Instituto Politécnico do Rio de Janeiro

Universidade Estadual do Rio de Janeiro

Curso de Graduação em Engenharia da Computação

Trabalho 2 - Métodos Numéricos para Equações Diferenciais

Professor: Helio Pedro Amaral Souto

Aluno: Victor Luis Teixeira Reis

Nova Friburgo

2024

1. Introdução

O trabalho consiste no uso de métodos numéricos e analíticos para obter a solução de um problema de propagação de uma frente de temperatura. Para isso, serão usadas várias formulações explícitas e implícitas de resolução de equações diferenciais parciais, com o objetivo de comparar, por meio do cálculo do seu Erro Médio Quadrado (EMQ) e de gráficos, a eficácia e acurácia de cada uma em relação à solução analítica do problema.

1.1 Apresentação do problema

O problema que será solucionado envolve a modelagem da propagação de uma frente de temperatura usando equações diferenciais parciais. Em tais sistemas, no instante de tempo $t = 0$, há uma frente descontínua de temperatura localizada em $x = 0$. A figura 1 ilustra esse tipo de problema:

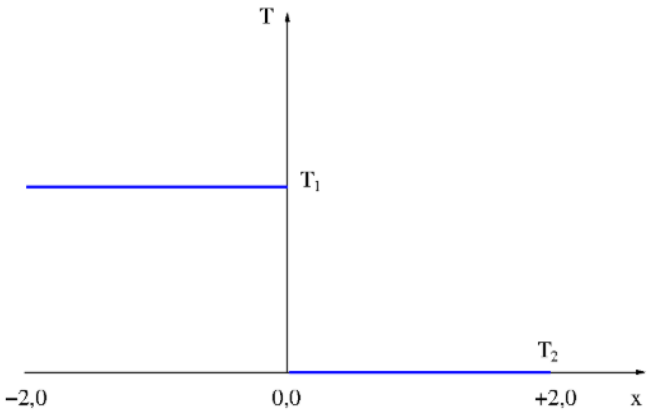

***Figura 1**: Perfil inicial da frente de temperatura*

Nesse caso, a modelagem do sistema é feita usando a equação de advecção-condução de calor unidimensional [2]:

$$\frac{\partial T}{\partial t} + u\frac{\partial T}{\partial x} = \alpha\frac{\partial^2 T}{\partial x^2} \quad u > 0 \tag{1}$$

Essa formulação é uma equação diferencial parcial cuja função $T(x,\ t)$ representa a temperatura em um ponto x, em um determinado instante de tempo t. O termo u é chamado de velocidade de advecção, enquanto α é a difusividade térmica.

Além disso, existem algumas condições iniciais e de contorno que devem ser respeitadas para que o problema possa ser devidamente resolvido. A condição inicial é que no instante $t = 0$, para toda posição $x < 0$, temos a temperatura $T_1 = 1,0$, enquanto que para toda posição $x > 0$, temos a temperatura $T_2 = 0,0$.

$$T(x,0) = \begin{cases} 1,0 & \text{se } x < 0 \\ 0,0 & \text{se } x > 0 \end{cases} \tag{2}$$

Já as condições de contorno atuam nos extremos do domínio espacial do problema e dizem que em qualquer momento na posição $x = -2,0$, a temperatura será T_1 e em $x = 2,0$, será T_2:

$$T(x,t) = \begin{cases} T_1 & \text{se } x = -2 \\ T_2 & \text{se } x = +2 \end{cases} \quad (3)$$

1.2 Apresentação da solução analítica

A solução analítica pode ser obtida para esse caso por meio de um processo de separação de variáveis [3] usando a equação (1) já descrita e considerando $T_1 = 1,0$ e $T_2 = 0,0$ [2]. A expressão final da solução é apresentada abaixo:

$$T(x,t) = 0,5 - \frac{2}{\pi}\sum_{k=1}^{\infty}\frac{1}{2k-1}\exp\left[\frac{-\alpha(2k-1)^2\pi^2 t}{L^2}\right]\text{sen}\left[(2k-1)\frac{\pi(x-ut)}{L}\right] \quad (4)$$

A variável L representa o tamanho do domínio espacial do problema, que vai de $x = -2,0$ até $x = 2,0$, logo $L = 4,0$.

Essa formulação (4) será utilizada ao longo do trabalho para compararmos com os resultados dos métodos numéricos, permitindo ver quais são mais acurados.

1.3 Apresentação do método numérico

O método numérico utilizado para aproximar a solução baseia-se em substituir as derivadas parciais da equação (1) por diferenças finitas centradas, avançadas ou por uma combinação desses esquemas. Neste trabalho, será utilizada uma formulação em que o termo advectivo da equação será aproximado por diferenças centradas ou avançadas de 1° ordem e o termo difusivo por uma diferença centrada à 3 pontos, resultando na seguinte expressão geral:

$$-\beta[(1-\delta)C+s]\phi_{j-1}^{n+1} + \{1+\beta[(1-2\delta)C+2s]\}\phi_j^{n+1} - \beta(-\delta C+s)\phi_{j+1}^{n+1}$$

$$=(1-\beta)[(1-\delta)C+s]\phi_{j-1}^{n} + \{1-(1-\beta)[(1-2\delta)C+2s]\}\phi_j^{n} - (1-\beta)(-\delta C+s)\phi_{j+1}^{n} \quad (5)$$

