Trabajo Final Ecuaciones en Derivadas Parciales

Pablo A. Osorio M.[1] and Jose M. Gil V.[2]

[1] *Universidad EAFIT,Ingenieria Matemática. Email: paosoriom@eafit.edu.co*
[2] *Universidad EAFIT,Ingenieria Matemática. Email: jmgilv@eafit.edu.co*

Abstract

En el desarrollo de este trabajo se evaluará e implementará el contenido desarrollado por [2], donde se propone un método de diferencias finitas para desarrollar un problema de frontera no-local para una ecuación del calor en dos dimensiones. En primera instancia se pretende explicar la solución del problema asociado a la no-localidad dada por el valor de μ. Posteriormente , usando este resultado se extiende el ejercicio a la clásica solución de la ecuación del calor bidimensional usando el método planteado por el autor e implementando un método alternativo de solución. Para todo el despliegue del análisis, se evalúan los errores correspondientes y las conclusiones convenientes para los mismos.

Palabras clave: Ecuación del calor,Diferencias finitas, Discretización,Ecuaciones diferenciales parciales, problemas de frontera no-locales.

Contents

Análisis de paper (2020)

Autores
Pablo Osorio y Jose Gil

Gráficas

Tablas

Códigos

1 Introducción

Como bien se detalla en [2], la solución de la ecuación de la ecuación del calor para problemas de difusión con condiciones de frontera no locales ha sido bastante estudiada en la literatura. El propósito del trabajo es analizar en esta medida los elementos desarrollados por el trabajo académico mencionado, donde, como desarrollo central del trabajo se presenta la siguiente expresión:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

Con la condición inicial

$$u(x, y, 0) = f(x, y), \text{ con las condiciones iniciales } 0 \leqslant x, y \leqslant 1$$

Las condiciones de frontera

$$\frac{\partial u(0, y, t)}{\partial x} = g_0(y, t), \;\; 0 < t \leqslant 1 \;\; 0 \leqslant y \leqslant 1$$

$$\frac{\partial u(1, y, t)}{\partial x} = g_1(y, t), \;\; 0 < t \leqslant 1 \;\; 0 \leqslant y \leqslant 1$$

$$u(x, 1, t) = h_1(x, t), \;\; 0 < t \leqslant 1 \;\; 0 \leqslant x \leqslant 1$$

$$u(x, 0, t) = h_0(x, t)\mu(t), \;\; 0 < t \leqslant 1 \;\; 0 \leqslant x \leqslant 1$$

Y la condición de frontera no local como

$$\int_0^1 \int_0^1 u(x, y, t) dx dy = m(t), \;\; 0 \leq x, y \leq 1$$

.

2 Análisis preliminar

Si bien el documento no es explicito completamente muchos de los métodos que utiliza para la consecución de sus procesos, a través de una exhausta evaluación el documento contiene errores mínimos de escritura. Adicionalmente, al momento de implementar el diseño planteado para la solución de diversos sistemas, algunos parecen no funcionar completamente, y adicionalmente, son emitidos sin una clara justificación.

3 Análisis de $\mu(t)$

En esta sección se tratará la forma que se implementan los esquemas planteados del documento base para la hallar una solucion aproximada de $\mu(t)$ [2].
El paper plantea que $\mu(t)$ tiene la siguiente relación 1, donde $v(0, t)$ es desconocido. Esta relación planteada no tiene su debida justificación en el paper [2], por lo cual se asume como valida.

$$\mu(t) = \frac{v(0, t)}{\int_0^1 h_0(x, t) dx} \tag{1}$$

donde $v(y, t)$ es de la forma 2, y para hallar los valores de $v(y, t)$ se plantea una solución numérica la cual es de la forma 3. Y con este nuevo planteamiento de procede a realizar un esquema de diferencias finitas para susolución

$$v(y, t) = \int_0^1 u(x, y, t) dx, 0 \leq x, y \leq 1 \tag{2}$$

$$\frac{\partial v}{\partial t} = \frac{\partial^2 v}{\partial y^2} + S(y, t) \tag{3}$$

Donde $S(y,t) = g_1(y,t) - g_0(y,t)$ y con las respectivas condiciones iniciales 4, condiciones de frontera 5 y condiciones no locales 6. Con S, F, G y m conocidas.

$$v(y,0) = F(y), 0 \leq y \leq 1 \tag{4}$$

$$v(1,t) = G(t), 0 < y \leq 1 \tag{5}$$

$$\int_0^1 v(y,t)dy = m(t), 0 \leq y \leq 1 \tag{6}$$

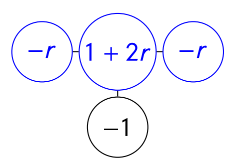
Ahora obtenemos un nuevo problema7 para la solución del problema numérico anterior 3, realizando la siguiente sustitución $w = (\partial v/\partial x)$ y se obtiene el problema 7

$$\frac{\partial w}{\partial t} = \frac{\partial^2 w}{\partial x^2} + H(x,t) \tag{7}$$

Donde $H(x,t) = \frac{\partial S(x,t)}{\partial x}$ y se conocen las condiciones iniciales, de frontera y frontera no local. Ahora con este nuevo problema 7 se procederá a realizar los esquemas matemáticos y numéricos para la solución de $\mu(t)$

3.1 Programación de $\mu(t)$ planteada

Para resolver este problema, primero se tiene que encontar los valores de W_i^n en la ecuación 7, y se hace por medio de un esquema implícito en tiempo y diferencias centradas en espacio (ver stencil del problema más adelante). Segundo, con los resultados de W_i^n se obtienen los resultados de v_i^n por medio de la relación planteada en 8. Y por ultimo, con los resultados anteriores se pueden encontrar los diferentes valores para μ^n con la relación planteada anteriormente 1



$$v(x,t) = g(t) - \int_x^1 w(z,t)dz \tag{8}$$

Al momento de realizar la programación , primero, se crea la malla temporal-espacial del problema , donde se plantean los limites del problema y el dominio, el espacio en $x \in [0,1]$ y el tiempo en $t \in [0,1]$, al discretizar tiempo y espacio en $M = 20$ intervalos para el espacio y $N = 1600$ intervalos para el tiempo; haciendo los pasas $h = 0.05$ paso en el espacio y el paso en el tiempo $K = 0.000625$ para tener estabilidad en el sistema.
Luego se crea una matriz $MW_{(M+1)(N+1)}$ la cual tendrá los valores para los diferentes W_i^n . Se crean los vectores con el tiempo y el espacio discretizado, cada uno con su respectivo tamaño de paso. La primera fila de WM son condiciones iniciales dadas por funciones del problema. Para el resto de los elementos se encontrarán a a partir de un esquema implícito en el tiempo y diferencias centradas en el espacio. Con las condiciones iniciales planteadas en el paper, resolviendo un sistema de ecuaciones para cada tiempo n que se resuelve un sistema en una matriz $A2_{(M+1)(M+1)}$. Después con los resultados de los W_i^n, se implementa el método de integración trapezoidal planteado en el documento 9 con los diferentes W_i^n en cada tiempo n para hallar los valores de V_i^n .
Al finalizar, con los valores de V_i^n se encuentran los valores de μ_0^n con una relación dada 1. Para el análisis de los resultados se gráfica la aproximación de la solución de $\mu(t)$ y la solución exacta de $\mu(t)$; además, se realiza una tabla de errores para las diferencias entre ambas funciones, los errores son planteados en 13

3.2 Resultados de la estructura planteada

Tras programar las estructuras del documento, se obtienen los errores mostrados en la tabla 1 y la gráfica de la solución aproximada para $\mu(t)$ en la gráfica 1, estos errores no coinciden con los mostrados en el documento de análisis, estos errores son mucho más grandes y los del paper son pequeños.

