-1,k} * & \left( \frac{1}{(\Delta r)^2} - \frac{1}{r_j \Delta r} \right) \\ t_{j,k} * & \left( \frac{-2}{(\Delta r)^2} + \frac{1}{r_j \Delta r} - \frac{2}{r_j^2 (\Delta \theta)^2} \right) \\ t_{j+1,k} * & \left( \frac{1}{(\Delta r)^2} \right) \\ t_{j,k-1} * & \left( \frac{1}{r_j^2 (\Delta \theta)^2} \right) \\ t_{j,k+1} * & \left( \frac{1}{r_j^2 (\Delta \theta)^2} \right) \end{aligned}$$

Por cuestiones de espacio, en adelante llamaremos  $M_{j,k}$  al factor que multiplica a la incógnita  $t_{j,k}$ ,  $M_{j-1,k}$  al que multiplica a  $t_{j-1,k}$  y así sucesivamente. Y resumiremos  $Mt_{j,k} = M_{j,k} * t_{j,k}$ ,  $Mt_{j-1,k} = M_{j-1,k} * t_{j-1,k}$ , etc.

- Analizar los “casos borde”: aquellos puntos donde la ecuación 2 no vale.

Para evitar confusiones, tomaremos  $\theta_0 = 0$  como el menor valor posible de  $\theta$  y  $\theta_{n-1}$  como el mayor, pues vale  $P_{j,n} = P_{j,0}$  para cualquier  $j$ .

Los casos interesantes para valores de  $j, k$  entonces son:

1. La pared interior del horno ( $j = 0; k = 0, \dots, n-1$ ). La ecuación en esos casos es

$$t_{0,k} = T_i(\theta_k)$$

2. La pared exterior del horno ( $j = m; k = 0, \dots, n-1$ ). La ecuación en esos casos es

$$t_{m,k} = T_e(\theta_k)$$

3. El valor mínimo de  $\theta$  ( $j = 0, \dots, m; k = 0$ ). Se debe reemplazar  $t_{j,k-1}$  por  $t_{j,n-1}$  en todas las ecuaciones correspondientes.

4. El valor máximo de  $\theta$  ( $j = 0, \dots, m; k = n-1$ ). Se debe reemplazar  $t_{j,k+1}$  por  $t_{j,0}$  en todas las ecuaciones correspondientes.

Estos últimos dos reemplazos se pueden resumir en

$$(j, k) \Rightarrow (j, k \bmod n)$$

- Combinar los puntos anteriores para plantear el sistema de ecuaciones a resolver:

$$\begin{aligned} t_{0,k} &= T_i(\theta_k) & \forall k = 0, \dots, n-1 \\ t_{m,k} &= T_e(\theta_k) & \forall k = 0, \dots, n-1 \\ Mt_{j-1,k} + Mt_{j,k} + Mt_{j+1,k} + Mt_{j,k-1} + Mt_{j,k+1} &= 0 & \forall j = 1, \dots, m-1; k = 1, \dots, n-2 \\ Mt_{j-1,0} + Mt_{j,0} + Mt_{j+1,0} + Mt_{j,n-1} + Mt_{j,1} &= 0 & \forall j = 1, \dots, m-1 \\ Mt_{j-1,n-1} + Mt_{j,n-1} + Mt_{j+1,n-1} + Mt_{j,n-2} + Mt_{j,0} &= 0 & \forall j = 1, \dots, m-1 \end{aligned}$$

Del mismo podemos obtener la matriz  $A$  (que tendrá 5 valores no nulos por fila a lo sumo) y el vector  $b$  (que será nulo en todas sus componentes salvo aquellas correspondientes a  $j = 0$  y  $j = m$ ).

- Pensar un orden para las incógnitas que permita asegurar que la matriz resultante sea *banda*. La importancia de esta propiedad se tratará en breve. El mismo es:

$$t_{0,0}; t_{0,1}; \dots; t_{0,n-1}; t_{1,0}; t_{1,1}; \dots; t_{j,n-1}; t_{j+1,0}; t_{j+1,1}; \dots; t_{m,n-2}; t_{m,n-1}$$

tanto para las filas como para las columnas. Este orden garantiza que la distancia máxima de un punto  $P_{j,k}$  hasta sus vecinos en la matriz es de  $n$  posiciones (para los casos  $P_{j+1,k}$  y  $P_{j-1,k}$ ) y por lo tanto el “ancho” de la banda es de  $2n$  (salvo para las filas que coinciden con la identidad, cuyo ancho es 1).

Una vez realizados estos pasos estamos en condiciones de plantear el sistema de ecuaciones  $Ax = b$ :

Lo primero que debemos notar es que como hay  $n * (m + 1)$  puntos diferentes tendremos  $n * (m + 1)$  incógnitas diferentes. Luego,  $A \in \mathbb{R}^{n(m+1) \times n(m+1)}$ : cada columna y cada fila de  $A$  corresponden a un punto  $P_{j,k}$  del sistema. Asimismo,  $x \in \mathbb{R}^{n(m+1)}$  y  $b \in \mathbb{R}^{n(m+1)}$ .

Lo segundo que debemos notar es que, por coincidir el orden elegido para filas y para columnas, el índice de la fila correspondiente al punto  $P_{j,k}$  coincide con el de la columna correspondiente a ese punto. Llamaremos a este índice  $i(j, k)$ . Notar que podemos computar  $i$  fácilmente como  $i(j, k) = j * n + k$  (suponiendo que indexamos por 0 tanto filas como columnas).

Por el orden elegido, las primeras  $n$  filas corresponden a los puntos  $P_{0,k}$  con  $k = 0, \dots, n - 1$ . Mirando el sistema de ecuaciones, las primeras  $n$  filas de  $A$  coinciden con la identidad (1 en la diagonal y 0 en el resto) y las primeras  $n$  filas de  $b$  coinciden con  $T_i(\theta_k)$ .

Lo mismo vale para las últimas  $n$  filas: corresponden a los puntos  $P_{m,k}$  con  $k = 0, \dots, n - 1$ , las filas correspondientes de  $A$  coinciden con la identidad y las componentes de  $b$  con  $T_e(\theta_k)$ .

Llegado este punto podemos definir completamente  $b$ : todas sus demás componentes son nulas (por ser 0 la solución al resto de las ecuaciones del sistema), por lo que resulta:

$$b = (T_i(0), T_i(1), \dots, T_i(n-1), 0, \dots, 0, T_e(0), T_e(1), \dots, T_e(n-1))$$

Para  $j \neq 0, j \neq m$ , las filas  $i(j, k)$  de  $A$  tendrán cinco componentes no-nulas (que corresponden a los vecinos de  $P_{j,k}$  en la discretización). Fijados  $j$  y  $k$  ( $0 \neq j \neq m, 0 \neq k \neq n - 1$ ), estas componentes serán  $i(j-1, k); i(j, k); i(j+1, k); i(j, k-1)$  e  $i(j, k+1)$  y coincidirán con lo que anteriormente llamamos  $M(j-1, k); M(j, k); M(j+1, k); M(j, k-1)$  y  $M(j, k+1)$  respectivamente.

Resta simplemente considerar los casos  $k = 0$  y  $k = n - 1$ , pero no revisten mayor complejidad que tomar módulo  $n$  después de las operaciones que involucren  $k$ .

## 2.4. Resolución del sistema de ecuaciones

### 2.4.1. Caracterización de la matriz

**Proposición 1.** *La matriz  $A$  es banda y diagonal dominante (no estricta) por filas.*

*Demostración.* El hecho de que es banda fue fundamentado al momento de construirla.

Resta ver que es diagonal dominante (no estricta) por filas, es decir, que

$$|a_{i,i}| \geq \sum_{j \neq i}^{n*(m+1)} |a_{i,j}| \quad \forall i = 1, \dots, n * (m + 1)$$

Las primeras y últimas  $n$  filas de  $A$  coinciden con la identidad, por lo que esto vale trivialmente. Dada cualquier otra fila  $i(j, k)$  con  $j \neq 0, j \neq m$  debemos recordar que hay solo 5 valores no nulos, por lo que en realidad queremos probar

$$|M_{j,k}| \geq |M_{j-1,k}| + |M_{j+1,k}| + |M_{j,k-1}| + |M_{j,k+1}|$$

Reemplazando por los valores de los multiplicadores, se obtiene

$$\left| \frac{-2}{(\Delta r)^2} + \frac{1}{r_j \Delta r} - \frac{2}{r_j^2 (\Delta \theta)^2} \right| \geq \left| \frac{1}{(\Delta r)^2} - \frac{1}{r_j \Delta r} \right| + \left| \frac{1}{(\Delta r)^2} \right| + 2 * \left| \frac{1}{r_j^2 (\Delta \theta)^2} \right|$$

Quitando el módulo en los valores claramente positivos y reordenando, tenemos

$$\left| \frac{1}{r_j \Delta r} - \frac{2}{(\Delta r)^2} - \frac{2}{r_j^2 (\Delta \theta)^2} \right| \geq \left| \frac{1}{(\Delta r)^2} - \frac{1}{r_j \Delta r} \right| + \frac{1}{(\Delta r)^2} + \frac{2}{r_j^2 (\Delta \theta)^2}$$

En este punto nos detuvimos a analizar cada caso en particular, pero pronto descubrimos que solo vale la pena uno: como sabemos  $j \neq 0$ , entonces  $r_j \geq \Delta r$  para cualquier valor de  $j$ , pues hasta el primer radio distinto al interior hay por lo menos un incremento en la discretización. Por lo tanto

$$\frac{1}{r_j \Delta r} \leq \frac{1}{(\Delta r)^2}$$

$$0 \leq \frac{1}{(\Delta r)^2} - \frac{1}{r_j \Delta r}$$

Por lo que podemos omitir el módulo de la derecha. Asimismo,

$$\frac{1}{r_j \Delta r} \leq \frac{1}{(\Delta r)^2}$$

$$\frac{1}{r_j \Delta r} < \frac{2}{(\Delta r)^2}$$

$$\frac{1}{r_j \Delta r} - \frac{2}{(\Delta r)^2} < 0$$

Y por ser  $\frac{2}{r_j^2(\Delta\theta)^2} > 0$  podemos afirmar

$$\frac{1}{r_j \Delta r} - \frac{2}{(\Delta r)^2} - \frac{2}{r_j^2(\Delta\theta)^2} < 0$$

Por lo que podemos deshacernos del módulo de la izquierda negando sus términos.

$$\frac{2}{(\Delta r)^2} - \frac{1}{r_j \Delta r} + \frac{2}{r_j^2(\Delta\theta)^2} \geq \frac{1}{(\Delta r)^2} - \frac{1}{r_j \Delta r} + \frac{1}{(\Delta r)^2} + \frac{2}{r_j^2(\Delta\theta)^2}$$

$$\frac{2}{(\Delta r)^2} - \frac{1}{r_j \Delta r} + \frac{2}{r_j^2(\Delta\theta)^2} \geq \frac{2}{(\Delta r)^2} - \frac{1}{r_j \Delta r} + \frac{2}{r_j^2(\Delta\theta)^2}$$

Con lo cual la desigualdad es trivialmente cierta (y de hecho resulta ser una igualdad, lo cual fundamenta por qué no es “estrictamente” diagonal dominante).  $\square$

#### 2.4.2. Resolución sin pivoteo

Nos interesa demostrar que es posible aplicar Eliminación Gaussiana sin pivoteo en la matriz  $A$ .