Com apenas uma equação matemática, conforme alteramos os valores das variáveis β e δ, podemos obter diferentes métodos para resolver o mesmo problema:

- $\beta = 0,0,\ \delta = 0,0 \rightarrow$ *Método Upwind*
- $\beta = 0,0,\ \delta = 0,5 \rightarrow$ *Método FTCS(Forward Time Centered Space)*
- $\beta = 0,0,\ \delta = 0,5(1-C) \rightarrow$ *Método Lax Wendroff*
- $\beta = 0,5,\ \delta = 0,0,\ 0,5,\ 0,5(1-C) \rightarrow$ *Método Crank Nicolson*
- $\beta = 1,0,\ \delta = 0,0,\ 0,5,\ 0,5(1-C) \rightarrow$ *Método Totalmente Implícito*

Além disso, os termos C e s são obtidos durante as manipulações algébricas e equivalem $C = u\Delta t/\Delta x$ e $s = \alpha\Delta t/\Delta x^2$, sendo bastante usados nas análises de estabilidade dos métodos, conforme mostra a tabela 1:

Método	β	δ	Condição
FTCS (Explícito)	0	0,5	$0 \leq C^2 \leq 2s \leq 1$
Upwind (Explícito)	0	0	$C + 2s \leq 1$
Lax-Wendroff (Explícito)	0	0,5(1-C)	$C^2 + 2s \leq 1$
Crank-Nicolson	0,5	todos	nenhuma
Totalmente Implícito	1	todos	nenhuma

***Tabela 1**: Condições de estabilidade para os métodos*

Essas condições definem os limites de funcionamento de alguns dos esquemas que serão usados. Aqueles que possuem $\beta = 0$ são denominados explícitos e possuem condições claras de estabilidade que se não forem seguidas resultam no método não funcionando da maneira correta. Já o Crank-Nicolson e o Totalmente Implícito não apresentam essas limitações.

Para resolver esses métodos, será usado o Algoritmo de Thomas, que serve para solucionar sistemas que podem ser representados como matrizes tridiagonais. A equação (5) pode ser escrita desta forma:

$$-c_j\phi_{j-1}^{n+1} + a_j\phi_j^{n+1} - b_j\phi_{j+1}^{n+1} = d_j^n$$

Em que:

- $a_j = 1 + \beta[(1 - 2\delta)C + 2s]$
- $b_j = \beta(-\delta C + s)$
- $c_j = \beta[(1 - \delta)C + s]$
- $d_j = (1 - \beta)[(1 - \delta)C + s]\phi^n{}_{j-1} + \{1 - (1 - \beta)[(1 - 2\delta)C + 2s]\}\phi^n{}_j + (1 - \beta)(-\delta C + s)\phi^n{}_{j+1}$

E resulta em:

$$\begin{bmatrix} a_1 & -b_1 & & & & & \\ -c_2 & a_2 & -b_2 & & & & \\ \ddots & \ddots & \ddots & & & & \\ & & -c_j & a_j & -b_j & & \\ & & \ddots & \ddots & \ddots & & \\ & & & & -c_{J-1} & a_{J-1} & -b_{J-1} \\ & & & & & -c_J & a_J \end{bmatrix} \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_j \\ \vdots \\ \phi_{J-1} \\ \phi_J \end{bmatrix}^{n+1} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_j \\ \vdots \\ d_{J-1} \\ d_J \end{bmatrix}$$

***Figura 2**: Matriz tridiagonal da equação geral (5)*

Os termos a_j, b_j, c_j e d_j são obtidos a partir própria equação (5), já o vetor ϕ será formado pela temperatura futura em cada posição da malha que representa o domínio espacial do problema. Quanto menor for o incremento Δx utilizado, maior será o tamanho desse vetor.

Portanto, é possível escrever a equação (5) como uma matriz tridiagonal. A resolução se baseará em, a cada incremento de tempo, calcular novos valores de d_j e resolver um sistema de equações usando o método TDMA, gradualmente atualizando a temperatura em cada ponto da malha até chegarmos ao tempo final desejado. O Algoritmo de Thomas é resumidamente explicado na figura 3:

1. Determinação dos valores de P_1 e Q_1: $P_1 = 0$ e $Q_1 = T_1$;
2. Cálculo dos valores de P_j e Q_j para $j = 2$ até $j = J - 1$:

$$P_j = \frac{b_j}{a_j - c_j P_{j-1}}$$

e

$$Q_j = \frac{d_j + c_j Q_{j-1}}{a_j - c_j P_{j-1}};$$

3. Determinação dos valores P_J e Q_J: $P_J = 0$ e $Q_J = T_2$;
4. Calcular a solução numérica do sistema algébrico, ϕ_j^{n+1}, para j variando de $J - 1$ até 1:

$$\phi_j^{n+1} = P_j \phi_{j+1}^{n+1} + Q_j.$$

***Figura 3**: Algoritmo de Thomas (TDMA)*

2. Resultados

Para a apresentação dos resultados, serão resolvidos diversos casos com diferentes condições físicas. Os resultados finais junto com o erro médio quadrático (EMQ) serão impressos na tela a cada execução dos métodos. O EMQ servirá como comparação dos resultados obtidos em relação à solução analítica e, quanto menor for seu valor, mais acurado terá sido a aproximação. Em todos os casos, os seguintes parâmetros físicos serão constantes:

Variável	Valor
α	$1,0 \times 10^{-1}$
L	$4,0$
T_1	$1,0$
T_2	$0,0$

***Tabela 2**: Parâmetros físicos*

2.1 Primeiro caso

Nesse caso, serão utilizados os seguintes valores para as variáveis:

- $\beta = 0$
- $C = 0$
- $s = 1/6 \; e \; 1$
- $t_{max} = 1,0$

Sendo $C = 0$, como Δx e Δt devem ser maiores que zero, logo $u = 0$. Como $\alpha = 0,1$, usando $s = 1/6$ e escolhendo arbitrariamente $\Delta t = 0,1$, podemos calcular que o incremento no espaço $\Delta x \approx 0,2449$ terá que ser usado para satisfazer as equações. Nessa configuração, o resultado obtido para os 3 métodos com $\beta = 0$ é:

```
EQUACAO DE ADVECCAO-DIFUSAO: FTCS (EXPLÍCITO)

JMAX= 17 TMAX= 1.00 T1= 1.00 T2= 0.00
BETA= 0.00000 DELTA= 0.50000
DELTA_T= 0.10000 DELTA_X= 0.24490
S= 0.16673 ALPHA= 0.10000
U= 0.00000 C= 0.00000

t= 0.000 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.917 0.500 0.083 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.100 TN = 1.000 1.000 1.000 1.000 1.000 1.000 0.986 0.861 0.500 0.139 0.014 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.200 TN = 1.000 1.000 1.000 1.000 1.000 0.998 0.968 0.822 0.500 0.178 0.032 0.002 0.000 0.000 0.000 0.000 0.000
t= 0.300 TN = 1.000 1.000 1.000 1.000 1.000 0.993 0.948 0.792 0.500 0.208 0.052 0.007 0.000 0.000 0.000 0.000 0.000
t= 0.400 TN = 1.000 1.000 1.000 1.000 0.999 0.987 0.930 0.770 0.500 0.230 0.070 0.013 0.001 0.000 0.000 0.000 0.000
t= 0.500 TN = 1.000 1.000 1.000 1.000 0.997 0.979 0.913 0.751 0.500 0.249 0.087 0.021 0.003 0.000 0.000 0.000 0.000
t= 0.600 TN = 1.000 1.000 1.000 0.999 0.994 0.971 0.897 0.736 0.500 0.264 0.103 0.029 0.006 0.001 0.000 0.000 0.000
t= 0.700 TN = 1.000 1.000 1.000 0.999 0.991 0.963 0.882 0.724 0.500 0.276 0.118 0.037 0.009 0.001 0.000 0.000 0.000
t= 0.800 TN = 1.000 1.000 1.000 0.998 0.988 0.954 0.869 0.713 0.500 0.287 0.131 0.046 0.012 0.002 0.000 0.000 0.000
t= 0.900 TN = 1.000 1.000 0.999 0.996 0.984 0.945 0.857 0.703 0.500 0.297 0.143 0.055 0.016 0.004 0.001 0.000 0.000
t= 1.000 TN = 1.000 1.000 0.999 0.995 0.979 0.937 0.846 0.695 0.500 0.305 0.154 0.063 0.021 0.005 0.001 0.000 0.000

t= 1.000 TE = 1.000 1.000 1.000 0.997 0.987 0.953 0.868 0.712 0.500 0.288 0.132 0.047 0.013 0.003 0.000 0.000 0.000

EMQ= 0.0113
```

***Figura 4**: Resultados usando FTCS*

```
EQUACAO DE ADVECCAO-DIFUSAO: UPWIND (EXPLÍCITO)

JMAX= 17 TMAX= 1.00 T1= 1.00 T2= 0.00
BETA= 0.00000 DELTA= 0.00000
DELTA_T= 0.10000 DELTA_X= 0.24490
S= 0.16673 ALPHA= 0.10000
U= 0.00000 C= 0.00000

t= 0.000 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.917 0.500 0.083 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.100 TN = 1.000 1.000 1.000 1.000 1.000 1.000 0.986 0.861 0.500 0.139 0.014 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.200 TN = 1.000 1.000 1.000 1.000 1.000 0.998 0.968 0.822 0.500 0.178 0.032 0.002 0.000 0.000 0.000 0.000 0.000
t= 0.300 TN = 1.000 1.000 1.000 1.000 1.000 0.993 0.948 0.792 0.500 0.208 0.052 0.007 0.000 0.000 0.000 0.000 0.000
t= 0.400 TN = 1.000 1.000 1.000 1.000 0.999 0.987 0.930 0.770 0.500 0.230 0.070 0.013 0.001 0.000 0.000 0.000 0.000
t= 0.500 TN = 1.000 1.000 1.000 1.000 0.997 0.979 0.913 0.751 0.500 0.249 0.087 0.021 0.003 0.000 0.000 0.000 0.000
t= 0.600 TN = 1.000 1.000 1.000 0.999 0.994 0.971 0.897 0.736 0.500 0.264 0.103 0.029 0.006 0.001 0.000 0.000 0.000
t= 0.700 TN = 1.000 1.000 1.000 0.999 0.991 0.963 0.882 0.724 0.500 0.276 0.118 0.037 0.009 0.001 0.000 0.000 0.000
t= 0.800 TN = 1.000 1.000 1.000 0.998 0.988 0.954 0.869 0.713 0.500 0.287 0.131 0.046 0.012 0.002 0.000 0.000 0.000
t= 0.900 TN = 1.000 1.000 0.999 0.996 0.984 0.945 0.857 0.703 0.500 0.297 0.143 0.055 0.016 0.004 0.001 0.000 0.000
t= 1.000 TN = 1.000 1.000 0.999 0.995 0.979 0.937 0.846 0.695 0.500 0.305 0.154 0.063 0.021 0.005 0.001 0.000 0.000

t= 1.000 TE = 1.000 1.000 1.000 0.997 0.987 0.953 0.868 0.712 0.500 0.288 0.132 0.047 0.013 0.003 0.000 0.000 0.000

EMQ= 0.0113
```

