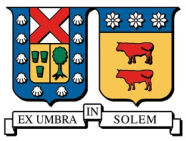


UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

Resolución de la Ecuación de Calor en una Pared

Métodos Numéricos

MEC270 - 2022

Departamento de Ing. Mecánica UTFSM

Martín Achondo Mercado

Rol: 201860005-9

Profesor: Romain Jean-Michel Gers

24 de Mayo 2022

Resumen

En este trabajo se intentó resolver la ecuación de calor numéricamente para determinar la mejor posición para situar un aislante. Para el problema se utilizaron 3 esquemas de discretización para resolver la ecuación de calor en una pared. La pared se simula con variación sinusoidal en el extremo externo y condición convectiva en el interior. Con esto, se evaluó la posibilidad de situar un aislante y su mejor ubicación para reducir la transferencia de calor. Como resultado se obtuvo que la mejor posición es en la pared interna dado que existe un alto de gradiente inicial producto de la condición inicial. De todas formas, esta conclusión es válida para el caso transiente dado que se obtuvo que en términos estacionarios, o cuasi-estacionarios, su posición no es relevante.

Índice

1. Introducción

En el problema se solicita resolver una ecuación diferencial parcial que modela la transferencia de calor con el objetivo de decidir la instalación de un aislante en un muero de una casa.

1.1. Planteamiento

Para el problema, se pide decidir si es mejor colocar un aislante por el lado interior o exterior del muro de una casa. De esta manera se evaluará la transferencia de calor para mantener el interior de la casa a temperatura constante de T_∞. La cara externa de la pared sufrirá variaciones sinusoidales simulando el día y la noche. Una imagen esquemática se presenta a continuación: Para la evaluación y decisión de la mejor configuración, se calculará el calor total

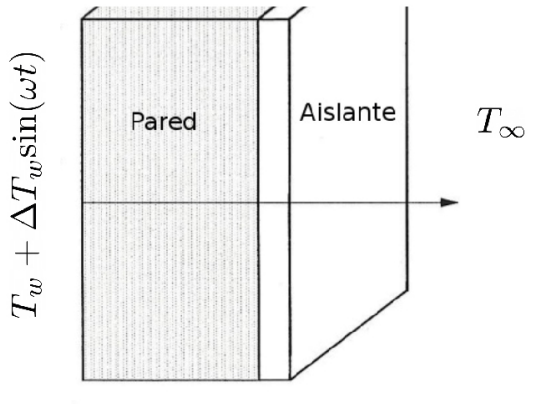


Figura 1: Imagen esquemática de la pared a modelar

transferido desde el interior de la casa en un plazo de 24 horas para el estado transiente y el cuasi-estacionario que se pueda alcanzar.

Se simulará la pared de concreto con propiedades:

Espesor	$\ell_{conc} = 0.45$ m
Difusividad Térmica	$\alpha_{conc} = 10^{-6}\text{m}^2/\text{s}$
Conductividad Térmica	$\lambda_{conc} = 2$ W/mK

Para el aislante, se simulará de madera con propiedades:

Espesor	$\ell_{mad} = 0.05$ m
Difusividad Térmica	$\alpha_{mad} = 10^{-7}\text{m}^2/\text{s}$
Conductividad Térmica	$\lambda_{mad} = 0.1$ W/mK

Junto a esto, para la zona externa se supondrá la temperatura de la pared con promedio de $T_w = 10°\text{C}$ y variación sinusoidal de amplitud $\Delta T_w = 5°\text{C}$ con periodo de 24 horas. Por último, la temperatura interior de la casa se asumirá constante de $T_\infty = 20°\text{C}$ con coeficiente de convección $h = 4\ \text{W/m}^2\text{k}$

2. Metodología

En este problema se pide resolver la ecuación de calor en una pared compuesta por un muro y un aislante. Para simplificar el problema, se considerará una transferencia de calor unidimensional, propiedades de los materiales constantes y la no existencia de generación de energía.

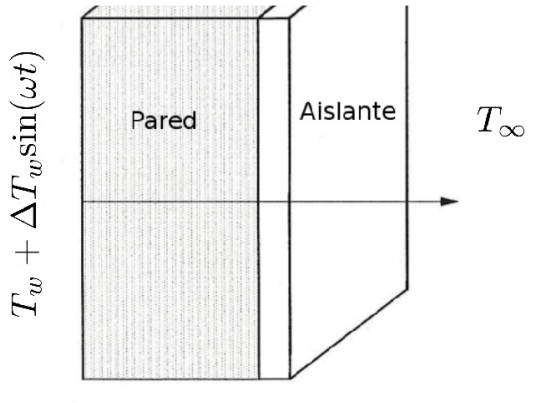


Figura 2: Representación de la pared a modelar

El problema, la ecuación con sus condiciones de borde e iniciales se puede plantear como:

$$\begin{cases} T(0,t) = T_w + \Delta T_w \sin(\omega t) & t > 0 \\ \dfrac{\partial T}{\partial t} = \alpha_1 \dfrac{\partial^2 T}{\partial x^2} & 0 < x < \ell_1 \, , \; t > 0 \\ -\lambda_1 \dfrac{\partial T}{\partial x} = -\lambda_2 \dfrac{\partial T}{\partial x} & x = \ell_1 \, , \; t > 0 \\ \dfrac{\partial T}{\partial t} = \alpha_2 \dfrac{\partial^2 T}{\partial x^2} & \ell_1 < x < \ell \, , \; t > 0 \\ -\lambda_2 \dfrac{\partial T}{\partial x} = h(T(\ell,t) - T_\infty) & x = \ell \, , \; t > 0 \end{cases} \tag{1}$$

Donde α_1 y λ_1 hacen referencia a la difusividad y conductividad térmica del material externo del muro (zona 1) y α_2, λ_2 al interno (zona 2). Para este trabajo se probarán las dos configuraciones, aislante en la zona externa y en la zona interna y se intentará justificar cual es más conveniente.

Se nota que para el borde izquierdo se tiene una condición de Dirichlet para la temperatura, la cual es oscilante.

$$T(0,t) = T_w + \Delta T_w \sin(\omega t) \tag{2}$$

En la interfaz de los dos materiales existe una condición de Neumann para la igualdad de flujo

de calor.

$$-\lambda_1 \frac{\partial T}{\partial x}\Big|_{x=\ell_1} = -\lambda_2 \frac{\partial T}{\partial x}\Big|_{x=\ell_1} \tag{3}$$

Para el borde externo, se tendrá una condición de borde de Robin igualando el calor por conducción con el calor por convección. Se elige esta forma dado que es la más realista. De todas formas, en los resultados se comparará el error al utilizar una condición de Dirichlet en este borde fijando T_∞

$$-\lambda_2 \frac{\partial T}{\partial x}\Big|_{x=\ell} = h(T(\ell, t) - T_\infty) \tag{4}$$

Para poder encontrar una aproximación teórica de la solución y así validar el código, se intentará obtener el comportamiento de la distribución de temperatura. Dada la condición periódica de la pared externa, la solución $T(x,t)$ de la ecuación 1 tiene que estar compuesto como:

$$T(x,t) = T_{tr}(x,t) + T_{qs}(x,t) \tag{5}$$

En donde T_{tr} corresponde al término transiente y T_{qs} corresponde al término cuasi-estacionario. Para el término transiente, se espera que luego de un tiempo considerable, el término se anule.

$$\lim_{t\to\infty} T_{tr}(x,t) = 0 \tag{6}$$

Además, se nota que la solución cuasi-estacionaria se puede descomponer como:

$$T_{qs}(x,t) = T_s(x) + A(x)\sin(\omega t + \phi(x)) \tag{7}$$

Lo que significa que después de un tiempo considerable, la temperatura en cada punto de la pared va a oscilar con una amplitud dependiente de la posición alrededor de su temperatura promedio. Analíticamente es sencillo obtener la componente estacionaria T_s. Basta resolver la ecuación:

$$\frac{\mathrm{d}^2 T_s}{\mathrm{d}x^2} = 0 \tag{8}$$

Con las siguientes condiciones de borde e iniciales:

$$\begin{cases} T(0,t) = T_w + \Delta T_w \sin(\omega t) & t > 0 \\ -\lambda_1 \dfrac{\mathrm{d}T_s}{\mathrm{d}x} = -\lambda_2 \dfrac{\mathrm{d}T_s}{\mathrm{d}x} & x = \ell_1\ ,\ t > 0 \\ -\lambda_2 \dfrac{\mathrm{d}T_s}{\mathrm{d}x} = h(T_s(\ell,t) - T_\infty) & x = \ell\ ,\ t > 0 \end{cases} \tag{9}$$

La solución tiene la siguiente forma:

$$T_s(x) = \begin{cases} \dfrac{h\lambda_2(T_\infty - T_w)}{\varphi}x + T_w & 0 \leq x \leq \ell_1 \\ \dfrac{h\lambda_1(T_\infty - T_w)}{\varphi}x + \dfrac{h(\ell\lambda_1 T_w - \ell_1 T_\infty(\lambda_1 - \lambda_2)) + \lambda_1\lambda_2}{\varphi} & \ell_1 < x \leq \ell \end{cases} \tag{10}$$

En donde φ tiene el valor de:

$$\varphi = h(\ell\lambda_1 - \ell_1(\lambda_1 - \lambda_2)) + \lambda_1\lambda_2 \tag{11}$$

Se nota claramente que el perfil de temperatura estacionario es lineal. Además, la solución cuasi-estacionaria oscilará respecto a esta solución estacionaria.

Para justificar que la solución oscilatoria es en efecto solución de la ecuación 1, basta remplazar T_{qs}, así se obtiene:

$$\frac{\partial T_{osc}}{\partial t} = \alpha\frac{\partial^2 T_{osc}}{\partial x^2} \tag{12}$$

En donde:

$$T_{osc}(x,t) = A(x)\sin(\omega t + \phi(x)) \tag{13}$$

Escribiendo este componente oscilatorio como:

$$A(x)\sin(\omega t + \phi(x)) = \mathrm{Im}\left(A(x)\mathrm{e}^{i\phi(x)}\mathrm{e}^{i\omega t}\right) \tag{14}$$

Con el siguiente cambio de variables:

$$U(x) = A(x)\mathrm{e}^{i\phi(x)} \tag{15}$$

Y reemplazando en 1, se obtiene:

$$\frac{\mathrm{d}^2U}{\mathrm{d}x^2} - \frac{i\omega}{\alpha}U = 0 \tag{16}$$

La cual en efecto tiene una solución dada por:

$$U(x) = C_1\mathrm{e}^{-\sqrt{\omega/2\alpha}(1+i)x} + C_2\mathrm{e}^{\sqrt{\omega/2\alpha}(1+i)x} \tag{17}$$

La función U entrega información sobre la amplitud de la oscilación en cada posición y su desfase. No es de interés obtenerla ya que el problema se resolverá numéricamente. Lo importante

es percatarse que la componente oscilante existe:

$$T_{osc}(x,t) = \mathrm{Im}\left(U(x)\mathrm{e}^{i\omega t}\right) \tag{18}$$

Y resuelve la ecuación diferencial 1.

Con la solución estacionaria obtenida y el conocimiento de la solución oscilante se podrá validar el código elaborado.

Respecto al flujo de calor, se nota que es de interés el flujo en la cara interna para evaluar cuanto calor pierde el ambiente interior al muro. El flujo entonces tendrá la siguiente forma:

$$\dot{q}_{qs} = -\lambda_2 \frac{\partial T_{qs}}{\partial x}\Big|_{x=\ell} \tag{19}$$

En donde:

$$\frac{\partial T_{qs}}{\partial x} = T_s'(x) + A'(x)\sin(\omega t + \phi(x)) + A(x)\phi'(x)\cos(\omega t + \phi(x)) \tag{20}$$

Dado que el flujo de calor es oscilatorio, será de interés el calor total transferido en un periodo (Δt) cuando se alcanza el estado cuasi-estacionario.

$$q_{qs} = \int_{\Delta t_p} \dot{q}_{qs}\,\mathrm{d}t = \int_{\Delta t_p} -\lambda_2 \frac{\partial T_{qs}}{\partial x}\Big|_{x=\ell}\,\mathrm{d}t \tag{21}$$

De donde se nota que la integral sobre un periodo del término oscilante es nulo. Así, el calor total queda como:

$$q_{qs} = -\lambda_2 \int_{\Delta t_p} T_s'(\ell)\,\mathrm{d}t \tag{22}$$

Y dado que se conoce el perfil estacionario, se puede evaluar directamente el calor transferido en un periodo.

$$q_{qs} = -\lambda_2 \frac{h\lambda_1(T_\infty - T_w)}{h(\ell\lambda_1 - \ell_1(\lambda_1 - \lambda_2)) + \lambda_1\lambda_2}\Delta t_p \tag{23}$$

Con este resultado se podrá plantear una segunda evaluación al código cuando alcanza el estado cuasi-estacionario. Notar en todo caso que con un simple cambio de variables se puede demostrar que este calor total es independiente de la posición del aislante. Esto se debe a se alcanza este estado cuasi-estacionario y además, la "resistencia térmica" es la misma independiente de la configuración.

