DEPARTAMENTO DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico I

Métodos Numéricos
Segundo Cuatrimestre de 2015

Integrante	LU	Correo electrónico
Iván Arcuschin	678/13	iarcuschin@gmail.com
Martín Jedwabny	885/13	martiniedva@gmail.com
José Massigoge	954/12	jmmassigoge@gmail.com
Iván Pondal	078/14	ivan.pondal@gmail.com

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Ciudad Universitaria - (Pabellón 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

1. Introducción

El objetivo de este Trabajo Práctico es implementar diferentes algoritmos de resolución de sistemas de ecuaciones lineales y experimentar con dichas implementaciones en el contexto de un problema de la vida real.

El problema a resolver es hallar la isoterma $500C^o$ en la pared de un Alto Horno. Para tal fin, deberemos particionar la pared del horno en puntos finitos, y luego resolver un sistema de ecuaciones lineales, en el cual cada punto de la pared interior y exterior del Horno es un dato, y las ecuaciones para los puntos internos satisfacen la ecuación del calor.

Los experimentos realizados se dividen en dos partes: Comportamiento del sistema y Evaluación de los métodos. En la primera parte, analizaremos con distintas instancias de prueba y se estudiará la proximidad de la isoterma buscada respecto de la pared exterior del horno. En la segunda parte, analizaremos el tiempo de computo requerido para la resolución del sistema en función de la granularidad de la discretización y analizaremos el escenario en el cual las temperaturas de los bordes varian a lo largo del tiempo.

2. Modelo

2.1. Descripción

El Alto Horno está definido por las siguientes variables:

- El radio de la pared exterior: $r_e \in \mathbb{R}$
- El radio de la pared interior: $r_i \in \mathbb{R}$
- La temperatura en cada punto de la pared: $T(r, \theta)$, donde (r, θ) se encuentra expresado en coordenadas polares, siendo r el radio y θ el ángulo polar de dicho punto.

  Son datos del problema, las temperaturas de la pared interior y exterior:

  - $T(r_i, \theta) = T_i \quad \textit{para todo punto } (r, \theta) \textit{ con } r \leq r_i$
  - $T(r_e, \theta) = T_e(\theta) \quad \textit{para todo punto } (r_e, \theta)$

La Figura 1 muestra las variables al tomar una sección circular del horno.

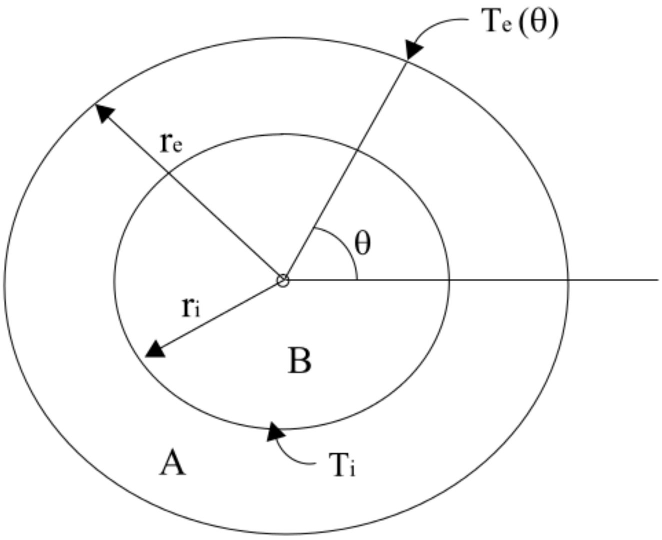


Figura 1: Sección circular del horno

En el estado estacionario, cada punto de la pared satisface la ecuación del calor:

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

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

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

Para encontrar estos valores, transformamos la ecuación (1) en un conjunto de ecuaciones lineales sobre las incógnitas $t_{j,k}$, evaluando (1) 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 (1) por medio de las siguientes fórmulas de diferencias finitas:

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

$$\frac{\partial T(r,\theta)}{\partial r}(r_j,\theta_k) \cong \frac{t_{j,k} - t_{j-1,k}}{\Delta r} \tag{3}$$

$$\frac{\partial^2 T(r,\theta)}{\partial \theta^2}(r_j,\theta_k) \cong \frac{t_{j,k-1} - 2t_{j,k} + t_{j,k+1}}{(\Delta\theta)^2} \tag{4}$$

Luego,

$$\frac{\partial^2 T(r,\theta)}{\partial r^2} + \frac{1}{r}\frac{\partial T(r,\theta)}{\partial r} + \frac{1}{r^2}\frac{\partial^2 T(r,\theta)}{\partial \theta^2} \cong \frac{t_{j-1,k} - 2t_{j,k} + t_{j+1,k}}{(\Delta r)^2} + \frac{1}{r}\frac{t_{j,k} - t_{j-1,k}}{\Delta r} + \frac{1}{r^2}\frac{t_{j,k-1} - 2t_{j,k} + t_{j,k+1}}{(\Delta\theta)^2} \tag{5}$$

Si agrupamos los términos para los t, la ecuación (5) nos queda:

$$\begin{aligned}&(\frac{1}{(\Delta\ r)^2} - \frac{1}{(\Delta\ r) * r})t_{j-1,k} + (\frac{1}{(\Delta\theta)^2 * r^2})t_{j,k-1} + (-\frac{2}{(\Delta\theta)^2 * r^2} - \frac{2}{(\Delta\ r)^2} + \\ &(\frac{1}{(\Delta\ r) * r})t_{j,k} + (\frac{1}{(\Delta\theta)^2 * r^2})t_{j,k+1} + (\frac{1}{(\Delta\ r)^2})t_{j+1,k} = 0\end{aligned} \tag{6}$$

2.2. Representación del sistema

Tenemos:

- $0 < j < m+1$: los indices de los radios
- $0 < k < n$: los indices de los ángulos
- $\alpha_{j,k} = \frac{1}{(\Delta\theta)^2 * r_j^2}$ donde $\Delta\theta = \theta_k - \theta_{k-1}$
- $\beta_{j,k} = \frac{1}{(\Delta\ r)^2}$ donde $\Delta r = r_j - r_{j-1}$
- $\gamma_{j,k} = \frac{1}{(\Delta\ r) * r_j}$ donde $\Delta r = r_j - r_{j-1}$

Entonces, a partir de la ecuación previa (6), podemos reemplazar usando los términos nuevos:

$$(\beta_{j,k} - \gamma_{j,k})t_{j-1,k} + (\alpha_{j,k})t_{j,k-1} + (-2\alpha_{j,k} - 2\beta_{j,k} + \gamma_{j,k})t_{j,k} + (\alpha_{j,k})t_{j,k+1} + (\beta_{j,k})t_{j+1,k} = 0 \tag{7}$$

A partir de lo detallado previamente, armamos el siguiente sistema de ecuaciones:

$$\begin{aligned} t_{0,0} &= T_i(0) \\ &\ldots \\ t_{0,n-1} &= T_i(n-1) \\ (\beta_{1,0} - \gamma_{1,0})t_{0,0} + (\alpha_{1,0})t_{1,n-1} + (-2\alpha_{1,0} - 2\beta_{1,0} + \gamma_{1,0})t_{1,0} + (\alpha_{1,0})t_{1,1} + (\beta_{1,0})t_{2,0} &= 0 \\ &\ldots \\ (\beta_{j,k} - \gamma_{j,k})t_{j-1,k} + (\alpha_{j,k})t_{j,k-1} + (-2\alpha_{j,k} - 2\beta_{j,k} + \gamma_{j,k})t_{j,k} + (\alpha_{j,k})t_{j,(k+1)\,\%n} + (\beta_{j,k})t_{j+1,k} &= 0 \\ &\ldots \\ (\beta_{m-1,n-1} - \gamma_{m-1,n-1})t_{m-2,n-1} + (\alpha_{m-1,n-1})t_{m-1,n-2} + \\ (-2\alpha_{m-1,n-1} - 2\beta_{m-1,n-1} + \gamma_{m-1,n-1})t_{m-1,n-1} + (\alpha_{m-1,n-1})t_{m-1,0} + (\beta_{m-1,n-1})t_{m,n-1} &= 0 \\ t_{m,0} &= T_e(0) \\ &\ldots \\ t_{m,n-1} &= T_e(n-1) \end{aligned}$$

En donde las primeras n ecuaciones, son las ecuaciones correspondientes al radio de la pared interior, r_i, para los distintos θ de la discretización. Luego para cada radio $r \neq r_i$ y $r \neq r_e$, se listan las ecuaciones correspondientes a los distintos θ de la discretización. Por último las últimas n ecuaciones son las ecuaciones correspondientes al radio de la pared exterior, r_e, para los distintos θ de la discretización.