***Figura 5**: Resultados usando Upwind*

Como nesse caso $C = 0$, o esquema de Lax-Wendroff com $delta = 0,5(1 - C)$ se torna exatamente igual ao esquema FTCS, logo os resultados são iguais. Tanto o esquema FTCS quanto o Upwind tiveram erros médios quadráticos iguais nas quatro primeiras casas decimais, logo foram bem equivalentes.

Usando agora $s = 1$, nenhum dos métodos explícitos implementados poderão ser usados, já que suas condições de estabilidade não são satisfeitas. Como mostrado na tabela 1, nos três métodos com $\beta = 0$, as condições não são respeitadas:

$$FTCS: 2 * s \leq 1 \rightarrow 2 * 1 \leq 1 \rightarrow 2 \geq 1$$

$$Upwind: C + 2 * s \leq 1 \rightarrow 0 + 2 * 1 \leq 1 \rightarrow 2 \geq 1$$

$$Lax\,Wendroff: C^2 + 2 * s \leq 1 \rightarrow 0^2 + 2 * 1 \leq 1 \rightarrow 2 \geq 1$$

2.2 Segundo Caso

Nesse caso, serão utilizados os seguintes valores para as variáveis:

- $\beta = 0$
- $u = 0,5\ e\ 4,\ 0$
- $\Delta t = 0,05$
- $\Delta x = 0,2$
- $\delta = 0,0,\ 0,5\ e\ 0,5(1 - C)$
- $t_{max} = 1,0$

Segue abaixo os resultados de todos os 3 métodos explícitos com $u = 0,5$:

```
EQUACAO DE ADVECCAO-DIFUSAO: FTCS (EXPLÍCITO)

JMAX= 21 TMAX= 1.00 T1= 1.00 T2= 0.00
BETA= 0.00000 DELTA= 0.50000
DELTA_T= 0.05000 DELTA_X= 0.20000
S= 0.12500 ALPHA= 0.10000
U= 0.50000 C= 0.12500

t= 0.000 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.969 0.562 0.094 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.050 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.949 0.609 0.176 0.018 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.100 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.995 0.937 0.646 0.247 0.046 0.003 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.150 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.993 0.930 0.676 0.309 0.081 0.011 0.001 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.200 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.990 0.926 0.700 0.364 0.120 0.024 0.003 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.250 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.988 0.924 0.722 0.412 0.159 0.040 0.006 0.001 0.000 0.000 0.000 0.000 0.000
t= 0.300 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.986 0.923 0.740 0.454 0.199 0.060 0.012 0.002 0.000 0.000 0.000 0.000 0.000
t= 0.350 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.984 0.923 0.757 0.492 0.238 0.083 0.021 0.004 0.000 0.000 0.000 0.000 0.000
t= 0.400 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.997 0.983 0.924 0.771 0.526 0.276 0.109 0.031 0.007 0.001 0.000 0.000 0.000 0.000
t= 0.450 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.997 0.982 0.926 0.785 0.556 0.312 0.135 0.044 0.011 0.002 0.000 0.000 0.000 0.000
t= 0.500 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.996 0.981 0.928 0.797 0.584 0.347 0.163 0.059 0.017 0.004 0.001 0.000 0.000 0.000
...
t= 1.000 TN = 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.994 0.981 0.948 0.879 0.763 0.605 0.429 0.268 0.146 0.069 0.028 0.010 0.003 0.000

t= 1.000 TE = 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.993 0.978 0.941 0.868 0.749 0.588 0.412 0.251 0.132 0.059 0.022 0.007 0.002 0.000

EMQ= 0.0088
```