Para su demostración usaremos los siguientes lemas:

**Lema 1.** *Aplicar un paso de la Eliminación Gaussiana sobre una matriz diagonal dominante preserva esa propiedad en la submatriz que resta triangular.*

*Idea de demostración.* Igual que la demostración vista en clase teórica de que un paso de la Eliminación Gaussiana sobre una matriz *estrictamente* diagonal dominante preserva esa propiedad en la submatriz que resta triangular, pero relajando la condición de *estrictamente* para pedir que simplemente preserve la propiedad de diagonal dominante (cambiar los  $<$  por  $\leq$ ).  $\square$

**Lema 2.** *A cada paso  $k$  de la Eliminación Gaussiana la fila  $F_k$  con la que trabaje tendrá un elemento no nulo en la columna  $k+n$ , o bien será una fila de la identidad.*

*Demostración.* Empecemos por ver que el ancho de la banda de la matriz es estricto para las filas que no coinciden con la identidad: toda fila  $F_k$  tiene siempre como último elemento no nulo el de la columna  $k+n$  o bien es una fila de la identidad. Esto surge directamente de observar el orden elegido para las incógnitas y viendo que todas tienen un elemento no nulo correspondiente al vecino  $j+1, k$ .

Ya vimos que en la matriz original  $A$  toda fila  $F_k$  o bien coincide con la identidad o bien tenía como último elemento no nulo el de la columna  $k+n$ . Veamos ahora que esto no se puede haber

alterado en ningún paso del algoritmo: de alterarse, quiere decir que a la fila  $F_k$  se le restó una fila  $F_x$  en algún paso  $x$ ,  $x < k$ , que tenía un elemento no nulo en la columna  $k + n$ . Pero por ser  $x < k$ , el último elemento no nulo de la fila  $F_x$  debe ser  $x + n < k + x$ . Absurdo, que provino de suponer que se podría alterar la propiedad en algún paso del algoritmo.  $\square$

**Proposición 2.** *A cada paso  $k$  de la Eliminación Gaussiana la fila  $k$  con la que trabaje tendrá un elemento no nulo en la columna  $k$ , y por lo tanto el algoritmo podrá continuar normalmente sin realizar pivoteo.*

*Demostración.* Queremos ver que al comenzar el paso  $k$  de la Eliminación Gaussiana  $a_{kk}^{(k-1)} \neq 0$ .

Por el lema 1, al comenzar el paso  $k$  de la Eliminación Gaussiana la submatriz que aún resta triangular (llamémosla  $B$ ) es diagonal dominante.  $B$  incluye parte de la fila  $F_k$ , en particular a partir de la columna  $k$  en adelante, por lo que  $a_{kk}$  es parte de su diagonal (en particular es su primer elemento).

A su vez, por el lema 2, al comenzar el paso  $k$  de la Eliminación Gaussiana la fila  $F_k$  tiene un elemento no nulo en su columna  $k + n$ . Es decir, el elemento  $a_{k(k+n)}$  es no nulo, y este elemento pertenece a  $B$ . Por lo que, por definición de diagonal dominante  $|a_{kk}| \geq |a_{k(k+1)}| \neq 0 \Rightarrow a_{kk} \neq 0$  como queríamos demostrar.  $\square$

## 2.5. Métodología de experimentación

En esta sección desarrollaremos los experimentos planteados, desde la generació de instancias de prueba hasta los criterios de medición utilizados y los detalles de los experimentos.

### 2.5.1. Consideraciones iniciales

- La pared interna del horno en teoría debería ser de 1500 grados constantes, nosotros consideramos más realista que al variar los ángulos de la pared interna, la temperatura varíe uniformemente entre  $1500 \pm \epsilon$ , con  $\epsilon = 50$  .
- Respecto a las temperaturas externas, varian entre 50 y 200 de manera aleatoria uniforme en los experimentos presentados a continuacion.

### 2.5.2. Generación de casos de prueba

Los casos de prueba son generados mediante un script de python llamado testgen.py, dentro de la carpeta tools. El mismo posee 3 modos de ejecución:

1. **Modo 1:** Dados por parámetro un radio interno, externo e isoterma fijas, toma además una cantidad de radios inicial, una final, una cantidad de ángulos inicial, una final, y para cada combinación de cantidades de ellos genera un número de instancias también provisto por parámetro. En cada instancia, las temperaturas internas y externas son generadas pseudoaleatoriamente, ejecutando el comando **random.uniform(1450, 1550)** para las temperaturas internas y **random.uniform(50, 200)**.
2. **Modo 2:** Dados un radio interno, externo, isoterma, número de ángulos y radios fijos por parámetro, genera un único archivo en el que varía las temperatura de esa discretización particular (usando los comandos descriptos anteriormente), generando un número de instancias provisto por parámetro.
3. **Modo 3:** Dados un radio interno, externo, isoterma, y número de radios fijos por parámetro, se le especifica una cantidad de radios inicial y una final. Para cada cantidad de radios genera un único vector de temperaturas internas y externas, y genera tantas instancias como se le especifique.

### 2.5.3. Métricas de performance

Se medirá y comparará la performance de ambos métodos en la resolución de diversas instancias del problema. El tiempo se mide en microsegundos, y corresponde al tiempo de procesamiento de CPU, dejando fuera de consideración el tiempo de entrada/salida. El objetivo último es encontrar una relación entre la complejidad teórica de ambos algoritmos y su tiempo de cómputo, y realizar la comparación de ambos métodos en distintos escenarios. Todos los casos considerados están especificados en la sección **Esquema de experimentación**. En todos los casos se mide el tiempo total de resolución en CPU de **todas** las instancias del archivo de entrada.

### 2.5.4. Método de posicionamiento estimado de la isoterma

El problema consiste en estimar, para cada ángulo, la posición radial de la isoterma  $\alpha$ .

Consideremos la función **contínua** de temperatura  $T(r, \theta)$  definida sobre las coordenadas polares  $r, \theta$ . Si fijamos  $\theta = \theta_i$  podemos definir la función  $g_{\theta_i}(r) = T(r, \theta_i)$  como la función de temperatura sobre todos los radios **contínuos** del ángulo  $\theta_i$  fijado. Dado que nosotros conocemos  $m + 1$  puntos aproximados de dicha función, esto reduce el problema a aproximar el elemento

$$z_\alpha \in \text{Dom}(g_{\theta_i}) = [R_i \dots R_e] \text{ tal que } g_{\theta_i}(z_\alpha) = \alpha.$$

El método propuesto consiste en aplicar el siguiente algoritmo a las aproximaciones discretas conocidas de las funciones  $g_{\theta_i}$ , para todos los ángulos.

**Proposición 3** (Algoritmo de estimación de la isoterma  $\alpha$  por ajuste lineal). *Sea  $\hat{g}_{\theta_i}$  la función discreta de aproximaciones de temperatura de un ángulo  $i$ .*

1. Buscar  $x_1, x_2 \in \text{Dom}(\hat{g}_{\theta_i})$  tales que  $\hat{g}_{\theta_i}(x_1) \leq \alpha \leq \hat{g}_{\theta_i}(x_2)$  y esta cota sea ajustada.

$$2. z_\alpha = x_1 + \left( \frac{x_2 - x_1}{\hat{g}_{\theta_i}(x_2) - \hat{g}_{\theta_i}(x_1)} \right) * (\alpha - \hat{g}_{\theta_i}(x_1))$$

3. Devolver  $z_\alpha$ .

$z_\alpha$  es la posición estimada linealmente de la isoterma tal que  $g_{\theta_i}(z_\alpha) = \alpha$ .

**Proposición 4** (Dominio de correctitud de la estimación). *El algoritmo anterior estima linealmente la isoterma  $\alpha$  entre  $x_1$  y  $x_2$  si:*

- $\min_{x \in \text{Dom}(\hat{g}_{\theta_i})} \hat{g}_{\theta_i}(x) \leq \alpha \leq \max_{x \in \text{Dom}(\hat{g}_{\theta_i})} \hat{g}_{\theta_i}(x)$ .

- $\alpha \notin \text{Im}(\hat{g}_{\theta_i})$ .

- En caso de que  $\alpha$  esté fuera del rango  $[min..max]$  por convención se establece que la isoterma se encuentra en una posición radial  $R_i - \epsilon$  o  $R_e + \epsilon$  según corresponda.

- En caso de que  $\alpha \in \text{Im}(\hat{g}_{\theta_i})$  entonces  $z = x_1 = x_2$  y no es necesario ajuste lineal.

**Proposición 5** (Idea intuitiva de la correctitud del algoritmo de estimación lineal). *Si se cumplen las hipótesis de la proposición anterior entonces la cota existe y el algoritmo procederá a hacer el cálculo aproximado del paso siguiente, que no es más que un ajuste lineal entre los dos puntos de las cotas. Es decir, estamos asumiendo que el calor se disipa linealmente entre 2 puntos cualesquiera de la función.*

**Nota:** Asumir esto no es necesariamente correcto, se podría hacer un análisis más fino graficando las funciones  $g_{\theta_i}$  y aplicando métodos más avanzados de estimación para que la curva quede más suave. Por simplicidad solo consideraremos el ajuste lineal en este trabajo.

### 2.5.5. Métricas de seguridad de la isoterma

Plantemos una métrica que estima la estabilidad o seguridad de la pared del horno estableciendo una relación relativa entre la posición de la isoterma y el radio externo.