Para representar el sistema de ecuaciones presentado, se utilizará una matriz cuadrada simple de tamaño $n(m+1)$, en donde las filas representan las ecuaciones detalladas previamente y las columnas cada punto de la discretización, $t_{j,k}$, implementada como un vector de vectores. A continuación, se muestra como quedaría la matriz para las ecuaciones de los puntos $t_{0,0}, t_{0,n-1}, t_{j,k}, t_{m,0}$ y $t_{m,n-1}$.

$$
\begin{array}{cc}
 & \begin{array}{ccccccccccccccc|c} \boldsymbol{t_{0,0}} & \cdots & \boldsymbol{t_{0,n-1}} & \cdots & t_{j-1,k} & \cdots & t_{j,k-1} & \boldsymbol{t_{j,k}} & t_{j,k+1} & \cdots & t_{j+1,k} & \cdots & \boldsymbol{t_{m,0}} & \cdots & \boldsymbol{t_{m,n-1}} & b \end{array} \\
\begin{array}{c} \boldsymbol{t_{0,0}} \\ \vdots \\ \boldsymbol{t_{0,n-1}} \\ \vdots \\ \boldsymbol{t_{j,k}} \\ \vdots \\ \boldsymbol{t_{m,0}} \\ \vdots \\ \boldsymbol{t_{m,n-1}} \end{array} &
\left[\begin{array}{ccccccccccccccc|c}
1 & \dots & 0 & \dots & 0 & \dots & 0 & 0 & 0 & \dots & 0 & \dots & 0 & \dots & 0 & t_{0,0} \\
\vdots & & \vdots & & \vdots & & \vdots & \vdots & \vdots & & \vdots & & \vdots & & \vdots & \\
0 & \dots & 1 & \dots & 0 & \dots & 0 & 0 & 0 & \dots & 0 & \dots & 0 & \dots & 0 & t_{0,n-1} \\
\vdots & & \vdots & & \vdots & & \vdots & \vdots & \vdots & & \vdots & & \vdots & & \vdots & \\
0 & \dots & 0 & \dots & \boldsymbol{\beta_{j,k} - \gamma_{j,k}} & \dots & \boldsymbol{\alpha_{j,k}} & \boldsymbol{-2\alpha_{j,k} - 2\beta_{j,k} + \gamma_{j,k}} & \boldsymbol{\alpha_{j,k}} & \dots & \boldsymbol{\beta_{j,k}} & \dots & 0 & \dots & 0 & 0 \\
\vdots & & \vdots & & \vdots & & \vdots & \vdots & \vdots & & \vdots & & \vdots & & \vdots & \\
0 & \dots & 0 & \dots & 0 & \dots & 0 & 0 & 0 & \dots & 0 & \dots & 1 & \dots & 0 & t_{m,0} \\
\vdots & & \vdots & & \vdots & & \vdots & \vdots & \vdots & & \vdots & & \vdots & & \vdots & \\
0 & \dots & 0 & \dots & 0 & \dots & 0 & 0 & 0 & \dots & 0 & \dots & 0 & \dots & 1 & t_{m,n-1}
\end{array}\right]
\end{array}
$$

Figura 2: Matriz del Sistema

Notese que, para las filas que representan las ecuaciones de los puntos de la pared distintos del interior e exterior, los coeficientes distintos de 0 se ubican en $t_{j-1,k}$, $t_{j,k}$, $t_{j+1,k}$, $t_{j,k-1}$ y $t_{j,k+1}$. Es importante destacar que el primer coeficiente distinto de 0 es siempre $\beta_{j,k} - \gamma_{j,k}$, mientras que el ultimo es siempre $\beta_{j,k}$, siendo la distancia entre ellos $2n$. Esto se debe al hecho de que, en nuestra disposicion de las ecuaciones del sistema en la matriz, en las columnas fijamos el radio y luego avanzamos con los distintos angulos para ese radio, para luego avanzar de radio, por lo cual $t_{j-1,k}$ y $t_{j+1,k}$ seran siempre el primer e ultimo coeficiente distinto de 0 en ese tipo de fila.
Esta caracteristica de la matriz la hace una matriz banda, en donde las diagonales $p, q = n$, estan compuestas por los valores $\beta_{j,k} - \gamma_{j,k}$ y $\beta_{j,k}$ respectivamente.

Ejemplo con $\boldsymbol{n, m+1 = 3}$:

$$
\begin{array}{cc}
 & \begin{array}{ccccccccc|c} t_{0,0} & t_{0,1} & t_{0,2} & t_{1,0} & t_{1,1} & t_{1,2} & t_{2,0} & t_{2,1} & t_{2,2} & b \end{array} \\
\begin{array}{c} t_{0,0} \\ t_{0,1} \\ t_{0,2} \\ t_{1,0} \\ t_{1,1} \\ t_{1,2} \\ t_{2,0} \\ t_{2,1} \\ t_{2,2} \end{array} &
\left[\begin{array}{ccccccccc|c}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & t_{0,0} \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & t_{0,1} \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & t_{0,2} \\
\beta_{1,0} - \gamma_{1,0} & 0 & 0 & -2\alpha_{1,0} - 2\beta_{1,0} + \gamma_{1,0} & \alpha_{1,0} & \alpha_{1,0} & \beta_{1,0} & 0 & 0 & 0 \\
0 & \beta_{1,1} - \gamma_{1,1} & 0 & \alpha_{1,1} & -2\alpha_{1,1} - 2\beta_{1,1} + \gamma_{1,1} & \alpha_{1,1} & 0 & \beta_{1,1} & 0 & 0 \\
0 & 0 & \beta_{1,2} - \gamma_{1,2} & \alpha_{1,2} & \alpha_{1,2} & -2\alpha_{1,2} - 2\beta_{1,2} + \gamma_{1,2} & 0 & 0 & \beta_{1,2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & t_{2,0} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & t_{2,1} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & t_{2,2}
\end{array}\right]
\end{array}
$$

3. Demostración: Eliminación Gaussiana sin pivoteo

Proposición 1. *Sea $A \in \mathbb{R}^{n(m+1)\times n(m+1)}$ la matriz obtenida para el sistema definido por las ecuaciones del Modelo, en donde $m+1$ y n corresponden a la cantidad de radios y angulos respectivamente de la discretizacion. Demostrar que es posible aplicar Eliminación Gaussiana sin pivoteo.*

Para poder demostrar la proposición, utilizamos los siguientes lemas, que demostramos a continuacion:

(L_1) A es una matriz banda.

(L_2) A es diagonal dominante (no estricta).

Lemma L_1. *A es una matriz banda*

Demostración. A partir del Modelo descripto en el punto anterior, vease Figura 2, podemos concluir que A es una matriz banda. □

Lemma L_2. *A es diagonal dominante (no estricta).*

Demostración. Por definición, una matriz es diagonal dominante (no estricta) cuando se cumple que, $\forall i = 0, 1, ..., n-1$:

$$|a_{i,i}| \geq \sum_{\substack{j=0 \\ j\neq i}}^{n-1} |a_{i,j}|$$

Esta desigualdad es evidente para las primeras y ultimas n filas, ya que el único valor distinto de 0 se encuentra en la diagonal. Falta ver el caso para el resto de A. Tenemos que probar que para fila, i, se cumple:

$$|-2\alpha - 2\beta + \gamma| \geq |\beta - \gamma| + |\alpha| + |\alpha| + |\beta|$$

Recordando que: $\alpha = \frac{1}{(\Delta\theta)^2 * r^2}$, $\beta = \frac{1}{(\Delta\ r)^2}$ y $\gamma = \frac{1}{(\Delta\ r) * r}$.

Entonces, por definición sabemos que $|\alpha| = \alpha$ y $|\beta| = \beta$.

Veamos que $|\beta - \gamma| = \beta - \gamma$. Supongamos que $\beta - \gamma < 0$:

$$\begin{aligned} \beta &< \gamma \\ \frac{1}{(\Delta\ r)^2} &< \frac{1}{(\Delta\ r) * r_j} \\ 1 &< \frac{\Delta\ r}{r_j} \\ 1 &< \frac{r_j - r_{j-1}}{r_j} \\ 1 &< 1 - \frac{r_{j-1}}{r_j} \end{aligned}$$

Como los radios son todos mayores a 0, llegamos a un absurdo, que vino de suponer $\beta - \gamma < 0$, por lo tanto $\beta - \gamma \geq 0$.