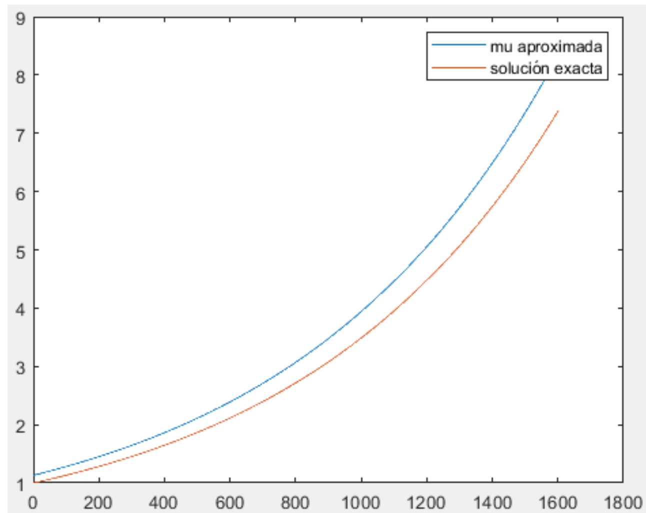


Figure 1. Aproximación de μ con modelo original del paper

t	Exacta	Error
0.1	1.221402	-0.160776
0.2	1.491824	-0.195608
0.3	1.822118	-0.238150
0.4	2.225540	-0.290112
0.5	2.718281	-0.353579
0.6	3.320116	-0.431097
0.7	4.055199	-0.525777
0.8	4.953032	-0.641421
0.9	6.049647	-0.782668
1	7.389056	-0.955187

Table 1. Tabla de errores programa original con método trapezoidal del paper

4 Análisis de $u(x, y, t)$

4.1 Programación de $u(x, y, t)$ planteada

Para la solución del ecuación del calor en dos dimensiones se tiene la siguiente formulación del problema:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

Con la condición inicial

$$u(x, y, 0) = f(x, y), \text{ con las condiciones iniciales } 0 \leqslant x, y \leqslant 1$$

y las condiciones de frontera

$$\frac{\partial u(0,y,t)}{\partial x} = g_0(y,t), \;\; 0 < t \leqslant 1 \;\; 0 \leqslant y \leqslant 1$$

$$\frac{\partial u(1,y,t)}{\partial x} = g_1(y,t), \;\; 0 < t \leqslant 1 \;\; 0 \leqslant y \leqslant 1$$

$$u(x,1,t) = h_1(x,t), \;\; 0 < t \leqslant 1 \;\; 0 \leqslant x \leqslant 1$$

$$u(x,0,t) = h_2(x,t) \;\; 0 < t \leqslant 1 \;\; 0 \leqslant x \leqslant 1$$

donde

$$h_2(x,t) = h_0(x,t)\mu(t)$$

[2]
En este sentido el dominio se discretiza de la siguiente manera:

$$[0,1]^2 x [0,T] \;\text{ como }\; M^2 x N$$

con paso de $h = \frac{1}{M}$ para ambas direcciones y un paso en el tiempo de $k = \frac{T}{N}$. Los puntos de la malla son dados por:

$$x_i = ih, \;\; i = 0,1,2,\cdots,M$$

$$y_j = jh, \;\; j = 0,1,2,\cdots,M$$

$$t_n = nk, \;\; n = 0,1,2,\cdots,N$$

De esta manera se declara un sistema explícito para el tiempo y centradas para el espacio, resultando la siguiente estructura:

$$u_{i,j}^{n+1} = s(u_{i+1,j}^n + u_{i,j+1}^n + u_{i-1,j}^n + u_{i,j-1}^n) + (1+4s)u_{i,j}^n$$

con $s = \frac{k}{h^2}$ Y con las siguientes condiciones fronteras

$$u_{i,0}^n = h_2(ih,nk), \;\; i = 0,1,2,\cdots,M$$

$$u_{i,M}^n = h_2(ih,nk), \;\; i = 0,1,2,\cdots,M$$

$$u_{M,j}^{n+1} = (1-2s)u_{M,j}^n + 2su_{M-1,j}^n + 2shg_1(jh,nk)$$

$$u_{0,j}^{n+1} = (1-2s)u_{0,j}^n + 2su_{1,j}^n + 2shg_0(jh,nk)$$

[2]
De esta manera se plantea un sistema de ecuaciones tridimensionales, tales que cada una represente un paso en el tiempo para su respectiva maya de x y y. Para evitar la repetición en las esquinas, se plantea que estas solo tienen el valor de los valores de las fronteras ofrecidas por la superior y la inferior.

4.2 Resultados de la estructura planteada

Planteando el sistema propuesto por el documento [2] de una manera apropiada, los resultados no son esperados, pues, el error comienza a propagarse para la derivada propuesta en $u_{0,j}^{n+1}$, que representa el lado izquierdo.2 Los resultados son evidentemente, con un error muy alto.2

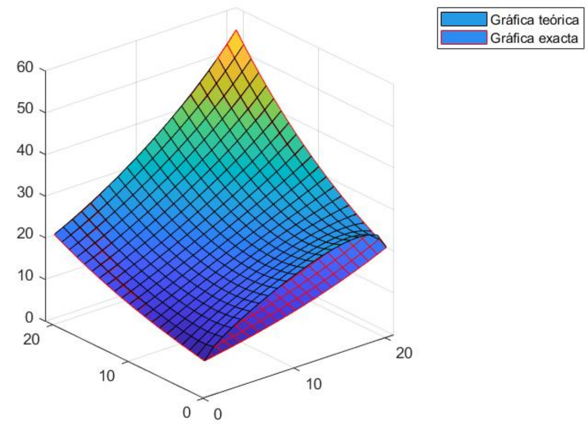


Figure 2. Aproximación de u con el método planteado para T=1

Ut	Uex	Error
7.3891	7.3891	0
9.7254	8.1662	1.5592
11.216	9.025	2.1908
12.518	9.9742	2.5435
13.73	11.023	2.7066
14.912	12.182	2.7295
16.11	13.464	2.6462
17.363	14.88	2.4837
18.71	16.445	2.2649
20.184	18.174	2.0101
21.823	20.086	1.7371
23.659	22.198	1.4613
25.727	24.533	1.195
28.06	27.113	0.94721
30.688	29.964	0.72388
33.643	33.115	0.52781
36.958	36.598	0.35942
40.665	40.447	0.21756
44.802	44.701	0.10084
49.413	49.402	0.010444
54.598	54.598	0

