ECUACIONES EN DERIVADAS PARCIALES PARABÓLICAS

LABORATORIO 3

Universidad Internacional de la Rioja (UNIR)

Cuatrimestre II

Máster en Ingeniería Matemática y Computación

Métodos Numéricos Aplicados 2

Autor: Javier Blanco Álvarez

RESUMEN

En la presente actividad se plantea un problema de EDP específica con condiciones de contorno mixtas definidas, con el que se busca la aproximación de su solución aplicando tres métodos numéricos: método explícito, método implícito y método Crank-Nicholson, implementados en MATLAB. El método explícito, que es condicionalmente estable, utiliza diferencias finitas para convertir la EDP en ecuaciones algebraicas, mientras que el método implícito, incondicionalmente estable, maneja mayores pasos de tiempo, pero requiere resolver sistemas de ecuaciones lineales. El método de Crank-Nicholson combina ventajas de ambos métodos, promediando valores para mejorar precisión y estabilidad. El análisis comparativo gráfico muestra que los métodos explícito e implícito producen resultados cercanos, aunque el explícito necesita una cuidadosa selección de pasos de tiempo para evitar inestabilidades. Crank-Nicholson destaca por su equilibrio entre estabilidad y precisión. En términos numéricos, se presentan errores absolutos en distintos nodos espaciales y tiempos, observándose que los errores aumentan con el tiempo y que los métodos numéricos pueden volverse menos precisos con mayores coordenadas temporales. Los errores más elevados se encuentran en nodos intermedios del intervalo espacial, indicando posibles inestabilidades debido a la discretización empleada. Cada método tiene ventajas y desventajas específicas, siendo Crank-Nicholson el más balanceado para la resolución de EDPs parabólicas.

Palabras Clave: Ecuaciones diferenciales parciales parabólicas, Método explícito, Método implícito, Crank-Nicholson, MATLAB.

1. Planteamiento del problema

Consideremos la siguiente ecuación diferencial parcial de segundo orden:

$$u_t = u_{xx} + t^2 u + xcos(xt)\ x \in [0,1];\ t > 0$$

$$u(x,0) = 0, x \in [0,1] \qquad u_x(0,t) = t;\ u(1,t) = sin(t)$$

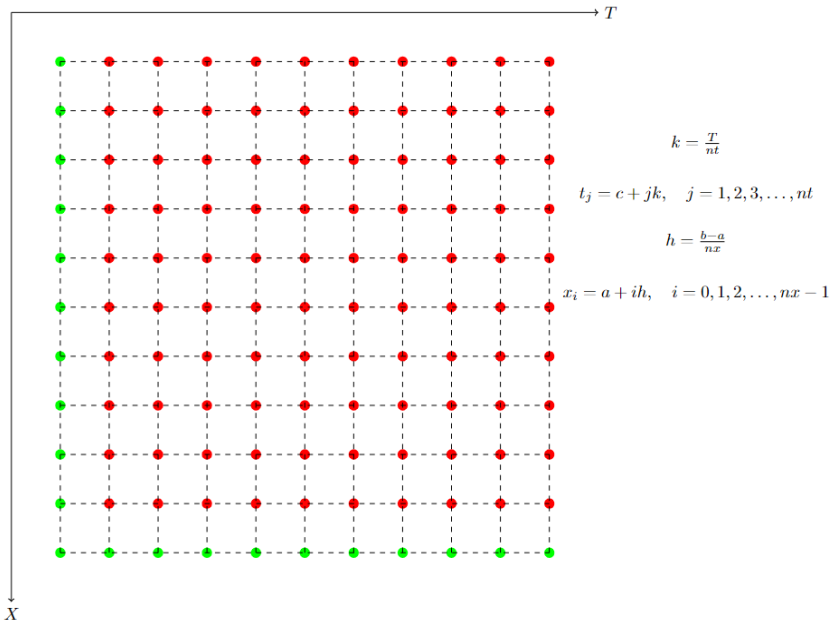

Figura 1. . Mallado de puntos sobre el problema descrito. Los puntos rojos representan la zona donde no conocemos la solución. Elaboración propia

Dadas las condiciones de contorno y la condición inicial podemos dibujar las zonas del mallado donde tenemos incógnitas:

Así, los puntos rojos representan las incógnitas del problema que vamos a resolver en esta práctica. La implementación de estos esquemas se llevará a cabo utilizando el entorno de desarrollo **MATLAB, versión 2021b.**

2. Método explícito

Para el método explicito discretizaremos la EDP aplicando diferencias finitas progresivas en u_t y centradas en u_{xx} para $i = 0,1,2\ldots nx-1$ y $j = 1,2,3\ldots nt$. Este método utiliza una aproximación hacia adelante en el tiempo y una aproximación centrada en el espacio. Esto convierte la EDP continua en ecuaciones algebraicas que se pueden resolver iterativamente (Lam, 1992). El método explícito es condicionalmente estable, con lo cual, para asegurar la estabilidad, el parámetro

λ debe satisfacer (Polyanin, 2002):

$$\lambda \leq \frac{1}{2}$$

2.1. Planteamiento.

La discretización del problema queda de la siguiente forma:

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

Multiplicamos ambos lados de la igualdad por k, establecemos $\lambda = \frac{k}{h^2}$ y despejamos $u_{(i,j+1)}$ quedando la siguiente expresión general:

$$\boldsymbol{u_{(i,j+1)} = (1 - 2\lambda + kt_j^2)u_{(i,j)} + \lambda(u_{(i+1,j)} + u_{(i-1,j)}) + kx_i cos(x_i t_j)}$$