Luego, deberiamos probar que la desigualdad se cumple para los siguientes casos:

1. $-2\alpha - 2\beta + \gamma \geq 0$
2. $-2\alpha - 2\beta + \gamma < 0$

Veamos caso por caso:

1. $\boldsymbol{-2\alpha - 2\beta + \gamma \geq 0}$:

$$\begin{aligned}
-2\alpha - 2\beta + \gamma &\geq \beta - \gamma + \alpha + \alpha + \beta \\
2\gamma &\geq 4\beta + 4\alpha \\
\gamma &\geq 2\beta + 2\alpha \\
\gamma - 2\beta - 2\alpha &\geq 0
\end{aligned}$$

   Que vale por ser exactamente la hipótesis del caso 1.

2. $\boldsymbol{-2\alpha - 2\beta + \gamma < 0}$:

$$\begin{aligned}
-2\alpha - 2\beta + \gamma &\leq -\beta + \gamma - \alpha - \alpha - \beta \\
\gamma - \gamma &\leq 2\beta - 2\beta + 2\alpha - 2\alpha \\
0 &\leq 0
\end{aligned}$$

   Que vale siempre.

□

Demostración Proposición 1. Por L_2 sabemos que A es diagonal dominante (no estricta) y por definición del Modelo sabemos que $a_{0,0} = 1$.
Sea $A^{(1)}$ la matriz resultante luego de aplicar un paso de la Eliminación Gaussiana. Para toda fila $i = 1, \ldots, n(m+1) - 1$ se cumple que:

$$a_{i,j}^{(1)} = a_{i,j}^{(0)} - \frac{a_{0,j}^{(0)} a_{i,j}^{(0)}}{a_{0,0}^{(0)}}, \text{para } 1 \leq j \leq n(m+1) - 1$$

Sabemos que $a_{i,0}^{(1)} = 0$. Luego:

$$\begin{aligned}
\sum_{\substack{j=1 \\ j \neq i}}^{n(m+1)-1} |a_{i,j}^1| &= \sum_{\substack{j=1 \\ j \neq i}}^{n(m+1)-1} |a_{i,j}^{(0)} - \frac{a_{0,j}^{(0)} a_{i,0}^{(0)}}{a_{0,0}^{(0)}}| \\
&\leq \sum_{\substack{j=1 \\ j \neq i}}^{n(m+1)-1} |a_{i,j}^{(0)}| + \sum_{\substack{j=1 \\ j \neq i}}^{n(m+1)-1} |\frac{a_{0,j}^{(0)} a_{i,0}^{(0)}}{a_{0,0}^{(0)}}| \\
&\leq |a_{i,i}^{(0)}| - |a_{i,0}^{(0)}| + \frac{|a_{i,0}^{(0)}|}{|a_{0,0}^{(0)}|} \sum_{\substack{j=1 \\ j \neq i}}^{n(m+1)-1} |a_{0,j}^{(0)}| \\
&\leq |a_{i,i}^{(0)}| - |a_{i,0}^{(0)}| + \frac{|a_{i,0}^{(0)}|}{|a_{0,0}^{(0)}|} (|a_{0,0}^{(0)}| - |a_{0,i}^{(0)}|) \\
&= |a_{i,i}^{(0)}| - \frac{|a_{i,0}^{(0)}||a_{0,i}^{(0)}|}{|a_{0,0}^{(0)}|} \\
&\leq |a_{i,i}^{(0)} - \frac{a_{i,0}^{(0)} a_{0,i}^{(0)}}{a_{0,0}^{(0)}}| = |a_{i,i}^{(1)}|
\end{aligned}$$

- Por lo tanto, el dominio diagonal no estricto se establece en los renglones $1, \ldots, n(m+1) - 1$, y como el primer renglon de $A^{(1)}$ y de A son iguales, $A^{(1)}$ sera diagonal dominante no estricto.
- Para poder aplicar un paso más de la Eliminación Gaussiana, es necesario que $a_{1,1}^{(1)} \neq 0$. Sabemos por definición del Modelo que $a_{1,1}^{(0)} = 1$, y como $a_{1,0}^{(0)} = 0$, por definición de la Eliminación Gaussiana $a_{1,1}^{(1)} = 1$. Esta situación se repite para las n primeras filas de A, ya que para $\forall i = 0, ..., n-1$ $\forall j = 0, ..., n(m+1) - 1 \wedge j \neq i, a_{i,j}^{(0)} = 0$, por lo tanto podemos afimar que podemos realizar los primeros $n - 1$ pasos de la Eliminación Gaussiana, y que la matriz $A^{(n-1)}$ será diagonal dominante no estricta (repitiendo el procedmiento hecho para $A^{(1)}$).

- Ahora veamos que sucede para los pasos $n \leq k \leq n(m+1) - n - 1$ de la Eliminación Gaussiana.
  - Para el paso $k = n$, utilizamos que A es una matriz banda (L_1), en particular la banda esta definida por las filas $i = n, .., n(m+1) - n - 1$, en donde el último coeficiente distinto de 0 de cada fila es β_i, el valor de la banda derecha q.
  - Sabemos que $\beta_i > 0$, por definición de los β. Al realizar el paso k, sabemos que la matriz resultante $A^{(k)}$ será diagonal dominante no estricta, (mismo procedimientos que en las filas precedentes) y también sabemos que $\beta_k^{(k)} = \beta_k^{(0)}$, debido al hecho que $\forall u = 0, .., k-1, a_{u,j}^{(u)} = 0$, donde j es el indice de la columna de $\beta_k^{(0)}$.
  - Como $A^{(k)}$ es diagonal dominante no estricta, sabemos que $a_{k,k}^{(k)} \geq \beta_k^{(k)}$, y como $\beta_k^{(k)} > 0$, entonces $a_{k,k}^{(k)} > 0$, por lo tanto podemos realizar un paso más de la Eliminación Gaussiana.
  - Esta situación se repite para el resto de las pasos $n + 1 \leq k \leq n(m+1) - n - 1$.
- Por último queda por ver que sucede con los pasos $n(m+1) - n \leq k \leq n(m+1) - 1$. Esta situación es idéntica al de los primeros n pasos, ya que en A $\forall i = n(m+1) - n, ..., n(m+1) - 1$ $\forall j = 0, ..., n(m+1) - 1 \wedge j \neq i, a_{i,j}^{(0)} = 0$. Es decir, el valor de la diagonal de estas filas no será alterado por la Eliminación Gaussiana, y como en A su valor es 1, podemos aplicar los pasos de la Eliminación Gaussiana.
- Por todo lo expuesto, podemos concluir que es posible aplicar a A el método de Eliminación Gaussiana sin pivoteo.

$\square$

4. Implementación

4.1. Eliminación Gaussiana

4.1.1. Descripción del método

El método de Eliminación Gaussiana consiste en una serie de pasos que permiten resolver un sistema de ecuaciones lineales de, en principio, n ecuaciones y n variables.
Sea A $\in \mathbb{R}^{n\times\ n}$ la matriz tal que el elemento en la fila i y columna j ($a_{i,j}$) representa el coeficiente de la variable j en la ecuación i. Y sea b $\in \mathbb{R}^n$ el vector tal que el elemento en la fila i (b_i) representa el termino independiente en la ecuación i.
Podemos dividir el método en 2 partes centrales:

1. Llevar la matriz A a una forma **Triangular Superior**, es decir, una matriz equivalente a A tal que tiene ceros debajo de los elementos de la diagonal. El siguiente pseudocódigo muestra como es el algoritmo para realizar esta tarea:

```
Para j desde 0 hasta n-1 hacer:
    Poner pivote = A[j][j]
    Para i desde j+1 hasta n-1 hacer:
        Poner coeficiente = A[i][j] / pivote
        Poner A[i][j] = 0
        Para k desde j+1 hasta n-1 hacer:
            Poner A[i][k] = A[i][k] - coeficiente * A[j][k]
        Fin para
        b[i] = b[i] - coeficiente * b[j]
    Fin para
Fin para
```

   Notese que no validamos que la variable “pivote” sea distinta de cero. Esto es así ya que por la forma en la que se modeló el problema el pivote siempre es distinto de cero.