Table 2. Tabla de errores método planteado con fronteras propuestas

5 Aportes y correcciones

5.1 Correcciones para $\mu(t)$

Método trapezoidal

En el documento original se tiene un método trapezoidal para la integración numérica[2] 9, el cual tiene un error con los índices, debido a que no abarca todos los términos y esto hace que se genere un gran error al momento de calcular V_i^n .

$$v_i^n = g^n - \frac{h}{2}(w_i^n + 2w_{i+1}^n + \cdots + 2w_{M-2}^n + w_{M-1}^n) \tag{9}$$

Al momento de realizar el modelo original, se tuvo presente este error y se corrigió en esta sección, ecuación 10, debido a que se pensó que era un error de escritura. Y porque al realizar los cálculos con el método planteado allí, los errores no permiten hacer un análisis apropiado, ver tabla 1, los cuales se pueden ver a simple vista en la figura 1. Por esto, se corrigió en la implementación a la ecución 10.

$$v_i^n = g^n - \frac{h}{2}(w_i^n + 2w_{i+1}^n + \cdots + 2w_{M-1}^n + w_M^n) \tag{10}$$

Y corrigiendo estos simples índices, se puede mejorar sustancialmente la gráfica y los errores, como se ve en la gráfica 3 y en la tabla 3

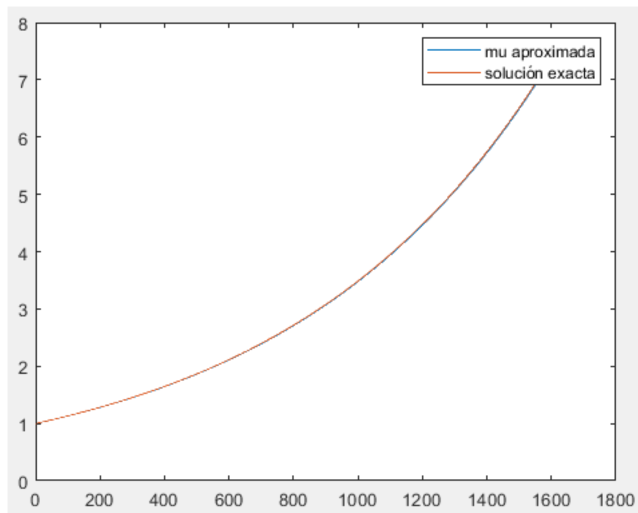


Figure 3. Aproximación de μ con método de integración trapezoidal del paper

t	Exacta	Error
0.1	1.221402	0.001241
0.2	1.491824	0.002321
0.3	1.822118	0.003641
0.4	2.225540	0.005253
0.5	2.718281	0.007222
0.6	3.320116	0.009627
0.7	4.055199	0.01256
0.8	4.953032	0.01615
0.9	6.049647	0.02053
1	7.389056	0.02588496

Table 3. Tabla de errores programa original con método trapezoidal corregido

5.2 Aportes para $\mu(t)$

algunas implementaciones realizadas en diferentes partes.

Cambio método numérico de integración

De la menera que se plantea inicialmente, se usa el método trapezoidal para el proceso de integración, en vez de usar este, usaremos el método de cuadratura de Gauss-Legendre, el cual consiste en aproximar la integral por medio de una sumatoria de términos, que son el producto de unos pesos w_i multiplicados por los valores de la función evaluada en los x_i , ver ecuación 11, estos pesos y estos puntos son dados por el polinomio de Legendre, ecuación 12.

$$\int_{-1}^{1} f(x)dx \approx \sum_{i=1}^{n} w_i f(x_i) \tag{11}$$

$$w_i = \frac{2}{(1-x_i^2)[P_n'(x_i)]} \tag{12}$$

Para obtener los pesos w_i y los puntos x_i usamos una función existente (**funcion**)[1].
Los errores de la aproximación de $\mu(t)$ con este método de integración se pueden ver en la tabla 4 y la gráfica donde se ve la diferencia en gráfica 4.
Con esta modificación empeora la solución usando los x_i generados por el polinomio de Legendre (12), pero al usar los x_i equiespaciados con el paso de h la solución mejora comparada con los x_i de Legendre. En la gráfica 5 se puede ver una mejor aproximación , reflejados igualmente en los errores de la tabla 5

t	Exacta	Error
0.1	1.221402	-0,626438
0.2	1.491824	-0,731397
0.3	1.822118	-0,829202
0.4	2.225540	-0,947648
0.5	2.718281	-1,092290
0.6	3.320116	-1,268954
0.7	4.055199	-1,484733
0.8	4.953032	-1,748285
0.9	6.049647	-2,070189
1	7.389056	-2,463364

Table 4. Tabla de errores con método de integración numérica cuadratura Gauss-Legendre

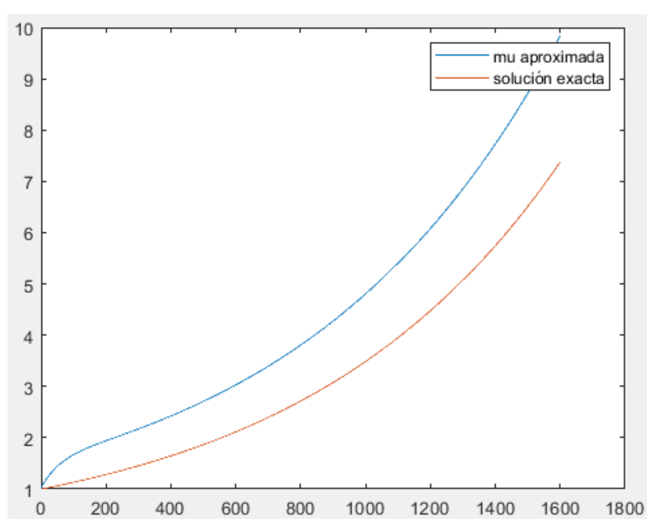


Figure 4. Aproximación de μ con método de integración cuadratura Gauss-Legendre

Cambio de método numérico explícito para calcular W_i^n

En este cambio se remplaza el método para encontrar los W_i^n con un esquema explícito en tiempo y diferencias centradas en espacio(ver stencil relacionado), ver tabla de errores 6 y la gráfica 6. La

t	Exacta	Error
0.1	1.221402	-0,028535
0.2	1.491824	-0,034049
0.3	1.822118	-0,040782
0.4	2.225540	-0,049005
0.5	2.718281	-0,059050
0.6	3.320116	-0,071318
0.7	4.055199	-0,086302
0.8	4.953032	-0,104604
0.9	6.049647	-0,126958
1	7.389056	-0,154261

Table 5. Tabla de errores con método de integración numérica cuadratura Gauss-Legendre, con x_i originales

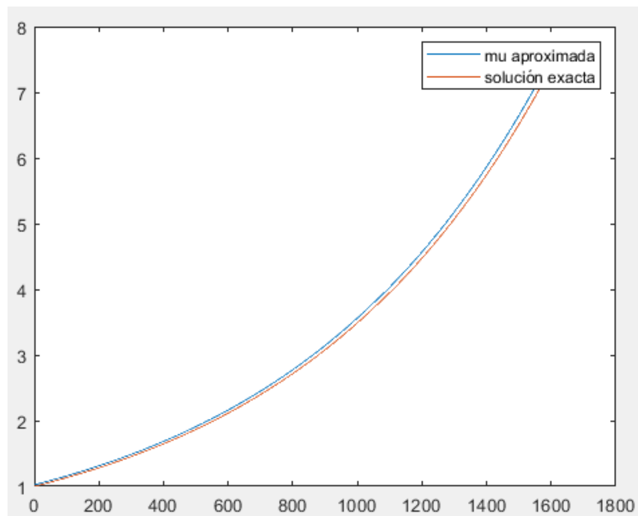
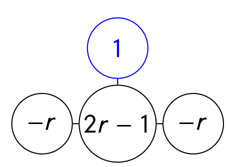


Figure 5. Aproximación de μ con método de integración cuadratura Gauss-Legendre con x_i originales

mejora no es considerable, por lo que este método no es el más recomendado para la solución de este sistema en este caso particular.