Seguidamente particularizamos para i=0, i=nx-1 para i=1 con una posición arbitraria de j

- ***Para i = 0***

$$u_{(0,j+1)} = (1 - 2\lambda + kt_j^2)u_{(0,j)} + \lambda(u_{(1,j)} + u_{(-1,j)}) + kx_1 cos(x_1 t_j)$$

El término $u_{(-1,j)}$ se sale del dominio de trabajo por lo que optamos por usar la primera condición de contorno (CC1)

$$u_x(0,t) = t$$

Discretizando con diferencias centradas y despejan $u_{(-1,j)}$ obtenemos:

$$u_{(-1,j)} = u_{(1,j)} - 2ht_j$$

Reemplazando en la formula general se obtiene

$$u_{(0,j+1)} = (1 - 2\lambda + kt_j^2)u_{(0,j)} + \lambda(u_{(1,j)} + u_{(1,j)} - 2ht_j) + kx_0 cos(x_0 t_j)$$

Operando y reordenando se obtiene:

$$u_{(0,j+1)} = (1 - 2\lambda + kt_j^2)u_{(0,j)} + \lambda(2u_{(1,j)} - 2ht_j) + kx_1 cos(x_1 t_j)$$

- ***Para i = 1***

$$u_{(1,j+1)} = (1 - 2\lambda + kt_j^2)u_{(1,j)} + \lambda(u_{(2,j)} + u_{(0,j)}) + kx_1 cos(x_1 t_j)$$

- ***Para i = nx-1***

$$u_{(nx-1,j+1)} = CC2 = sin(t_{j+1})$$

2.2. Resultados

El método explícito recibe los parámetros mostrados en la tabla 1.

Tabla 1. Parámetros de entrada para el método del disparo con secante.

Parámetros	Valor
α (Alpha)	1
Intervalo inferior (a)	0
Intervalo superior (b)	1
Tiempo máximo (Tmax)	0.5
Tamaño de paso temporal (h)	0.1
Tamaño de paso espacial (k)	0.0005
Criterio de estabilidad (λ)	0.05

Como $\lambda \leq {}^{1}/_{2}$ entonces decimos que el método explícito es computacionalmente estable para este problema en particular. Con esto se logra mantener una convergencia $O(k + h^2)$. En la tabla 2 se muestran los resultados de solución aproximada de la EDP parabólica en distintas secciones temporales (t).

Tabla 2. Valores aproximados de unidades espaciales x y U(x,t) para distintas secciones temporales (t=1/8, ¼, ½, ¾ de Tmax) utilizando el método explicito.

Nodos espaciales (X)	U(x, Tmax/8)	U(x, Tmax/4)	U(x, Tmax/2)	U(x, 3Tmax/4)	U(x, Tmax)
0.0	-0.00013375	-0.00015316	-0.00013017	-0.00011923	-0.00014334
0.1	0.0060622	0.012297	0.024821	0.037331	0.049855
0.2	0.012251	0.024746	0.049758	0.074728	0.099726
0.3	0.018435	0.037193	0.074667	0.11202	0.14935
0.4	0.02462	0.049637	0.099532	0.14916	0.19859
0.5	0.03081	0.062078	0.12434	0.18609	0.24734
0.6	0.037012	0.074515	0.14907	0.22276	0.29547
0.7	0.043229	0.086946	0.17371	0.25912	0.34286
0.8	0.049462	0.09937	0.19824	0.29511	0.38939
0.9	0.055708	0.11178	0.22265	0.33069	0.43495
1.0	0.06196	0.12418	0.24692	0.36581	0.47943

3. Método Implícito

Se trata de un método incondicionalmente estable con orden convergencia $O(k + h^2)$. Para discretizar aplicaremos diferencias finitas regresivas en u_t y centradas en u_{xx}. La aplicación de este método implica la resolución del sistema lineal (Polyanin, 2002):

$$Au^{(j)} = u^{(j-1)} + d_j$$

3.1. Planteamiento

Con la discretización de la EDP siguiendo el método implícito se obtiene:

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

Tras seguir las mismas operaciones del método anterior (estableciendo expresión de λ) y agrupando incógnitas en el lado izquierdo de la igualdad y términos conocidos en el lado derecho, se obtiene la siguiente expresión general

$$\mathbf{-\lambda u_{(i+1,j)} + (1 + 2\lambda - k t_j^2) u_{(i,j)} - \lambda u_{(i-1,j)} = u_{(i,j-1)} + k x_i cos(x_i t_j)}$$

Particularizando se tiene:

- ***Para i = 0***

$$-\lambda u_{(1,j)} + (1 + 2\lambda - k t_j^2) u_{(0,j)} - \lambda u_{(-1,j)} = u_{(0,j-1)} + k x_0 cos(x_0 t_j)$$

De la CC1 se sabe se obtiene la expresión de $u_{(-1,j)}$

$$u_{(-1,j)} = u_{(1,j)} - 2h t_j$$

Así:

$$-\lambda u_{(1,j)} + (1 + 2\lambda - k t_j^2) u_{(0,j)} - \lambda u_{(1,j)} + 2\lambda h t_j = u_{(0,j-1)} + k x_0 cos(x_0 t_j)$$