**Proposición 6** (Métrica de seguridad del horno basada en la posición relativa de la isoterma). *Sea  $iso_\alpha$  la solución del sistema, es decir, el conjunto de radios para cada ángulo que indica la posición radial de la isoterma.*

*Consideremos  $\Delta_{iso_\alpha} = \left( \frac{f(iso_\alpha) - R_i}{R_e - R_i} \right)$ , donde  $f(iso_\alpha)$  es una función de la isoterma, en nuestro caso, consideramos que el máximo o el promedio son buenas métricas.*

*Salvo casos patológicos, vale que  $0 \leq \Delta_{iso_\alpha} \leq 1$ . Luego basta establecer un límite de seguridad  $0 \leq \gamma_0 \leq 1$  tal que si vale  $\Delta_{iso_\alpha} > \gamma_0$  se considera inestable o insegura la pared del horno.*

## 2.6. Esquema de experimentación

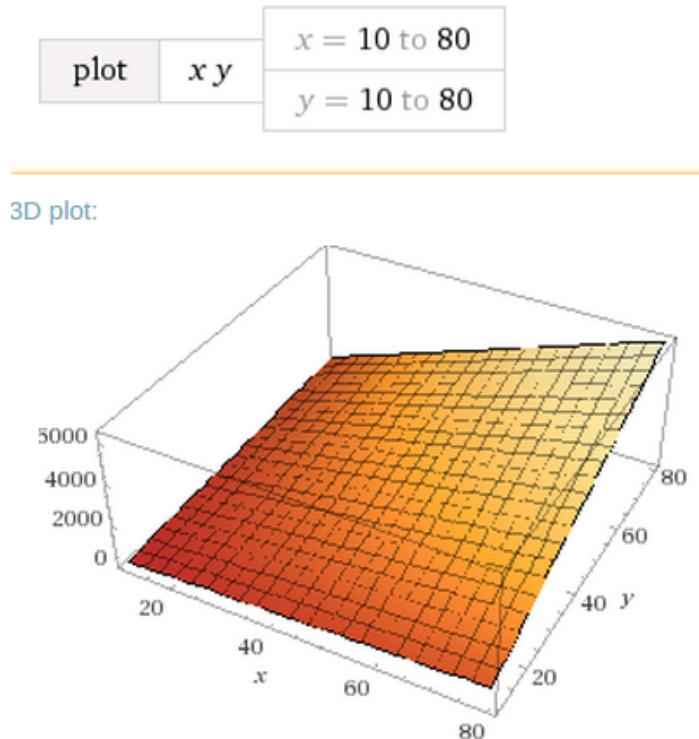
### 2.6.1. Experimentos focalizados en la performance

**Proposición 7** (La performance es invariante bajo cambios de datos iniciales cualitativos del problema). *La variación de las condiciones iniciales del problema, es decir, temperaturas internas y externas no cambian el tiempo de resolución del sistema de ecuaciones. (Utilizando los algoritmos estándares sin optimizaciones). Esto se debe a que la matriz resultante no cambia su dimensión al cambiar estos parámetros y la complejidad de los algoritmos reside en el tamaño de la matriz y no sus elementos.*

**Proposición 8** (Relación entre variables de la discretización y tamaño de la matriz del sistema). *La performance de los algoritmos se puede medir incrementando las variables ( $n, m+1$ ) del problema conjuntamente.*

Dado que en la sección de desarrollo especificamos que  $A \in \mathbb{R}^{n(m+1) \times n(m+1)}$  donde  $n$  y  $m+1$  representan los parámetros de la discretización. Tenemos que la dimensión de la matriz es  $n(m+1) \times n(m+1)$ . Además, sabemos que los algoritmos tienen una complejidad teórica definida sobre una variable  $k = n(m+1)$ , que expresa el tamaño de una matriz cuadrada  $A \in \mathbb{R}^{k \times k}$  (ie. LU y Gauss son  $\mathcal{O}(k^3)$ ), luego podemos afirmar que la complejidad teórica de Gauss y LU se puede escribir como  $\mathcal{O}((n * (m+1))^3) = \mathcal{O}((n * m + n)^3) = \mathcal{O}((n * m)^3)$  pues ( $m \geq 1$ ).

Veamos cómo se comporta la función  $z = n * m$  con valores de  $n, m$  en el rango [10, 80].



En el gráfico puede verse, que fijando  $n$  y variando  $m$ , o al revés, se obtiene un crecimiento lineal

en esa dirección. Asimismo, variando  $n$  y  $m$  al mismo tiempo, se obtiene un crecimiento lineal (más pronunciado) por la diagonal entre  $(0, 0)$  y  $(80, 80)$

Dado que nosotros queremos ver como se comportan los algoritmos cuando  $k = m * n$  aumenta, nos parece representativo solo tomar combinaciones de  $n, m$  que varíen conjuntamente, es decir, movernos por la diagonal del plano entre  $(0, 0)$  y  $(n_{max}, m_{max})$ .

Los experimentos de performance se dividen en dos secciones:

1. **Performance de los métodos en función de la discretización para una instancia:** Se generaron diversas discretizaciones del problema, variando la cantidad de radios y ángulos, conjuntamente, de 5 a 60, con una instancia para cada discretización. El objetivo es analizar la evolución del tiempo de cómputo a mayor granularidad de la discretización, y comprobar empíricamente que la complejidad de ambos algoritmos es del orden cúbico. Asimismo, factorización LU debería tomar más tiempo pues su constante es más grande al realizar más operaciones que simple triangulación.
2. **Performance de EG vs LU variando la cantidad de instancias y la granularidad de la discretización:** Para varias cantidades de instancias por archivo de test, se generan muchas discretizaciones diferentes. El objetivo de este test es analizar qué ocurre con el tiempo de ejecución cuando se resuelven muchas instancias utilizando EG o LU. También se evalúa la influencia de las diferentes discretizaciones para cada cantidad de instancias.

#### **2.6.2. Evolución de la temperatura y posición de la isoterma con distintas discretizaciones**

Para evaluar la calidad de nuestros estimadores y la evolución de la posición de la isoterma planteamos los siguientes experimentos:

Experimentos de estimación de isoterma con distintas discretizaciones:

1. **Variación de cantidad de radios.** La cantidad de ángulos, las temperaturas (internas y externas) y los demás parámetros se mantienen invariantes entre los tests de este experimento
  - Se plantean los radios internos, externos y la isoterma buscada y se mantienen fijos durante todo el experimento.
  - Se plantean un conjunto de temperaturas internas y externas **aleatorias uniformes** iniciales y se mantienen fijas durante todo el experimento.
  - Se plantea un número de ángulos de la discretización y se mantiene fijo durante todo el experimento.
  - Se plantea un rango  $[r_{min}, r_{max}]$  que denota la cobertura de discretizaciones distintas del experimento.
  - Se generan archivos de entrada  $test_i$  variando únicamente la cantidad de radios de la discretización a utilizar con una instancia por archivo.
  - Se ejecutan todos los archivos de entrada con el método de resolución más conveniente.
  - Se grafica, para cada archivo de test, la posición de la isoterma y el mapa de temperaturas del horno.
  - Se considera un video que tiene por frames los gráficos ordenados en el rango  $[r_{min}..r_{max}]$  del ítem anterior relacionado con la posición de la isoterma.
  - Se considera un video que tiene por frames los gráficos ordenados en el rango  $[r_{min}..r_{max}]$  del ítem anterior relacionado con la temperatura de la pared del horno.
  - Se grafica una función en el plano que denota la **máxima** posición relativa de la isoterma en la pared del horno a medida que varía ordenadamente el rango  $[r_{min}..r_{max}]$ .
  - Se grafica una función en el plano que denota la posición relativa **promedio** de la isoterma en la pared del horno a medida que varía ordenadamente el rango  $[r_{min}..r_{max}]$ .

Estos ultimos 2 ítems corresponden a cuantificar el error de la estimación lineal de las funciones  $g_{\theta_i}$  a medida que aumentan los radios, y en consecuencia teniendo más puntos discretos en las funciones  $\hat{g}_{\theta_i}$ .

**2. Variación de cantidad de ángulos.** La cantidad de radios, las temperaturas (internas y externas) y los demás parametros, se mantienen invariantes entre los tests de este experimento

- Se plantean los radios internos, externos y la isoterma buscada y se mantienen fijos durante todo el experimento.
- Se plantean un conjunto de temperaturas internas y externas **constantes** (si son aleatorias no asegurar misma solución entre tests del experimento) y se mantienen fijas durante todo el experimento, usándose en la creación de cada archivo de input con diferente cantidad de ángulos.
- Se plantea un numero de radios de la discretización y se mantiene fijo durante todo el experimento.
- Se plantea un rango  $[\theta_{min}, \theta_{max}]$  que denota la cobertura de discretizaciones distintas del experimento.
- Se generan archivos de entrada  $test_i$  variando únicamente la cantidad de ángulos de la discretización a utilizar con una instancia por archivo.
- Se ejecutan todos los archivos de entrada con el método de resolución más conveniente.
- Se grafica, para cada archivo de test, la posición de la isoterma y el mapa de temperaturas del horno.
- Se considera un video que tiene por frames los graficos ordenados en el rango  $[\theta_{min}, \theta_{max}]$  del ítem anterior relacionado con la posición de la isoterma.
- Se considera un video que tiene por frames los graficos ordenados en el rango  $[\theta_{min}, \theta_{max}]$  del ítem anterior relacionado con la temperatura de la pared del horno.

**3. Variación conjunta de cantidad de radios y ángulos.** Las temperaturas (internas y externas) y los demás parametros, se mantienen invariantes entre los tests de este experimento

- Se plantean los radios internos, externos y la isoterma buscada y se mantienen fijos durante todo el experimento.
- Se plantean un conjunto de temperaturas internas y externas **constantes** (si son aleatorias no asegurar misma solución entre tests del experimento) y se mantienen fijas durante todo el experimento.
- Se plantea un rango  $[r_{min}, r_{max}]$  que denota la cobertura de discretizaciones distintas del experimento.
- Se plantea un rango  $[\theta_{min}, \theta_{max}]$  que denota la cobertura de discretizaciones distintas del experimento.
- Se generan archivos de entrada  $test_i$  variando ambos rangos mencionados anteriormente en la discretización a utilizar con una instancia por archivo.
- Se ejecutan todos los archivos de entrada con el método de resolución más conveniente.
- Se grafica, para cada archivo de test, la posición de la isoterma y el mapa de temperaturas del horno.
- Se considera un video que tiene por frames los graficos ordenados convenientemente según granularidad, del ítem anterior relacionado con la posición de la isoterma.
- Se considera un video que tiene por frames los graficos ordenados convenientemente según granularidad, del ítem anterior relacionado con la temperatura de la pared del horno.

Los experimentos aquí planteados, tienen la misma solución entre tests, variando únicamente las discretizaciones, esto nos debería dar la pauta de como cambia la posición estimada de la isoterma al cambiar la discretización.

Se espera poder obtener conclusiones acerca de la suavidad de la curva estimada de la isoterma a medida que disminuye la granularidad de la discretización. Asimismo en las funciones que grafican el máximo y el promedio, se espera poder obtener conclusiones similares respecto a la variacion radial de la curva polar de la isoterma.

### **2.6.3. Evolución de la posición de la isoterna con distintas discretizaciones y temperaturas externas**

Experimentos de estimación de estabilidad de la pared del horno:

1. Para distintas discretizaciones, se plantea una variacion gradual en las temperaturas externas entre 50 y 1350 grados. Esto modifica la posición estimada de la isoterna 500, asimismo cambiando el coeficiente de seguridad mencionado más arriba.