2.1. Discretización

La ecuación 1 se resolverá numéricamente utilizando 3 esquemas de discretización para obtener la distribución de temperatura a lo largo del muro para distintos instantes de tiempo por el método de diferencias finitas. Así, se podrá comparar el orden de convergencia de cada esquema. Para ambas zonas, 1 y 2, se utilizará el mismo espaciamiento espacial Δx para mantener el orden de convergencia de cada esquema. La cantidad de nodos entonces será:

$$N = \frac{\ell}{\Delta x} + 1 \tag{24}$$

De esta manera cada nodo $j = 1, 2, 3, \cdots, k-1, k, k+1, \cdots, N-1, N$. Es importante elegir bien el espaciamiento espacial para que calce un nodo en la interfaz. Este nodo se representará como $j = k$. El paso temporal será Δt, por lo que la cantidad de pasos temporales contando el instante inicial será de:

$$N_t = \frac{t_f}{\Delta t} + 1 \tag{25}$$

Un número adimensional importante que tiene relación con la discretización del problema es el número de Fourier, que puede ser definido para cada zona $i = 1, 2$ como:

$$\sigma_i = \alpha_i \frac{\Delta t}{\Delta x^2} \tag{26}$$

Con esto, una representación de la discretización espacial y temporal a realizar se visualiza esquemáticamente en la siguiente imagen.

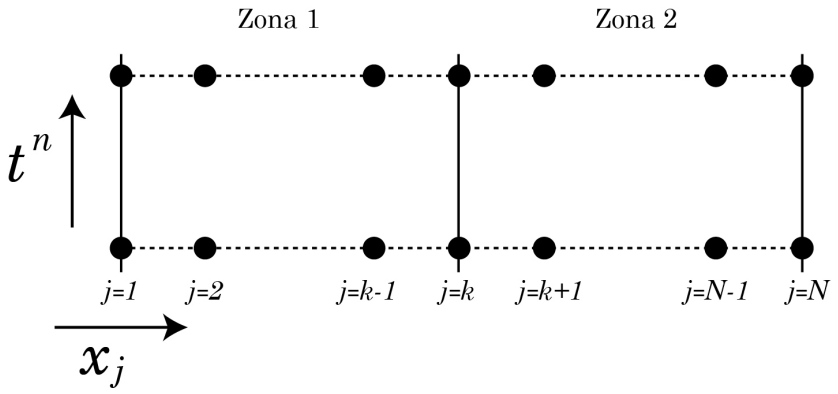


Figura 3: Esquema de discretización

Para todos los esquemas a presentar, se discretizará la segunda derivada de la temperatura como:

$$\frac{\partial^2 T}{\partial x^2} = \frac{T_{j-1} - 2T_j + T_{j+1}}{\Delta x^2} + \mathcal{O}(\Delta x^2) \tag{27}$$

Las condiciones de borde se discretizarán con esquemas de orden 1 y 2 (las derivadas).

- Borde en pared exterior (Dirichlet), $j = 1$

$$T_1^n = T_w + \Delta T_w \sin(\omega t^n) \tag{28}$$

- Interfaz entre los dos materiales (Neumann), $j = k$
  - Esquema orden 1:

$$\lambda_1 \frac{T_k^n - T_{k-1}^n}{\Delta x} = \lambda_2 \frac{T_{k+1}^n - T_k^n}{\Delta x} \tag{29}$$

  - Esquema orden 2:

$$\lambda_1 \frac{T_{k-2}^n - 4T_{k-1}^n + 3T_k^n}{2\Delta x} = \lambda_2 \frac{-3T_k^n + 4T_{k+1}^n - T_{k+2}^n}{2\Delta x} \tag{30}$$

- Borde interior (Robin), $j = N$
  - Esquema orden 1:

$$-\lambda_2 \frac{T_N^n - T_{N-1}^n}{\Delta x} = h(T_N^n - T_\infty) \tag{31}$$

  - Esquema orden 2:

$$-\lambda_2 \frac{T_{N-2}^n - 4T_{N-1}^n + 3T_N^n}{2\Delta x} = h(T_N^n - T_\infty) \tag{32}$$

La condición inicial será de fijar la misma temperatura en toda la pared.

$$T_j^0 = T_\infty \;\; 1 \leq j \leq N \tag{33}$$

Ahora se detallarán los esquemas de discretización de la ecuación de calor y los esquemas que se utilizarán en sus condiciones de borde.

Esquema Euler Explícito Discretiza la ecuación de calor utilizando los valores en el paso anterior. Así, la ecuación queda como:

$$\frac{T^{n+1} - T^n}{\Delta t} = f(t^n, T^n) \tag{34}$$

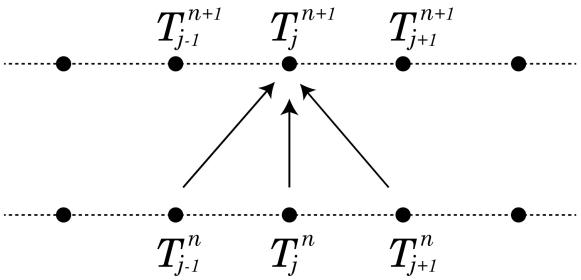


Figura 4: Esquema de discretización Euler Explícito

Así, la ecuación queda como:

$$\frac{T_j^{n+1} - T_j^n}{\Delta t} = \alpha_i \frac{T_{j-1}^n - 2T_j^n + T_{j+1}^n}{\Delta x^2} \tag{35}$$

Despejando T^{n+1} y utilizando las condiciones de borde de orden 1, se forman las siguientes ecuaciones para cada nodo j. Con esto, el método queda con orden $\mathcal{O}(\Delta x, \Delta t)$, primer orden en tiempo y espacio.

$$\begin{cases} T_1^{n+1} = T_w + \Delta T_w \sin\left(\omega t^{n+1}\right) & j = 1 \\ T_j^{n+1} = T_j^n + \sigma_1 (T_{j-1}^n - 2T_j^n + T_{j+1}^n) & 1 < j < k \\ T_j^{n+1} = T_j^n + \sigma_2 (T_{j-1}^n - 2T_j^n + T_{j+1}^n) & k < j < N \\ T_k^{n+1} = \dfrac{\lambda_1 T_{k-1}^{n+1} + \lambda_2 T_{k+1}^{n+1}}{\lambda_1 + \lambda_2} & j = k \\ T_N^{n+1} = \dfrac{\lambda_2 T_{N-1}^{n+1} + \Delta x h T_\infty}{\Delta x h \lambda_2} & j = N \end{cases} \tag{36}$$

Esquema Euler Implícito Discretiza la ecuación de calor utilizando los valores en el nodo del paso anterior y los cercanos en el paso actual. Así, la ecuación queda como:

$$\frac{T^{n+1} - T^n}{\Delta t} = f(t^{n+1}, T^{n+1}) \tag{37}$$

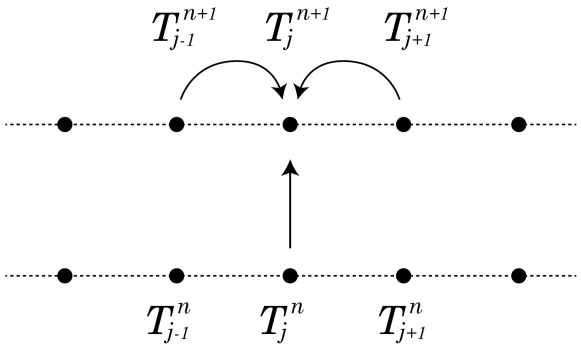


Figura 5: Esquema de discretización Euler Implícito

Así, la ecuación queda como:

$$\frac{T_j^{n+1} - T_j^n}{\Delta t} = \alpha_i \frac{T_{j-1}^{n+1} - 2T_j^{n+1} + T_{j+1}^{n+1}}{\Delta x^2} \tag{38}$$

Despejando T^{n+1} y utilizando las condiciones de borde de orden 2, se forman las siguientes ecuaciones para cada nodo j. Con esto, el método queda con orden $\mathcal{O}(\Delta x^2, \Delta t)$, segundo orden en espacio y primero en el tiempo.

$$\begin{cases} T_1^{n+1} = T_w + \Delta T_w \sin\left(\omega t^{n+1}\right) & j = 1 \\ -T_{j-1}^{n+1} + \left(2 + \dfrac{1}{\sigma_1}\right) T_j^{n+1} - T_{j+1}^{n+1} = \dfrac{1}{\sigma_1} T_j^n & 1 < j < k \\ \lambda_1 T_{k-2}^{n+1} - 4\lambda_1 T_{k-1}^{n+1} + 3(\lambda_1 + \lambda_2) T_k^{n+1} - 4\lambda_2 T_{k+1}^{n+1} + \lambda_2 T_{k+2}^{n+1} = 0 & j = k \\ -T_{j-1}^{n+1} + \left(2 + \dfrac{1}{\sigma_2}\right) T_j^{n+1} - T_{j+1}^{n+1} = \dfrac{1}{\sigma_2} T_j^n & k < j < N \\ \theta T_{N-2}^{n+1} - 4\theta T_{N-1}^{n+1} + (3\theta + h) T_N^{n+1} = h T_\infty & j = N \end{cases} \tag{39}$$

Notar que se utiliza $\theta = \frac{\lambda_2}{2\Delta x}$. El sistema de ecuaciones que se forma es pentadiagonal y será resuelto con el algoritmo de Thomas para sistemas tridiagonales por bloques.

Esquema Euler Crank Nicolson Discretiza la ecuación de calor utilizando los valores del paso anterior y actual. Así, la ecuación queda como:

$$\frac{T^{n+1}-T^n}{\Delta t}=\frac{1}{2}\left(f(t^{n+1},T^{n+1})+f(t^n,T^n)\right) \tag{40}$$

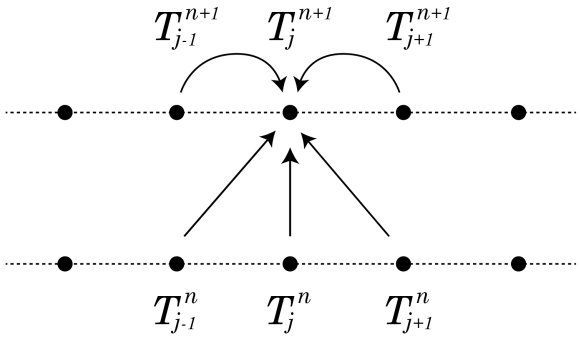


Figura 6: Esquema de discretización Crank Nicolson

Así, la ecuación queda como:

$$\frac{T_j^{n+1}-T_j^n}{\Delta t}=\frac{\alpha_i}{2}\left(\frac{T_{j-1}^{n+1}-2T_j^{n+1}+T_{j+1}^{n+1}}{\Delta x^2}+\frac{T_{j-1}^{n}-2T_j^{n}+T_{j+1}^{n}}{\Delta x^2}\right) \tag{41}$$

Despejando T^{n+1} y utilizando las condiciones de borde de orden 2, se forman las siguientes ecuaciones para cada nodo j. Con esto, el método queda con orden $\mathcal{O}(\Delta x^2,\Delta t^2)$, segundo orden en espacio y tiempo.

$$\begin{cases} T_1^{n+1}=T_w+\Delta T_w\sin\left(\omega t^{n+1}\right) & j=1 \\ -T_{j-1}^{n+1}+\left(2+\dfrac{2}{\sigma_1}\right)T_j^{n+1}-T_{j+1}^{n+1}=T_{j-1}^{n}+\left(2-\dfrac{2}{\sigma_1}\right)T_j^{n}+T_{j+1}^{n} & 1<j<k \\ \lambda_1 T_{k-2}^{n+1}-4\lambda_1 T_{k-1}^{n+1}+3(\lambda_1+\lambda_2)T_k^{n+1}-4\lambda_2 T_{k+1}^{n+1}+\lambda_2 T_{k+2}^{n+1}=0 & j=k \\ -T_{j-1}^{n+1}+\left(2+\dfrac{2}{\sigma_2}\right)T_j^{n+1}-T_{j+1}^{n+1}=T_{j-1}^{n}+\left(2-\dfrac{2}{\sigma_2}\right)T_j^{n}+T_{j+1}^{n} & k<j<N \\ \theta T_{N-2}^{n+1}-4\theta T_{N-1}^{n+1}+(3\theta+h)T_N^{n+1}=hT_\infty & j=N \end{cases} \tag{42}$$

El sistema de ecuaciones que se forma es pentadiagonal y será resuelto con el algoritmo de Thomas para sistemas tridiagonales por bloques.