Reordenando

$$\mathbf{-\lambda u_{(1,j)} + (1 + 2\lambda - k t_j^2) u_{(0,j)} - \lambda u_{(1,j)} = u_{(0,j-1)} - 2\lambda h t_j + k x_0 cos(x_0 t_j)}$$

- ***Para i = 1***

$$-\lambda u_{(2,j)} + (1 + 2\lambda - k t_j^2) u_{(1,j)} - \lambda u_{(0,j)} = u_{(1,j-1)} + k x_1 cos(x_1 t_j)$$

- ***Para i = nx-1***

$$-\lambda u_{(nx,j)} + (1 + 2\lambda - k t_j^2) u_{(nx-1,j)} - \lambda u_{(nx-2,j)} = u_{(nx-1,j-1)} + k x_{nx-1} cos(x_{nx-1} t_j)$$

Con la CC2 se sabe que:

$$u_{(nx,j)} = sin(t_j)$$

Con los términos conocidos al lado derecho de la igualdad se tiene:

$$\mathbf{-\lambda u_{(nx-2,j)} + (1 + 2\lambda - k t_j^2) u_{(nx-1,j)} = u_{(nx-1,j-1)} + \lambda sin(t_j) + k x_{nx-1} cos(x_{nx-1} t_j)}$$

Las matrices $A, u^{(j)}, u^{(j-1)} y\ d_j$ para este EDP quedan de la siguiente forma:

$$A = \begin{bmatrix} 1+2\lambda-kt_j^2 & -2\lambda & \dots & 0 & 0 \\ -\lambda & 1+2\lambda-kt_j^2 & -\lambda & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & 1+2\lambda-kt_j^2 & -\lambda \\ 0 & 0 & \dots & -\lambda & 1+2\lambda-kt_j^2 \end{bmatrix}$$

$$u^{(j)} = \begin{bmatrix} u_{(0,j)} \\ u_{(1,j)} \\ \vdots \\ \vdots \\ u_{(nx-1,j)} \end{bmatrix}$$

$$d_j = \begin{bmatrix} -2\lambda \mathrm{ht_j} + \mathrm{kx_0}\cos(\mathrm{x_0 t_j}) \\ \mathrm{kx_1}\cos(\mathrm{x_1 t_j}) \\ \vdots \\ \mathrm{kx_{nx-2}}\cos(\mathrm{x_{nx-2} t_j}) \\ \lambda\sin(\mathrm{t_j}) + \mathrm{kx_{nx-1}}\cos(\mathrm{x_{nx-1} t_j}) \end{bmatrix}$$

$$u^{(j-1)} = \begin{bmatrix} u_{(0,j-1)} \\ u_{(1,j-1)} \\ \vdots \\ \vdots \\ u_{(nx-1,j-1)} \end{bmatrix}$$

3.2. Resultados

Los parámetros de entrada para este método fueron los mismo utilizados en el método explícito (ver tabla 1). Al tratarse de un método incondicionalmente estable el tamaño de paso espacial. En la tabla 3 se muestran los resultados de la solución aproximada de la EDP parabólica de esta actividad.

Tabla 3. Valores aproximados de unidades espaciales x y U(x,t) para distintas secciones temporales (t=1/8, ¼, ½, ¾ de Tmax) utilizando el método implícito

Nodos espaciales (X)	U(x, Tmax/8)	U(x, Tmax/4)	U(x, Tmax/2)	U(x, 3Tmax/4)	U(x, Tmax)
0.0	-5,58E-04	-6,37E-03	-7,01E-02	-2,85E-01	-7,76E-01
0.1	0.0061999	0.012449	0.024943	0.03742	0.04992
0.2	0.0124	0.024897	0.049876	0.074814	0.099788
0.3	0.018599	0.037341	0.074778	0.1121	0.1494
0.4	0.024797	0.049779	0.099632	0.14923	0.19864
0.5	0.030995	0.062209	0.12442	0.18615	0.24738
0.6	0.037191	0.07463	0.14914	0.22281	0.29551
0.7	0.043386	0.087039	0.17376	0.25915	0.34289
0.8	0.04958	0.099435	0.19828	0.29513	0.38941
0.9	0.055771	0.11182	0.22267	0.3307	0.43496
1.0	0.06196	0.12418	0.24692	0.36581	0.47943

4. Método de Crank-Nicholson

Este método se basa en resolver el siguiente sistema matricial (García & Armando, 2007):

$$Au^{(j+1)} = Bu^{(j)} + d_j$$

4.1. Planteamiento

Aplicamos diferencias progresivas en u_t para t_j y regresivas para t_{j+1}. Aplicaremos diferencias centradas para u_{xx} y luego haremos una media aritmética.