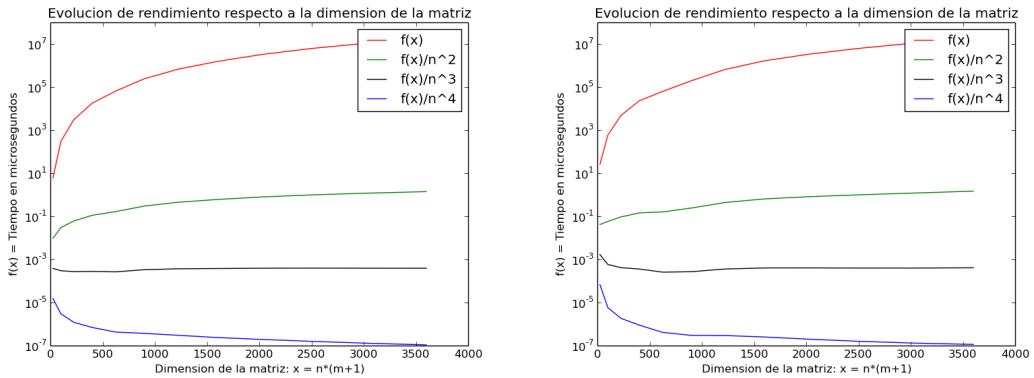
### 3. Resultados

#### 3.1. Performance

##### 3.1.1. Performance de los métodos en función de la discretización para una instancia

Utilizamos una heurística para poder determinar mejor el orden cúbico de la curva resultado. Sea  $f(x)$  el valor del tiempo de cómputo, graficar  $f(x)/x^k$ , siendo  $k \in [2, 3, 4]$ . De ser  $f(x)$  de orden cúbico, esperaríamos que para el polinomio de orden 2 el resultado sea una función creciente, para el de orden 3 el resultado se asemeje a una constante y para el de orden 4, una función decreciente.

**1 instancia por archivo de test**

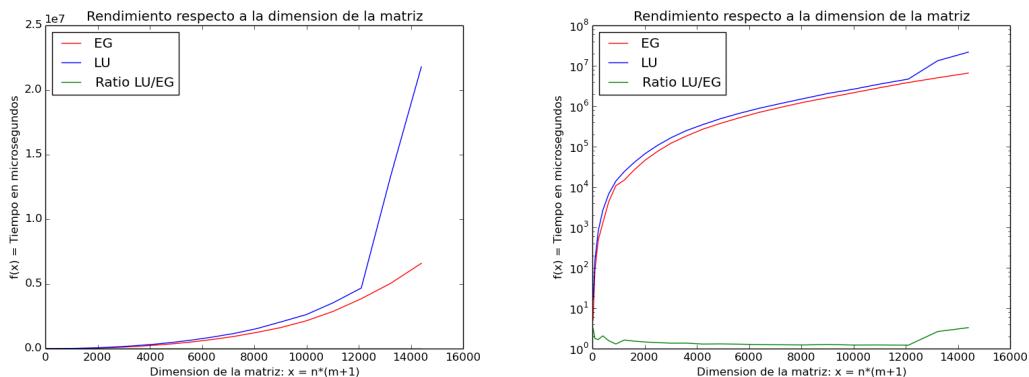


A pesar de ser una heurística, podemos corroborar que la curva sobre cubo se asemeja mucho a una constante, mientras que al cuadrado y a la cuarta funciones crecientes y decrecientes, respectivamente. Por lo que podemos decir que probamos empíricamente que los algoritmos son del orden cúbico.

##### 3.1.2. Performance de EG vs LU variando la cantidad de instancias y la granularidad de la discretizacion

Para comenzar esta sección, compararemos la performance de EG vs LU variando las discretizaciones, con archivos de entrada de una sola instancia. A continuación se presentan los gráficos de comparación entre EG y LU, en diferentes escalas.

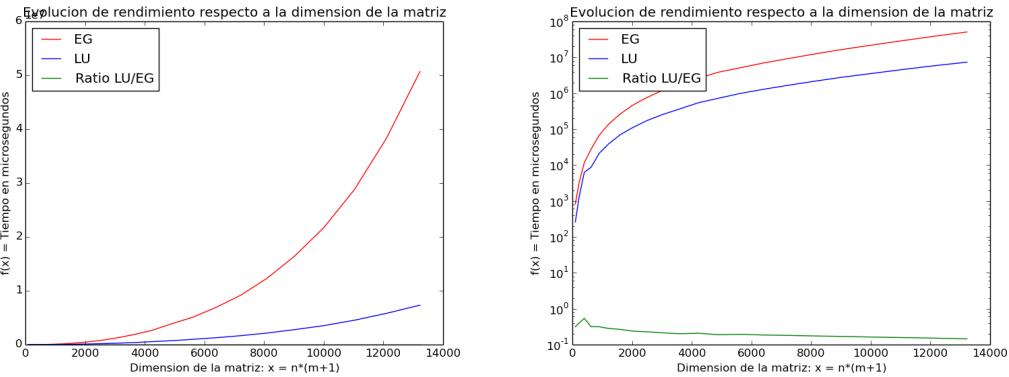
**1 instancia por archivo de test**



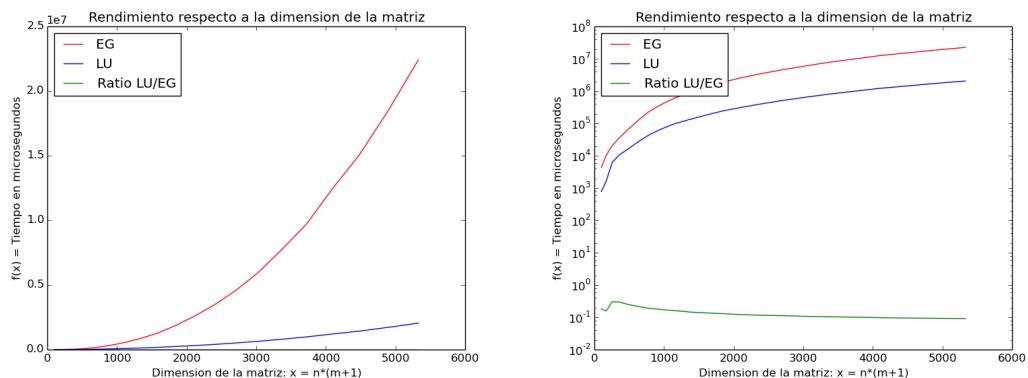
Dado que factorización LU, es idéntico a EG, pero guardando los coeficientes en la matriz L, tiene sentido que para una sola instancia sea más costoso realizar la factorización lu y resolver el sistema que simplemente resolver usando EG.

A continuacion, se presentan graficos comparativos entre LU y EG para distintas cantidades de instancias. Lo que se observa es que, la brecha entre EG y LU se acentúa cada vez más a medida que aumenta la cantidad de instancias(ver graficos con escala lineal). Esto se debe a que la complejidad teorica de resolver k instancias (misma matriz A, distinto vector b, en un sistema  $Ax=b$ ) usando EG es  $\mathcal{O}(k * (n + m)^3)$ . Por otro lado, la complejidad de resolver k instancias usando LU es  $\mathcal{O}((n + m)^3 + k * (n + m)^2)$ , es decir: Complejidad cúbica para hallar la descomposicion LU, y luego k resoluciones de los sistemas triangulares  $Ly = b$  y  $Ux = y$  que cuestan orden cuadrático cada uno.

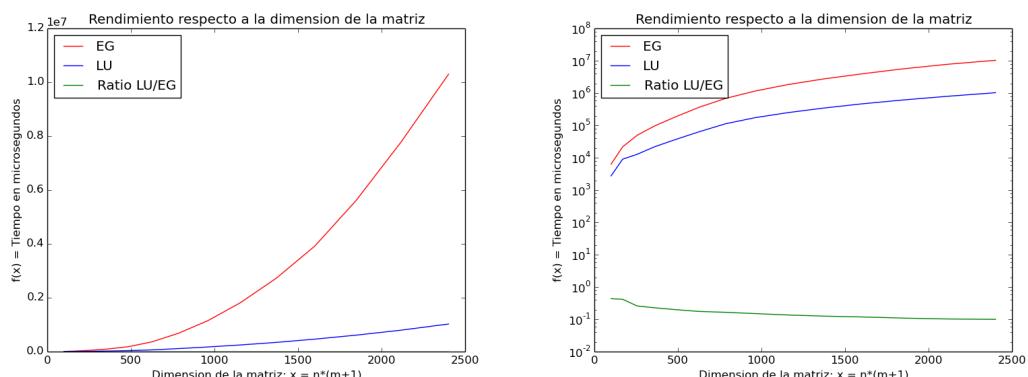
### 10 instancias por archivo de test



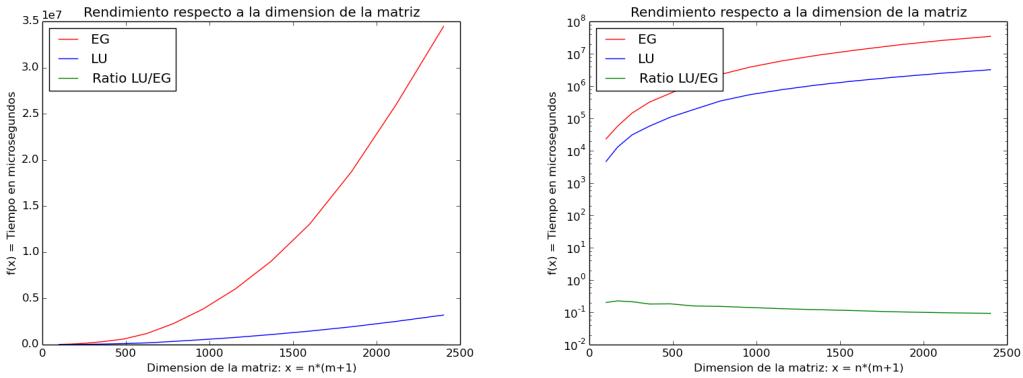
### 50 instancias por archivo de test



### 150 instancias por archivo de test



### 500 instancias por archivo de test



Tambien se observa en los graficos con escala logaritmica que la brecha entre EG y LU para una cantidad de instancias fija se acentua a medida que se aumenta el tamaño de la matriz del sistema, pues el ratio LU/EG decrece. Creemos que esto se debe a que al dejar  $k$  fijo y aumentar  $n + m$ , tenemos que  $\mathcal{O}((n + m)^3 + k * (n + m)^2)$  es mas pequeño que  $\mathcal{O}(k * (n + m)^3)$  en terminos empíricos.

#### Consideración Adicional:

Luego de la experimentacion, se nos ocurrio una optimizacion de la implementación de los algoritmos, se agregó la siguiente sentencia condicional al código de EG y la parte de triangulacion de LU:

```

for (int j = i+1; j < numfilas; j++) {
    if (abs(_A[j][i]) < EPSILON) {
        continue;
    }
}
.
.
.
```

Es decir, si el elemento a analizar de la matriz es un cero (con tolerancia epsilon, en nuestro caso  $\exp(10, -9)$ ), el algoritmo no realiza el cálculo del coeficiente multiplicador ni tampoco opera sobre la fila multiplicando cada elemento por éste. Al agregar esta sentencia y realizar los experimentos de fitteo de orden de complejidad, obtuvimos como resultado que la complejidad empírica de ambos algoritmos era de orden cuadrático. Creemos que esto se debe a la condición banda de la matriz, y que en muchos casos esta condición de corte evita que el algoritmo ingrese en la tercera iteracion anidada.