Para el **flujo de calor**, dado que es de interés la transferencia en la pared interna, se puede aprovecha de la condición de borde utilizada para calcularlo como:

$$\dot{q}_N^n = h(T_N^n - T_\infty) \tag{43}$$

De todas formas, la comparación se realizará con el calor total transferido en un periodo. Dado esto, es de relevancia calcular la integral:

$$q_N = \int \dot{q}_N \, \mathrm{d}t \tag{44}$$

Esta integral se aproximará utilizando el método del trapecio el cual tiene orden $\mathcal{O}(\Delta t^3)$. Así, para un periodo completo, el calor total transferido en el borde interno queda como:

$$q_N = \frac{\Delta t}{2} \sum_n q_N^n + q_N^{n+1} \tag{45}$$

Algoritmo de Thomas Dado que en los esquemas de Euler Implícito y Crank Nicolson presentados aparece un sistema pentadiagonal por las aproximaciones de segundo orden en las condiciones de borde, es importante utilizar un método eficiente para resolverlo. Dado esto, se utilizará el algoritmo de Thomas para sistemas tridiagonales por bloques. Esquemáticamente, se está resolviendo el sistema de M ecuaciones matriciales:

$$A_i X_{i-1} + C_i X_i + B_i X_{i+1} = F_i \tag{46}$$

En donde A, B, C son matrices de 3 x 3 y X, F vectores de 3 componentes. El algoritmo de resolución tiene dos pasos. Primero, se calcula la matriz Λ y el vector β:

$$\begin{cases} \Lambda_1 = G_1 B_1 \\ \Lambda_i = G_i B_i \quad i = 2, \cdots, M-1 \end{cases} \tag{47}$$

Y también:

$$\begin{cases} \beta_1 = G_1 F_1 \\ \beta_i = G_i(F_i - A_i \beta_{i-1}) \quad i = 2, \cdots, M \end{cases} \tag{48}$$

En donde $G_1 = C_1^{-1}$ y $G_i = (C_i - A_i \Lambda_i)^{-1}$. Con esto, se puede encontrar el vector solución X con la iteración:

$$\begin{cases} X_M = \beta_M \\ X_i = \beta_i - \Lambda_i X_{i+1} \quad i = M-1, \cdots, 1 \end{cases} \tag{49}$$

2.2. Planteamiento de Resultados

En las simulaciones para pasos de tiempo de 1 y 10 segundos y espaciamientos en la posición de 0.02 y 0.005 metros. Estos pasos serán utilizados en los 3 esquemas presentados anteriormente para modelar el perfil de temperatura en el estado transiente y el cuasi-estacionario. De esta forma se podrá comparar la convergenci de los esquemas con su costo computacional para así evaluar la conveniencia de utilizar uno o el otro.

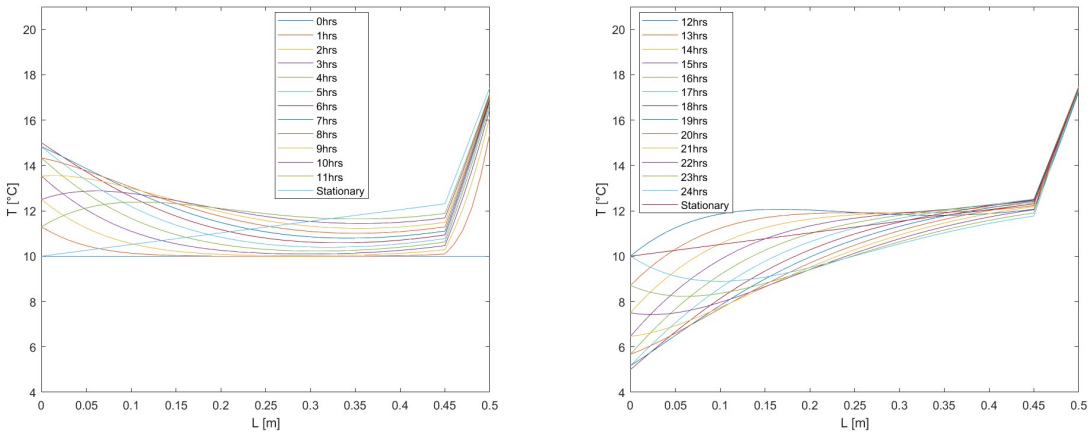
Además, se calculará el calor total perdido por el interior en estas dos condiciones (transiente y cuasi-estacionario). Por último, se compararán los resultados con la teoría presentada al comienzo para validar los resultados y así intentar encontrar la amplitud del término oscilante en la temperatura. Todo esto para verificar si es más conveniente situar el aislante dentro o fuera de la pared. Además, se compararán los resultados con el caso si no existiece aislante.

3. Resultados

Se presentan los resultados para la modelación de la transferencia de calor en el muro. Se presentarán las distribuciones de temperatura para las configuraciones de aislante interior, exterior, sin aislante y condición de temperatura fija. En todas las gráficas se presentará la distribución para el caso estacionario como comparación. Por último, se presentan los gráficos de la distribución de temperatura y flujo de calor en la pared interior junto al calor total transferido en 24 horas.

3.0.1. Transferencia de Calor en Estado Transiente

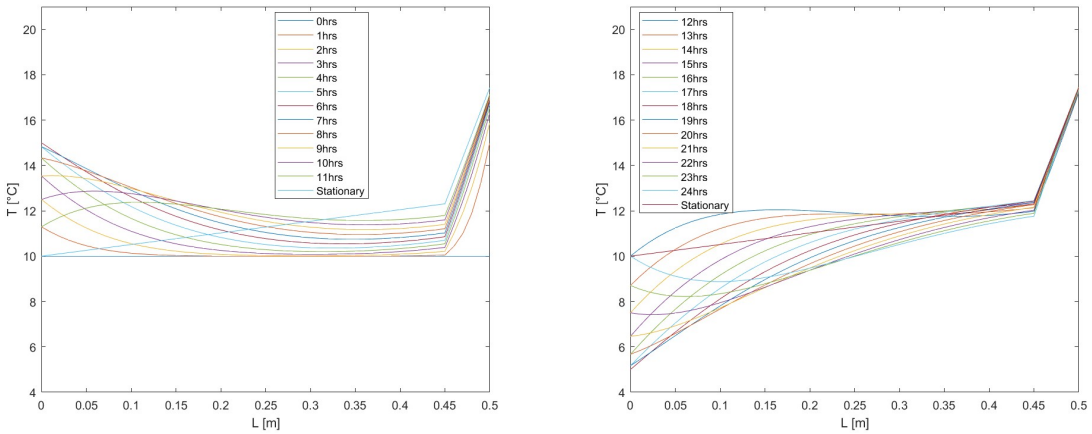
Variación esquema de Discretización Aislante en Interior



(a) 0 - 11 horas

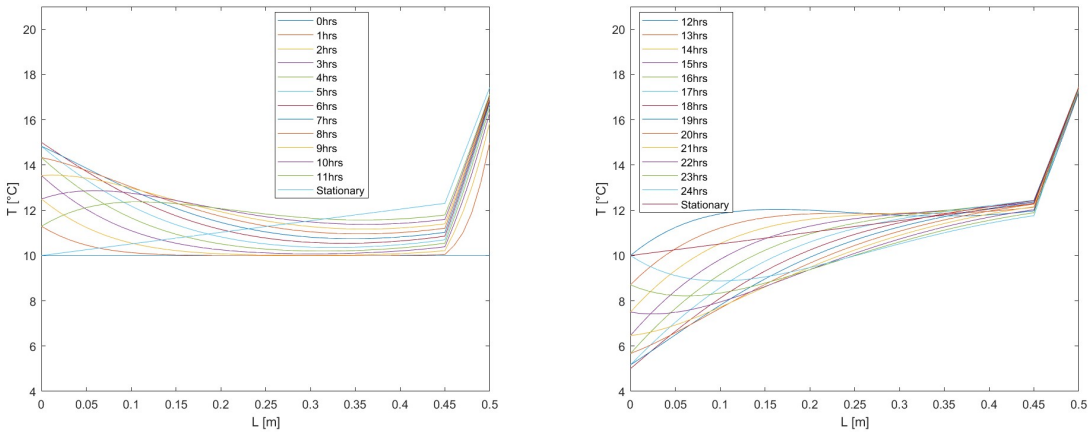
(b) 12 - 24 horas

Figura 7: Distribución de temperatura para las primeras 24 horas con aislante en cara interior con Euler Explícito con $\Delta x = 0.005$ y $\Delta t = 1$



(a) 0 - 11 horas (b) 12 - 24 horas

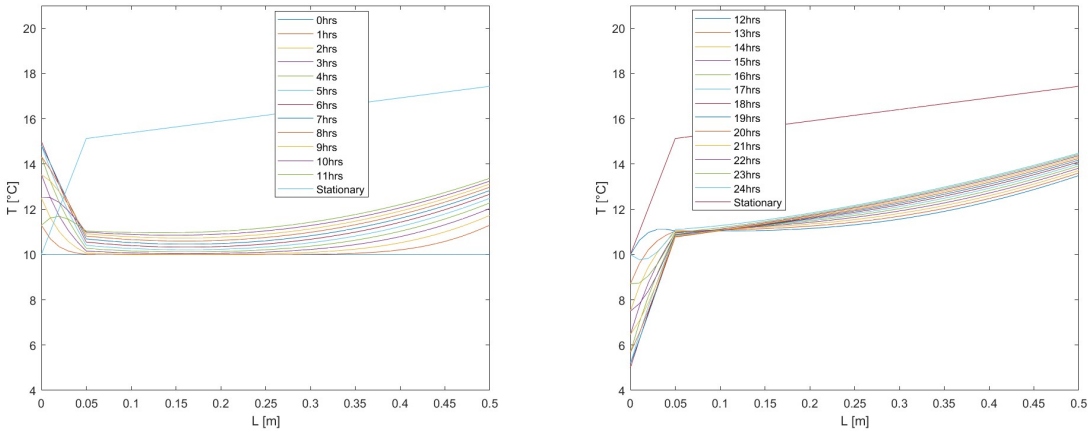
Figura 8: Distribución de temperatura para las primeras 24 horas con aislante en cara interior con Euler Implícito $\Delta x = 0.01$ y $\Delta t = 5$



(a) 0 - 11 horas (b) 12 - 24 horas

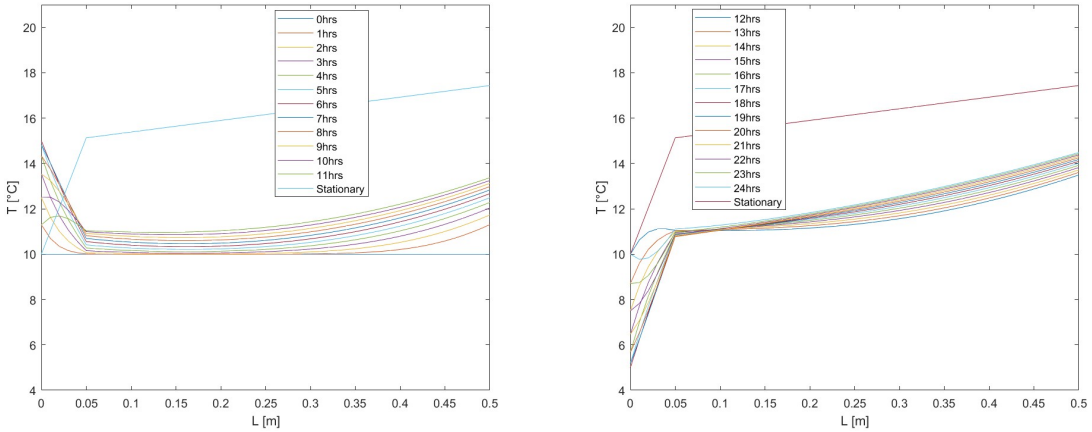
Figura 9: Distribución de temperatura para las primeras 24 horas con aislante en cara interior con Crank Nicolson $\Delta x = 0.01$ y $\Delta t = 10$