Cambio de función para evaluar integrales

A lo largo del código usado para resolver el problema, se usa la función de MATLAB $int(\cdot)$, pero se intenta reemplazar esta función por otra para obtener más precisión, la cual es $quad(\cdot)$, Al realizar este cambio para el calculo exacto de integrales definidas, no se obtuvo una gran diferencia entre funciones, ver tabla 7 y compararla con la tabla del código original 3; el mismo efecto se presenta con las graficas 7 y 1

Tomar el error como una constante

Es conveniente plantear un método para mejorar los errores, de esta manera se plantea manejar los mismos como una constante en la solución, por lo que la solución original 1 se le plantea como si fuera c veces la solución exacta, por lo cual ser realiza la siguiente relación: $\mu_{exacta} = c \cdot \mu_{aproximada}$

t	Exacta	Error
0.1	1.221402	-0.049106
0.2	1.491824	-0.059213
0.3	1.822118	-0.071558
0.4	2.225540	-0.086636
0.5	2.718281	-0.105052
0.6	3.320116	-0.127545
0.7	4.055199	-0.155019
0.8	4.953032	-0.188575
0.9	6.049647	-0.229561
1.0	7.389056	-0.279621

Table 6. Tabla de errores con método Forward

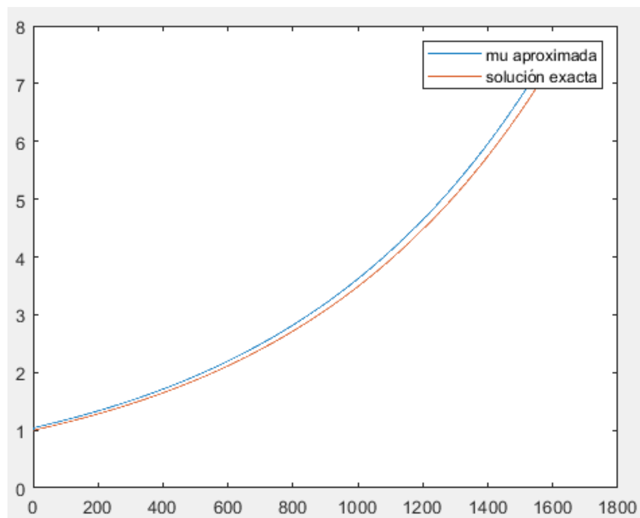


Figure 6. Aproximación de μ con método Forward

t	Exacta	Error
0.1	1.221402	0.00124151
0.2	1.491824	0.00232190
0.3	1.822118	0.00364163
0.4	2.225540	0.00525355
0.5	2.718281	0.00722235
0.6	3.320116	0.00962705
0.7	4.055199	0.01256416
0.8	4.953032	0.01615156
0.9	6.049647	0.02053321
1.0	7.389056	0.02588497

Table 7. Tabla de errores con función de integración $quad(\cdot)$

y de esta, se concluye que $c = 1.003138$. Se puede apreciar el cambio en el gráfico 8 , y los errores disminuyen 8. En la gráfica 9 se pueden ver los errores de $\mu(t)$, tomando la diferencia de errores como una constante vs los errores teóricos que presenta el documento. Cabe aclarar que ninguna gráfica anterior presenta errores tan cercanos a los teóricos(esto son, los presentados por el documento analizado). Esta es la mejor aproximación.

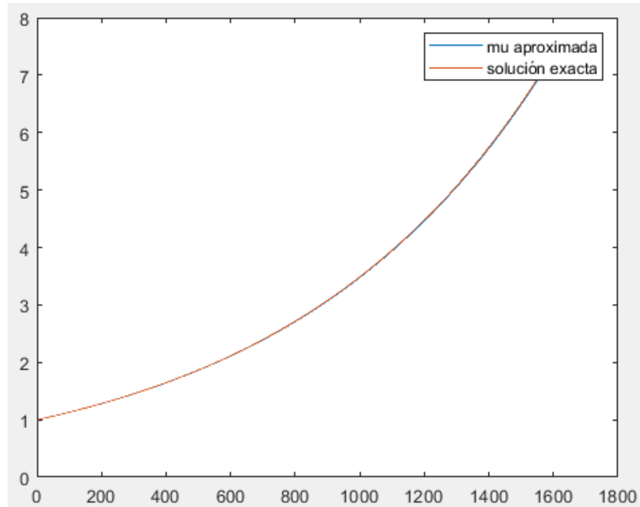


Figure 7. Aproximación de μ con función de integración $quad(\cdot)$

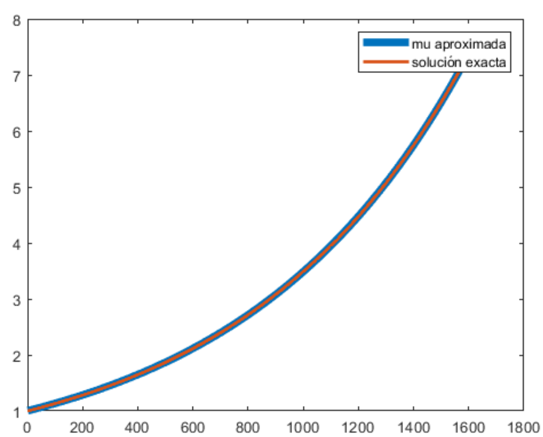


Figure 8. Aproximación de μ con errores como constante

t	Exacta	Error
0.1	1.221402	-0.002587
0.2	1.491824	-0.002352
0.3	1.822118	-0.002064
0.4	2.225540	-0.001713
0.5	2.718281	-0.001285
0.6	3.320116	-0.000761
0.7	4.055199	-0.000122
0.8	4.953032	0.000659
0.9	6.049647	0.001613
1.0	7.389056	0.002778

Table 8. Tabla de errores con errores como constante

5.3 Correcciones para $u_{i,j}^n$

Con el propósito de desarrollar un buen modelo para el problema, se cambian entonces las condiciones de frontera desarrolladas con método numérico (izquierda y derecha) por las condiciones