***Figura 6**: Resultados usando FTCS*

```
EQUACAO DE ADVECCAO-DIFUSAO: UPWIND (EXPLÍCITO)

JMAX= 21 TMAX= 1.00 T1= 1.00 T2= 0.00
BETA= 0.00000 DELTA= 0.00000
DELTA_T= 0.05000 DELTA_X= 0.20000
S= 0.12500 ALPHA= 0.10000
U= 0.50000 C= 0.12500

t= 0.000 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.938 0.562 0.125 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.050 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.992 0.906 0.602 0.219 0.031 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.100 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.983 0.890 0.630 0.291 0.074 0.008 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.150 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.997 0.976 0.881 0.652 0.349 0.120 0.023 0.002 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.200 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.995 0.969 0.876 0.672 0.396 0.165 0.045 0.007 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.250 TN = 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.993 0.964 0.874 0.688 0.436 0.208 0.070 0.016 0.002 0.000 0.000 0.000 0.000 0.000
t= 0.300 TN = 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.991 0.960 0.873 0.703 0.471 0.248 0.098 0.028 0.005 0.001 0.000 0.000 0.000 0.000
t= 0.350 TN = 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.989 0.957 0.874 0.716 0.501 0.285 0.127 0.042 0.010 0.002 0.000 0.000 0.000 0.000
t= 0.400 TN = 1.000 1.000 1.000 1.000 1.000 1.000 0.997 0.987 0.954 0.875 0.729 0.528 0.319 0.156 0.059 0.017 0.004 0.001 0.000 0.000 0.000
t= 0.450 TN = 1.000 1.000 1.000 1.000 1.000 0.999 0.997 0.986 0.953 0.876 0.740 0.552 0.351 0.184 0.078 0.026 0.007 0.001 0.000 0.000 0.000
t= 0.500 TN = 1.000 1.000 1.000 1.000 1.000 0.999 0.996 0.984 0.951 0.878 0.751 0.574 0.380 0.213 0.098 0.037 0.011 0.002 0.000 0.000 0.000
...
t= 1.000 TN = 1.000 1.000 1.000 1.000 0.999 0.997 0.991 0.978 0.952 0.904 0.827 0.721 0.589 0.447 0.311 0.197 0.112 0.057 0.026 0.010 0.000

t= 1.000 TE = 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.993 0.978 0.941 0.868 0.749 0.588 0.412 0.251 0.132 0.059 0.022 0.007 0.002 0.000

EMQ= 0.0296
```

***Figura 7**: Resultados usando Upwind*

```
EQUACAO DE ADVECCAO-DIFUSAO: LAX-WENDROFF (EXPLÍCITO)

JMAX= 21 TMAX= 1.00 T1= 1.00 T2= 0.00
BETA= 0.00000 DELTA= 0.43750
DELTA_T= 0.05000 DELTA_X= 0.20000
S= 0.12500 ALPHA= 0.10000
U= 0.50000 C= 0.12500

t= 0.000 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.965 0.562 0.098 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.050 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.943 0.608 0.182 0.019 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.100 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.994 0.930 0.644 0.254 0.049 0.004 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.150 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.991 0.923 0.672 0.315 0.086 0.012 0.001 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.200 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.988 0.918 0.696 0.369 0.126 0.026 0.003 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.250 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.985 0.916 0.717 0.416 0.166 0.044 0.007 0.001 0.000 0.000 0.000 0.000 0.000
t= 0.300 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.983 0.916 0.734 0.457 0.206 0.065 0.014 0.002 0.000 0.000 0.000 0.000 0.000
t= 0.350 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.997 0.981 0.916 0.750 0.494 0.245 0.089 0.023 0.004 0.000 0.000 0.000 0.000 0.000
t= 0.400 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.996 0.980 0.917 0.765 0.526 0.283 0.115 0.035 0.008 0.001 0.000 0.000 0.000 0.000
t= 0.450 TN = 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.996 0.978 0.919 0.778 0.556 0.319 0.142 0.048 0.012 0.002 0.000 0.000 0.000 0.000
t= 0.500 TN = 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.995 0.978 0.920 0.790 0.582 0.353 0.170 0.064 0.019 0.004 0.001 0.000 0.000 0.000
...
t= 1.000 TN = 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.993 0.978 0.942 0.871 0.757 0.603 0.432 0.275 0.154 0.075 0.032 0.012 0.004 0.000

t= 1.000 TE = 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.993 0.978 0.941 0.868 0.749 0.588 0.412 0.251 0.132 0.059 0.022 0.007 0.002 0.000

EMQ= 0.0101
```