#### 3.1.3. Evolución estimación de la isotermia y temperatura

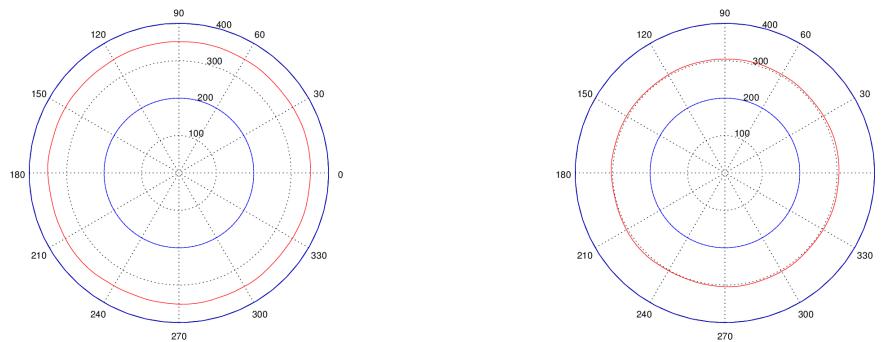
Se presentarán los resultados de los experimentos en el mismo orden en que fueron planteados en la sección de desarrollo. Se realizará el análisis de los mismos en este mismo apartado.

1.
  - **Temperaturas internas y externas:** aleatorias uniformes entre [50 ... 200] y [1450 ... 1550], pero fijas entre tests.
  - **Radio interno:** 200
  - **Radio externo:** 400
  - **Cantidad radios:** [5 ... 100]
  - **Cantidad ángulos:** 100

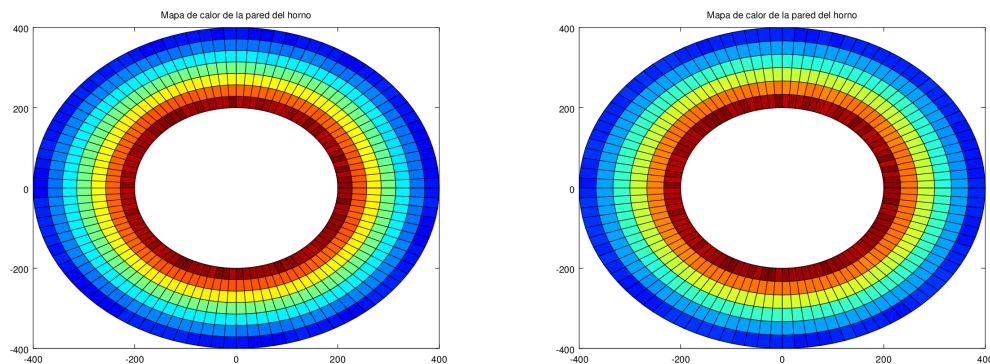
■ **Isotermas buscadas:** 500

Se adjunta con el trabajo práctico un video que expone la evolución del sistema mientras se incrementa la cantidad de radios. Expondremos estáticamente algunos frames, pero es conveniente ver el video primero. Se encuentra en la misma carpeta que el pdf. (variación\_radial\_isomap.mp4, variación\_radial\_heatmap.mp4).

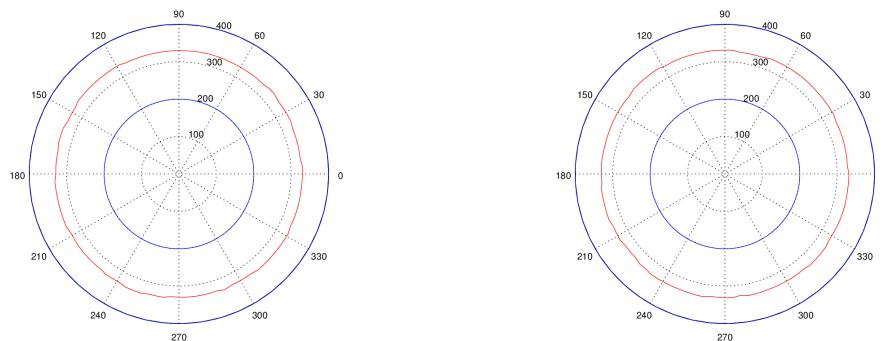
**Variación de la estimación de la isotermas entre 5 y 6 radios de discretización**



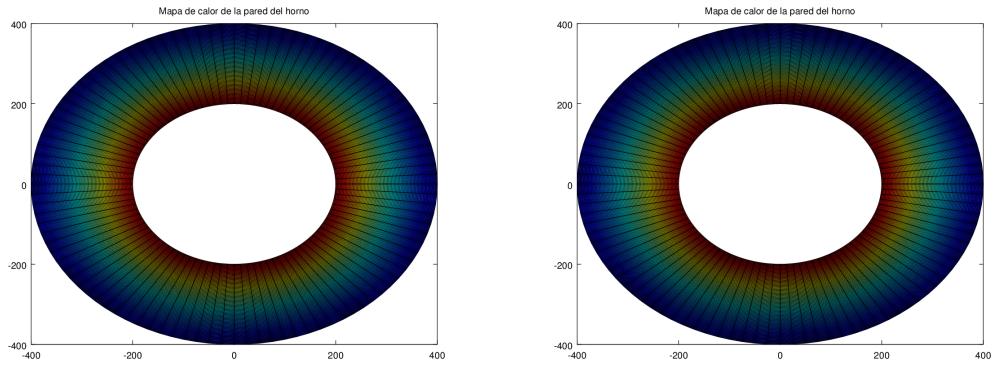
**Variación de la temperatura entre 6 y 7 radios de discretización**



**Variación de la estimación de la isotermas entre 99 y 100 radios de discretización**

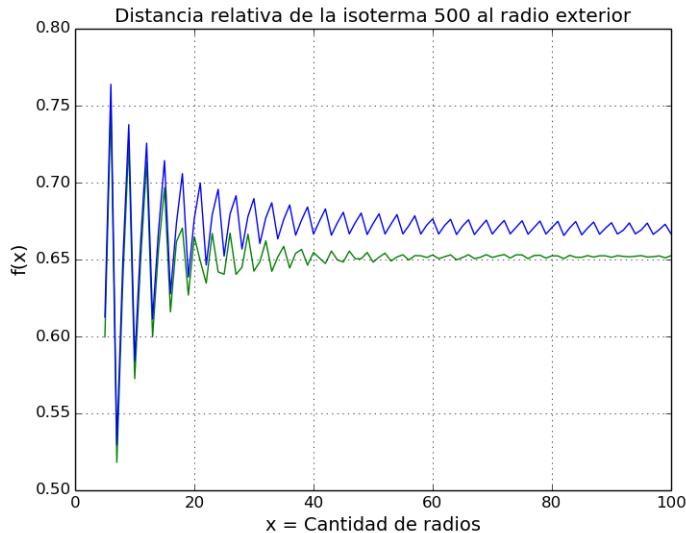


**Variación de la temperatura entre 99 y 100 radios de discretización**



Se observa es que a medida que se aumenta la cantidad de radios de la discretización, la variación radial de la curva de la isoterma disminuye entre tests, es decir, se hace más fina la estimación, de forma tal que entre  $i$  e  $i + 1$  radios la diferencia de la posición de la isoterma es menor a medida que  $i$  crece. Para ver mejor esto se graficaron, para cada test de  $i$  cantidad de radios de la discretización, el máximo y el promedio radial de la isoterma.

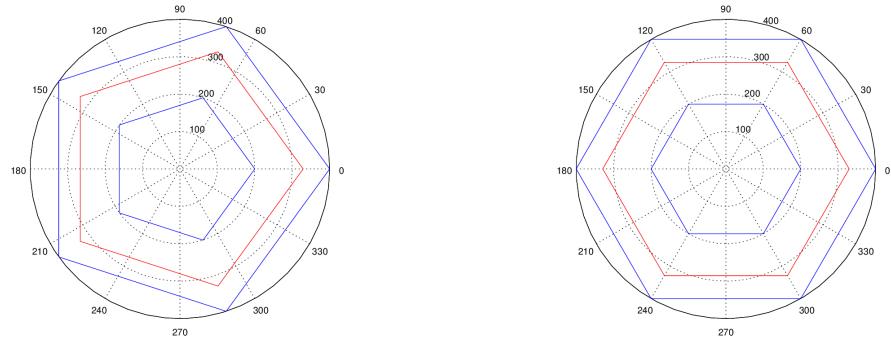
#### Evolución de la variación radial de la isoterma con cantidad creciente de radios



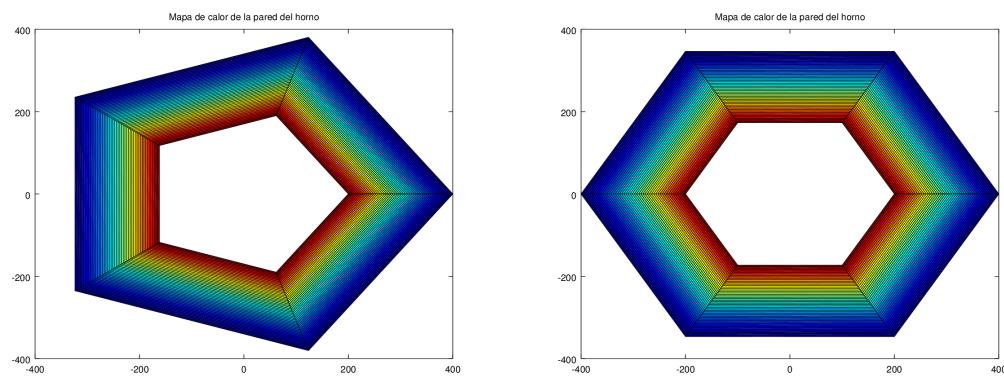
2.
  - **Temperaturas internas y externas:** constantes, 100 y 1500. Esto es para que tenga la misma solución cada test del experimento.
  - **Radio interno:** 200
  - **Radio externo:** 400
  - **Cantidad radios:** 50
  - **Cantidad ángulos:** [5...50]
  - **Isoterma buscada:** 500

Se adjunta con el trabajo práctico un video que expone la evolución del sistema mientras se incrementa la cantidad de radios. Expondremos estáticamente algunos frames, pero es conveniente ver el video primero. Se encuentra en la misma carpeta que el pdf. (variación\_angular\_isomap.mp4, variación\_angular\_heatmap.mp4).