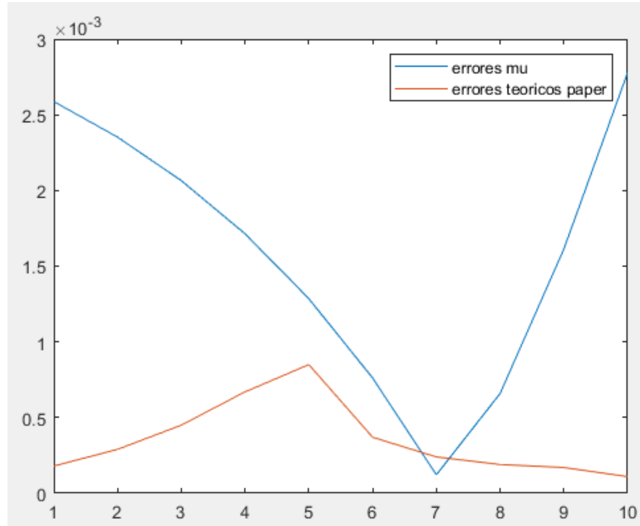


Figure 9. errores teóricos del documento y errores de dados por $\mu(t)$ en 8

exactas planteadas por la función que las modela, esto es, las funciones g_0 y g_1. De esta manera, las condiciones quedan de la siguiente manera:

$$u^n_{M,j} = g_1(jh, nk)$$

$$u^n_{0,j} = g_0(jh, nk)$$

En este sentido obtenemos los siguientes resultados: 9

Ut	Uex	Error
7.3891	7.3891	0
8.1661	8.1662	-4.018e-05
9.0249	9.025	-1.317e-04
9.9739	9.9742	-2.545e-04
11.023	11.023	-3.963e-04
12.182	12.182	-5.469e-04
13.463	13.464	-6.975e-04
14.879	14.88	-8.401e-04
16.444	16.445	-9.674e-04
18.173	18.174	-1.073e-03
20.084	20.086	-1.149e-03
22.197	22.198	-1.193e-03
24.531	24.533	-1.198e-03
27.111	27.113	-1.161e-03
29.963	29.964	-1.080e-03
33.114	33.115	-9.543e-04
36.597	36.598	-7.855e-04
40.447	40.447	-5.800e-04
44.701	44.701	-3.517e-04
49.402	49.402	-1.306e-04
54.598	54.598	0

Table 9. Tabla de errores para solución de u usando la función exacta en las condiciones de frontera

Y obtenemos así mismo la siguiente gráfica para $T = 1$ 10

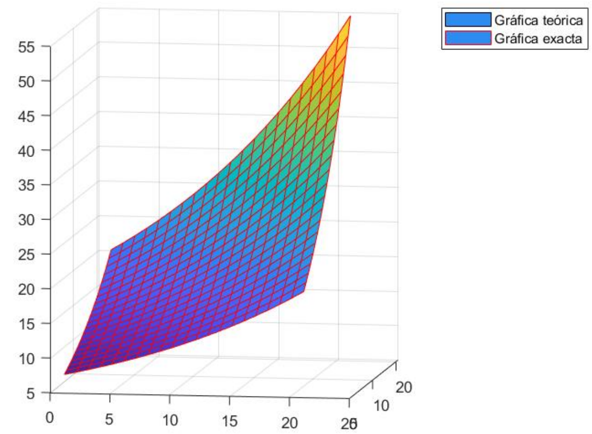


Figure 10. Aproximación de u con fronteras exactas

5.4 Aportes para $U_{i,j}^n$

En primera instancia, el método planteado en el documento citado se desarrolla a partir de un esquema explicito para discretizar el tiempo. En este sentido resulta pertinente evaluar cual sería el desarrollo de un esquema que se desarrolle de manera implícita para el tiempo. Veamos como sería su desarrollo.

Esquema euler implícito

Dado que se desarrollará el sistema implícito en el tiempo, el espacio seguirá usando un esquema de diferencias centradas.En este sentido el desarrollo sería el siguiente:

$$\frac{u_{i,j}^t - u_{i,j}^{t-1}}{k} = \frac{1}{h^2}(u_{i-1,j}^t - 2u_{i,j}^t + u_{i+1,j}^t + u_{i,j-1}^t - 2u_{i,j}^t + u_{i,j+1}^t)$$

$$u_{i,j}^t - u_{i,j}^{t-1} = s(u_{i-1,j}^t - 2u_{i,j}^t + u_{i+1,j}^t + u_{i,j-1}^t - 2u_{i,j}^t + u_{i,j+1}^t), \text{ con } s = \frac{k}{h^2}$$

$$u_{i,j}^t - s(u_{i-1,j}^t - 2u_{i,j}^t + u_{i+1,j}^t + u_{i,j-1}^t - 2u_{i,j}^t + u_{i,j+1}^t) = u_{i,j}^{t-1},$$

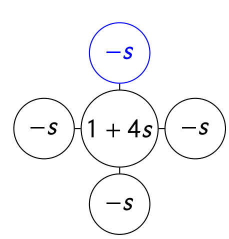
donde

$$i = 1, 2, \cdots, M-1$$

$$j = 1, 2, \cdots, M-1$$

$$t = 1, 2, \cdots, N$$

Tenemos un stencil de la siguiente manera para el sistema de ecuaciones planteado



De esta manera, podemos dividir el espacio de solución de la siguiente manera para generar nuestra matriz de solución más fácilmente. El que se presentará a continuación es un ejemplo donde $M\,x\,M = 20x20$.11 De esta manera, podemos definir la matriz A, con dimensiones $(M+1)^2x(M+1)^2$,

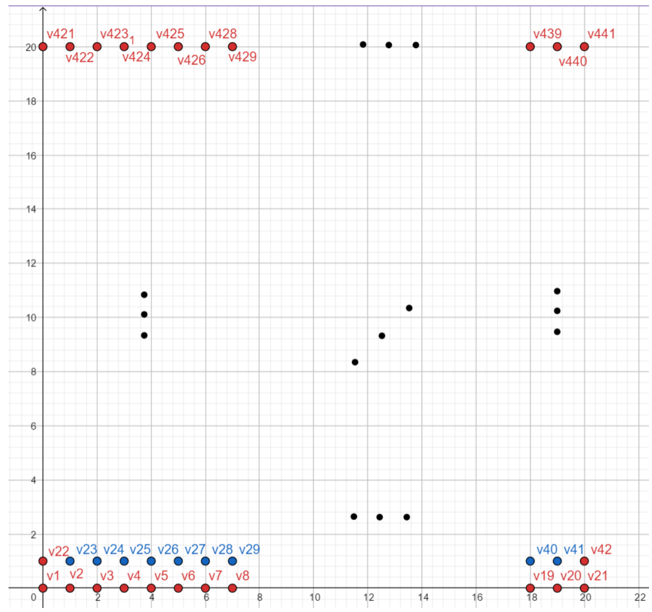


Figure 11. Esquema del grid para sistema implícito. Los puntos rojos representan las fronteras.

especificada para el ejemplo de la siguiente manera: .