Variación esquema de Discretización Aislante en Exterior



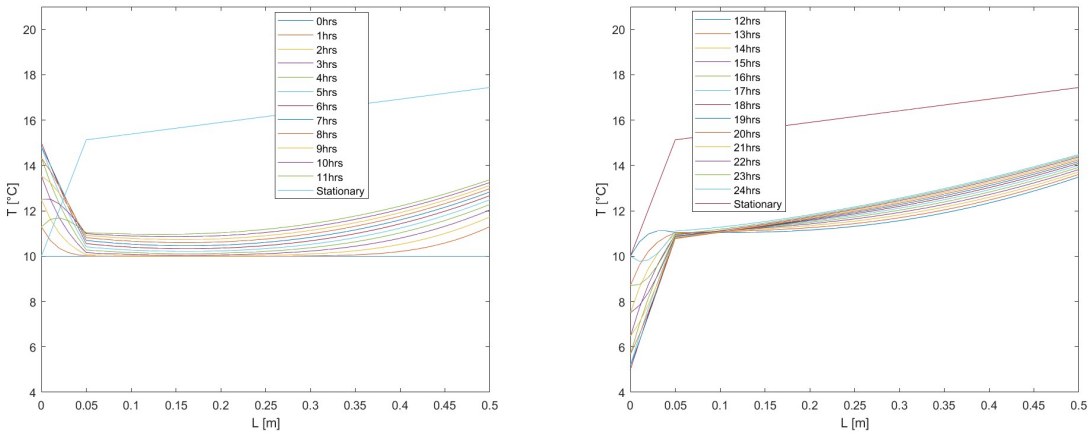
(a) 0 - 11 horas (b) 12 - 24 horas

Figura 10: Distribución de temperatura para las primeras 24 horas con aislante en cara exterior con Euler Explícito con $\Delta x = 0.005$ y $\Delta t = 1$



(a) 0 - 11 horas (b) 12 - 24 horas

Figura 11: Distribución de temperatura para las primeras 24 horas con aislante en cara exterior con Euler Implícito $\Delta x = 0.01$ y $\Delta t = 5$

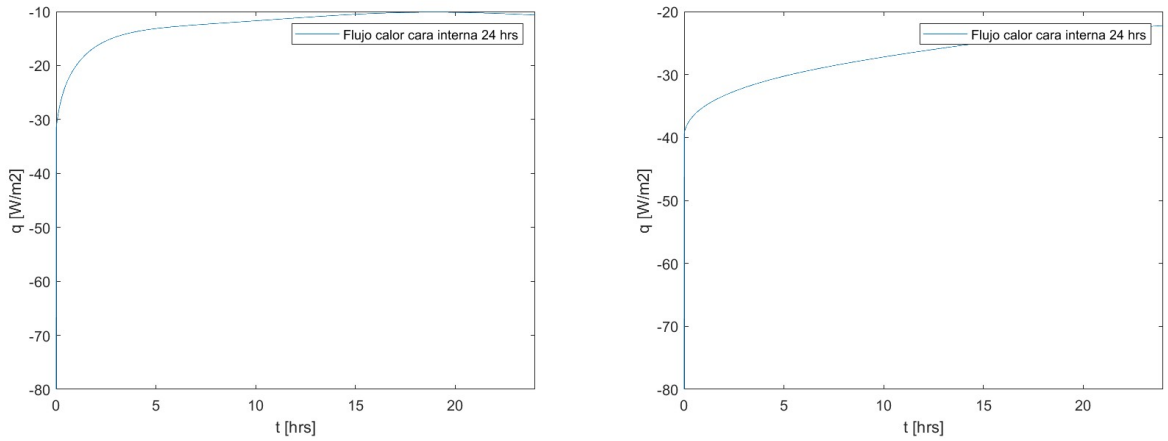


(a) 0 - 11 horas

(b) 12 - 24 horas

Figura 12: Distribución de temperatura para las primeras 24 horas con aislante en cara exterior con Crank Nicolson $\Delta x = 0.01$ y $\Delta t = 10$

Flujo de Calor y Calor Total en Cara Interna



(a) Aislante interno (b) Aislante externo

Figura 13: Flujo de Calor en la cara interna en las primeras 24 horas para ambas configuraciones

Tabla 1: Tabla para calor total transferido en las primeras 24 horas con aislante interior

Esquemas	Euler Explícito	Euler Implícito	Crank Nicolson
q_N [KJ/m2]	1034	1035	1035

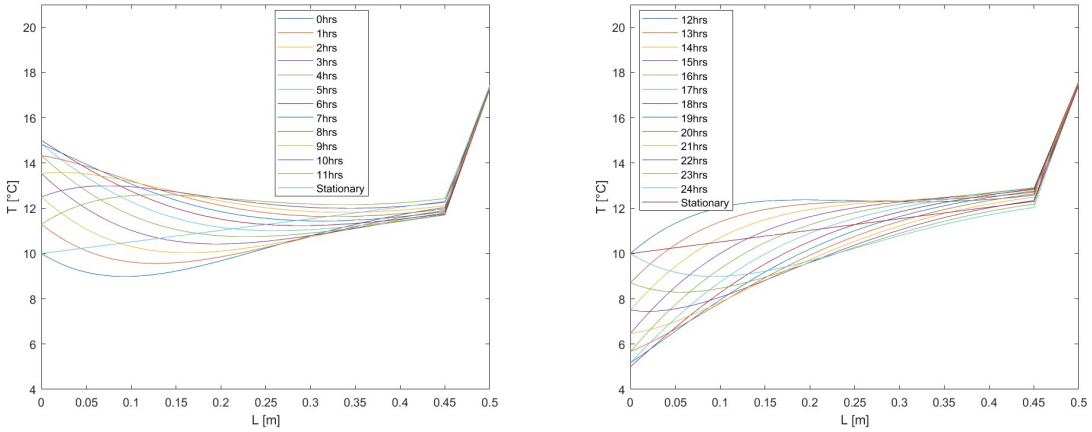
Tabla 2: Tabla para calor total transferido en las primeras 24 horas con aislante exterior

Esquemas	Euler Explícito	Euler Implícito	Crank Nicolson
q_N [KJ/m2]	2321	2322	2322

3.0.2. Transferencia de Calor en Estado Estacionario

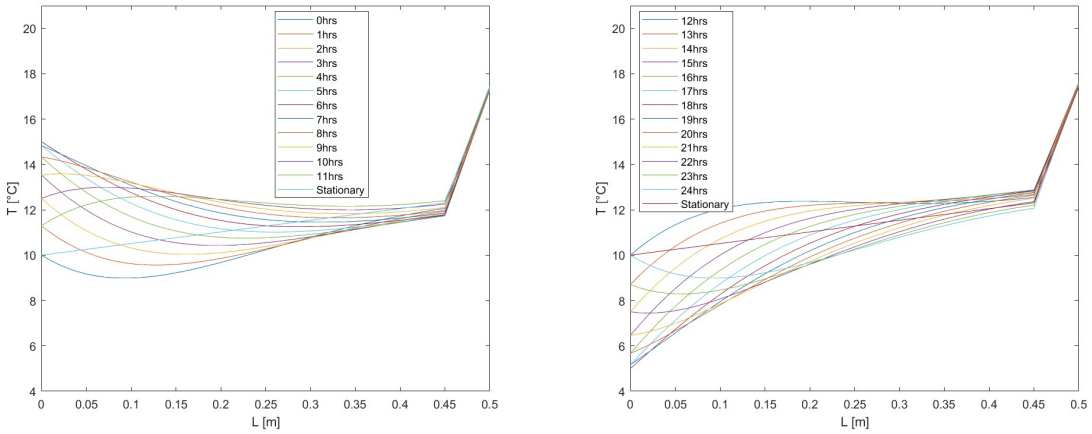
Se modelan para las 24 horas próximas a 10 días.

Variación esquema de Discretización Aislante en Interior



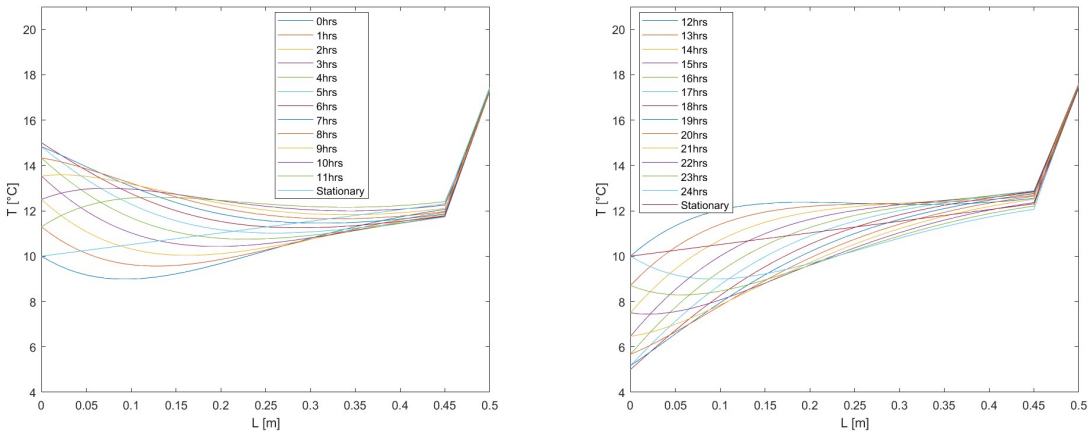
(a) 0 - 11 horas (b) 12 - 24 horas

Figura 14: Distribución de temperatura para las próximas 24 horas pasados 10 días con aislante en cara interior con Euler Explícito con $\Delta x = 0.005$ y $\Delta t = 1$



(a) 0 - 11 horas (b) 12 - 24 horas

Figura 15: Distribución de temperatura para las próximas 24 horas pasados 10 días con aislante en cara interior con Euler Implícito $\Delta x = 0.01$ y $\Delta t = 5$

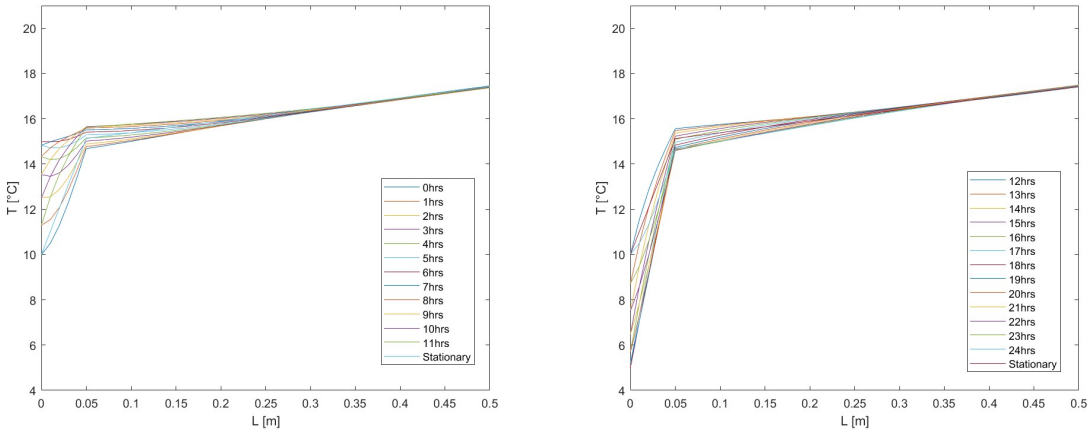


(a) 0 - 11 horas

(b) 12 - 24 horas

Figura 16: Distribución de temperatura para las próximas 24 horas pasados 10 días con aislante en cara interior con Crank Nicolson $\Delta x = 0.01$ y $\Delta t = 10$

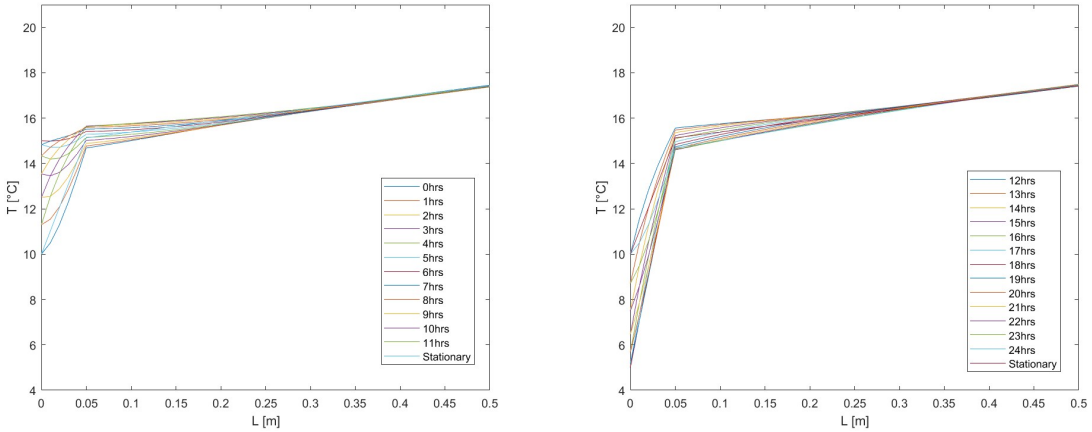
Variación esquema de Discretización Aislante en Exterior