- **Para t_j**:

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

- **Para $t_j + 1$**:

$$\frac{u_{i,j+1} - u_{i,j}}{k} = \frac{u_{i+1,j+1} - 2u_{i,j+1} + u_{i-1,j+1}}{h^2} + t_j^2 u_{i,j+1} + x_i cos(x_i t_{j+1})$$

La media aritmética entre ambas expresiones resulta en:

$$\frac{u_{i,j+1} - u_{i,j}}{k} = \frac{1}{2}[\frac{u_{i+1,j+1} - 2u_{i,j+1} + u_{i-1,j+1}}{h^2} + \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + t_j^2 u_{i,j} + t_j^2 u_{i,j+1} + x_i cos(x_i t_j) + x_i cos(x_i t_{j+1})]$$

Para simplificar expresiones diremos que:

$$D = x_i cos(x_i t_j) + x_i cos(x_i t_{j+1})$$

Por tanto:

$$\frac{u_{i,j+1} - u_{i,j}}{k} = \frac{1}{2}[\frac{u_{i+1,j+1} - 2u_{i,j+1} + u_{i-1,j+1}}{h^2} + \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + t_j^2 u_{i,j} + t_j^2 u_{i,j+1} + D]$$

Decimos también que:

$$\lambda = \frac{k}{h^2}$$

Posicionando los términos de $j + 1$ al lado izquierdo de la igualdad, agrupando y aplicando factor común nos queda:

$$(\mathbf{1} + \boldsymbol{\lambda} - \frac{\boldsymbol{k}}{\mathbf{2}}\boldsymbol{t_j^2})\boldsymbol{u_{i,j+1}} - \frac{\boldsymbol{\lambda}}{\mathbf{2}}(\boldsymbol{u_{i+1,j+1}} + \boldsymbol{u_{i-1,j+1}}) = (\mathbf{1} - \boldsymbol{\lambda} + \frac{\boldsymbol{k}}{\mathbf{2}}\boldsymbol{t_j^2})\boldsymbol{u_{i,j}} + \frac{\boldsymbol{\lambda}}{\mathbf{2}}(\boldsymbol{u_{i+1,j}} + \boldsymbol{u_{i-1,j}}) + \frac{\boldsymbol{k}}{\mathbf{2}}\boldsymbol{D}$$

- ***Para i = 0***

$$-\frac{\lambda}{2}u_{1,j+1} + (1+\lambda-\frac{k}{2}t_j^2)u_{0,j+1} - \frac{\lambda}{2}u_{-1,j+1} = \frac{\lambda}{2}u_{1,j} + (1-\lambda+\frac{k}{2}t_j^2)u_{0,j} + \frac{\lambda}{2}u_{-1,j} + \frac{k}{2}D$$

Términos $u_{-1,j}$ y $u_{-1,j+1}$se salen del dominio por tanto recurrimos a condiciones de contorno:

$$u_x(0,t) = t$$

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

$$u_{i-1,j+1} = u_{i+1,j+1} - 2ht_{j+1}$$

$$u_{i-1,j} = u_{i+1,j} - 2ht_j$$

Reemplazamos ambas expresiones en la ecuación anterior:

$$-\frac{\lambda}{2}u_{1,j+1} + (1+\lambda-\frac{k}{2}t_j^2)u_{0,j+1} - \frac{\lambda}{2}u_{1,j+1} + \lambda ht_{j+1} = \frac{\lambda}{2}u_{1,j} + (1-\lambda+\frac{k}{2}t_j^2)u_{0,j} + \frac{\lambda}{2}u_{1,j} - \lambda ht_j + \frac{k}{2}D$$

Reordenando:

$$\mathbf{(1+\lambda-\frac{k}{2}t_j^2)u_{0,j+1} - \lambda u_{1,j+1} = (1-\lambda+\frac{k}{2}t_j^2)u_{0,j} + \lambda u_{1,j} - \lambda ht_j - \lambda ht_{j+1} + \frac{k}{2}D}$$

- ***Para i = 1:***

$$\mathbf{-\frac{\lambda}{2}u_{0,j+1} + (1+\lambda-\frac{k}{2}t_j^2)u_{1,j+1} - \frac{\lambda}{2}u_{2,j+1} = \frac{\lambda}{2}u_{0,j} + (1-\lambda+\frac{k}{2}t_j^2)u_{1,j} + \frac{\lambda}{2}u_{2,j} + \frac{k}{2}D}$$

- ***Para i = nx-1***

$$-\frac{\lambda}{2}u_{nx-2,j+1} + (1+\lambda-\frac{k}{2}t_j^2)u_{nx-1,j+1} - \frac{\lambda}{2}u_{nx,j+1} = \frac{\lambda}{2}u_{nx-2,j} + (1-\lambda+\frac{k}{2}t_j^2)u_{nx-1,j} + \frac{\lambda}{2}u_{nx,j} + \frac{k}{2}D$$

Los valores de $u_{nx,j+1}$ y $u_{nx,j}$ son conocidos:

$$u_{nx,j+1} = sin(t_{j+1}) \qquad u_{nx,j} = sin(t_j)$$

Términos conocidos se agrupan a la derecha de la igualdad, por tanto:

$$\mathbf{(1+\lambda-\frac{k}{2}t_j^2)u_{nx-1,j+1} - \frac{\lambda}{2}u_{nx-2,j+1} = (1-\lambda+\frac{k}{2}t_j^2)u_{nx-1,j} + \frac{\lambda}{2}u_{nx-2,j} + \frac{\lambda}{2}(sin(t_j)) + \frac{\lambda}{2}(sin(t_{j+1})) + \frac{k}{2}D}$$