$$
A = \begin{array}{cc}
 & \begin{array}{ccccccccccccccccc} v_1 & v_2 & v_3 & v_4 & \cdots & v_{22} & v_{23} & v_{24} & v_{25} & \cdots & v_{43} & v_{44} & v_{45} & \cdots & v_{421} & \cdots & v_{441} \end{array} \\
\begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ \vdots \\ v_{22} \\ v_{23} \\ v_{24} \\ v_{25} \\ \vdots \\ v_{43} \\ v_{44} \\ v_{45} \\ \vdots \\ v_{421} \\ \vdots \\ v_{441} \end{array} &
\begin{bmatrix}
1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & \cdots & 0 \\
0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & \cdots & 0 \\
0 & 0 & 1 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & \cdots & 0 \\
0 & 0 & 0 & 1 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & \cdots & 0 \\
 & & & & \ddots & & & & & \cdots & & & & \cdots & & \cdots & \\
0 & 0 & 0 & 0 & \cdots & 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & \cdots & 0 \\
0 & -s & 0 & 0 & \cdots & -s & 1+4s & -s & 0 & \cdots & -s & 0 & 0 & \cdots & 0 & \cdots & 0 \\
0 & 0 & -s & 0 & \cdots & 0 & -s & 1+4s & -s & \cdots & 0 & -s & 0 & \cdots & 0 & \cdots & 0 \\
0 & 0 & 0 & -s & \cdots & 0 & 0 & -s & 1+4s & \cdots & 0 & 0 & -s & \cdots & 0 & \cdots & 0 \\
 & & & & \ddots & & & & & \ddots & & & & \ddots & & \cdots & \\
0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & \cdots & 0 & -s & 0 & 0 & \cdots & -s & 1+4s & -s & \cdots & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & \cdots & 0 & 0 & -s & 0 & \cdots & 0 & -s & 1+4s & \cdots & 0 & \cdots & 0 \\
 & & & & \ddots & & & & & \ddots & & & & \ddots & & \cdots & \\
0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 & \cdots & 0 \\
 & & & & \ddots & & & & & \ddots & & & & \ddots & & \cdots & \\
0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & \cdots & 1
\end{bmatrix}
\end{array}
$$

El vector de u anexo a la matriz para el sistema es el siguiente:

$$\begin{bmatrix} u^t_{0,0} \\ u^t_{1,0} \\ u^t_{2,0} \\ \vdots \\ u^t_{0,1} \\ u^t_{1,1} \\ u^t_{2,1} \\ \vdots \\ u^t_{M-1,M} \\ u^t_{M,M} \end{bmatrix}$$

Para este caso particular obtenemos

$$\begin{bmatrix} u^t_{0,0} \\ u^t_{1,0} \\ u^t_{2,0} \\ \vdots \\ u^t_{0,1} \\ u^t_{1,1} \\ u^t_{2,1} \\ \vdots \\ u^t_{19,20} \\ u^t_{20,20} \end{bmatrix}$$

Cabe recalcar que los 1 presentes en la matriz A hacen referencia a las condiciones de frontera. Aquellos que se presentan al principio y final de la diagonal total se refieren a las condiciones de frontera inferior y superior respectivamente. Aquellos 1 que se presentan en el principio y final de la sub-matriz tri-diagonal son aquellos que se refieren a la frontera izquierda y derecha respectivamente. Con esto en mente, construyamos el vector **b**.

Frontera inferior

$$b_f = u^t_{f,0}, \\ f = 0, 1, 2, \cdots, M$$

Frontera superior

$$b_j = u^t_{j,M}, \\ j = M^2 - M, \cdots, M^2$$

Frontera izquierda. Aquí se ponen los subíndices que pertenecen a la parte izquierda de la malla.(Sin esquinas)

$$b_l = u^t_{0,l}, \\ l = (M+1) * 1, (M+1) * 2, \cdots, (M+1) * (M-1)$$

Frontera derecha. Para esta frontera se usarán los índices de la frontera izquierda.

$$b_r = u^t_{M,r}, \quad r = l + (M)$$

Valores de la sub-matriz. Igualados a su correspondiente valor en el tiempo anterior. Sus subíndices son todos los de la sub-matriz sin la frontera respectiva.

$$
\begin{gathered}
b_g = u_{i,j}^{t-1}\ , \\
i = 1, 2, 3, \cdots, M-1 \\
j = 1, 2, 3, \cdots, M-1, \\
g = \{0, 1, 2, \cdots, M^2\} - \{f\} - \{j\} - \{l\} - \{r\}
\end{gathered}
$$

Es claro que todos estos valores igualados son conocidos. Ahora resolviendo este sistema para cada uno de los pasos temporales planteados, obtenemos los siguientes resultados. Gráfica12 y tabla10

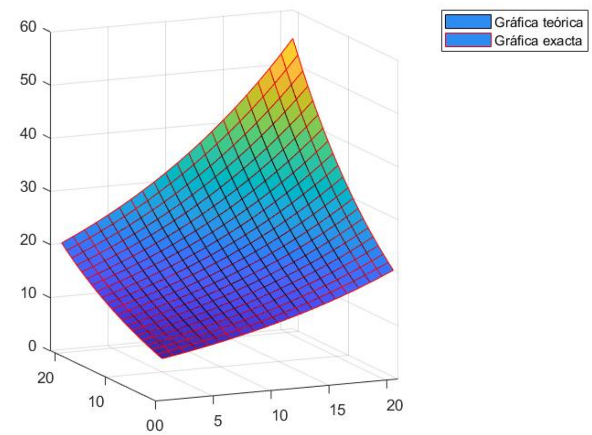


Figure 12. Aproximación de u por esquema implícito

Ut	Uex	Error
7.3891	7.3891	0
8.1663	8.1662	8.029e-05
9.0253	9.025	2.631e-04
9.9747	9.9742	5.087e-04
11.024	11.023	7.920e-04
12.184	12.182	1.093e-03
13.465	13.464	1.394e-03
14.881	14.88	1.679e-03
16.447	16.445	1.933e-03
18.176	18.174	2.143e-03
20.088	20.086	2.297e-03
22.2	22.198	2.384e-03
24.535	24.533	2.394e-03
27.115	27.113	2.320e-03
29.966	29.964	2.159e-03
33.117	33.115	1.907e-03
36.6	36.598	1.570e-03
40.448	40.447	1.159e-03
44.702	44.701	7.028e-04
49.403	49.402	2.609e-04
54.598	54.598	0

Table 10. Tabla de errores para solución de u usando esquema implícito

5.5 Analisis Von Neumann $U_{i,j}^{n}$

Para el análisis Von Neumann respectivo al sistema backward planteado, se formula un sistema vectorizado dado la aproximación de Laplace otorgada por un problema de calor en dos dimensiones. De esta manera obtenemos lo siguiente.