2. **Resolver el sistema equivalente**. Para esto, vamos a utilizar que la matriz es Triangular Superior. La idea es empezar despejando el valor de la n-ésima variable, luego usar este valor para despejar la $(n-1)$-ésima variable, y así sucesivamente hasta la primera variable. En pseudocódigo:

```
Poner X = vector de n elementos
Para i desde n-1 hasta 0 hacer:
    Poner X[i] = b[i]
    Para j desde i+1 hasta n-1 hacer:
        Poner X[i] = X[i] - U[i][j] * X[j]
    Fin para
    Poner X[i] = X[i] / U[i][i]
Fin para
```

   Donde U es la matriz que calculamos en el paso 1.

4.1.2. Utilizando que la matriz es Banda

Si miramos la matriz con la cual representamos el modelo del problema, podemos ver que alrededor de los elementos de la diagonal hay una “banda” de tamaño $2n$. Es decir, si quisieramos poner elementos debajo del elemento $a_{i,i}$, nos bastaría con modificar las filas desde $i+1$ hasta $i+2n+1$, ya que $\forall\ a_{j,i},\ j > i+2n+1 \implies a_{j,i} = 0$.
Usando esto podemos optimizar significativamente el primer paso de la Eliminación Gaussiana, que consiste en hallar la matriz equivalente Triangular Superior. El pseudocódigo es el siguiente:

```
Para j desde 0 hasta n-1 hacer:
    Poner pivote = A[j][j]
```

```
    Poner inicioBanda = max(i+1, n)
    Poner finBanda = min(n, inicioBanda + n)
    Para i desde inicioBanda hasta finBanda hacer:
        Si A[i][j] != 0 hacer:
            Poner coeficiente = A[i][j] / pivote
            Poner A[i][j] = 0
            Para k desde j+1 hasta n-1 hacer:
                Poner A[i][k] = A[i][k] - coeficiente * A[j][k]
            Fin para
            b[i] = b[i] - coeficiente * b[j]
        Fin si
    Fin para
Fin para
```

4.2. Factorización LU

Dada la matriz A $\in \mathbb{R}^{n\times\ n}$, los vectores x $\in \mathbb{R}^n$ y b $\in \mathbb{R}^n$ y la ecuación $Ax = b$, esta técnica de resolución descompone la matriz en cuestión de la siguiente manera:

$A = LU$ con L,U $\in \mathbb{R}^{n\times\ n}$
L es triangular inferior con 1's en los elementos de su diagonal
U es triangular superior
Y por lo tanto, $Ux = y$, $Ly = b$ reemplazando la ecuación original

Ahora, como vimos en las clases prácticas y teóricas, estas dos matrices son únicas y la forma de obtenerla es haciendo los pasos de eliminación gaussiana (descripta anteriormente) y guardarnos los coeficientes por los cuáles vamos multiplicando las filas para triangular las de abajo. A modo de ejemplo:

$$\underbrace{\begin{pmatrix} 5 & -1 & 2 \\ 10 & 3 & 7 \\ 15 & 17 & 19 \end{pmatrix}}_{A} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 4 & 1 \end{pmatrix}}_{L} \times \underbrace{\begin{pmatrix} 5 & -1 & 2 \\ 0 & 5 & 3 \\ 0 & 0 & 1 \end{pmatrix}}_{U}$$

En términos de espacio, guardar las matrices L y U se puede hacer en una 'misma' matriz si se la interpreta de forma diferente. Lo que hay que tener en cuenta que la única parte relevante de U es de la diagonal para arriba (con la diagonal inclusive), pero de L ya sabemos que la diagonal contiene 1's y solo nos interesa saber la parte de abajo. Entonces la matriz anterior se podría guardar así en memoria:

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 4 & 1 \end{pmatrix}}_{L} \underbrace{\begin{pmatrix} 5 & -1 & 2 \\ 0 & 5 & 3 \\ 0 & 0 & 1 \end{pmatrix}}_{U} \leftarrow \begin{pmatrix} 5 & -1 & 2 \\ 2 & 5 & 3 \\ 3 & 4 & 1 \end{pmatrix}$$

Además, como ya demostramos que la matriz se puede triangular mediante eliminación gaussiana sin pivoteo, sabemos que la factorización LU existe (y no tenemos que tomar matrices de permutación).
Teniendo todo eso en cuenta, encontramos la factorización (con el formato explicado antes) con este procedimiento:

```
Sea A la matriz que queremos factorizar
Sea n la cantidad de filas (y columnas, porque es cuadrada) de A
Sea M una matriz de n*n (donde guardaremos la respuesta)

FactorizarLU(A, n, M)
    Copiar la primer fila de A hacia M
    Para i desde 0 hasta n-2
        Para j desde i+1 hasta n-1
            Poner c = A[j][i]/A[i][i]
            Poner M[j][i] = c
            Para k desde i+1 hasta n-1
                M[j][k]-= c*A[i][k]
    devolver M
```

Ya teniendo la factorización LU de A, solo quedaría resolver las ecuaciones $Ux = y$, $Ly = b$ adaptando los algoritmos clásicos de triangulación de sistemas lineales para que tomen en cuenta el formato con el cual guardamos la factorización LU de A en una sola matriz:

```
Sean A, n y M definidos como el pseudocodigo anterior
Sea b el vector de n elementos correspondiente a la ecuacion a resolver

ResolverSistema(A,n,b)
    M = Crear matriz cuadrada de n*n
    FactorizarLU(A,n,M)
    y = Crear vector de n elementos
    ResolverTriangularInferiorLU(M,b,n,y) \\ Ly = b
    x = Crear vector de n elementos
    ResolverTriangularSuperiorLU(M,y,n,x) \\ Ux = y
    devolver x

ResolverTriangularInferiorLU(M,y,n,x)
    Para i desde 0 hasta n-1
        Poner x[i] = b[i]
        Para j desde 0 hasta i-1
            Poner x[i] = x[i] - M[i][j] * x[j]
        // la diagonal es 1, asi que no hay que dividir por nada

ResolverTriangularSuperiorLU(M,b,n,y) \\ seria igual al procedimiento de
    la seccion 4.1.1.2
    Para i desde n-1 hasta 0
        Poner x[i] = b[i]
        Para j desde i+1 hasta n-1
            Poner x[i] = x[i] - M[i][j] * x[j]
        Poner x[i] = x[i] / M[i][i]
```

Finalmente, es importante aclarar que la ventaja de utilizar la factorización LU de una matriz para resolver sistemas de ecuaciones lineales radica en que una vez computada la factorización, solo queda resolver sistemas triangulares. Es decir, si bien encontrar la factorización LU no es una operación barata, una vez que la obtenemos, solo nos queda resolver las ecuaciones $Ux = y$, $Ly = b$ donde U y L son triangulares. Por lo tanto, una vez que calculamos L y U no hace falta hacer eliminación gaussiana devuelta aunque cambiemos el 'b', a diferencia de la eliminación gaussiana normal.

4.3. Determinación de la Isoterma

Recordemos que nuestra discretización particiona una sección circular del Alto Horno de la siguiente forma:

- $0 = \theta_0 < \theta_1 < ... < \theta_n = 2\pi$ en n ángulos discretos, y
- $r_i = r_0 < r_1 < ... < r_m = r_e$ en $m + 1$ radios discretos

Luego, para cada ángulo j tenemos los puntos: $t_{i,j}$ con $0 \leq i \leq m$.
Entonces, hallar la isoterma C equivale a, para cada ángulo j, hallar el radio r_C tal que $T(r_C, \theta_j) = C$.

4.3.1. Promedio simple

Este método consiste en, dado un ángulo j, buscar un punto $t_{i,j}$ en la solución del sistema tal que $t_{i,j} \leq C \leq t_{i+1,j}$.
Una vez hallado este punto, tenemos que $r_C = \frac{r_i + r_{i+1}}{2}$.