Las matrices: A, B, $u^{(j+1)}$, $u^{(j)}$ y d_j quedan expresadas de la siguiente forma:

$$A = \begin{bmatrix} 1+\lambda-\frac{\mathrm{k}}{2}t_j^2 & -\lambda & 0 & \dots & 0 \\ -\frac{\lambda}{2} & 1+\lambda-\frac{\mathrm{k}}{2}t_j^2 & -\frac{\lambda}{2} & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & 1+\lambda-\frac{\mathrm{k}}{2}t_j^2 & -\frac{\lambda}{2} \\ 0 & \dots & 0 & -\frac{\lambda}{2} & 1+\lambda-\frac{\mathrm{k}}{2}t_j^2 \end{bmatrix}$$

$$B = \begin{bmatrix} 1-\lambda+\frac{\mathrm{k}}{2}t_j^2 & \lambda & 0 & \dots & 0 \\ \frac{\lambda}{2} & 1-\lambda+\frac{\mathrm{k}}{2}t_j^2 & \frac{\lambda}{2} & \vdots & \vdots \\ \vdots & \vdots & \ddots & \frac{\lambda}{2} & \vdots \\ \vdots & \vdots & \vdots & 1-\lambda+\frac{\mathrm{k}}{2}t_j^2 & \frac{\lambda}{2} \\ 0 & \dots & 0 & \frac{\lambda}{2} & 1-\lambda+\frac{\mathrm{k}}{2}t_j^2 \end{bmatrix}$$

$$u^{(j+1)} = \begin{bmatrix} u_{0,j+1} \\ u_{1,j+1} \\ \vdots \\ \vdots \\ u_{nx-1,j+1} \end{bmatrix} \qquad u^{(j)} = \begin{bmatrix} u_{0,j} \\ u_{1,j} \\ \vdots \\ \vdots \\ u_{nx-1,j} \end{bmatrix}$$

$$d_j = \begin{bmatrix} -\lambda \mathrm{h}(\mathrm{t_j}+\mathrm{t_{j+1}}) \\ 0 \\ \vdots \\ \vdots \\ \frac{\lambda}{2}(\sin(\mathrm{t_j}))+\frac{\lambda}{2}(\sin(\mathrm{t_{j+1}})) \end{bmatrix} + \begin{bmatrix} \frac{k}{2}((x_0 cos(x_0 t_j) + x_0 cos(x_0 t_{j+1})) \\ \frac{k}{2}((x_1 cos(x_1 t_j) + x_1 cos(x_1 t_{j+1})) \\ \vdots \\ \vdots \\ \frac{k}{2}((x_{nx-1} cos(x_{nx-1} t_j) + x_{nx-1} cos(x_{nx-1} t_{j+1})) \end{bmatrix}$$

4.2. Resultados

En la tabla 4 se muestran los resultados obtenidos tras aplicar el método de Cran-Nicholson en el problema planteado. A diferencia de los métodos anteriores, para Crank-Nicholson se requiere aproximar la solución considerando Tmax = 1, en lugar de 0.5.

Tabla 4. Valores aproximados de unidades espaciales x y U(x,t) para distintas secciones temporales (t=1/8, ¼, ½, ¾ de Tmax) utilizando el método de Crank-Nicholson.

Nodos espaciales (X)	U(x, Tmax/8)	U(x, Tmax/4)	U(x, Tmax/2)	U(x, 3Tmax/4)	U(x, Tmax)
0.0	0.045838	0.078367	0.12186	0.1518	0.18024
0.1	0.057248	0.10186	0.16979	0.22432	0.27745
0.2	0.067138	0.123	0.21417	0.29219	0.36863
0.3	0.075887	0.14219	0.25533	0.35551	0.45337
0.4	0.083801	0.15978	0.29361	0.4144	0.53132
0.5	0.091119	0.17609	0.32933	0.46898	0.60218
0.6	0.098023	0.19139	0.36275	0.51936	0.66567
0.7	0.10465	0.20589	0.39416	0.56566	0.72156
0.8	0.11109	0.21979	0.42379	0.60796	0.76962
0.9	0.11742	0.23326	0.45185	0.64636	0.80966
1.0	0.12269	0.24547	0.47767	0.68017	0.84093

5. Análisis

Los métodos estudiados en este laboratorio se caracterizan por tener ventajas y desventajas que pueden hacerlos más convenientes y flexibles según el contexto en los que se necesite. Por consiguiente, procedemos hacer un análisis comparativo entre los tres métodos y observar el comportamiento gráficos y numérico de estos a la hora de aproximar la solución de la EDP parabólica.

5.1. Comparativa Gráfica

A nivel gráfico, podemos comparar los resultados de la solución obtenida en el tiempo t=Tmax con cada uno de los métodos aplicados. En la figura 2, se observa que los métodos explícito e implícito muestran valores aproximados muy cercanos entre sí (recordemos que Crank-Nicholson se evalúa hasta Tmax=1. Esto demuestra que, aunque el método explícito debe aplicarse cumpliendo con un criterio de convergencia específico (ver apartado 2.1), también puede arrojar resultados bastante fiables al evaluar ecuaciones de aplicación directa, en lugar de resolver sistemas de ecuaciones matriciales, como es el caso del método implícito.