$$E_{l,j}^{m} = \xi^{m} e^{i(\theta\alpha)\cdot\binom{l}{j}h}$$

[3] [4] De esta manera, es posible expandirlo así:

$$E_{l,j}^{m} = \xi^{m} e^{i(\theta)lh} e^{i(\alpha)jh}$$

Y aplicándolo a nuestro problema

$$(1+4s)\xi^{m} e^{i(\theta)lh} e^{i(\alpha)jh} - s(\xi^{m} e^{i(\theta)(l-1)h} e^{i(\alpha)jh} + \xi^{m} e^{i(\theta)(l+1)h} e^{i(\alpha)jh} + \xi^{m} e^{i(\theta)lh} e^{i(\alpha)(j-1)h} + \cdots$$
$$\cdots \xi^{m} e^{i(\theta)lh} e^{i(\alpha)(j+1)h}) = \xi^{m-1} e^{i(\theta)lh} e^{i(\alpha)jh}$$

con $s = \frac{k}{h^2}$

$$(1+4s)\xi^{m} e^{i(\theta)lh} e^{i(\alpha)jh} - s\xi^{m}(e^{i(\theta)(l)h} e^{i(\alpha)jh} e^{-i(\theta)h} + e^{i(\theta)lh} e^{i(\alpha)jh} e^{i(\theta)h} + e^{i(\theta)lh} e^{i(\alpha)jh} e^{-i(\alpha)h} + \cdots$$
$$\cdots + e^{i(\theta)lh} e^{i(\alpha)jh} e^{i(\alpha)h}) = \xi^{m-1} e^{i(\theta)lh} e^{i(\alpha)jh}$$
$$(1+4s)\xi^{m} - s\xi^{m}(e^{-i(\theta)h} + e^{i(\theta)h} + e^{-i(\alpha)h} + e^{i(\alpha)h}) = \xi^{m-1}$$
$$\xi[(1+4s) - s(\cos h\theta + \cos h\alpha)] = 1$$
$$\xi = \frac{1}{[(1+4s) - s(\cos h\theta + \cos h\alpha)]}$$
$$\xi = \frac{1}{[1 + s(4 - (\cos h\theta + \cos h\alpha))]}$$
$$|\xi| < 1$$

De esta manera, es directo evaluar que el sistema es incondicionalmente estable.

6 Análisis del error

A lo largo del trabajo presentado, los errores son medidos de la siguiente manera:

$$Error(i) = soluci\acute{o}nExacta(i) - Soluci\acute{o}nAproximada(i) \tag{13}$$

Donde $i = 1, 2, \ldots, M+1$

6.1 Análisis de errores con el programa original

A pesar de implementar los esquemas originales del paper [2] sin ninguna modificación, no se logran los mismos resultados planteados en el mismo [2], se piensa que es debido a algunas funciones usadas al evaluar integrales y derivadas que podrían afectar los resultados numérico,s o por algunos detalles en la escritura de las estructuras, que al hacer pequeños cambios en los índices, generan grandes repercusiones en las soluciones; por esto, se implementan variaciones en los esquemas y en las formas de integrar, para certificar resultados y ver variaciones.

A pesar de hacer los cambios correspondientes en los métodos de integración y evaluación, no se alcanzan los resultados planteados en el paper [2].

7 Conclusiones

El contenido desarrollado en este documento refleja un buen análisis del paper [2], donde no solo se hace una intromisión en los métodos que allí se emplean, sino que se permiten corregir ciertos errores planteados y adicionalmente se ofrece una alternativa de solución a los mismos, en adición al aporte mediante otro sistema de solución para el sistema central del problema y su respectivo análisis, lo que permite, en efecto, el tema allí planteado sea no solo más afable de entender, sino que permita dar una contribución a la temática.

8 Funciones dadas para la solución del problema

$$f(x) = e^{(x+y)} \tag{14}$$

$$g_0(y,t) = e^{(y+2t)} \tag{15}$$

$$g_1(y,t) = e^{(1+y+2t)} \tag{16}$$

$$h_0(x,t) = e^{x} \tag{17}$$

$$h_1(x,t) = e^{(1+x+2t)} \tag{18}$$

$$m(t) = e^{(2t)}(e-1)^2 \tag{19}$$

$$\mu(t) = e^{(2t)} \tag{20}$$

9 Anexos

En esta sección se plasman algunos códigos usados en MATLAB para la solución de los problemas.

Código de MATLAB

$\mu(t)$ según paper

Este primer código presenta el código original, es decir, el paper sin modificaciones.

```
MW(1,:)=fl(X);%condiciones iniciales

%MATRIZ A2
 I=speye(M-1,M-1);
 E=sparse(2:M-1,1:M-2,1,M-1,M-1);
 IC=sparse(1:M-2,2:M-1,1,M-1,M-1);
 A=-r*E-r*IC+(1+2*r)*I;
%Condiciones frontera
 A2(2:20,2:20)=A;
 A2(21,21)=3;
 A2(1,1)= -1;
 A2(1,21)=1;
 A2(2,1)=-r;
 A2(20,21)=-r;
 A2(21,19)=1;
 A2(21,20)=-4;

%Metodo original de paper
 for n=2:N+1
    %vector de vb
     vb=zeros(M+1,1);
     vb(1,1)=qt(T(n));
```

```
        for i=1:M-1
            vb(i+1,1)=MW(n-1,i+1)+k*phi(X(i+1),T(n));%paper
        end
        vb(M+1,1)=2*h*gm(T(n));
        %vector incognitas w
        wp=A2\vb;
        wp=wp';
        MW(n,:)=wp;
end

%metodo trapezoidal
for n=1:N+1
    vpw=MW(n,:);
    V(n)=gm(T(n))-(h/2)*(vpw(1)+vpw(M)+2*sum(vpw(2:end-2)));%paper
end

%Evaluar integrales con int()
for i=1:N+1
    mu(i)=(V(i))/(phi0(0,T(i)));
end

figure(1)
plot(mu,'LineWidth',2)
hold on
plot(exp(2*T),'LineWidth',2)
legend('mu aproximada','soluci n exacta')
```

Correcciones para $\mu(t)$

Se presenta la corrección del método trapezoidal implementado

```
%metodo trapezoidal corregido
for n=1:N+1
    vpw=MW(n,:);
    V(n)=gm(T(n))-(h/2)*(vpw(1)+vpw(M+1)+2*sum(vpw(2:end-1)));
end
```

Aportes para $\mu(t)$

Se muestra los códigos implementados para las diferentes modificaciones, como método de cuadratura Gauss-Legendre, esquema explicito, función $quad(\cdot)$ y tomar el error como constante.

```
%Metodo explicito
 for n=2:N+1
     %vector de b
     bf=zeros(M+1,1);

     bf(1,1)=qt(T(n));
     bf(M+1,1)=2*h*gm(T(n));
     for i=1:M-1
         bf(i+1,1)=r*MW(n,i)-(2*r-1)*MW(n,i+1)+r*MW(n,i+2)+k*phi(X(i+1),T(n))
```

```
        end

        %vector incognitas w
        MW(n,:)=bf';
    end


%Integraci n cuadratura Gauss-Legendre
for n=1:N+1
    vpw=MW(n,:);
    V(n)=gm(T(n))-sum(vpw.*w);
end

%Evaluar integrales con quad()
for i=1:N+1
    mu2(i)=(V(i))/quad(@(x)h0(x,T(i)),0,1);
end



figure(1)
plot(mu,'LineWidth',2)
hold on
plot(exp(2*T),'LineWidth',2)
legend('mu aproximada','soluci n exacta')


%Soluci n exacta
for i=0:N
    solex(i+1)=exp(2*T(i+1));
end
div2=solex/mu;%Implementada
mucon=div2.*mu;
plot(mucon)
hold on
plot(exp(2*T))
```