4.3.2. Búsqueda binaria mediante sistemas de ecuaciones

Lo que hacemos en este método es, una vez halladas las temperaturas del alto horno según nuestra discretización, sub-discretizar los radios hasta encontrar la isoterma. Esto se realiza encontrando, para cada angulo, dos radios entre los cuáles está la isoterma. Sean r_{i1} y r_{i2} estos radios, creamos un nuevo sistema de ecuaciones con tres radios: los dos anteriores (de los cuales ya sabemos su temperatura) y el radio medio entre esos dos (cuya temperatura queremos hallar). De alguna manera se puede ver como un nuevo horno discretizado con 3 radios, r_{i1} y r_{i2} y el del medio. Este procedimiento se realiza varias veces (tomando radios medios todo el tiempo) asi convergiendo a la isoterma de la misma forma que en el algoritmo de Búsqueda Binaria. Es decir, ya teniendo $t_{i,j}$ para todo i,j con $0 \leq i \leq m$ y $0 \leq j \leq n - 1$, hacemos lo siguiente:

```
Sea PRECISION = 0.00001 \\ valor configurable, en este caso el mismo que
    nuestro codigo

CalcularIsotermaBinaria
    Poner solucion = Crear vector de n elementos (para contener el radio
        en el que esta la isoterma en cada angulo)
    Para j desde 0 hasta n-1
        Poner i2 = el primer radio que cumple t_{i2,j} < isoterma
        Poner i1 = i2-1 \\tener en cuenta que a menores radio, mayores
           temperaturas
        Poner a2 = r_{i2}
        Poner a1 = r_{i1}
        Mientras (Valor Absoluto(T(a1,θ_j)-isoterma) > PRECISION)
            Poner ah = (a1+a2)/2
            Hallar T(ah,θ_j)
            Si (T(ah,θ_j) < isoterma) {
                Poner a2 = ah
            Sino
                Poner a1 = ah
            }
        Poner solucion[j] = a1
    devolver solucion
```

4.3.3. Regresión lineal (Linear fit)

Este método utiliza el algoritmo de regresión lineal para, dado un ángulo j, y usando todos los puntos $t_{i,j}$ con $0 \leq i \leq m$, hallar una función lineal que aproxime dichos puntos lo mejor posible. Como la función que estamos buscando es lineal, es de la forma: $y(x) = a + bx$, donde b es el coeficiente principal, a el termino independiente, x es un radio sobre el ángulo j e $y(x)$ es la temperatura para dicho radio.

Luego, el algoritmo de regresion lineal basicamente utiliza la minimización de la suma de las distancias al cuadradado desde los puntos a la función lineal. Esto se logra calculando la derivada con respecto a a y b y fijando estos en cero.
Entonces, si definimos:

$$\overline{x} = \frac{1}{m}\sum_{i=0}^{m} r_i \qquad \overline{y} = \frac{1}{m}\sum_{i=0}^{m} t_{i,j}$$

$$S_x = \sum_{i=0}^{m} (r_i - \overline{x})^2 \qquad S_{xy} = \sum_{i=0}^{m} (r_i - \overline{x})(t_{i,j} - \overline{y})$$

Tenemos que:

$$b = \frac{S_{xy}}{S_x} \qquad a = \overline{y} - b\overline{x}$$

Una vez obtenidos a y b, para hallar la isoterma C en el ángulo j, basta con calcular:

$$r_C = |C - a|/b$$

En pseudocódigo:

```
Poner solucion = vector de n elementos
Para j desde 0 hasta n hacer:
    Poner avgX = 0
    Poner avgY = 0
    Para i desde 0 hasta m hacer:
        Poner avgX = avgX + r_i
        Poner avgY = avgY + t_{i,j}
    Fin para
    Poner avgX = avgX / m
    Poner avgY = avgY / m
    Poner numerador = 0
    Poner denominador = 0
    Para i desde 0 hasta m hacer:
        Poner numerador = numerador + (r_i - avgX) * (t_{i,j} - avgY)
        Poner denominador = denominador + (r_i - avgX) * (r_i - avgX)
    Fin para
    Si denominador == 0 hacer:
        Poner denominador = 1
    Fin si
    Poner coeficiente = numerador / denominador
    Poner independiente = avgY - slope * coeficiente
    Poner solucion[j] = abs(C - independiente) / coeficiente
Fin para
```

4.4. Evaluación del peligro de la estructura

Una vez obtenida la isoterma C, queremos evaluar la peligrosidad de la estructura en función de la distancia de la isoterma a la pared externa del horno. En este sentido, estamos asumiendo que la temperatura C es elevada y que mientras más cercana está la temperatura de la pared externa a C, entonces más peligrosa es la estructura.
En base a esto, proponemos dos medidas distintas para evaluar la peligrosidad.

4.4.1. Proximidad porcentual simple

Para cada ángulo j, podemos calcular el coeficiente porcentual $\Delta_j(C) = (r_e - r_C)/(r_e - r_i)$, donde r_e es el radio de la pared externa del horno, r_i el radio de la pared interna, y r_C el radio de la isoterma C para el ángulo j.
Notese que $r_i \leq r_C \leq r_e$, y por lo tanto si $r_C = r_i \implies \Delta_j(C) = 1$, y si $r_C = r_e \implies \Delta_j(C) = 0$.
De esta forma, podemos definir un ε_C, con $0 < \varepsilon_C < 1$, tal que decimos que la estructura se encuentra en peligro si:

$$\varepsilon_C \geq \min_{1 \leq j \leq n-1} (\Delta_j(C))$$

4.4.2. Proximidad porcentual promediada

En la medida anterior, podría pasar que para un j' dado $\Delta_{j'}(C) < \varepsilon_C$ pero el resto de los $\Delta_j(C)$ sean mayores a ε_C, en cuyo caso, igualmente la estructura sería catalogada como peligrosa.
Entonces, querríamos dar una medida de la peligrosidad de la estructura que tome en cuenta todos los ángulos. Para esto, vamos a tomar el promedio de todos los $\Delta_j(C)$, definidos como en la medida anterior para cada ángulo j, y decimos que la estructura se encuentra en peligro si:

$$\Delta(C) = \frac{\sum_{j=1}^{n} \Delta_j(C)}{n} \leq \varepsilon_C$$

5. Experimentación

5.1. Instancias de prueba

Para ser lo más realista posible, se investigó[1] acerca de los diferentes tamaños de Altos Hornos, así como de las temperaturas que alcanzan. En base a esto, se armaron 3 instancias de prueba distintas (las discretizaciones se eligen después):

- Alto Horno de Plomo:
  - Radio pared interna: $r_i = 5$
  - Radio pared externa: $r_e = 6$
  - Temperatura pared interna: $T(r_i, \theta_j) = 327\ C^{o}$, $\forall\ 1 \leq j \leq n$
  - Temperatura pared externa: $T(r_e, \theta_j) = 20\ C^{o}$, $\forall\ 1 \leq j \leq n$
- Alto Horno de Zinc:
  - Radio pared interna: $r_i = 7$
  - Radio pared externa: $r_e = 9$
  - Temperatura pared interna: $T(r_i, \theta_j) = 419{,}5\ C^{o}$, $\forall\ 1 \leq j \leq n$
  - Temperatura pared externa: $T(r_e, \theta_j) = 20\ C^{o}$, $\forall\ 1 \leq j \leq n$
- Alto Horno de Hierro:
  - Radio pared interna: $r_i = 11$
  - Radio pared externa: $r_e = 15$
  - Temperatura pared interna: $T(r_i, \theta_j) = 1538\ C^{o}$, $\forall\ 1 \leq j \leq n$
  - Temperatura pared externa: $T(r_e, \theta_j) = 20\ C^{o}$, $\forall\ 1 \leq j \leq n$

5.2. Número de condición

Antes de empezar a experimentar, queremos saber para cada instancia de prueba que tamaño de discretizaciones son aceptables, en terminos del Número de Condición (K). En el caso de que este fuera muy grande, al resolver el sistema no tendríamos garantía de que la solución hallada fuera efectivamente buena. Esta medida se calcula de la forma:

$$K(A) = ||A||_\infty * ||A^{-1}||_\infty = \left(\max_{1 \leq i \leq n\times(m+1)} \sum_{j=1}^{n\times(m+1)} |a_{i,j}| \right) * \left(\max_{1 \leq i \leq n\times(m+1)} \sum_{j=1}^{n\times(m+1)} |a^{-1}_{i,j}| \right)$$

Tomando una discretización inicial de 30 ángulos ($n = 30$) y 30 radios ($m = 30$), tenemos que:

- Para el Alto Horno de Plomo, el número de condición es: 1678.42
- Para el Alto Horno de Zinc, el número de condición es: 419.448
- Para el Alto Horno de Hierro, el número de condición es: 104.844

Pero observemos cual es el espesor del horno para cada instancia de prueba:

- Para el Alto Horno de Plomo, el espesor de la pared es de $r_e - r_i = 1$.
- Para el Alto Horno de Zinc, el espesor de la pared es de $r_e - r_i = 2$.
- Para el Alto Horno de Hierro, el espesor de la pared es de $r_e - r_i = 4$.