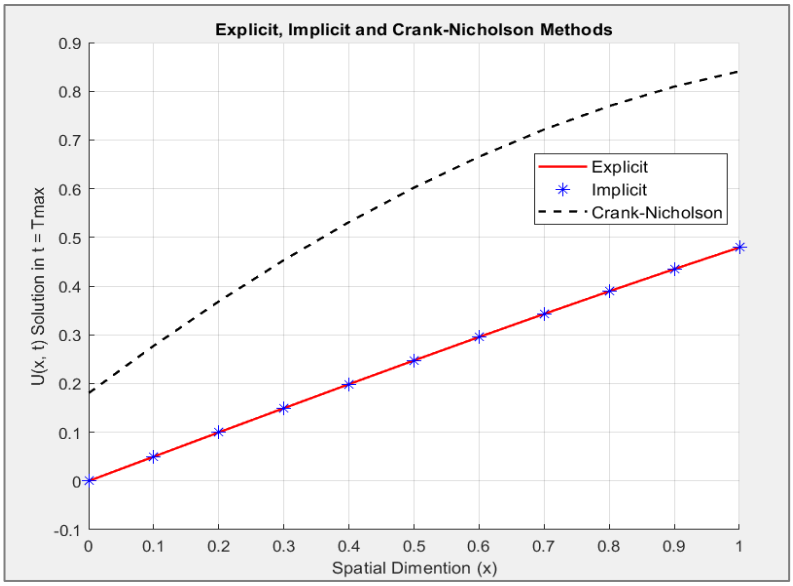

Figura 2. Gráfico comparativo de los métodos Explícito, implícito y Crank-Nicholson utilizados pata t=Tmax. Elaboración propia

La figura 2 también muestra el resultado aproximado de la EDP utilizando Crank-Nicholson con Tmax = 1; lo que muestra la diferencia de resultados con respecto a los otros dos métodos, dado que el espacio temporal es mayor.

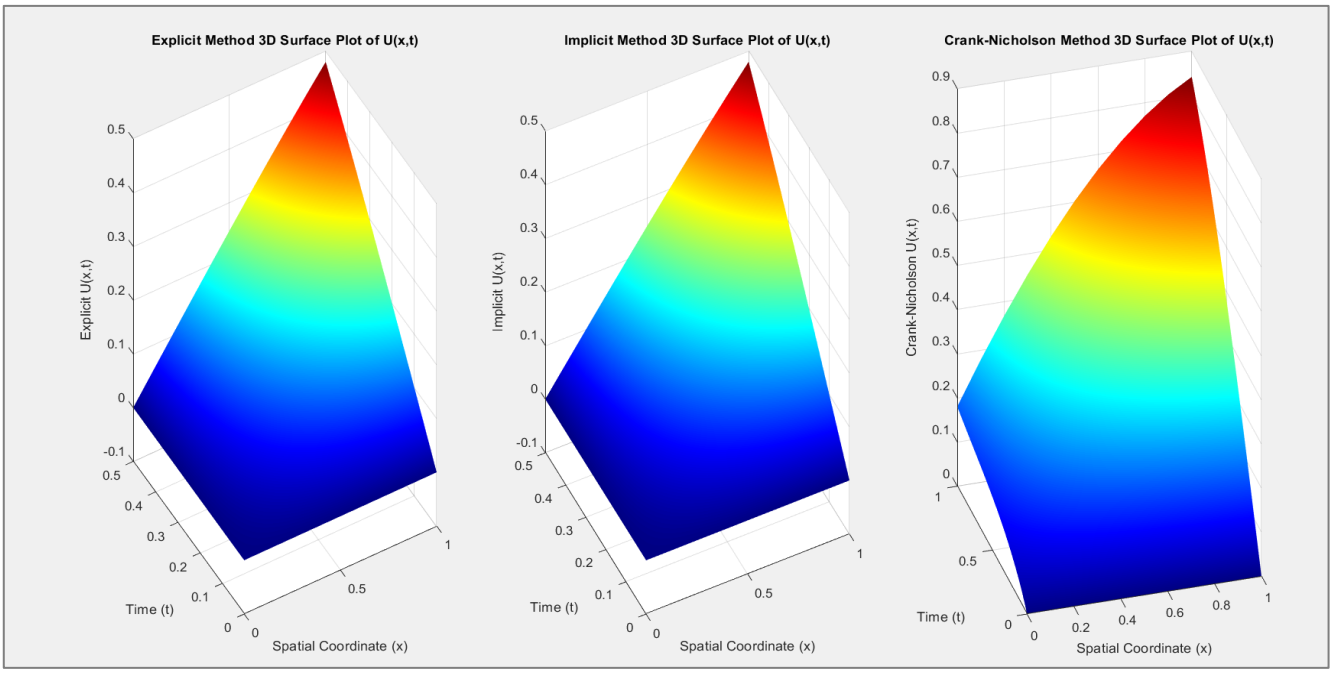

Figura 3. Representación en 3D de las soluciones aproximadas por método Explícito, implícito y Crank-Nicholson utilizados para t=Tmax. Elaboración propia

En la figura 3 se comparan los tres métodos numéricos. El método explícito, mostrado en la izquierda, es simple de implementar, pero su estabilidad depende del criterio de convergencia de Courant-Friedrichs-Lewy (CFL), requiriendo pasos de tiempo pequeños, lo que puede ser costoso computacionalmente (Rubio, 2005). El método implícito, en el centro, es estable incondicionalmente y puede manejar pasos de tiempo grandes, pero requiere resolver sistemas de ecuaciones lineales en cada paso, aumentando la carga computacional.

El método de Crank-Nicholson, a la derecha, combina las ventajas de ambos métodos anteriores, promediando los valores entre los pasos de tiempo para mejorar la precisión mientras mantiene la estabilidad. Este método ofrece una transición más uniforme y es útil para equilibrar precisión y estabilidad, permitiendo pasos de tiempo relativamente grandes sin comprometer la exactitud. Comparando los tres, todos producen resultados razonables, pero el método explícito necesita una cuidadosa selección de los pasos de tiempo y espaciales para evitar inestabilidades, el implícito es robusto, pero computacionalmente intensivo, y Crank-Nicholson se destaca por su equilibrio entre estabilidad y precisión, siendo ideal para muchos problemas prácticos en la resolución de EDPs parabólicas.