(a) 0 - 11 horas

(b) 12 - 24 horas

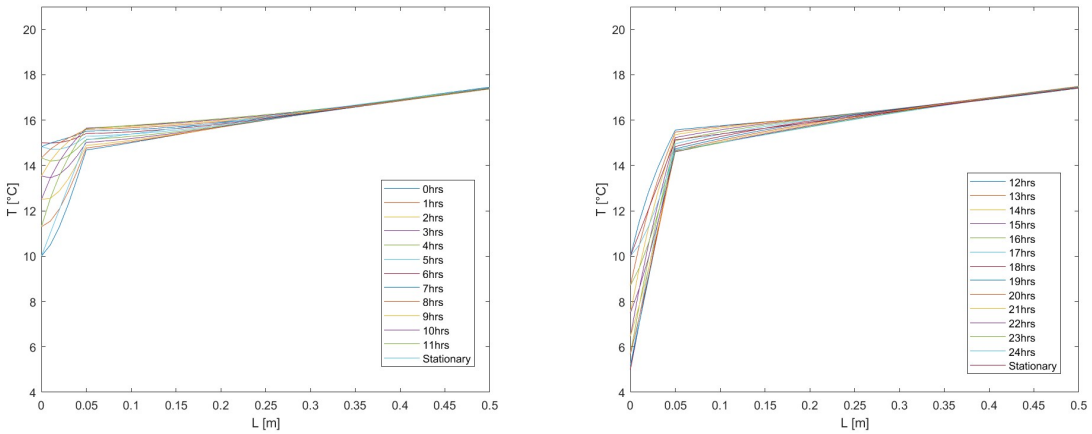
Figura 17: Distribución de temperatura para las próximas 24 horas a 10 días con aislante en cara exterior con Euler Explícito con $\Delta x = 0.005$ y $\Delta t = 1$



(a) 0 - 11 horas

(b) 12 - 24 horas

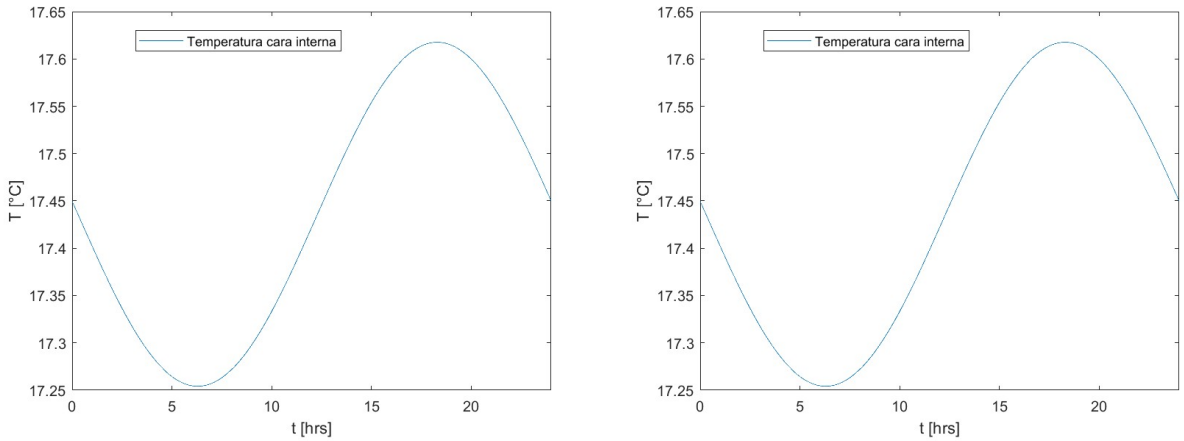
Figura 18: Distribución de temperatura para las próximas 24 horas a 10 días con aislante en cara exterior con Euler Implícito $\Delta x = 0.01$ y $\Delta t = 5$



(a) 0 - 11 horas (b) 12 - 24 horas

Figura 19: Distribución de temperatura para las próximas 24 horas a 10 días con aislante en cara exterior con Crank Nicolson $\Delta x = 0.01$ y $\Delta t = 10$

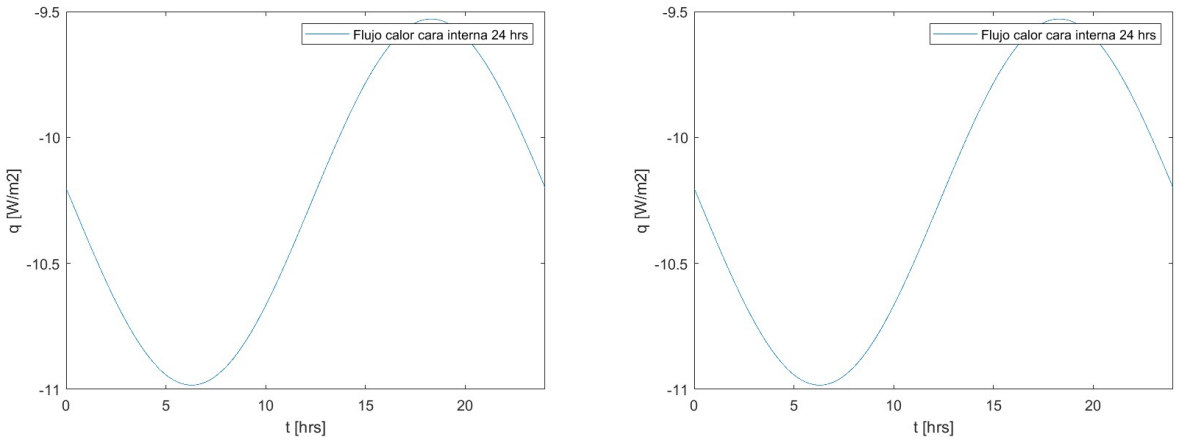
Temperatura en cara externa en 24 horas a 10 días en estado estacionario



(a) Aislante interno (b) Aislante externo

Figura 20: Temperatura en la cara interna en las próximas 24 horas a 10 días para ambas configuraciones

Flujo de Calor y Calor Total en Cara Interna



(a) Aislante interno (b) Aislante externo

Figura 21: Flujo de Calor en la cara interna en las próximas 24 horas a 10 días para ambas configuraciones

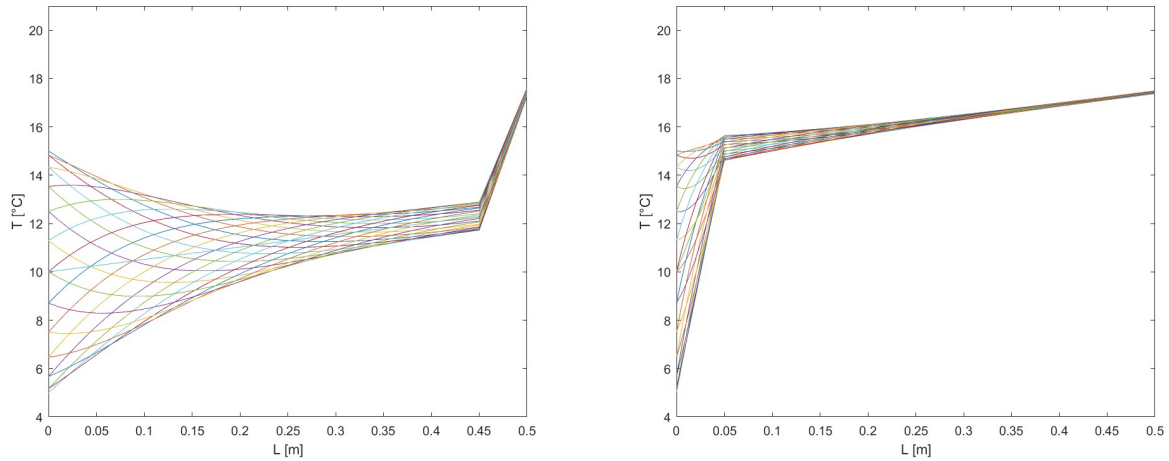
Tabla 3: Tabla para calor total transferido en las próximas 24 horas a 10 días con aislante interior

Esquemas	Euler Explícito	Euler Implícito	Crank Nicolson	Teórico
q_N [KJ/m2]	886.16	886.17	886.117	886.17

Tabla 4: Tabla para calor total transferido en las próximas 24 horas a 10 días con aislante exterior

Esquemas	Euler Explícito	Euler Implícito	Crank Nicolson	Teórico
q_N [KJ/m2]	887.22	886.17	886.17	886.17

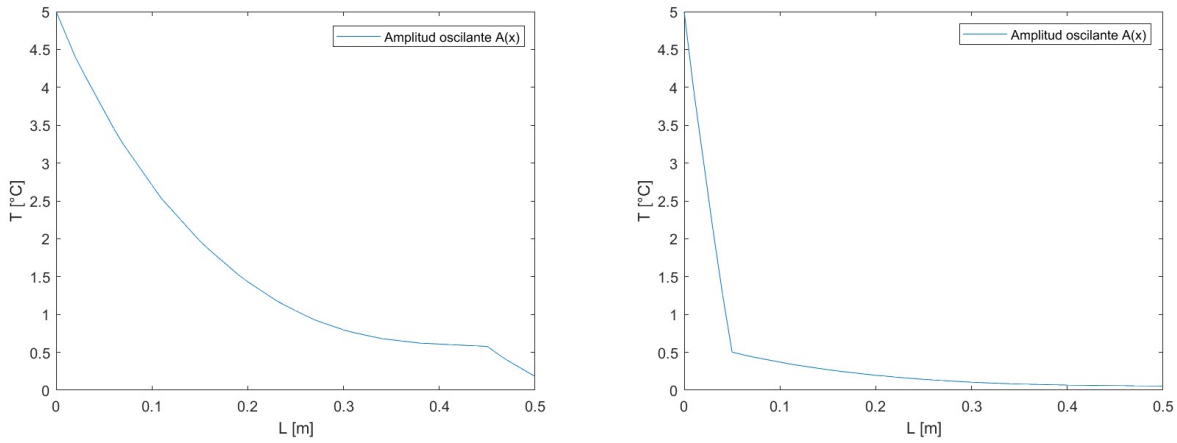
Variación de Temperatura en 1 día completo



(a) Aislante interno (b) Aislante externo

Figura 22: Variación de Temperatura en 1 día completo pasado 10 días

Amplitud de oscilación para estado estacionario

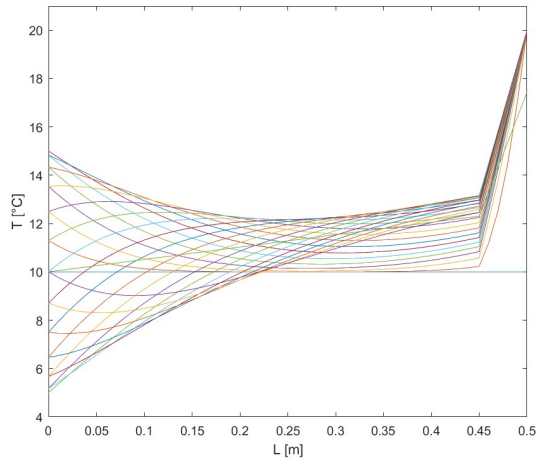


(a) Aislante interno (b) Aislante externo

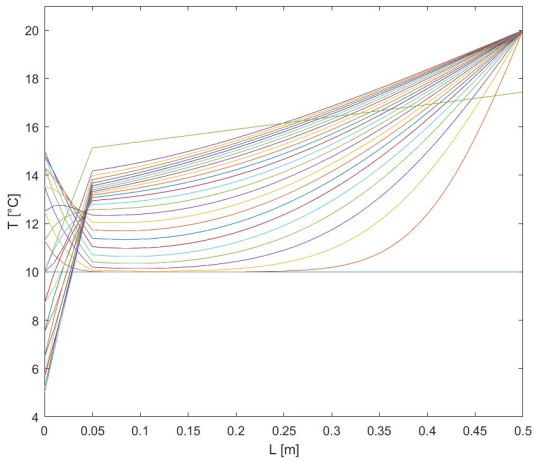
Figura 23: Amplitud de oscilación $A(x)$ para la temperatura