[1] http://www.britannica.com/technology/blast-furnace

Luego, plantemos la siguiente **Hipotesis:** *el número de condición aumenta con la cantidad de ecuaciones (relacionado a la cantidad de angulos y radios) y disminuye al aumentar el espesor de la pared (diferencia entre el radio externo e interno)*. Intuitivamente, podemos pensar el espesor de la pared como el espacio a resolver, y al aumentar las ecuaciones aumenta la redundancia (disminuye la indepencia) del sistema. Podemos entonces probar con una discretización de 60 ángulos ($n = 60$) y 60 radios ($m = 60$);

- Para el Alto Horno de Plomo, el número de condición es: 6957.49
- Para el Alto Horno de Zinc, el número de condición es: 1739.99
- Para el Alto Horno de Hierro, el número de condición es: 435.281

Vemos que el resultado corrobora nuestra hipotesis.
Más aún, el mayor número de condición (con las discretizaciones vistas) es 6957.49, que es relativamente aceptable[2].

5.3. Calidad de las soluciones

Ahora que ya vimos que el Número de Condición para los hornos y discretizaciones propuestos es aceptable, podemos ver efectivamente cual es la calidad de la solución:
Llamemos $\tilde{x}$ al resultado de resolver el sistema $Ax = b$, y $\tilde{b}$ al resultado de multiplicar A por $\tilde{x}$, donde A es la matriz del sistema y b es el termino independiente, dado por las temperaturas de la pared exterior, la pared interior y el término independiente de la ecuación del calor.

Luego, podemos definir $\Delta(b, \tilde{b}) = \max\limits_{1 \leq i \leq n \times (m+1)} |b[i] - \tilde{b}[i]|$.
Al resolver los sistemas con el método de Eliminación Gaussiana obtenemos los siguientes resultados:

Instancia de prueba	$\Delta(b, \tilde{b})$
H. Plomo - 30x30	1.74623e-10
H. Plomo - 60x60	4.36557e-11
H. Zinc - 30x30	5.82077e-11
H. Zinc - 60x60	9.31323e-10
H. Hierro - 30x30	2.32831e-10
H. Hierro - 60x60	2.32831e-10

Cuadro 1: Calidad de las soluciones para las distintas instancias de pruebas y discretizaciones.

Viendo los distintos $\Delta(b, \tilde{b})$ podemos concluir que tenemos, en el peor caso, una precision de 10^{-10}, lo cual es muy aceptable.

5.4. Comportamiento del sistema

5.4.1. Distintas discretizaciones

En primer lugar, para cada horno mencionado en las instancias de pruebas, vamos a definir una isoterma.

- Para el Alto Horno de Plomo, la isoterma buscada será de: $200\ C^o$
- Para el Alto Horno de Zinc, la isoterma buscada será de: $350\ C^o$
- Para el Alto Horno de Hierro, la isoterma buscada será de: $1300\ C^o$

Luego, para cada horno se resolvió el sistema de ecuaciones mediante factorización LU, y se utilizaron los distintos métodos propuestos en la sección Implementación para hallar las isotermas correspondientes a partir de las soluciones de los sistemas.

[2]Fuente: *Numerical Mathematics and Computing, by Cheney and Kincaid.*, con un número de condición de 10^k, se pierde k dígitos de precisión. Como el formato *double* maneja una precisión de al menos 15 dígitos, los valores obtenidos son aceptables.

Los resultados obtenidos se muestran a continuación:

Instancia de prueba	Isoterma Promedio	Isoterma Regresión Lineal	Isoterma Búsqueda Binaria
H. Plomo - 30x30	5.3965	5.3988	5.3916
H. Plomo - 60x60	5.3983	5.3986	5.3916
H. Zinc - 30x30	7.3103	7.3060	7.3126
H. Zinc - 60x60	7.3220	7.3053	7.3127
H. Hierro - 30x30	11.4827	11.5226	11.5477
H. Hierro - 60x60	11.5762	11.5208	11.5479

Cuadro 2: Resultados obtenidos para las distintas instancias de prueba
y distintos métodos para hallar la isoterma.

Nota: como las instancias de prueba tienen la misma temperatura de la pared interior para todos sus ángulos, y la misma temperatura de la pared exterior para todos sus ángulos, el radio de la isoterma tiene el mismo valor para todos los ángulos. Es por eso que solo se presenta un valor en el Cuadro 2 y no n valores.

Y, a modo de ejemplo, las figuras 3 y 4 muestran para el Alto Horno de Hierro la úbicación de la isoterma con respecto a las paredes, y la evolución de la temperatura dentro de las mismas. Los gráficos para los otros Hornos son muy similares por lo que se omiten.

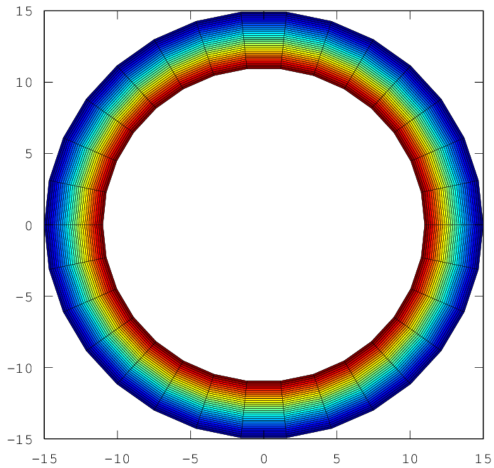


Figura 3: Evolución de las temperaturas para el Alto Horno de Hierro

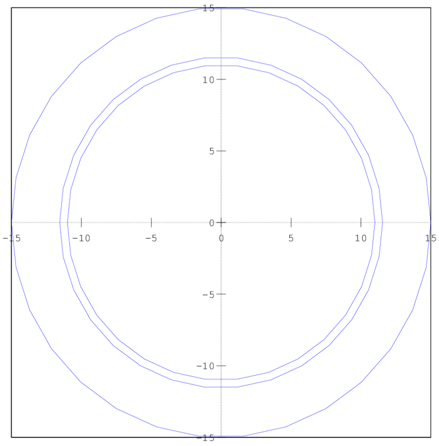


Figura 4: Ubicación de la isoterma para el Alto Horno de Hierro
(utilizando el método de Búsqueda Binaria)

Analizando los valores obtenidos en el Cuadro 2:

- En todas las instancias de pruebas, la diferencia entre la discretización de 30x30 y la discretización de 60x60 no supera nunca $0{,}1$, con lo que podemos concluir que el aumento en la discretización es significativo a nivel de 1 decimal. Esto es importante ya que en el Alto Horno de Plomo por ejemplo, el espesor es de 1 metro por lo que la isoterma hallada con una discretización de 30x30 podría variar $\pm 0{,}1m$, que en este espesor es exactamente $\pm 0{,}1\,\%$.

- Teniendo en cuenta que al realizar la discretización de la ecuación del calor para construir la representación del sistema tenemos cierta perdida de precisión, y que la naturaleza misma de la arítmetica finita que realiza el computador puede llevar a errores de precisión, no podemos garantizar que las isotermas halladas sean 100 % confiables.

  Sin embargo, al comparar los distintos métodos para hallar la isoterma, dentro del mismo error de precisión, podemos ver que el mejor método es el de *Búsqueda Binaria*, ya que con una discretización de 30x30 obtiene unos resultados muy similares (mirando los primeros 4 decimales) que con una discretización de 60x60. En el otro extremo, tenemos que el método de *Promedio* es el más burdo, teniendo una variación importante entre las diferentes discretizaciones. Esto último lo atribuimos que este método no tiene en cuenta la forma en la que se se resuelve el sistema original, realizando un simple promedio entre dos cotas de la isoterma, mientras que el método *Búsqueda Binaria* utiliza internamente el método de Eliminación Gaussiana.

  En un punto intermedio se encuentra el método de *Regresión Lineal*, que varía al aumentar la discretización pero cada vez menos, estabilizandose en un valor a medida que se aumenta la discretización.

5.4.2. Proximidad de la isoterma

Para estudiar la proximidad de la isoterma buscada respecto de la pared exterior del horno vamos a usar las siguientes instancias de prueba:

- Alto Horno de Plomo, variando la temperatura de la pared externa a $180\ C^o$, con una discretización de 15 ángulos y 15 radios.