***Figura 8**: Resultados usando Lax-Wendroff*

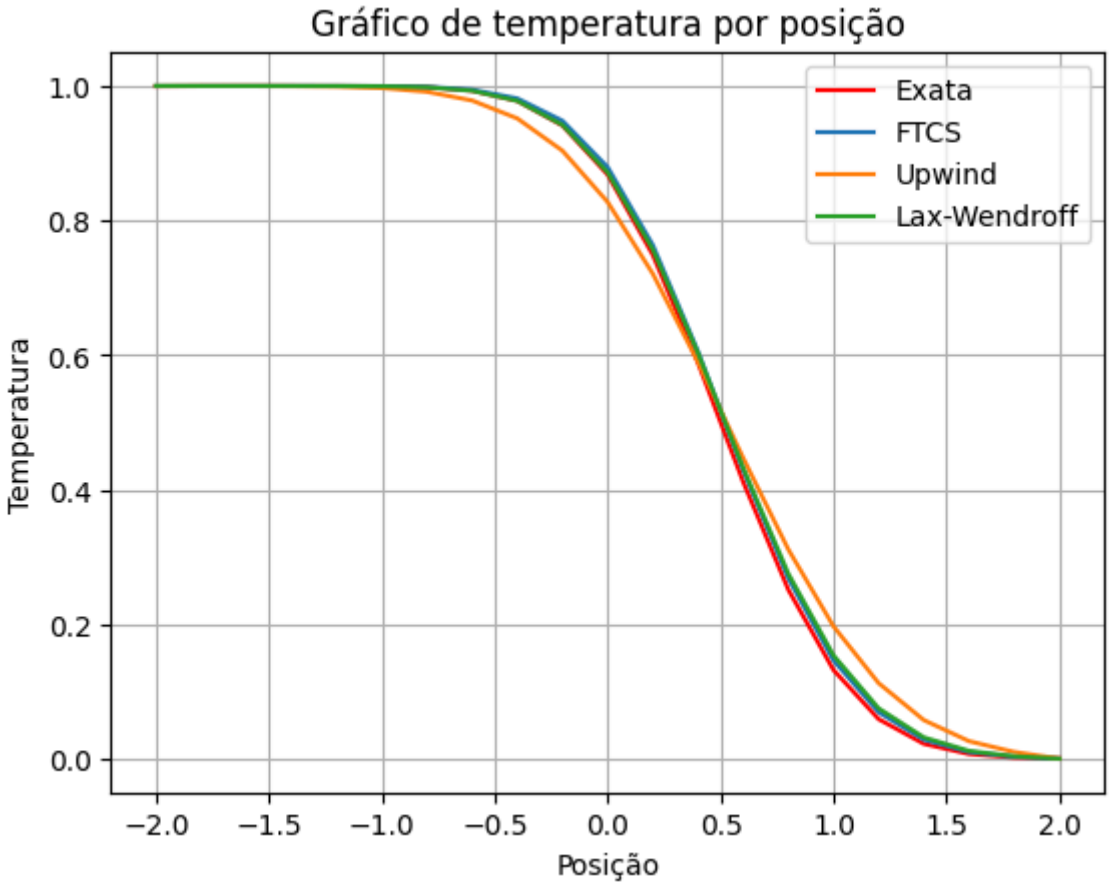

***Figura 9**: Gráfico com todos os resultados do caso 2*

Nesta rodada de resultados, o método que mostrou maior acurácia foi o FTCS, com um EMQ igual a 0,0088. O esquema Upwind ficou um pouco diferente da solução exata em alguns pontos, devido à difusão numérica que seu erro de truncamento tende a acrescentar no resultado em algumas condições físicas. O esquema de Lax-Wendroff ficou como um meio termo, sendo um pouco pior do que o FTCS nesse caso.

Com $u = 4,0$, C será igual a 1 e s será igual a 0,125, fazendo com que os 3 métodos explícitos não possam ser utilizados, já que suas condições de estabilidade não são satisfeitas:

$$FTCS: C^2 \leq 2 * s \rightarrow 1^2 \leq 2 * 0,125 \rightarrow 1 \geq 0,25$$

$$Upwind: C + 2 * s \leq 1 \rightarrow 1 + 2 * 0,125 \leq 1 \rightarrow 1,25 \geq 1$$

$$Lax\,Wendroff: C^2 + 2 * s \leq 1 \rightarrow 1^2 + 2 * 0,125 \leq 1 \rightarrow 1,25 \geq 1$$

2.3 Terceiro Caso

Nesse caso, serão utilizados os seguintes valores para as variáveis:

- $\beta = 1$
- $u = 1,0$
- $\Delta t = 0,025$
- $\Delta x = 0,2$

- $\delta = 0,0,\ 0,5\ e\ 0,5(1-C)$
- $t_{max} = 0,5$

Segue abaixo os resultados:

```
EQUACAO DE ADVECCAO-DIFUSAO: TOTALMENTE IMPLÍCITO

JMAX= 21 TMAX= 1.00 T1= 1.00 T2= 0.00
BETA= 1.00000 DELTA= 0.00000
DELTA_T= 0.02500 DELTA_X= 0.20000
S= 0.06250 ALPHA= 0.10000
U= 1.00000 C= 0.12500

t= 0.000 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.977 0.551 0.083 0.013 0.002 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.025 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.997 0.961 0.593 0.157 0.034 0.007 0.001 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.050 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.995 0.950 0.628 0.223 0.061 0.015 0.003 0.001 0.000 0.000 0.000 0.000 0.000
t= 0.075 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.993 0.941 0.657 0.282 0.093 0.026 0.007 0.002 0.000 0.000 0.000 0.000 0.000
t= 0.100 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.991 0.936 0.683 0.334 0.126 0.040 0.011 0.003 0.001 0.000 0.000 0.000 0.000
t= 0.125 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.989 0.932 0.705 0.381 0.161 0.057 0.018 0.005 0.001 0.000 0.000 0.000 0.000
t= 0.150 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.988 0.930 0.725 0.423 0.196 0.077 0.026 0.008 0.002 0.001 0.000 0.000 0.000
t= 0.175 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.986 0.929 0.743 0.462 0.231 0.098 0.036 0.012 0.004 0.001 0.000 0.000 0.000
t= 0.200 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.985 0.929 0.758 0.496 0.265 0.120 0.048 0.017 0.006 0.002 0.001 0.000 0.000
t= 0.225 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.997 0.984 0.930 0.772 0.528 0.299 0.144 0.061 0.023 0.008 0.003 0.001 0.000 0.000
t= 0.250 TN = 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.997 0.983 0.930 0.785 0.557 0.331 0.169 0.076 0.031 0.011 0.004 0.001 0.000 0.000
...
t= 0.500 TN = 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.995 0.982 0.947 0.872 0.746 0.583 0.414 0.267 0.157 0.085 0.043 0.020 0.008 0.000

t= 0.500 TE = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.987 0.943 0.829 0.624 0.376 0.171 0.057 0.013 0.002 0.000 0.000 0.000

EMQ= 0.0453
```