$U(x,y,t)$ según paper

Se realiza la solución siguiento el paper con exactitud.

```
%Generamos matriz
Ut=zeros(length(X),length(Y),length(T));
%Generamos condiciones iniciales
for i=0:M
    for j=0:M
        Ut(i+1,j+1,1)=f(X(i+1),Y(j+1));
    end
end

```

```
%Calculamos la nueva matriz
for t=2:N+1
    for i=0:M
        for j=0:M
            if j==0
             %Condici n para frontera inferior
                Ut(i+1,0+1,t)=h2(X(i+1),T(t));
            elseif j==M
              %Condici n para frontera superior
                 Ut(i+1,M+1,t)=h1(X(i+1),T(t));
            elseif i==0
                %Condici n para frontera izquierda(sin esquinas)
                Ut(0+1,j+1,t)=(1-2*s)*Ut(0+1,j+1,t-1)+2*s ...
                *Ut(2,j+1,t-1)+2*h*s*g0(Y(j+1),T(t-1));%original
            elseif i==M
                %Condici n para frontera derecha (sin esquinas)
                Ut(M+1,j+1,t)=(1-2*s)*Ut(M+1,j+1,t-1)+2*s ...
                *Ut(M,j+1,t-1)+2*s*h*g1(Y(j+1),T(t-1));
            else
                Ut(i+1,j+1,t)=s*(Ut(i+2,j+1,t-1)+Ut(i+1,j+2,t-1)+ ...
                Ut(i,j+1,t-1)+Ut(i+1,j,t-1))+(1-4*s)*Ut(i+1,j+1,t-1);
            end

        end
    end
end

%Soluci n exacta
for t=1:N+1
    for i=1:M+1
        for j=1:M+1
            Uext(i,j,t)=exp(X(i)+Y(j)+2*T(t));
        end
    end
end

%Errores
Ta_rk=table(diag(Ut(:,:,1601)),diag(Uext(:,:,1601)),diag(Ut(:,:,1601)) ...
-diag(Uext(:,:,1601)),'VariableNames',{'Ut','Uex','Error'});

%Gr fica
surf(Ut(:,:,1601));
hold on
surf(Uext(:,:,1601),'EdgeColor','r')
xlabel('x')
ylabel('y')
zlabel('z')
```

Correcciones para $U(x,y,t)$

Se cambia la forma de asignar las condiciones de frontera por la función exacta.

```
%Calculamos la nueva matriz
for t=2:N+1%Paso temporal
    for i=0:M
        for j=0:M
            if j==0
             %Condici n para frontera inferior
               Ut(i+1,0+1,t)=h2(X(i+1),T(t));
            elseif j==M
              %Condici n para frontera superior
                Ut(i+1,M+1,t)=h1(X(i+1),T(t));
            elseif i==0
              %Condici n para frontera izquierda (sin esquinas)
               Ut(0+1,j+1,t)=g0(Y(j+1),T(t));%correcci n
            elseif i==M
               %Condici n para frontera derecha (sin esquinas)
                Ut(M+1,j+1,t)=g1(Y(j+1),T(t));%correcci n
            else
               Ut(i+1,j+1,t)=s*(Ut(i+2,j+1,t-1)+Ut(i+1,j+2,t-1) ...
               +Ut(i,j+1,t-1)+Ut(i+1,j,t-1))+(1-4*s)*Ut(i+1,j+1,t-1);
            end

        end
    end
end
```

Aporte para $U(x,y,t)$

El esquema propuesto e implementado, un esquema implícito.

```
%Generamos matriz
Ut=zeros(length(X),length(Y),length(T));
%Generamos condiciones iniciales
for i=0:M
    for j=0:M
        Ut(i+1,j+1,1)=f(X(i+1),Y(j+1));
    end
end


for t=2:N+1
    %Construimos la matriz A
    A=zeros((M+1)^2,(M+1)^2);
    %Llenamos la matriz con las condiciones de frontera superior e inferior
    %Inferior
    for i=0:M
        for j=0:M
            if(i==j)
                A(i+1,j+1)=1;
```

```
            end
        end
    end
    %Superior
    for i=(M+1)^2-20:(M+1)^2
        for j=(M+1)^2-20:(M+1)^2
            if(i==j)
                A(i,j)=1;
            end
        end
    end
    %Frontera Izquiera y derecha
    %Vectos de subindices (Sin esquinas)
    sub_izq=M+2:M+1:((M+1)^2)-(M+1);
    sub_der=sub_izq+M;
    for i=sub_izq
        A(i,i)=1;
    end
    for i=sub_der
        A(i,i)=1;
    end
    %Llenamos la submatriz de coeficientes
    for i=1:(M+1)^2
        if A(i,i)~=1
            A(i,i)=1+4*s;
            A(i,i+1)=-s;
            A(i,i-1)=-s;
            A(i,i+(M+1))=-s;
            A(i,i-(M+1))=-s;
        end
    end
    %Construimos la matriz b
    b=NaN((M+1)^2,1);
    %Limites superiores e inferior
    for i=0:M%Sup
        b(i+1,1)=h2(X(i+1),T(t));
    end
    index=0;
    for i=(M+1)^2-20:(M+1)^2%Inf
        b(i,1)=h1(X(index+1),T(t));
        index=index+1;
    end
    %Limites izquierdo y derecho
    aux=1;
    for i=sub_izq %Izquierdo
        b(i,1)=g0(Y(aux+1),T(t));
        aux=aux+1;
    end
    aux=1;
```

```
    for i=sub_der %Derecho
        b(i,1)=g1(Y(aux+1),T(t));
        aux=aux+1;
    end
    %Terminamos de rellenar el vectr b con los valores de los tiempos
    %anteriores correspondientes
    pos_ver=zeros(M+1,M+1)%Matriz que me ubica los vertices en la posici n r
    acum=1;
    for i=0:M
        for j=0:M
            pos_ver(i+1,j+1)=acum;
            acum=acum+1;
        end
    end
    for i=1:length(b)
        if isnan(b(i,1))
            [l h]=find(i==pos_ver);
            b(i,1)=Ut(l,h,t-1);
        end
    end
    %Resolvemos el sistema
    Wact=A\b;
    %Ubicamos soluciones en su posici n correspondiente
    for i=1:length(Wact)
        [l h]=find(i==pos_ver);
        Ut(l,h,t)=Wact(i);
    end
end
```

10 Referencias

References

[1] Greg von Winckel, Legendre-Gauss Quadrature Weights and Nodes, https://www.mathworks.com/matlabcentral/fileexchange/4540-legendre-gauss-quadrature-weights-and-node.

[2] Mehdi Dehghan, 2000,A finite diference method for a non-local boundary value problem for two-dimensional heat equation

[3] 18.336 Numerical Methods for Partial Differential Equations, lectura 14 "Von Neumann Stability Analysis", Spring 2009, MIT OpenCourseWare http://ocw.mit.edu

[4] Mehwish Naz Rajput1, Asif Ali Shaikh2and Shakeel Ahmed KambohAsian, Research Journal of Mathematics, 2020, Article no.ARJOM.54909 "Computational Analysis of the Stability of 2D Heat Equation on Elliptical Domain Using Finite Difference Method "

[5] Randall J. LeVeque, 2007, Finite Difference Methodsfor Ordinary and PartialDifferential Equations