5.2. Comparativa numérica (error absoluto)

Los resultados de la tabla 5 presentan los errores absolutos calculados entre los métodos explicito e implícito en diferentes nodos espaciales (X) y en distintos momentos temporales (Tmax/8, Tmax/4, Tmax/2, 3Tmax/4, Tmax).

La primera columna muestra los nodos espaciales, que van desde 0 a 1 con un incremento de 0.1, mientras que las siguientes columnas muestran los errores absolutos correspondientes a esos nodos en diferentes instantes de tiempo. En el nodo espacial X=0, los errores son consistentemente bajos, comenzando desde 1.337×10^{-4} en Tmax/8 y alcanzando 6.58×10^{-1} en Tmax.

Este patrón es similar para otros nodos, mostrando un incremento en el error absoluto conforme avanza el tiempo. Sin embargo, los errores en Tmax son notoriamente altos en comparación con los valores iniciales, lo que indica que los métodos numéricos empleados pueden volverse menos precisos a medida que la coordenada temporal aumenta.

En nodos como X=0.5, los errores son más elevados en tiempos intermedios Tmax/2y 3Tmax/4, alcanzando valores como 8.74×10^{-1} y 6.22×10^{-2} respectivamente, pero disminuyen en TmaxT . Esto sugiere que, en la mitad del intervalo espacial, los métodos numéricos pueden estar experimentando mayores inestabilidades, posiblemente debido a la naturaleza del problema o la discretización espacial y temporal utilizada.

Sin embargo, en nodos cercanos como X=0.9, los errores decrecen desde 6.30×10^{-1} en Tmax/8 hasta 7.42×10^{-2} en Tmax, mostrando una mejora en la precisión hacia el final del intervalo temporal.

Tabla 5. Resumen de resultados del método del Disparo por Newton y secante para problemas de contorno

Nodos espaciales (X)	Errores Absolutos				
	Tmax/8	Tmax/4	Tmax/2	3Tmax/4	Tmax
0.0	0.0001337	0.00015253	0.00012315	9,07E-01	6,58E-01
0.1	0.00013776	0.00015225	0.00012169	8,95E-02	6,48E-01
0.2	0.00014885	0.00015108	0.00011731	8,60E-01	6,20E-01
0.3	0.00016377	0.00014807	0.00011006	8,01E-01	5,75E-01
0.4	0.00017765	0.00014191	0.00010003	7,21E-01	5,13E-01
0.5	0.00018469	0.00013125	8,74E-01	6,22E-02	4,37E-01
0.6	0.00017923	0.00011505	7,24E-01	5,08E-02	3,50E-02
0.7	0.00015721	9,29E-02	5,56E-01	3,82E-01	2,57E-01
0.8	0.00011753	6,53E-01	3,74E-01	2,51E-01	1,62E-01
0.9	6,30E-01	3,37E-01	1,86E-01	1,21E-01	7,42E-02
1.0	0	0	0	0	0

REFERENCIAS

García, D., & Armando, A. (2007). *Esquema de discretización de Crank-Nicholson aplicado a balances de calor en un volumen de control (parte ii)*. Retrieved 6 16, 2024, from http://redalyc.org/pdf/4455/445543754004.pdf

Lam, C. Y. (1992). Spreadsheet approach to partial differential equations part 2: parabolic and hyperbolic equations. *International Journal of Engineering Education, 8*(5), 371-383. Retrieved 6 16, 2024, from https://dialnet.unirioja.es/servlet/articulo?codigo=6542953

Polyanin, A. D. (2002). *Handbook of linear partial differential equations for engineers and scientists.* CRC Press. Retrieved 6 16, 2024, from https://books.google.com/books?id=NLnwhsevQGEC&pg=PA476#v=onepage&q&f=false

Rubio, R. G. (2005). *Métodos de diferencias finitas incodicionalmente estables para la resolución de las ecuaciones de maxwell en el dominio del tiempo*. Retrieved 6 16, 2024, from https://dialnet.unirioja.es/servlet/tesis?codigo=108732

- **Función Matlab para Método Explícito**

```
function [u, x, t] = ExplicitMethodPDE(alpha, ci, a, b, nx, Tmax, nt)

    % ExplicitMethodPDE Solves a parabolic PDE using the explicit method with non-Dirichlet
conditions
    %
    % Inputs:
    %   alpha - Coefficient in the PDE
    %   ci - Initial condition function handle
    %   a - Left boundary of the spatial domain
    %   b - Right boundary of the spatial domain
    %   nx - Number of spatial points
    %   Tmax - Maximum time
    %   nt - Number of time steps
    %
    % Outputs:
    %   u - Solution matrix (spatial points x time steps)
    %   x - Spatial grid points
    %   t - Time grid points


    % Compute spatial and temporal step sizes
    h=(b-a)/nx;
    x=a:h:b;
    k = Tmax / nt;
    t = 0:k:Tmax;

    % Evaluate the initial condition at spatial grid points
    cix = feval(ci, x);
    u = zeros(nx+1, nt);
    u(:, 1) = cix';

    % Stability criterion
    lambda = k * alpha^2 / h^2;
    if lambda > 1/2
        disp('No se cumple el criterio de convergencia')
    else
        disp('sin problema')
    end

    for j = 2:nt
        u(1, j+1) = (1 - 2*lambda + k*(t(j)^2)) * u(1, j) + lambda * (2*u(2, j) - (2*h*t(j))) + k *
x(1) * cos(x(1) * t(j));
        u(2:nx, j+1) = (1 - 2*lambda + k * (t(j)^2)) .* u(2:nx, j) + lambda * (u(3:nx+1, j) +
u(1:nx-1, j)) + k * x(2:nx)' .* cos(x(2:nx)' * t(j));
        u(nx+1, j+1) = sin(t(j+1));
    end
end
```