***Figura 10**: Resultados usando Método Implícito e $\delta = 0,0$*

```
EQUACAO DE ADVECCAO-DIFUSAO: TOTALMENTE IMPLÍCITO

JMAX= 21 TMAX= 1.00 T1= 1.00 T2= 0.00
BETA= 1.00000 DELTA= 0.50000
DELTA_T= 0.02500 DELTA_X= 0.20000
S= 0.06250 ALPHA= 0.10000
U= 1.00000 C= 0.12500

t= 0.000 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.556 0.062 0.007 0.001 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.025 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.605 0.122 0.020 0.003 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.050 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.649 0.181 0.038 0.007 0.001 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.075 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.688 0.237 0.060 0.013 0.002 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.100 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.723 0.291 0.085 0.021 0.004 0.001 0.000 0.000 0.000 0.000 0.000
t= 0.125 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.753 0.342 0.114 0.031 0.007 0.002 0.000 0.000 0.000 0.000 0.000
t= 0.150 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.781 0.391 0.145 0.044 0.011 0.003 0.001 0.000 0.000 0.000 0.000
t= 0.175 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.805 0.437 0.177 0.059 0.017 0.004 0.001 0.000 0.000 0.000 0.000
t= 0.200 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.827 0.480 0.211 0.075 0.023 0.006 0.002 0.000 0.000 0.000 0.000
t= 0.225 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.846 0.521 0.245 0.094 0.031 0.009 0.002 0.001 0.000 0.000 0.000
t= 0.250 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.863 0.559 0.280 0.115 0.040 0.013 0.004 0.001 0.000 0.000 0.000
...
t= 0.500 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.958 0.817 0.599 0.376 0.206 0.099 0.043 0.017 0.006 0.002 0.000

t= 0.500 TE = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.987 0.943 0.829 0.624 0.376 0.171 0.057 0.013 0.002 0.000 0.000 0.000

EMQ= 0.0159
```

***Figura 11**: Resultados usando Método Implícito e $\delta = 0,5$*

```
EQUACAO DE ADVECCAO-DIFUSAO: TOTALMENTE IMPLÍCITO

JMAX= 21 TMAX= 1.00 T1= 1.00 T2= 0.00
BETA= 1.00000 DELTA= 0.43750
DELTA_T= 0.02500 DELTA_X= 0.20000
S= 0.06250 ALPHA= 0.10000
U= 1.00000 C= 0.12500

t= 0.000 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.997 0.555 0.065 0.008 0.001 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.025 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.995 0.603 0.127 0.021 0.003 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.050 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.993 0.646 0.187 0.041 0.008 0.001 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.075 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.992 0.683 0.244 0.064 0.014 0.003 0.001 0.000 0.000 0.000 0.000 0.000
t= 0.100 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.991 0.716 0.298 0.091 0.023 0.005 0.001 0.000 0.000 0.000 0.000 0.000
t= 0.125 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.990 0.746 0.349 0.121 0.034 0.009 0.002 0.000 0.000 0.000 0.000 0.000
t= 0.150 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.990 0.772 0.397 0.152 0.048 0.013 0.003 0.001 0.000 0.000 0.000 0.000
t= 0.175 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.989 0.795 0.442 0.185 0.064 0.019 0.005 0.001 0.000 0.000 0.000 0.000
t= 0.200 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.989 0.815 0.484 0.219 0.082 0.026 0.007 0.002 0.000 0.000 0.000 0.000
t= 0.225 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.990 0.834 0.523 0.254 0.101 0.035 0.011 0.003 0.001 0.000 0.000 0.000
t= 0.250 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.990 0.850 0.559 0.289 0.123 0.045 0.015 0.004 0.001 0.000 0.000 0.000
...
t= 0.500 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.994 0.944 0.805 0.597 0.383 0.215 0.108 0.049 0.020 0.008 0.003 0.000

t= 0.500 TE = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.987 0.943 0.829 0.624 0.376 0.171 0.057 0.013 0.002 0.000 0.000 0.000

EMQ= 0.0189
```