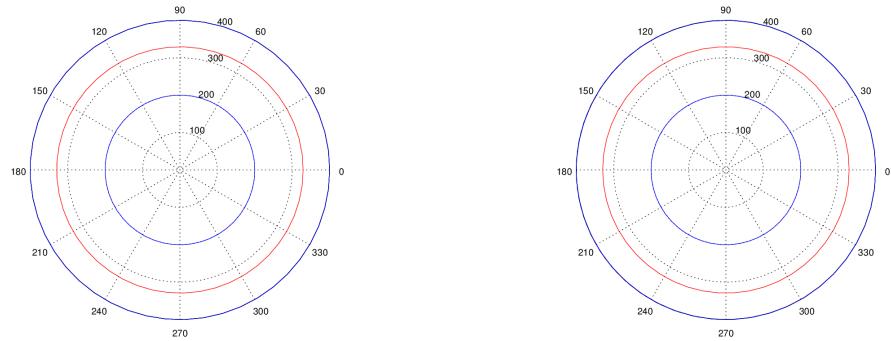
#### Variación de la estimación de la isoterma entre 5 y 6 ángulos de discretización



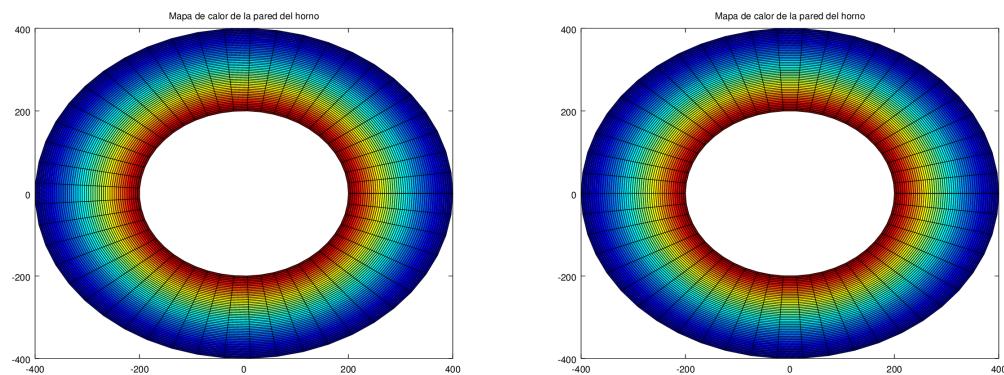
**Variación de la temperatura entre 5 y 6 ángulos de discretización**



**Variación de la estimación de la isoterna entre 49 y 50 ángulos de discretización**



**Variación de la temperatura entre 49 y 50 ángulos de discretización**



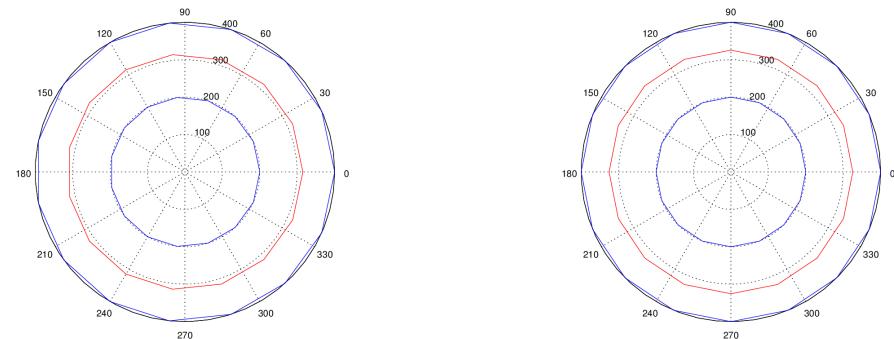
Aquí el radio es el mismo, pero se gana en precisión al tener más ángulos por no tener que

linealizar la posición de la isoterma angularmente. Nuevamente, la posición entre dos tests consecutivos se estabiliza al aumentar la cantidad de ángulos. Tambien se observa que al cambiar el  $\Delta_\theta$  los ángulos entre tests consecutivos no son los mismos.

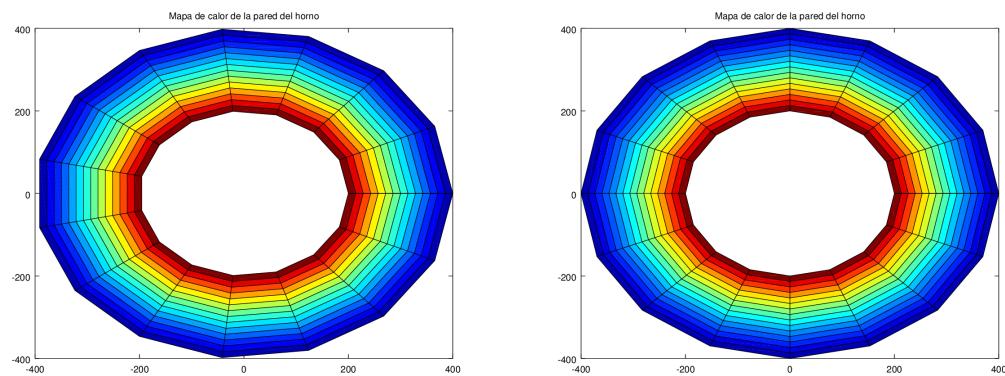
3.
  - **Temperaturas internas y externas:** constantes, 100 y 1500. Esto es para que tenga la misma solución cada test del experimento.
  - **Radio interno:** 200
  - **Radio externo:** 400
  - **Cantidad radios:** [15...60]
  - **Cantidad ángulos:** [15...60]
  - **Isoterma buscada:** 500

Se adjunta con el trabajo práctico un video que expone la evolución del sistema mientras se incrementa la cantidad de radios. Expondremos estáticamente algunos frames, pero es conveniente ver el video primero. Se encuentra en la misma carpeta que el pdf. (variación\_doble\_isomap.mp4, variación\_doble\_heatmap.mp4).

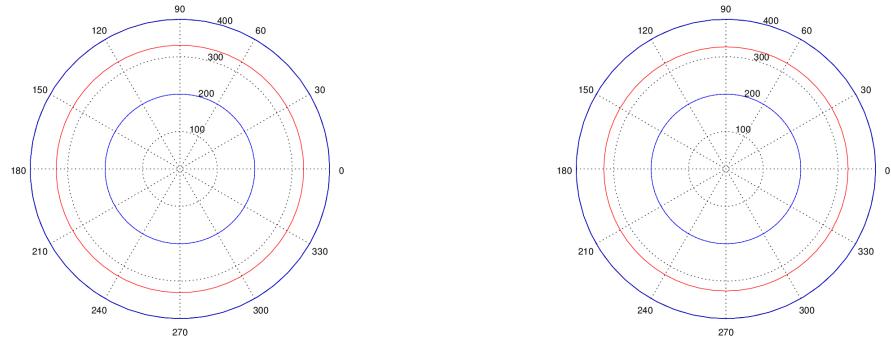
#### Variación de la estimación de la isoterma entre 15 y 16 radios, ángulos de discretización



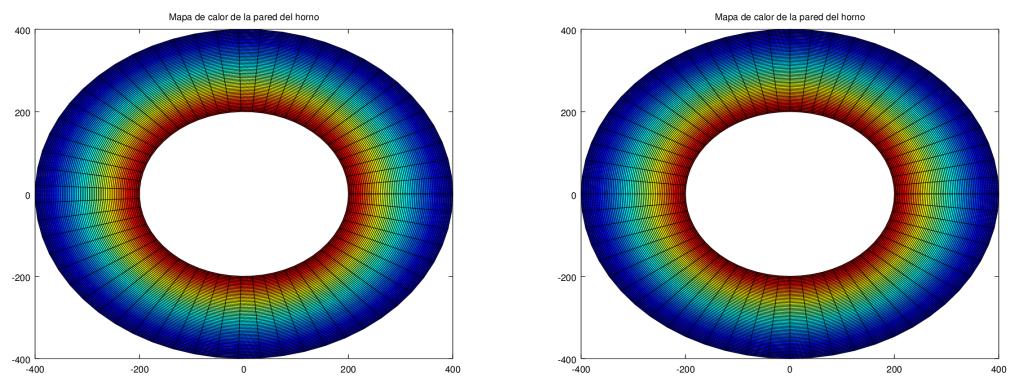
#### Variación de la temperatura entre 59 y 60 radios, ángulos de discretización



#### Variación de la estimación de la isoterma entre 15 y 16 radios, ángulos de discretización



**Variación de la temperatura entre 59 y 60 radios, ángulos de discretización**



En este último ejemplo ocurren ambos fenómenos al mismo tiempo, hay una variación radial menor a medida que crecen los radios y la curva se suaviza al aumentar los ángulos.

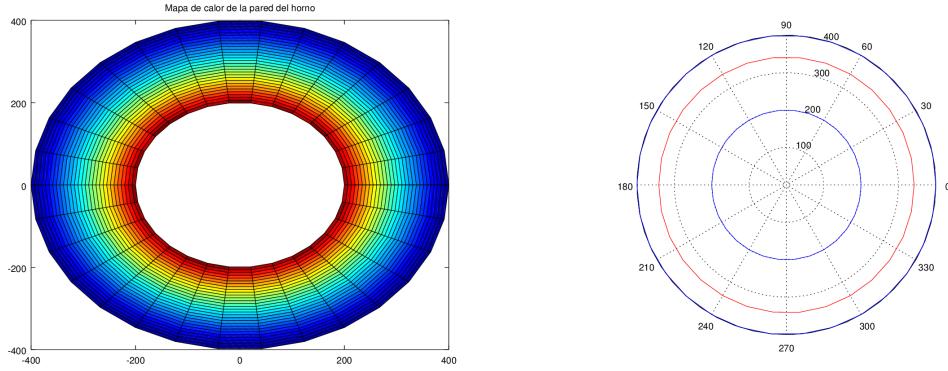
Efectivamente, podemos concluir que mientras más fina sea la discretización, se obtendrán resultados más estables y confiables acerca de la estimación. Uno de los motivos es porque habrá menos puntos para interpolar en la posición de la isoterma y el otro porque se tiene más información de la temperatura de la pared del horno.

### 3.1.4. Estimación de estabilidad de la pared del horno

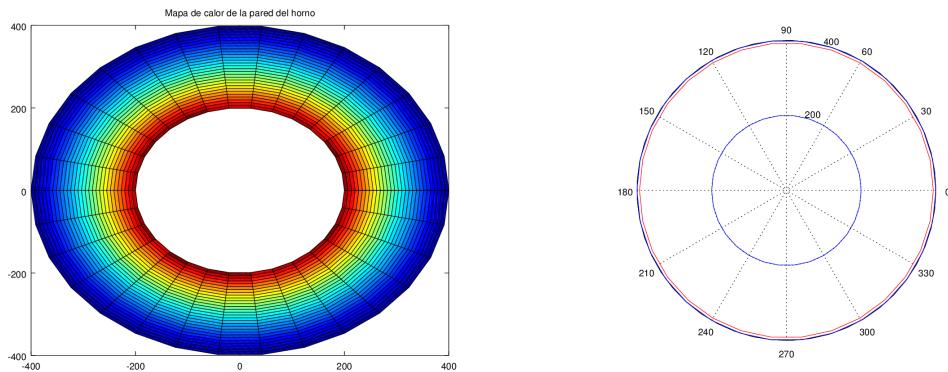
Para este experimento utilizaremos discretizaciones finas, ya vimos en los experimentos anteriores que esto provee mayor confiabilidad en las estimaciones, para discretizaciones más gruesas las estimaciones de seguridad serán menos exactas.