- **Función Matlab para Método Implícito**

```
function [U, x, t] = ImplicitMethodPDE(alpha, ci, a, b, nx, Tmax, nt)
    % ImplicitMethodPDE Solves a parabolic PDE using the implicit method
    %
    % Inputs:
    %   alpha - Coefficient in the PDE
    %   ci - Initial condition function handle
    %   a - Left boundary of the spatial domain
    %   b - Right boundary of the spatial domain
    %   nx - Number of spatial points
    %   Tmax - Maximum time
    %   nt - Number of time steps
    %
    % Outputs:
    %   U - Solution matrix (spatial points x time steps)
    %   x - Spatial grid points
```

```
    %   t - Time grid points

    % Compute spatial and temporal step sizes
    h=(b-a)/nx;     x=a:h:b;
    k=Tmax/nt;      t=0:k:Tmax;

    % Stability criterion
    lambda=k*alpha^2/h^2;


    % Initialize solution matrix and set initial condition
    U = zeros(nx+1, nt);
    cix= feval(ci ,x);
    U(:,1)=cix';


    % Initialize tridiagonal matrix components
    ds = -lambda*ones(nx-1,1);
    di = ds;
    ds(1) = -2*lambda;
    dp = ones(nx,1);
    d = ones(nx,1);

    % Time-stepping loop to solve the PDE
    for j = 2:nt+1
        dp(1) = 1+(2*lambda) - k*(t(j)^2);
        dp(nx) = 1+(2*lambda) - k*(t(j)^2);
        dp(2:nx-1) = 1+(2*lambda) - k*(t(j)^2);

        d(1) =  k*(x(1) * cos(x(1) * t(j))) - (2 * h * lambda * t(j));
        d(nx) = lambda * sin(t(j)) + k*(x(nx) * cos(x(nx) * t(j)));
        d(2:nx-1) = k.*(x(2:nx-1) .* cos(x(2:nx-1) * t(j)));
        dj = U(1:nx,j-1) + d;

        % Solve the tridiagonal system using Crout's method
        z = Crout(dp, ds, di, dj);
        U(1:nx, j) = z;
        U(nx+1, j) = sin(t(j));
    end

end
```

- **Función Matlab para Método Crank-Nicholson**

```
function [U,x, t]=CrankNicholsonPDE(alpha,ci,a,b,nx,Tmax,nt)
    % CrankNicholsonPDE Solves a parabolic PDE using the Crank-Nicholson method
    %
    % Inputs:
    %   alpha - Coefficient in the PDE
    %   ci - Initial condition function handle
    %   a - Left boundary of the spatial domain
    %   b - Right boundary of the spatial domain
    %   nx - Number of spatial points
    %   Tmax - Maximum time
    %   nt - Number of time steps
    %
    % Outputs:
    %   U - Solution matrix (spatial points x time steps)
    %   x - Spatial grid points
    %   t - Time grid points

    % Compute spatial and temporal step sizes
    h=(b-a)/nx;
    x=a:h:b;
    k=Tmax/nt;
    t=0:k:Tmax;


    % Initialize solution matrix and set initial condition
    U=zeros(nx+1,nt);
```

```
    cix=feval(ci,x);
    U(:,1)=cix';

    % Stability criterion
    lambda=k*alpha^2/h^2;

    % Initialize tridiagonal matrix components for the Crank-Nicholson method
    ds=(-lambda/2)*ones(nx-1,1);
    di=ds;
    ds(1)=-lambda;

    bs=(lambda/2)*ones(nx-1,1);
    bi=bs;
    bs(1)=lambda;
    d = ones(nx,1);


    % Time-stepping loop to solve the PDE
    for j=1:nt

        dp=(1+lambda)-((k/2)*t(j)^2)*ones(nx,1);
        bp=(1-lambda)+((k/2)*t(j)^2)*ones(nx,1);

        % Construct the B matrix
        B=diag(bp)+diag(bs,1)+diag(bi,-1);

        % Construct the dj vector
        d(1)=(-lambda*h)*(t(j)+(t(j+1))) + (k/2)*(x(1)*(cos(x(1)*t(j)))+cos(x(1)*t(j+1)));
        d(nx)=(lambda/2)*(sin(t(j))+sin((t(j+1)))) +
(k/2)*(x(nx)*(cos(x(nx)*t(j)))+cos(x(nx)*t(j+1)));
        d(2:nx-1)=(k/2)*(x(2:nx-1).*(cos(x(2:nx-1)*t(j)))+cos(x(2:nx-1)*t(j+1)));
        dj=(B*U(1:nx,j)) + d;


        % Solve the tridiagonal system using Crout's method
        z=Crout(dp,ds,di,dj);

        U(1:nx,j+1)=z;
        U(nx+1,j+1)=sin(t(j));
    end
end
```