3.0.3. Condición temperatura fija

Variación al utilizar condición de Temperatura fija en la pared interior

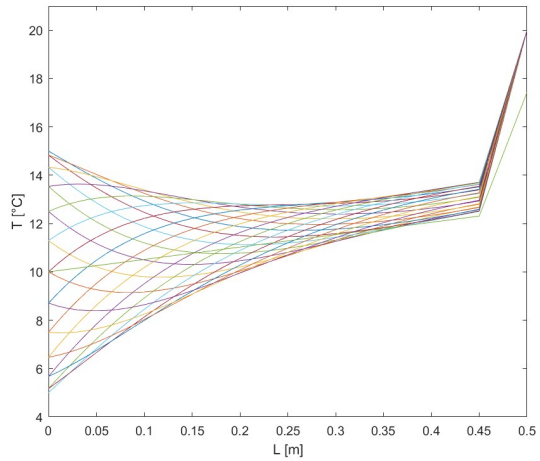


(a) Aislante interno

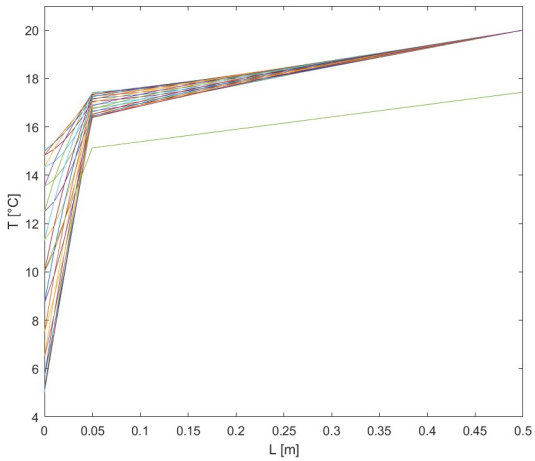


(b) Aislante externo

Figura 24: Variación de Temperatura en las primeras 24 horas (estado transiente)



(a) Aislante interno



(b) Aislante externo

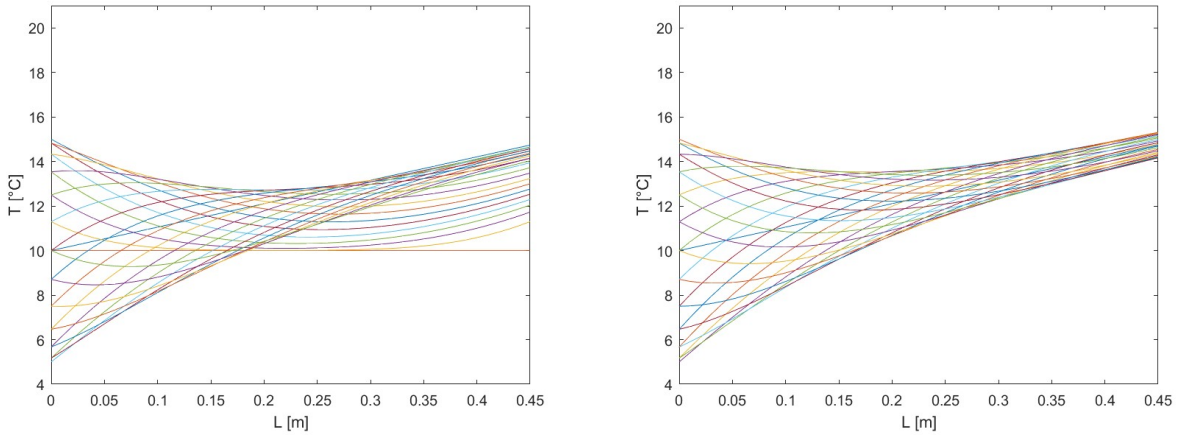
Figura 25: Variación de Temperatura en 24 horas (estado estacionario)

Tabla 5: Tabla para calor total transferido en las próximas 24 horas a 10 días para temperatura fija

Calor total	Asilante adentro	Aislante afuera
q_N [KJ/m2]	1192	1192

3.0.4. Caso Sin Aislante

Variaciones para caso sin aislante



(a) Transiente (b) Estacionario

Figura 26: Variación de Temperatura en 24 horas (estado transiente y estacionario)

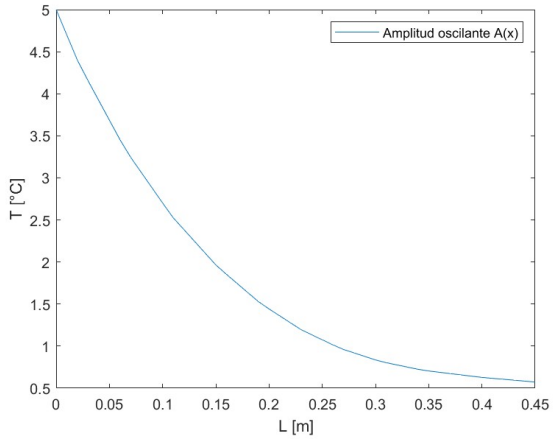


Figura 27: Amplitud $A(x)$ para caso sin aislante

Tabla 6: Tabla para calor total transferido en las próximas 24 en estado transiente y estacionario

Calor total	Transiente	Estacionario
q_N [KJ/m2]	2273	1818

3.0.5. Tiempos de Cálculo

Tiempo de Cálculo para distintos pasos de tiempo con $\Delta x = 0.01$

Tabla 7: Tiempos de cálculo en segundos para modelar 15 días con distintos pasos de tiempo

Método	$\Delta t = 15$	$\Delta t = 60$	$\Delta t = 120$	$\Delta t = 1200$	$\Delta t = 3600$
Euler Explícito	1.5	-	-	-	-
Euler Implícito	42	9	4.9	1.1	0.73
Crank Nicolson	43	8	5.2	1.1	0.73

4. Análisis

Luego de tener las simulaciones realizadas, se nota como concuerdan con la teoría presentada al comienzo. Para ambas configuraciones la temperatura de la pared tiende a la distribución estacionaria. En estado transiente, pareciera que cuando se fija el aislante en la zona interna se tiende al estado estacionario más rápido que con la configuración con el aislante fuera. Esto se puede deber a la gran diferencia de temperatura que existe justo en los bordes del aislante. Se estima que pasado 10 días, ambas configuraciones alcanzan el estado estacionario. En la figura 22 presentadas se puede visualizar como los perfiles de temperatura oscilan en torno al estado estacionario presentado en la metodología en la ecuación 20, lo que valida los resultados. Además, se muestra como existe mayor oscilación para el caso de aislante interno. Esa oscilación es sinusoidal, la cual se ve reflejada en las figuras 20 y 21 para la temperatura y el flujo de calor en el borde interno. La amplitud de oscilación pudo ser obtenida satisfactoriamente en la figura 23, de donde se muestra como esta es mayor en la pared exterior, en donde existe la condición periódica. Notar que para la pared interior la amplitud no es nula pero si bastante baja.

Para el cálculo del calor transferido, se nota que la configuración con aislante interno es bastante mejor ya que transfiere un 55 % de calor menos que el caso de aislante externo. Esto se puede deber a que el caso de aislante interno alcanza más rápido el caso estacionario. Ahora en condiciones estacionarias (10 días), ambas configuraciones transfieren el mismo calor, lo que concuerda con la teoría. Este calor es de 886 [kJ/m2] comparado con 1034 [kJ/m2] y 2322 [kJ/m2] que corresponden a aislante interior y exterior en estado transiente respectivamente. Si se comparan los resultados para el caso sin aislante, se nota que en estado transiente situar el aislante afuera aumenta en un 2 % el calor total, mientras que situarlo adentro disminuye en un 54 %. Claramente confirma que en estado transiente es preferible situar el aislante en la zona interior. Ya en estado estacionario, ambas configuraciones transfieren un 55 % menos calor que para el caso sin aislante. Estos resultados demuestran que utilizar un aislante funciona para disminuir la transferencia de calor dado que se aumenta la "resistencia térmica". Sin embargo, el donde situarlo no es trivial dado que en ciertos intervalos de tiempo, como las primeras 24 horas, puede aumentar el calor transferido. Como se dijo para este caso es preferible situar el aislante en la zona interior. Es importante destacar que esto se debe netamente a la condición inicial. Por esto, situarlo en la zona donde existe el mayor gradiente de temperatura tiene sentido. Si el perfil de temperatura inicial hubiera sido distinto, por ejemplo de $T_\infty = 20°C$, el aislante debería haber estado en la cara externa. De todas formas, la diferencia solo ocurrirá en el estado transiente dado que a largo plazo (estacionario) se demostró que cualquier configuración terminará transfiriendo lo mismo.

Para los casos de temperatura fija en el borde interno, se nota como se alejan bastante de la realidad, generando bastantes diferencias para las distribuciones de temperatura respecto al caso con condición convectiva. Además, se alejan del perfil estacionario. Para el cálculo del calor transferido en estado estacionario, se estiman variaciones respecto al teórico de 38 % lo que es bastante si se necesita exactitud en el valor.

Respecto a los esquemas de discretización, los 3 métodos utilizados fueron satisfactorios para modelar la situación propuesta. Los esquemas fueron aplicados a un espaciamiento espacial de $\Delta x = 0.01$ y $\Delta x = 0.005$ y un paso temporal de 1, 5 y 10 segundos. En las figuras presentadas no se notan diferencias y en el cálculo de calor varían los valores en aproximadamente en un 0.0001 %, lo que en este escenario puede ser insignificante. No se quiso disminuir más el espaciamiento espacial para poder simular de buena manera el aislante, el cual es bastante pequeño comparado con el dominio del problema. La discretización realizada necesita un nodo justo en la interfaz y un Δx constante para no disminuir el orden del método. Esta puede ser una gran desventaja del método de diferencias finitas. Por último, se varió el paso temporal con valores de 15, 60, 120, 1200, 3600 segundos para los 3 esquemas. Para $\Delta t = 15$, se nota el gran costo computacional que tienen los esquemas de Euler Implícito y Crank Nicolson, dado que resuelven un sistema lineal en cada iteración. Es por esto que sus tiempos de cálculo son aprox. 28 veces el tiempo de Euler Explícito. A medida que se aumenta el paso temporal con el espaciamiento en la posición fija, los métodos implícitos empiezan a demorarse cada vez menos llegando a tiempos menores a 1 segundo y con perfiles de temperatura muy parecidos a los otros casos. Además, se nota como el método de Euler Explícito diverge al aumentar el paso temporal dada su estabilidad condicional dada por el número de Fourier. Con lo analizado se puede deslumbrar las ventajas de cada método y sus rangos de aplicación, en donde el Explícito permite cálculos bastante exactos y rápidos para tiempos totales bajos. En cambio los métodos implícitos pueden ser utilizados para simular grandes intervalos de tiempo con grandes pasos temporales sin perder mucha precisión. Por último, el costo computacional del método de Crank Nicolson y Euler Implícito son bastante similares y por ende, se prefiere el método de Crank Nicolson para simulaciones con altos pasos temporales dado su orden 2 en el tiempo. Cabe mencionar que los tiempos de cálculo para las simulaciones hubieran sido mayores si se hubiera trabajado con matrices llenas en comparación con el algoritmo de Thomas utilizado.

5. Conclusión

Mediante el análisis realizado para el problema resuelto, se puede concluir que existe una gran variedad de métodos dependiendo del problema a resolver. Los métodos numéricos pueden ser útiles para resolver ecuaciones diferenciales ordinarias y/o parciales que modelan problemas físicos reales, como lo es por ejemplo la ecuación de calor.

Para el problema, se concluye que se pudo modelar satisfactoriamente la transferencia de calor en el muro. Con los distintos esquemas de discretización y parámetros numéricos se pudo concluir que el aislante sirve para reducir la transferencia de calor. Además, su posición no debe ser aleatoria para reducir el flujo de calor en el estado transiente hasta alcanzar el estacionario. Se infiere que este debe ir siempre en la posición con mayor gradiente de temperatura dada por la condición inicial. Para el caso descrito en este trabajo, para reducir el flujo de calor lo más conveniente es situar el aislante en la pared interna, logrando reducir casi un 55 % el calor total en el primer día respecto al caso sin aislante. Respecto a los esquemas, se pudo evidenciar la estabilidad condicional del Metodo de Euler Explícito, el cual divergió para grandes pasos de tiempo. Por otra parte, los métodos implícitos, Euler Implícito y Crank Nicolson lograron buenas precisiones utilizando incluso pasos de tiempo de 1 hora, permitiendo así modelar incluso semanas. De estos 2 últimos, se prefiere el Método de Crank Nicolson dado su similar costo computacional y su convergencia de segundo orden en el tiempo.