- **Temperaturas internas:** [200, 500, 750]
- **Temperaturas externas:** [1450 ... 1550] aleatorias uniformes.
- **Radio interno:** 200
- **Radio externo:** 400
- **Cantidad radios:** 30
- **Cantidad ángulos:** 30
- **Isoterma buscada:** 500

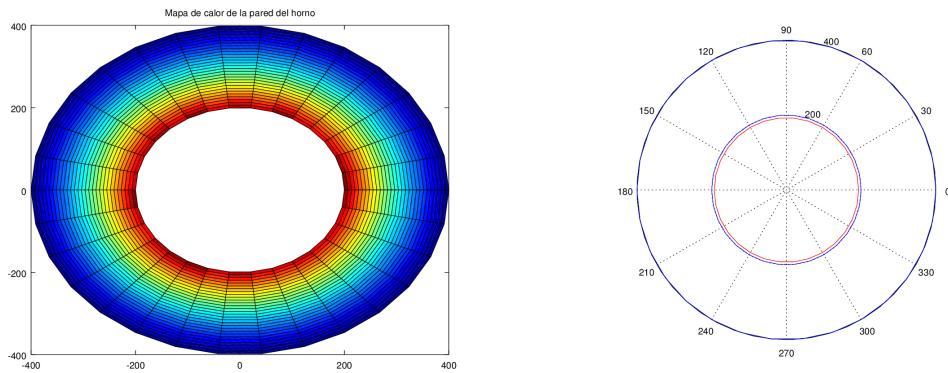
### Temperaturas externas: 200



### Temperaturas externas: 500



### Temperaturas externas: 750



La escala de color de los mapas de temperaturas se hace en base al mínimo y máximo de la muestra, es por esto que la variación de temperaturas externas no provee una variación en los colores de los radios del borde.

Asimismo, se ve que la posición de la isoterma en los casos 200, 500 se posiciona según lo esperado dentro de la pared del horno. Mientras que en el caso 750, al ser  $750 > 500$ , por convención posicionamos la isoterma en  $R_i - \epsilon$ .

En la siguiente tabla se puede observar que las métricas de seguridad nos dan una pauta acerca de la posición promedio y maxima de la isoterma dentro de la pared del horno. Si utilizamos  $\gamma_0 = 0,75$  como límite de seguridad, podemos establecer conclusiones acerca de la seguridad: en el caso donde hay 200 grados en el exterior es seguro, mientras que 500 grados, no lo es.

$T_e$	$\Delta_{max_{iso500}}$	$\Delta_{prom_{iso500}}$	Seguro bajo $\gamma_0 = 0,75$
200	0.709145	0.71037	Si
500	0.965517	0.965517	No

## 4. Conclusiones

En este trabajo buscamos comparar la eficiencia temporal de dos algoritmos de resolución de sistemas lineales. Consideramos confirmada la hipótesis inicial de que, si bien Eliminación Gaussiana tiene una eficiencia similar a Factorización LU para un único sistema, cuando consideramos una familia de sistemas que varían solo en el vector y no en la matriz, la Factorización LU es considerablemente más rápida.

Sobre el problema que motivó esta comparación, el cálculo de una isoterma en una corona circular con temperaturas conocidas solamente en su circunferencia interna y externa, pudimos verificar que usando una discretización lo suficientemente fina, el mismo tiende a estabilizarse rápidamente, dándonos cierta confianza sobre el método elegido (interpolación lineal).

## 5. Trabajo Futuro

Como posible trabajo futuro resta comparar este con otros métodos de interpolación, por ejemplo logarítmica. Asimismo, podrían estudiarse otras formas de interpolación de la isoterma, por ejemplo interpolación respecto a la dimensión angular del problema, donde dados  $k$  y  $k + \Delta_\theta$  se puede aproximar  $k + \frac{\Delta_\theta}{2}$  con alguna técnica mas avanzada para dar mas suavidad a la curva sin aumentar la discretización. Queda pendiente tambien la implementación de estructuras de datos y algoritmos que aprovechen la condición de banda de la matriz, esta mejora cambiaría drásticamente el orden de complejidad espacial y variaría tambien el orden de complejidad temporal. Por otro lado, de estos algoritmos no serían triviales y requerirían un análisis más fino.

Lo que sí pudimos experimentar un poco pero no nos alcanzó el tiempo para poner la información en el informe fue, al poner una condición en los algoritmos de LU y eliminación gaussiana que saltee el procesamiento de una fila si ya hay un cero bajo la diagonal en esa fila, produce que el tiempo empírico de computo se reduzca a tiempo cuadrático. Creemos que esto se debe a la condición banda de la matriz, y que en muchos casos esta condición de corte evita que el algoritmo ingrese en la tercera iteración anidada.

## 6. Apéndice

### 6.1. Apéndice A: Enunciado

Laboratorio de Métodos Numéricos - Segundo Cuatrimestre 2015  
Trabajo Práctico Número 1: Con 15  $\theta$ s discretizo alto horno...

#### Introducción

Consideremos la sección horizontal de un horno de acero cilíndrico, como en la Figura 1. El sector A es la pared del horno, y el sector B es el horno propiamente dicho, en el cual se funde el acero a temperaturas elevadas. Tanto el borde externo como el borde interno de la pared forman círculos. Suponemos que la temperatura del acero dentro del horno (o sea, dentro de B) es constante e igual a 1500°C.

Tenemos sensores ubicados en la parte externa del horno para medir la temperatura de la pared externa del mismo, que habitualmente se encuentra entre 50°C y 200°C. El problema que debemos resolver consiste en estimar la isoterma de 500°C dentro de la pared del horno, para estimar la resistencia de la misma. Si esta isoterma está demasiado cerca de la pared externa del horno, existe peligro de que la estructura externa de la pared colapse.

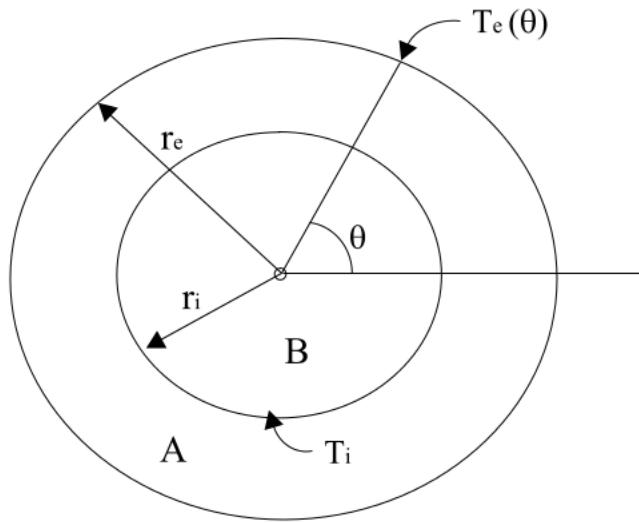


Figura 1: Sección circular del horno

El objetivo del trabajo práctico es implementar un programa que calcule la isoterma solicitada, conociendo las dimensiones del horno y las mediciones de temperatura en la pared exterior.

#### El Modelo

Sea  $r_e \in \mathbb{R}$  el radio exterior de la pared y sea  $r_i \in \mathbb{R}$  el radio interior de la pared. Llámese  $T(r, \theta)$  a la temperatura en el punto dado por las coordenadas polares  $(r, \theta)$ , siendo  $r$  el radio y  $\theta$  el ángulo polar de dicho punto. En el estado estacionario, esta temperatura satisface la ecuación del calor:

$$\frac{\partial^2 T(r, \theta)}{\partial r^2} + \frac{1}{r} \frac{\partial T(r, \theta)}{\partial r} + \frac{1}{r^2} \frac{\partial^2 T(r, \theta)}{\partial \theta^2} = 0 \quad (3)$$

Si llamamos  $T_i \in \mathbb{R}$  a la temperatura en el interior del horno (sector B) y  $T_e : [0, 2\pi] \rightarrow \mathbb{R}$  a la función de temperatura en el borde exterior del horno (de modo tal que el punto  $(r_e, \theta)$  tiene temperatura  $T_e(\theta)$ ), entonces tenemos que

$$T(r, \theta) = T_i \quad \text{para todo punto } (r, \theta) \text{ con } r \leq r_i \quad (4)$$

$$T(r_e, \theta) = T_e(\theta) \quad \text{para todo punto } (r_e, \theta) \quad (5)$$

El problema en derivadas parciales dado por la primera ecuación con las condiciones de contorno presentadas recientemente, permite encontrar la función  $T$  de temperatura en el interior del horno (sector A), en función de los datos mencionados en esta sección.

Para resolver este problema computacionalmente, discretizamos el dominio del problema (el sector A) en coordenadas polares. Consideramos una partición  $0 = \theta_0 < \theta_1 < \dots < \theta_n = 2\pi$  en  $n$  ángulos discretos con  $\theta_k - \theta_{k-1} = \Delta\theta$  para  $k = 1, \dots, n$ , y una partición  $r_i = r_0 < r_1 < \dots < r_m = r_e$  en  $m + 1$  radios discretos con  $r_j - r_{j-1} = \Delta r$  para  $j = 1, \dots, m$ .

El problema ahora consiste en determinar el valor de la función  $T$  en los puntos de la discretización  $(r_j, \theta_k)$  que se encuentren dentro del sector A. Llamemos  $t_{jk} = T(r_j, \theta_k)$  al valor (desconocido) de la función  $T$  en el punto  $(r_j, \theta_k)$ .

Para encontrar estos valores, transformamos la ecuación (2) en un conjunto de ecuaciones lineales sobre las incógnitas  $t_{jk}$ , evaluando (2) en todos los puntos de la discretización que se encuentren dentro del sector A. Al hacer esta evaluación, aproximamos las derivadas parciales de  $T$  en (2) por medio de las siguientes fórmulas de diferencias finitas:

$$\frac{\partial^2 T(r, \theta)}{\partial r^2}(r_j, \theta_k) \cong \frac{t_{j-1,k} - 2t_{jk} + t_{j+1,k}}{(\Delta r)^2} \quad (6)$$

$$\frac{\partial T(r, \theta)}{\partial r}(r_j, \theta_k) \cong \frac{t_{j,k} - t_{j-1,k}}{\Delta r} \quad (7)$$

$$\frac{\partial^2 T(r, \theta)}{\partial \theta^2}(r_j, \theta_k) \cong \frac{t_{j,k-1} - 2t_{jk} + t_{j,k+1}}{(\Delta\theta)^2} \quad (8)$$

Es importante notar que los valores de las incógnitas son conocidos para los puntos que se encuentran sobre el borde exterior de la pared, y para los puntos que se encuentren dentro del sector B. Al realizar este procedimiento, obtenemos un sistema de ecuaciones lineales que modela el problema discretizado. La resolución de este sistema permite obtener una aproximación de los valores de la función  $T$  en los puntos de la discretización.