- Alto Horno de Plomo, variando la temperatura de la pared externa a $180\ C^o$, con una discretización de 30 ángulos y 30 radios.

- Alto Horno de Plomo, variando la temperatura de la pared externa a $180\ C^o$, con una discretización de 60 ángulos y 60 radios.

Se experimentará con los 3 métodos propuestos para hallar isotermas (Promedio, Regresión Lineal y Búsqueda Binaria) y con las 2 medidas de peligrosidad propuestas (Proximidad Porcentual Simple y Promediada).
La siguiente tabla muestra los resultados obtenidos:

Instancia de prueba	Isoterma Promedio	Isoterma Regresión Lineal	Isoterma Búsqueda Binaria
H. Plomo - 15x15	5.8214	5.8500	5.8529
H. Plomo - 30x30	5.8448	5.8495	5.8529
H. Plomo - 60x60	5.8559	5.8493	5.8529

Cuadro 3: Resultados obtenidos para las distintas instancias de prueba
y distintos métodos para hallar la isoterma.

Luego, si tomamos un $\varepsilon_C = 0{,}25$ tenemos que para todos lo métodos y discretizaciones la estructura se encuentra en peligro ($r_e - (r_e - r_i) * 0{,}25 = 5{,}75$). Y tomando un $\varepsilon_C = 0{,}10$ tenemos que para todos lo métodos y discretizaciones la estructura **no** se encuentra en peligro ($r_e - (r_e - r_i) * 0{,}10 = 5{,}90$).
Por último, tomando un valor intermedio de $\varepsilon_C = 0{,}15$ ($r_e - (r_e - r_i) * 0{,}15 = 5{,}85$) tenemos que:

Instancia de prueba	Isoterma Promedio	Isoterma Regresión Lineal	Isoterma Búsqueda Binaria
H. Plomo - 15x15	Fuera de peligro	**Peligrosa**	**Peligrosa**
H. Plomo - 30x30	Fuera de peligro	Fuera de peligro	**Peligrosa**
H. Plomo - 60x60	**Peligrosa**	Fuera de peligro	**Peligrosa**

Cuadro 4: Evaluación de la peligrosidad de la estructura con $\varepsilon_C = 0{,}15$

Nota: en el Cuadro 4 es indistinto si la medida de peligrosidad utilizada es la Proximidad Porcentual Simple o Promediada, ya que las instancias de prueba tienen la misma temperatura de la pared interior para todos sus ángulos, la misma temperatura de la pared exterior para todos sus ángulos,y por lo tanto el radio de la isoterma tiene el mismo valor para todos los ángulos, por lo que es equivalente realizar el promedio de todos los ángulos a evaluar cualquiera de los ángulos de la isoterma.

Analizando los valores obtenidos en el Cuadro 3:

- Para el método de *Búsqueda Binaria* no influyó los primero 4 decimales que se varíe la discretización. Esto lo atribuimos a que éste método utiliza la misma mecánica que el método de Eliminación Gaussiana y continúa hasta que satisface una precisión determinada, por lo que es entendible que las distintas discretizaciones den resultados parecidos en los primeros decimales.

- El método de *Regresión Lineal* converge a medida que aumentamos la discretización, cada vez variando menos. Esto lo atribuimos a que a medida que aumentamos la discretización, también aumenta la cantidad de puntos que tiene en cuenta el método para cada ángulo, convergiendo en una función lineal en particular.

- Para el método de *Promedio*, la variación entre las distintas discretizacion es bastante, pero podemos ver que para la última discretización (60x60), este método calcula una isoterma muy similar al método de *Búsqueda Binaria*.

- En base a esto último, es interesante notar que lo que parece ser el valor más confiable de la isoterma ($5{,}85\ldots$) es aproximado tanto utilizando *Búsqueda Binaria* con discretización 15x15 como haciendo *Promedio* con discretización 60x60. En este aspecto, podemos inferir que el primero de los dos, por la forma en la que funciona, puede partir de cualquier discretización y arrivar a resultados parecidos.

- Para finalizar, tomando el promedio de las distintas isotermas halladas para la discretización de 60x60 tenemos: $5{,}8527$, que usando un $\varepsilon_C = 0{,}15$ definiriamos como **Peligrosa**, lo que se corresponde con el promedio de la “peligrosidad”: *Promedio* y *Búsqueda Binaria* dicen que la estructura es **Peligrosa** y *Regresión Lineal* dice que está fuera de peligro.

5.5. Evaluación de los métodos

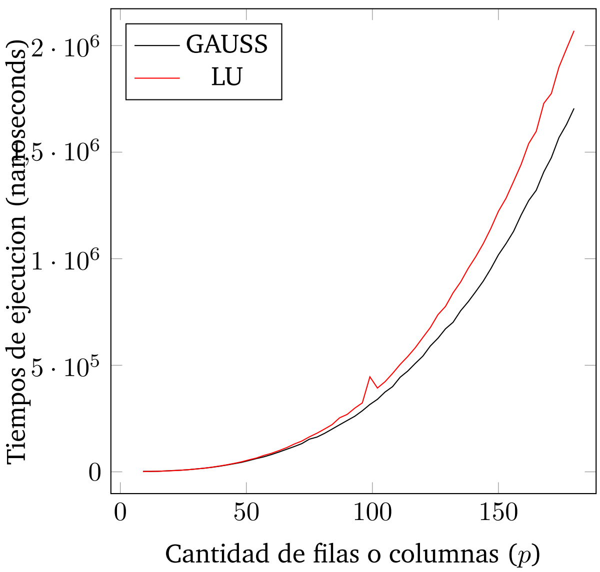
5.5.1. Tiempo de cómputo

En esta sección queremos evaluar los tiempos de cómputo de los métodos implementados, teniendo como hipótesis la complejidad teórica calculada para nuestras implementaciones, siendo la misma $\mathcal{O}(p^3)$ para ambos métodos, donde $p = n * (m + 1)$. A su vez queremos ver si existen diferencias de tiempos de cómputo entre los métodos.

Para concretar este objetivo, realizamos el siguiente experimento:

- Generamos casos tests a partir de los valores del Alto Horno de Hierro, vease Sección 5.1, variando aleatoriamente la cantidad de puntos de la discretización y seteando el $ninst$ en 1.
- El tamaño de la matriz generada, a partir de la entrada, varia de $9 * 9$ hasta $180 * 180$.
- Los tiempos de ejecución se midieron con la biblioteca `chrono` y estos fueron convertidos a nanosegundos.
- Para cada caso de test, se midió la ejecución de los métodos Eliminación Gaussiana y LU.
- Para cada tamaño de la matriz, generamos 30 muestras para las cuales se promediaron los resultados obtenidos para cada método.
- No se tuvo en cuenta la optimizacion generada cuando se tiene en cuenta la caracteristica de banda de la matriz.

Los resultados obtenidos fueron los siguientes:



A partir de la información suministrada por el gráfico precedente, estaríamos tentados a afimar que la complejidad experimental de los métodos es $\mathcal{O}(p^3)$, debido a la formas de las curvas, y que Gauss es más rápido que LU, lo cual no es correcto en este estadio del análisis. Para poder concluir que, efectivamente, la complejidad experimental coincide con la teórica, debemos realizar un paso más en el análisis, que consiste en tomar los tiempos de la experimentación y dividirlos por su correspondiente p elevado al cubo.