6. Referencias

[1] Chapra, S; Canale, R.(2007). *Métodos numéricos para ingenieros*, 5^a Edición.

[2] Gers, R. Apuntes de clases : *MEC270-Métodos numéricos en ingeniería mecánica.*

[3] Quarteroni, A; Sacco, R; Saleri, F. (2000). *Numerical Mathematics.* Springer.

[4] Cengel, Y. (2007). *Transferencia de Calor y Masa.* 3^a Edición.

[5] Hancock, M (2006). *The 1-D Heat Equation*

7. Anexos

Se presentan los códigos elaborados para la resolución de la ecuación de calor.

7.0.1. Código Principal Pregunta 2

```
1
2  % Programa principal EDP Calor
3  % Caso convectivo en interior
4
5  format compact
6
7  % Modificacion aislante interior o exterior
8  aisl_in = true;
9
10 if aisl_in
11     lambda_1 = 2;
12     pC_1 = 2*10^6;
13     lambda_2 = 0.1;
14     pC_2 = 10^6;
15     L1 = 0.45;
16     L2 = 0.05;
17 else
18     lambda_2 = 2;
19     pC_2 = 2*10^6;
20     lambda_1 = 0.1;
21     pC_1 = 10^6;
22     L2 = 0.45;
23     L1 = 0.05;
24 end
25
26 alpha_1 = lambda_1/pC_1;
27 alpha_2 = lambda_2/pC_2;
28 h = 4;
29 prop = [alpha_1,alpha_2,lambda_1,lambda_2,h];
30
31 Tw = 10;
32 dTw = 5;
33 T_inf = 20;
34 prob = [Tw,dTw,T_inf];
35
36 % parametros numericos
37
38 dx = 0.01;  %% 0.005   0.01
39 dt = 10;    %1   10 15
40
41 t = 0;
42 hrs = 49;
```

```
43 hrsi = 25;
44 tf = hrs*3600;
45
46 N1 = floor(L1/dx);
47 N2 = floor(L2/dx);
48 N = N1 + N2 + 1;
49 Nt = hrs + 1;
50
51 Ti = ones(1,N)*Tw;
52 L = linspace(0,L1+L2,N);
53 k = N1 + 1;
54
55 % outputs para figuras en pared interior
56
57 dt2 = 30;
58 if dt2<dt
59     dt2 = dt;
60 end
61 Nt2f = hrs*3600/dt2;
62 Nt2i = (hrs-24)*3600/dt2+1;
63
64
65 %%%%% EULER EXPLICITO %%%%%%%
66
67 [T_out1,TN] = Euler_Explicito(Ti,N,Nt,t,k,dx,dt,tf,prob,prop,dt2,Nt2f,
       Nt2i);
68 f1 = figure(1);
69 plot_hrs(L,T_out1,hrsi,hrs)
70 hold on
71
72 [q,qj] = Heat_flux_int(TN,dt2,prob,prop);
73
74 T_out = Stationary(prob,prop,L1,L1+L2,k,N);
75 plot(L,T_out,'DisplayName','Stationary')
76 hold on
77 legend('Location', 'Best')
78 set(f1, 'Position', [150 90 650 530]);
79
80 %%%%% EULER IMPLICITO %%%%%%
81
82 [T_out2,TN] = Euler_Implicito(Ti,N,Nt,t,k,dx,dt,tf,prob,prop,dt2,Nt2f,
       Nt2i);
83 f2 = figure(2);
84 plot_hrs(L,T_out2,hrsi,hrs)
85 hold on
86
87 [q,qj] = Heat_flux_int(TN,dt2,prob,prop);
88
```

```
89  T_out = Stationary(prob,prop,L1,L1+L2,k,N);
90  plot(L,T_out,'DisplayName','Stationary')
91  hold on
92  legend('Location', 'Best')
93  set(f2, 'Position', [150 90 650 530]);
94
95  %%%%% CRANK NICOLSON %%%%%%%
96
97
98  [T_out3,TN] = Crank_Nicolson(Ti,N,Nt,t,k,dx,dt,tf,prob,prop,dt2,Nt2f,
        Nt2i);
99  f3 = figure(3);
100 plot_hrs(L,T_out3,hrsi,hrs)
101 hold on
102
103 [q,qj] = Heat_flux_int(TN,dt2,prob,prop);
104
105 T_out = Stationary(prob,prop,L1,L1+L2,k,N);
106 plot(L,T_out,'DisplayName','Stationary')
107 hold on
108 legend('Location', 'Best')
109 set(f3, 'Position', [150 90 650 530]);
110
111 %%%%%% GRAFICO CALOR EN CARA INTERIOR %%%%%
112
113 figure(4)
114 t = linspace(0,24,length(qj));
115 plot(t,qj,'DisplayName','Flujo calor cara interna 24 hrs')
116 ylabel('q [W/m2]')
117 xlabel('t [hrs]')
118 xlim([0 24])
119 legend
120
121 %%%%% GRAFICO TEMPERATURA EN  INTERIOR %%%%%
122
123 figure(5)
124 plot(t,TN,'DisplayName','Temperatura cara iterna')
125 ylabel('T [ C ]')
126 xlabel('t [hrs]')
127 xlim([0 24])
128 legend
129
130 %%%%% AMPLITUD TMPERATURA OSCILANTE %%%%
131
132 figure(6)
133 Tsin = T_out3(hrs-24:hrs,:);
134 T_tot = zeros(1,N);
135 for j=1:N
```

```
136     xx = max(Tsin(:,j));
137     T_tot(j) = xx;
138 end
139 plot(L,T_tot-T_out)
140 xlabel('L [m]')
141 ylabel('T [ C ]')
142 legend('Amplitud oscilante A(x)')
143
144
145
146 %%%%%%%%%%%%%%%%%%%% FUNCTIONS
       %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
147
148 % Funcion plots
149
150 function plot_hrs(L,T,ti,tf)
151     if ti==0
152         ti=1;
153         tf = tf + 1;
154     end
155     for j=ti:tf
156         %a = mod(j,24);
157         % if a==0
158         %     a=24
159         % elseif a==1
160         %     a=25
161         % end
162         plot(L,T(j,:),'DisplayName',strcat(int2str(j-1),'hrs'))
163         hold on
164     end
165     xlabel('L [m]')
166     ylabel('T [ C ]')
167     legend
168     ylim([4 21])
169 end
```

7.0.2. Función Euler Explícito

```
1
2
3 % Euler explicito
4
5 function [T_out,TN_out] = Euler_Explicito(Ti,N,Nt,t,k,dx,dt,tf,prob,
      prop,dt2,Nt2f,Nt2i)
6
7     r = num2cell(prop);
8     [alpha_1,alpha_2,lambda_1,lambda_2,h] = deal(r{:});
9     r = num2cell(prob);
10    [Tw,dTw,T_inf] = deal(r{:});
```

```
11
12      T = zeros(1,N);
13      T_old = Ti;
14      T_out = zeros(Nt,N);
15      TN = zeros(1,Nt2f);
16      z = 1;
17      zz = 1;
18      T_out(z,:) = Ti(:);
19
20      while t<tf
21          t = t + dt;
22          for i=1:N
23              if i==1
24                  T(i) = Tw + dTw*sin((2*pi*t/3600)/(24));
25                  %T(i) = Tw;
26              elseif i>1 && i<k
27                  T(i) = T_old(i) + (dt/(dx^2))*(T_old(i+1)-2*T_old(i)+
                        T_old(i-1))*alpha_1;
28              elseif i>k && i<N
29                  T(i) = T_old(i) + (dt/(dx^2))*(T_old(i+1)-2*T_old(i)+
                        T_old(i-1))*alpha_2;
30              elseif i==N
31                  %T(i) = T_inf;
32                  T(i) = (lambda_2*T(i-1)/dx + h*T_inf)/(h+lambda_2/dx);
33              end
34          end
35          T(k) = (lambda_2*T(k+1)+lambda_1*T(k-1))/(lambda_1+lambda_2);
36          T_old(:) = T(:);
37          if mod(t,3600)==0
38              z = z + 1;
39              T_out(z,:) = T(:);
40          end
41          if mod(t,dt2) == 0
42              zz = zz + 1;
43              TN(zz) = T(N);
44          end
45      end
46      TN_out = TN(Nt2i:Nt2f);
47  end
```

7.0.3. Función Euler Implícito

```
1
2   % Euler implicito
3
4   function [T_out,TN_out] = Euler_Implicito(Ti,N,Nt,t,k,dx,dt,tf,prob,
        prop,dt2,Nt2f,Nt2i)
5
6       r = num2cell(prop);
```

```
7       [alpha_1,alpha_2,lambda_1,lambda_2,h] = deal(r{:});
8       r = num2cell(prob);
9       [Tw,dTw,T_inf] = deal(r{:});
10
11      T_old(:) = Ti(:);
12      T_out = zeros(Nt,N);
13      TN = zeros(1,Nt2f);
14      z = 1;
15      zzz = 1;
16      T_out(z,:) = Ti(:);
17
18      sigma_1 = alpha_1*dt/(dx^2);
19      sigma_2 = alpha_2*dt/(dx^2);
20      theta = lambda_2/(2*dx);
21      zz = 0;
22      if mod(N,3) == 2
23          zz = 1;
24          M = floor(N/3) + zz;
25      elseif mod(N,3) == 0
26          M = floor(N/3);
27      end
28      kk = floor(k/3);
29      if mod(k,3)==0
30          kk = kk - 1;
31      end
32
33      A = zeros(M,3,3);
34      B = zeros(M,3,3);
35      C = zeros(M,3,3);
36      F = zeros(M,3);
37      for i=1:M
38          if i==1
39              C(i,:,:) = [1 0 0; -1 2+1/sigma_1 -1; 0 -1 2+1/sigma_1];
40              B(i,:,:) = [0 0 0; 0 0 0; -1 0 0];
41          elseif i<=kk
42              A(i,:,:) = [0 0 -1; 0 0 0; 0 0 0];
43              C(i,:,:) = [2+1/sigma_1 -1 0; -1 2+1/sigma_1 -1; 0 -1 2+1/
                    sigma_1];
44              B(i,:,:) = [0 0 0; 0 0 0; -1 0 0];
45          elseif i==kk+1
46              if mod(k,3)==0
47                  A(i,:,:) = [0 0 -1; 0 0 0; 0 0 0];
48                  C(i,:,:) = [2+1/sigma_1 -1 0; -1 2+1/sigma_1 -1;
                        lambda_1 -4*lambda_1 3*(lambda_1+lambda_2)];
49                  B(i,:,:) = [0 0 0; 0 0 0; -4*lambda_2 lambda_2 0];
50              else
51                  A(i,:,:) = [0 lambda_1 -4*lambda_1; 0 0 0; 0 0 0];
52                  C(i,:,:) = [3*(lambda_1+lambda_2) -4*lambda_2 lambda_2;
```

```
                    -1 2+1/sigma_2 -1; 0 -1 2+1/sigma_2];
53                  B(i,:,:) = [0 0 0; 0 0 0; -1 0 0];
54              end
55          elseif i>kk+1 && i<M
56              A(i,:,:) = [0 0 -1; 0 0 0; 0 0 0];
57              C(i,:,:) = [2+1/sigma_2 -1 0; -1 2+1/sigma_2 -1; 0 -1 2+1/
                    sigma_2];
58              B(i,:,:) = [0 0 0; 0 0 0; -1 0 0];
59          elseif i==M
60              if mod(N,3) == 2
61                  A(i,:,:) = [0 0 -1; 0 0 theta; 0 0 0];
62                  C(i,:,:) = [2+1/sigma_2 -1 0; -4*theta 3*theta+h 0; 0 0
                        1];
63              elseif mod(N,3) == 0
64                  A(i,:,:) = [0 0 -1; 0 0 0; 0 0 0];
65                  C(i,:,:) = [2+1/sigma_2 -1 0; -1 2+1/sigma_2 -1; theta
                        -4*theta 3*theta+h];
66              end
67          end
68      end
69
70      while t<tf
71          t = t + dt;
72          for i=1:M
73              if i==1
74                  F(i,1) = Tw + dTw*sin((2*pi*t/3600)/(24));
75                  F(i,2) = T_old(i*3-1)/sigma_1;
76                  F(i,3) = T_old(i*3)/sigma_1;
77              elseif i<=kk
78                  F(i,1) = T_old(i*3-2)/sigma_1;
79                  F(i,2) = T_old(i*3-1)/sigma_1;
80                  F(i,3) = T_old(i*3)/sigma_1;
81              elseif i==kk+1
82                  if mod(k,3)==0
83                      F(i,1) = T_old(i*3-2)/sigma_1;
84                      F(i,2) = T_old(i*3-1)/sigma_1;
85                      F(i,3) = 0;
86                  else
87                      F(i,1) = 0;
88                      F(i,2) = T_old(i*3-1)/sigma_2;
89                      F(i,3) = T_old(i*3)/sigma_2;
90                  end
91              elseif i>kk+1 && i<M
92                  F(i,1) = T_old(i*3-2)/sigma_2;
93                  F(i,2) = T_old(i*3-1)/sigma_2;
94                  F(i,3) = T_old(i*3)/sigma_2;
95              elseif i==M
96                  if mod(N,3) == 2
```

```
97                      F(i,1) = T_old(i*3-2)/sigma_2;
98                      F(i,2) =  h*T_inf;
99                      F(i,3) = 1;
100                 elseif mod(N,3) == 0
101                     F(i,1) = T_old(i*3-2)/sigma_2;
102                     F(i,2) = T_old(i*3-1)/sigma_2;
103                     F(i,3) = h*T_inf;
104                 end
105             end
106         end
107         T_star = Thomas_algorithm(A,B,C,F,M);
108         T_old(:) = T_star(1:M*3-zz);
109         if mod(t,3600)==0
110             z = z + 1;
111             T_out(z,:) = T_old(:);
112         end
113         if mod(t,dt2) == 0
114             zzz = zzz + 1;
115             TN(zzz) = T_old(N);
116         end
117     end
118     TN_out = TN(Nt2i:Nt2f);
119 end
```