### Enunciado

Se debe implementar un programa en C o C++ que tome como entrada los parámetros del problema ( $r_i, r_e, m + 1, n$ , valor de la isoterma buscada,  $T_i, T_e(\theta)$ ) que calcule la temperatura dentro de la pared del horno utilizando el modelo propuesto en la sección anterior y que encuentre la isoterma buscada en función del resultado obtenido del sistema de ecuaciones. El método para determinar la posición de la isoterma queda a libre elección de cada grupo y debe ser explicado en detalle en el informe.

El programa debe formular el sistema obtenido a partir de las ecuaciones (1) - (6) y considerar dos métodos posibles para su resolución: mediante el algoritmo clásico de Eliminación Gaussiana y la Factorización LU. Finalmente, el programa escribirá en un archivo la solución obtenida con el formato especificado en la siguiente sección.

Como ya se ha visto en la materia, no es posible aplicar los métodos propuestos para la resolución a cualquier sistema de ecuaciones. Sin embargo, la matriz del sistema considerado en el presente trabajo cumple con ser diagonal dominante (no estricto) y que, ordenando las variables y ecuaciones convenientemente, es posible armar un sistema de ecuaciones cuya matriz posee la propiedad de ser *banda*. Luego, se pide demostrar (o al menos dar un esquema de la demostración) el siguiente resultado e incluirlo en el informe:

**Proposición 9.** *Sea  $A \in \mathbb{R}^{n \times n}$  la matriz obtenida para el sistema definido por (1)-(6). Demostrar que es posible aplicar Eliminación Gaussiana sin pivoteo.<sup>2</sup>*

<sup>2</sup>Sugerencia: Notar que la matriz es diagonal dominante (no estrictamente) y analizar qué sucede al aplicar un paso de Eliminación Gaussiana con los elementos de una fila.

La solución del sistema de ecuaciones permitirá saber la temperatura en los puntos de la discretización. Sin embargo, nuestro interés es calcular la isoterma 500, para poder determinar si la estructura se encuentra en peligro. Luego, se pide lo siguiente:

- Dada la solución del sistema de ecuaciones, proponer una forma de estimar en cada ángulo de la discretización la posición de la isoterma 500.
- En función de la aproximación de la isoterma, proponer una forma (o medida) a utilizar para evaluar la peligrosidad de la estructura en función de la distancia a la pared externa del horno.

En función de la experimentación, se busca realizar dos estudios complementarios: por un lado, analizar cómo se comporta el sistema y, por otro, cuáles son los requerimientos computacionales de los métodos. Se pide como mínimo realizar los siguientes experimentos:

1. Comportamiento del sistema.
  - Considerar al menos dos instancias de prueba, generando distintas discretizaciones para cada una de ellas y comparando la ubicación de la isoterma buscada respecto de la pared externa del horno. Se sugiere presentar gráficos de temperatura o curvas de nivel para los mismos, ya sea utilizando las herramientas provistas por la cátedra o implementando sus propias herramientas de graficación.
  - Estudiar la proximidad de la isoterma buscada respecto de la pared exterior del horno en función de distintas granularidades de discretización y las condiciones de borde.
2. Evaluación de los métodos.
  - Analizar el tiempo de cómputo requerido para obtener la solución del sistema en función de la granularidad de la discretización. Se sugiere presentar los resultados mediante gráficos de tiempo de cómputo en función de alguna de las variables del problema.
  - Considerar un escenario similar al propuesto en el experimento 1. pero donde las condiciones de borde (i.e.,  $T_i$  y  $T_e(\theta)$ ) cambian en distintos instantes de tiempo. En este caso, buscamos obtener la secuencia de estados de la temperatura en la pared del horno, y la respectiva ubicación de la isoterma especificada. Para ello, se considera una secuencia de  $ninst$  vectores con las condiciones de borde, y las temperaturas en cada estado es la solución del correspondiente sistema de ecuaciones. Se pide formular al menos un experimento de este tipo, aplicar los métodos de resolución propuestos de forma conveniente y compararlos en términos de tiempo total de cómputo requerido para distintos valores de  $ninst$ .

De manera opcional, aquellos grupos que quieran ir un poco más allá pueden considerar trabajar y desarrollar alguno(s) de los siguientes puntos extra:

1. Notar que el sistema resultante tiene estructura *banda*. Proponer una estructura para aprovechar este hecho en términos de la *complejidad espacial* y como se adaptarían los algoritmos de Eliminación Gaussiana y Factorización LU para reducir la cantidad de operaciones a realizar.
2. Implementar dicha estructura y las adaptaciones necesarias para el algoritmo de Eliminación Gaussiana.
3. Implementar dicha estructura y las adaptaciones necesarias para el algoritmo de Factorización LU.

Finalmente, se deberá presentar un informe que incluya una descripción detallada de los métodos implementados y las decisiones tomadas, el método propuesto para el cálculo de la isoterma buscada y los experimentos realizados, junto con el correspondiente análisis y siguiendo las pautas definidas en el archivo `pautas.pdf`.

### **Programa y formato de archivos**

Se deberán entregar los archivos fuentes que contengan la resolución del trabajo práctico. El ejecutable tomará tres parámetros por línea de comando, que serán el archivo de entrada, el archivo de salida, y el método a ejecutar (0 EG, 1 LU).

El archivo de entrada tendrá la siguiente estructura:

- La primera línea contendrá los valores  $r_i$ ,  $r_e$ ,  $m + 1$ ,  $n$ ,  $iso$ ,  $ninst$ , donde  $iso$  representa el valor de la isoterma buscada y  $ninst$  es la cantidad de instancias del problema a resolver para los parámetros dados.
- A continuación, el archivo contendrá  $ninst$  líneas, cada una de ellas con  $2n$  valores, los primeros  $n$  indicando los valores de la temperatura en la pared interna, i.e.,  $T_i(\theta_0), T_i(\theta_1), \dots, T_i(\theta_{n-1})$ , seguidos de  $n$  valores de la temperatura en la pared externa, i.e.,  $T_e(\theta_0), T_e(\theta_1), \dots, T_e(\theta_{n-1})$ .

El archivo de salida obligatorio tendrá el vector solución del sistema reportando una componente del mismo por línea. En caso de  $ninst > 1$ , los vectores serán reportados uno debajo del otro.

Junto con el presente enunciado, se adjunta una serie de scripts hechos en python y un conjunto instancias de test que deberán ser utilizados para la compilación y un testeo básico de la implementación. Se recomienda leer el archivo README.txt con el detalle sobre su utilización.

### Fechas de entrega

- *Formato Electrónico:* Jueves 3 de Septiembre de 2015, hasta las 23:59 hs, enviando el trabajo (informe + código) a la dirección `metnum.lab@gmail.com`. El subject del email debe comenzar con el texto [TP1] seguido de la lista de apellidos de los integrantes del grupo.
- *Formato físico:* Viernes 4 de Septiembre de 2015, de 17:30 a 18:00 hs.

**Importante:** El horario es estricto. Los correos recibidos después de la hora indicada serán considerados re-entrega. Los grupos deben ser de 3 o 4 personas, sin excepción. Es indispensable que los trabajos pasen satisfactoriamente los casos de test provistos por la cátedra.

## 6.2. Apéndice B: Código relevante

Listing 1: Eliminacion gaussiana

```
// Triangular la matriz ampliada del sistema
for (int i = 0; i < numcolumnas - 1; i++) {

    for (int j = i+1; j < numfilas; j++) {

        // Aumenta la performance este chequeo
        if (_A[j][i] == 0) {
            // Ya hay un cero alli , no hay nada que hacer
            continue;
        }

        // Calculo el coeficiente multiplicador
        double m = _A[j][i] / _A[i][i];

        // Opero sobre la fila
        for (int k = i; k < numcolumnas; k++) {
            _A[j][k] -= m * _A[i][k];
        }

        // Tambien hay que modificar el vector b !
        _b[j] -= m * _b[i];
    }
}

// Calculo el vector X de soluciones con backward substitution .
for (int i = numfilas - 1; i >= 0; i--) {
    // Obtener suma de la fila por el b
    double sumaAcum = 0;
    for (int j = i+1; j < numcolumnas; j++) {
        if (i<numfilas && j < numcolumnas) {
            sumaAcum += _A[i][j] * vec_sol[j];
        }
    }

    // Despejar el xi
    vec_sol[i] = (_b[i] - sumaAcum) / _A[i][i];
}
```

Listing 2: Factorizacion LU

```
void SistemaEcuaciones::resolver_con_LU( FactorizacionLU& factorizacion ,
vector<double> &resX) {
    int sizeB = _b.size();
    vector<double> resY(sizeB);

    int numfilas = _A.get_filas();

    // Calculo resY
    for (int i = 0 ; i < numfilas; i++) {
        double suma = 0;
        for (int j = 0; j < i; j++) {
            suma += factorizacion._L[i][j] * resY[j];
        }
        resY[i] = (_b[i] - suma) / factorizacion._L[i][i];
    }
}
```

```

// Calculo resX
for (int i = numfilas - 1 ; i>=0; i--) {
    double suma = 0;
    for (int j=i+1; j < numfilas; j++) {
        suma += factorizacion._U[i][j] * resX[j];
    }
    resX[i] = (resY[i] - suma) / factorizacion._U[i][i];
}

void SistemaEcuaciones::factorizar_LU( FactorizacionLU& lu) {
    int numfilas = _A.get_filas();
    int numcolumnas = _A.get_columnas();

    // L identidad
    lu._L.resize(numfilas , numcolumnas);

    for (int i = 0; i<numfilas; i++) {
        lu._L[i][i] = 1;
    }

    // U comienza siendo la matriz A
    lu._U = _A;

    for (int i = 0; i < numcolumnas - 1; i++) {
        for (int j = i+1; j < numfilas; j++) {

            // Aumenta la performance este chequeo
            if (lu._U[j][i] == 0) {
                // Ya hay un cero alli , no hay nada que hacer
                continue;
            }

            // Calculo el coeficiente multiplicador para L y U
            double multiplicadorU = (lu._U[j][i] / lu._U[i][i]);

            // Opero sobre la fila
            for (int k = i; k < numcolumnas; k++) {
                // modiflico los elementos de U
                lu._U[j][k] -= multiplicadorU * lu._U[i][k];
            }

            // Guardo los elementos de L
            lu._L[j][i] = multiplicadorU;
        }
    }
}

```