En los siguientes graficos mostramos este procedimiento:

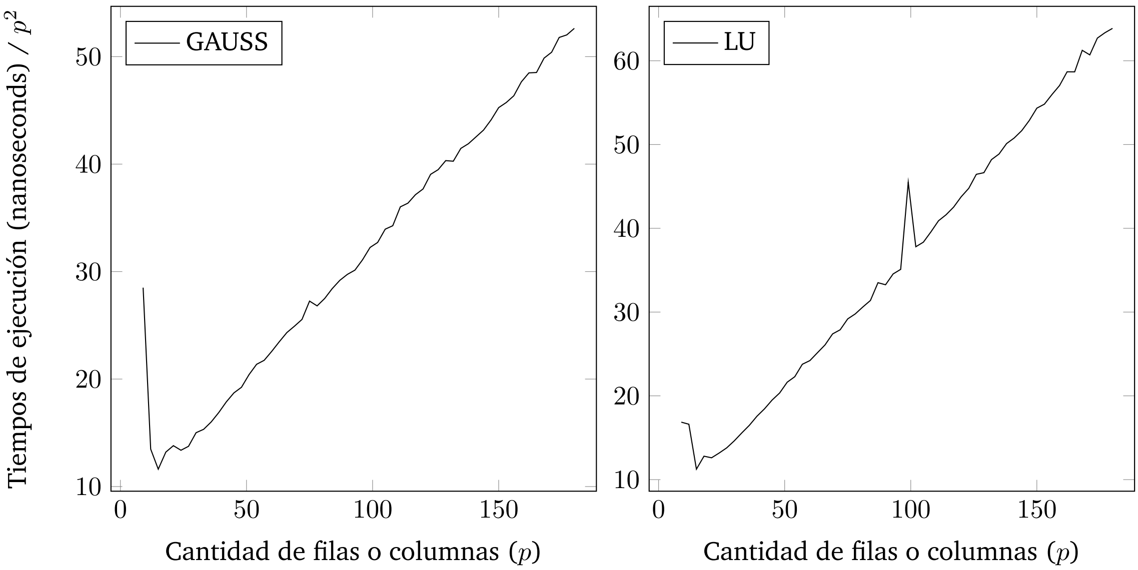


Figura 5: Dividiendo los tiempos por p^2

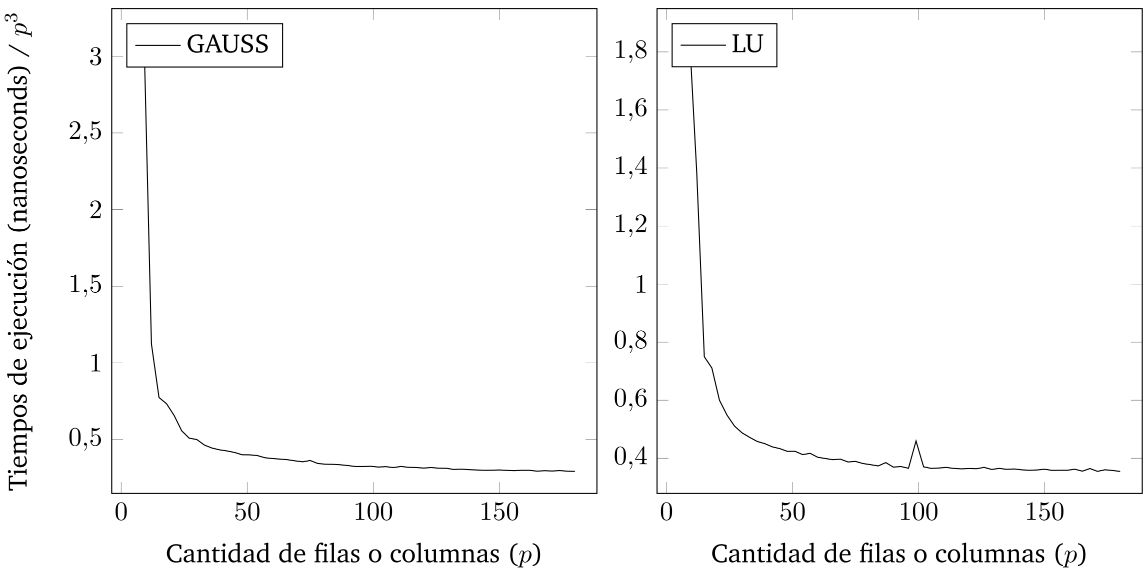


Figura 6: Dividiendo los tiempos por p^3

Como podemos ver en los ultimos graficos, al dividir los tiempos por p^3, estos tienden a un número constante mayor a 0. Por lo tanto los métodos tendrían una complejidad de $\mathcal{O}(c_1 * p^3)$ y $\mathcal{O}(c_2 * p^3)$ para la Eliminación Gaussiana y LU respectivamente, donde c_1 y c_2 son las constantes a las cuales convergen los graficos, y $c_1 \leq c_2$. Por lo tanto podemos concluir que la complejidad experimental coincide con la complejidad teórica propuesta, corroborando la hipótesis planteada al inicio de la seccion, y que no existen diferencias en terminos asintoticos entre ambos métodos.

5.5.2. Variación a lo largo del tiempo

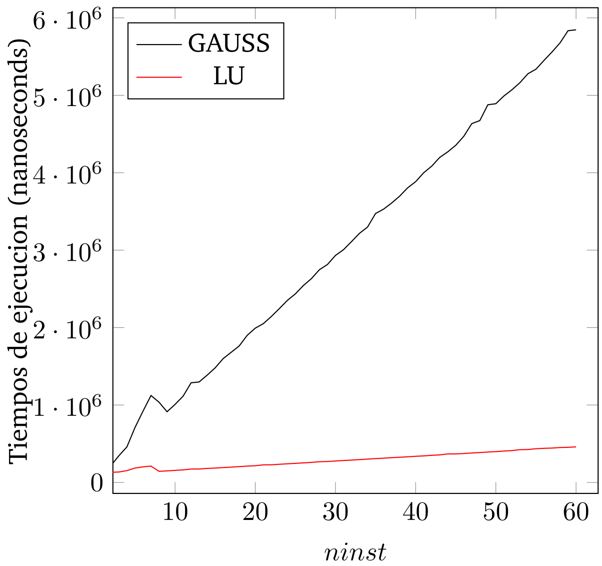
En esta sección nos interesa comparar los métodos implementados con respecto al tiempo computacional que insumen para resolver sistemas en los cuales hay variación del b. La hipótesis que se busca corroborar es que el método LU es mas rápido en este tipo de instancias que la Eliminación Gaussiana, debido a la reutilización de las matrices L y U.

En este caso, realizamos el siguiente experimento:

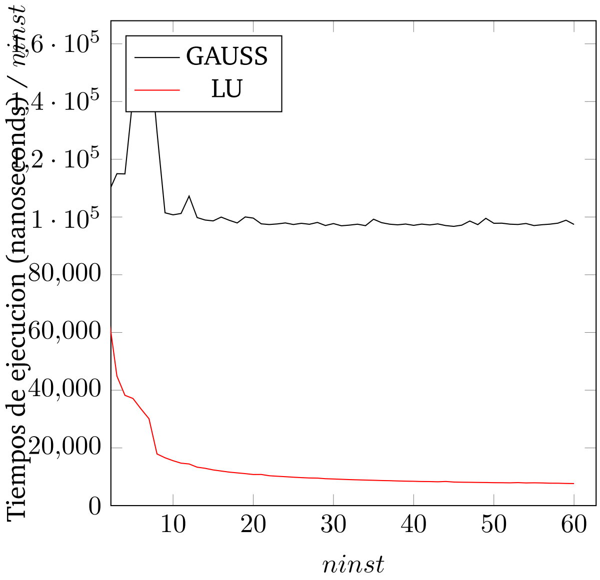
- Fijamos el $p = n * (m + 1)$ tal que el tamaño de la matriz generada es de $60 * 60$.

- Variamos el $ninst$ de 2 hasta 30.
- Utilizamos el r_i, r_e e iso del Alto Horno de Hierro, vease Sección 5.1.
- Generamos los valores de T_i y $T_e(\theta)$ para cada $ninst$ de forma aleatoria dentro de los siguientes intervalos:
  - $600 \leq T_i \leq 2000$
  - $20 \leq T_e(\theta) \leq 100$
- Los tiempos de ejecución se midieron con la biblioteca `chrono` y estos fueron convertidos a nanosegundos.
- Para cada caso de test, se midió la ejecución de los métodos Eliminación Gaussiana y LU.
- Para cada valor del $ninst$, generamos 30 muestras para las cuales se promediaron los resultados obtenidos para cada método.
- No se tuvo en cuenta la optimizacion generada cuando se tiene en cuenta la caracteristica de banda de la matriz.

Los resultados obtenidos fueron los siguientes:



Queda en evidencia la diferencia considerable entre Eliminación Gaussiana y LU, la cual se va acentuando a medida que aumenta el $ninst$. Podemos llevar al análisis un paso más y dividir los tiempos de cómputo por su respectivo $ninst$:



El gráfico muestra como, a medida que se aumenta el $ninst$, el costo promedio (o costo *amortizado*) de resolver un sistema de ecuaciones para un b en particular disminuye en el caso de LU, mientras que en Gauss se mantiene constante. A partir de estas afirmaciones, podemos concluir que, efectivamente, LU es más rápido que Gauss en este tipo de instancias dinámicas.

6. Conclusión