7.0.4. Función Crank Nicolson

```
1
2  % Crank Nicolson
3
4  function [T_out,TN_out] = Crank_Nicolson(Ti,N,Nt,t,k,dx,dt,tf,prob,prop
       ,dt2,Nt2f,Nt2i)
5
6      r = num2cell(prop);
7      [alpha_1,alpha_2,lambda_1,lambda_2,h] = deal(r{:});
8      r = num2cell(prob);
9      [Tw,dTw,T_inf] = deal(r{:});
10
11     T_old(:) = Ti(:);
12     T_out = zeros(Nt,N);
13     TN = zeros(1,Nt2f);
14     z = 1;
15     zzz = 1;
16     T_out(z,:) = Ti(:);
17
18     sigma_1 = alpha_1*dt/(dx^2);
19     sigma_2 = alpha_2*dt/(dx^2);
20     theta = lambda_2/(2*dx);
21     zz = 0;
22     if mod(N,3) == 2
```

```
23          zz = 1;
24          M = floor(N/3) + zz;
25      elseif mod(N,3) == 0
26          M = floor(N/3);
27      end
28      kk = floor(k/3);
29      if mod(k,3)==0
30          kk = kk - 1;
31      end
32
33      A = zeros(M,3,3);
34      B = zeros(M,3,3);
35      C = zeros(M,3,3);
36      F = zeros(M,3);
37      for i=1:M
38          if i==1
39              C(i,:,:) = [1 0 0; -1 2+2/sigma_1 -1; 0 -1 2+2/sigma_1];
40              B(i,:,:) = [0 0 0; 0 0 0; -1 0 0];
41          elseif i<=kk
42              A(i,:,:) = [0 0 -1; 0 0 0; 0 0 0];
43              C(i,:,:) = [2+2/sigma_1 -1 0; -1 2+2/sigma_1 -1; 0 -1 2+2/
                    sigma_1];
44              B(i,:,:) = [0 0 0; 0 0 0; -1 0 0];
45          elseif i==kk+1
46              if mod(k,3)==0
47                  A(i,:,:) = [0 0 -1; 0 0 0; 0 0 0];
48                  C(i,:,:) = [2+2/sigma_1 -1 0; -1 2+2/sigma_1 -1;
                        lambda_1 -4*lambda_1 3*(lambda_1+lambda_2)];
49                  B(i,:,:) = [0 0 0; 0 0 0; -4*lambda_2 lambda_2 0];
50              else
51                  A(i,:,:) = [0 lambda_1 -4*lambda_1; 0 0 0; 0 0 0];
52                  C(i,:,:) = [3*(lambda_1+lambda_2) -4*lambda_2 lambda_2;
                        -1 2+2/sigma_2 -1; 0 -1 2+2/sigma_2];
53                  B(i,:,:) = [0 0 0; 0 0 0; -1 0 0];
54              end
55          elseif i>kk+1 && i<M
56              A(i,:,:) = [0 0 -1; 0 0 0; 0 0 0];
57              C(i,:,:) = [2+2/sigma_2 -1 0; -1 2+2/sigma_2 -1; 0 -1 2+2/
                    sigma_2];
58              B(i,:,:) = [0 0 0; 0 0 0; -1 0 0];
59          elseif i==M
60              if mod(N,3) == 2
61                  A(i,:,:) = [0 0 -1; 0 0 theta; 0 0 0];
62                  C(i,:,:) = [2+2/sigma_2 -1 0; -4*theta 3*theta+h 0; 0 0
                        1];
63              elseif mod(N,3) == 0
64                  A(i,:,:) = [0 0 -1; 0 0 0; 0 0 0];
65                  C(i,:,:) = [2+2/sigma_2 -1 0; -1 2+2/sigma_2 -1; theta
```

```
                    -4*theta 3*theta+h];
66              end
67          end
68      end
69
70      while t<tf
71          t = t + dt;
72          for i=1:M
73              if i==1
74                  F(i,1) = Tw + dTw*sin((2*pi*t/3600)/(24));
75                  F(i,2) = T_old(i*3-2)+(2/sigma_1-2)*T_old(i*3-1)+T_old(
                        i*3);
76                  F(i,3) = T_old(i*3-1)+(2/sigma_1-2)*T_old(i*3)+T_old(i
                        *3+1);
77              elseif i<=kk
78                  F(i,1) = T_old(i*3-3)+(2/sigma_1-2)*T_old(i*3-2)+T_old(
                        i*3-1);
79                  F(i,2) = T_old(i*3-2)+(2/sigma_1-2)*T_old(i*3-1)+T_old(
                        i*3);
80                  F(i,3) = T_old(i*3-1)+(2/sigma_1-2)*T_old(i*3)+T_old(i
                        *3+1);
81              elseif i==kk+1
82                  if mod(k,3)==0
83                      F(i,1) = T_old(i*3-3)+(2/sigma_1-2)*T_old(i*3-2)+
                            T_old(i*3-1);
84                      F(i,2) = T_old(i*3-2)+(2/sigma_1-2)*T_old(i*3-1)+
                            T_old(i*3);
85                      F(i,3) = 0;
86                  else
87                      F(i,1) = 0;
88                      F(i,2) = T_old(i*3-2)+(2/sigma_2-2)*T_old(i*3-1)+
                            T_old(i*3);
89                      F(i,3) = T_old(i*3-1)+(2/sigma_2-2)*T_old(i*3)+
                            T_old(i*3+1);
90                  end
91              elseif i>kk+1 && i<M
92                  F(i,1) = T_old(i*3-3)+(2/sigma_2-2)*T_old(i*3-2)+T_old(
                        i*3-1);
93                  F(i,2) = T_old(i*3-2)+(2/sigma_2-2)*T_old(i*3-1)+T_old(
                        i*3);
94                  F(i,3) = T_old(i*3-1)+(2/sigma_2-2)*T_old(i*3)+T_old(i
                        *3+1);
95              elseif i==M
96                  if mod(N,3) == 2
97                      F(i,1) = T_old(i*3-3)+(2/sigma_2-2)*T_old(i*3-2)+
                            T_old(i*3-1);
98                      F(i,2) =  h*T_inf;
99                      F(i,3) = 1;
```

```
100                 elseif mod(N,3) == 0
101                     F(i,1) = T_old(i*3-3)+(2/sigma_2-2)*T_old(i*3-2)+
                            T_old(i*3-1);
102                     F(i,2) = T_old(i*3-2)+(2/sigma_2-2)*T_old(i*3-1)+
                            T_old(i*3);
103                     F(i,3) = h*T_inf;
104                 end
105             end
106         end
107         T_star = Thomas_algorithm(A,B,C,F,M);
108         T_old(:) = T_star(1:M*3-zz);
109         if mod(t,3600)==0
110             z = z + 1;
111             T_out(z,:) = T_old(:);
112         end
113         if mod(t,dt2) == 0
114             zzz = zzz + 1;
115             TN(zzz) = T_old(N);
116         end
117     end
118     TN_out = TN(Nt2i:Nt2f);
119 end
```

7.0.5. Función Cálculo del Calor con Método del Trapecio

```
1
2
3  % Calculo del flujo de calor en interior
4
5  function [q,qj] = Heat_flux_int(T,dt,prob,prop)
6
7      r = num2cell(prop);
8      [~,~,~,~,h] = deal(r{:});
9      r = num2cell(prob);
10     [~,~,T_inf] = deal(r{:});
11     qj(:) = h*(T(:)-T_inf);
12
13     q = Total_heat(qj,dt);
14
15     function [qt] = Total_heat(qj,dt)
16         sum = 0;
17         for j=1:length(qj)-1
18             sum = sum + qj(j) + qj(j+1);
19         end
20         qt = sum*dt/2;
21     end
22
23 end
```

7.0.6. Función Distribución Temperatura Estacionaria

```
% Perfil de temperatura estacionaria teorica

function [T_out] = Stationary(prob,prop,L1,L,k,N)

    r = num2cell(prop);
    [~,~,lambda_1,lambda_2,h] = deal(r{:});
    r = num2cell(prob);
    [Tw,~,T_inf] = deal(r{:});

    x = linspace(0,L,N);

    phi = h*(L*lambda_1-L1*(lambda_1-lambda_2))+lambda_1*lambda_2;
    c1 = h*lambda_2*(T_inf-Tw)/phi;
    c2 = h*lambda_1*(T_inf-Tw)/phi;
    c3 = (h*(L*lambda_1*Tw-L1*(lambda_1-lambda_2)*T_inf)+lambda_1*
       lambda_2*Tw)/phi;

    T_out = zeros(1,N);
    T_out(1:k) = c1*x(1:k) + Tw;
    T_out(k+1:N) = c2*x(k+1:N) + c3;

end
```

7.0.7. Algoritmo de Thomas

```

% Algoritmo de thomas por bloques

function [X_out] = Thomas(A,B,C,F,n)

    psi_1 = zeros(n,3,3);
    psi_2 = zeros(n,3);
    X = zeros(n,3);
    X_out = zeros(1,n*3);

    psi_1(1,:,:) = (to_mat(C,1))\to_mat(B,1);
    psi_2(1,:) = transpose((to_mat(C,1))\to_vec(F,1)');


    for i=2:n-1
        psi_1(i,:,:) = (to_mat(C,i)-to_mat(A,i)*to_mat(psi_1,i-1))\
           to_mat(B,i);
        psi_2(i,:) = transpose((to_mat(C,i)-to_mat(A,i)*to_mat(psi_1,i
           -1))\transpose(to_vec(F,i)-transpose(to_mat(A,i)*to_vec(
           psi_2,i-1)')));
    end
    psi_2(n,:) = transpose((to_mat(C,n)-to_mat(A,n)*to_mat(psi_1,n-1))\
       transpose(to_vec(F,n)-transpose(to_mat(A,n)*to_vec(psi_2,n-1)'))
```

```
        );
20
21      X(n,:) = to_vec(psi_2,n);
22      for ss=1:n-1
23          i = n-ss;
24          X(i,:) = to_vec(psi_2,i) - transpose(to_mat(psi_1,i)*to_vec(X,i
                +1)');
25      end
26
27      for i=1:n
28          X_out(i*3-2) = X(i,1);
29          X_out(i*3-1) = X(i,2);
30          X_out(i*3) = X(i,3);
31      end
32
33
34      function [Z] = to_mat(X,q)
35          Z = zeros(3,3);
36          for l=1:3
37              for j=1:3
38                  Z(l,j) = X(q,l,j);
39              end
40          end
41      end
42
43      function [Z] = to_vec(X,q)
44          Z = zeros(1,3);
45          for l=1:3
46              Z(l) = X(q,l);
47          end
48      end
49
50  end
```