***Figura 12**: Resultados usando Método Implícito e $\delta = 0,5(1 - C)$*

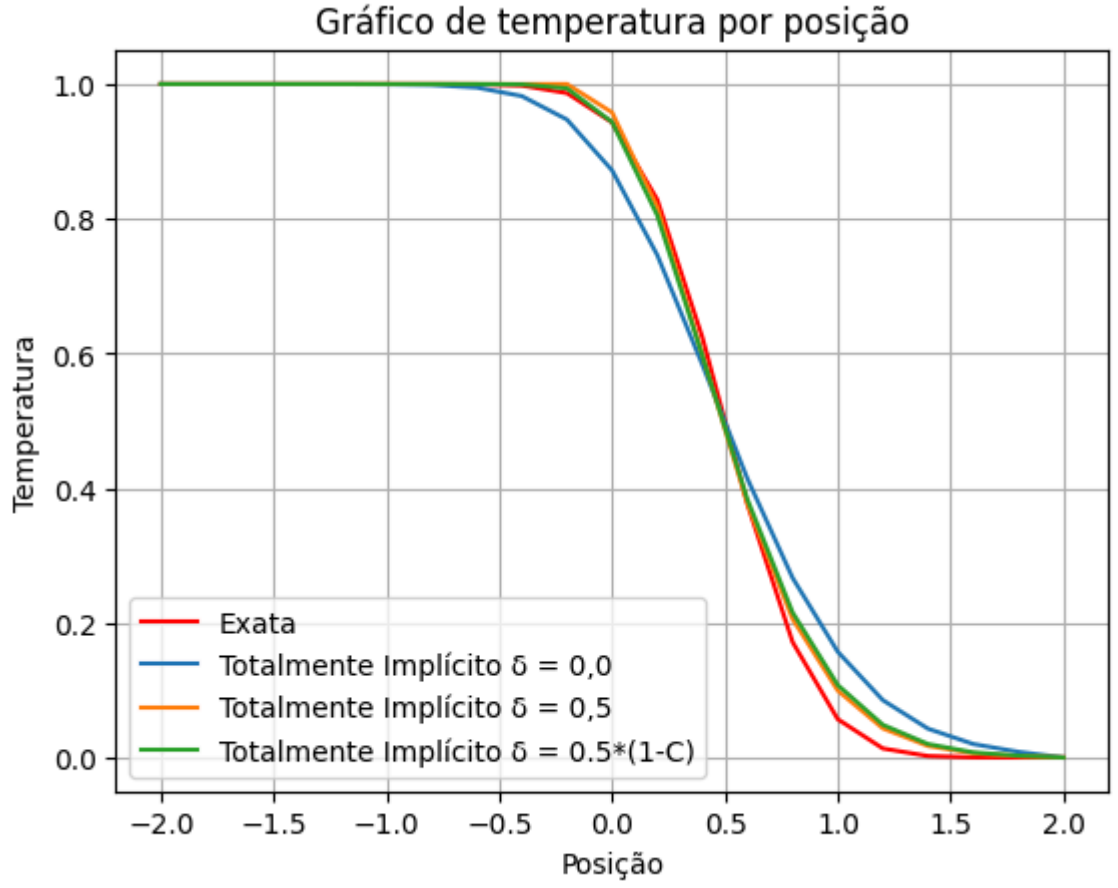

***Figura 13**: Gráfico com todos os resultados do caso 3*

Entre os resultados obtidos, o método totalmente implícito com $\delta = 0,0$ foi o que teve os piores resultados, com um $EMQ = 0,0453$. Isso pode ser explicado novamente pois, com esse valor de delta, o termo advectivo é aproximado usando uma diferença avançada, o que acaba inserindo uma difusividade artificial no método que não existe na vida real. Isso causa o esquema a divergir um pouco mais da solução ideal do que os outros métodos. Já o que aproxima o termo advectivo com uma diferença centrada ($\delta = 0,5$) teve

o melhor resultado, já que não sofre desta difusividade artificial. O esquema com $\delta = 0,5(1 - C)$ ficou novamente como um resultado intermediário entre os dois outros métodos.

2.4 Quarto Caso

Nesse caso, serão utilizados os seguintes valores para as variáveis:

- $\beta = 1/2 = 0,5$
- $u = 1,0$
- $\Delta t = 0,025$
- $\Delta x = 0,2$
- $\delta = 0,0,\ 0,5\ e\ 0,5(1 - C)$
- $t_{max} = 0,5$

Segue abaixo os resultados:

```
 EQUACAO DE ADVECCAO-DIFUSAO: CRANK-NICOLSON

 JMAX= 21 TMAX= 1.00 T1= 1.00 T2= 0.00
 BETA= 0.50000 DELTA= 0.00000
 DELTA_T= 0.02500 DELTA_X= 0.20000
 S= 0.06250 ALPHA= 0.10000
 U= 1.00000 C= 0.12500

 t= 0.000 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.974 0.556 0.088 0.007 0.001 0.000 0.000 0.000 0.000 0.000 0.000 0.000
 t= 0.025 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.997 0.956 0.600 0.166 0.027 0.003 0.000 0.000 0.000 0.000 0.000 0.000 0.000
 t= 0.050 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.995 0.944 0.636 0.234 0.055 0.009 0.001 0.000 0.000 0.000 0.000 0.000 0.000
 t= 0.075 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.993 0.936 0.666 0.295 0.087 0.019 0.003 0.001 0.000 0.000 0.000 0.000 0.000
 t= 0.100 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.991 0.931 0.692 0.348 0.123 0.033 0.007 0.001 0.000 0.000 0.000 0.000 0.000
 t= 0.125 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.989 0.928 0.713 0.396 0.160 0.050 0.013 0.003 0.001 0.000 0.000 0.000 0.000
 t= 0.150 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.987 0.927 0.733 0.438 0.197 0.069 0.020 0.005 0.001 0.000 0.000 0.000 0.000
 t= 0.175 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.985 0.926 0.750 0.476 0.234 0.091 0.029 0.008 0.002 0.000 0.000 0.000 0.000
 t= 0.200 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.997 0.984 0.927 0.765 0.511 0.270 0.115 0.040 0.012 0.003 0.001 0.000 0.000 0.000
 t= 0.225 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.997 0.983 0.927 0.779 0.542 0.305 0.140 0.054 0.017 0.005 0.001 0.000 0.000 0.000
 t= 0.250 TN = 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.997 0.982 0.929 0.791 0.570 0.338 0.166 0.068 0.024 0.007 0.002 0.001 0.000 0.000
 ...
 t= 0.500 TN = 1.000 1.000 1.000 1.000 1.000 1.000 0.999 0.995 0.982 0.948 0.876 0.755 0.594 0.421 0.267 0.152 0.078 0.036 0.015 0.006 0.000

 t= 0.500 TE = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.987 0.943 0.829 0.624 0.376 0.171 0.057 0.013 0.002 0.000 0.000 0.000

 EMQ= 0.0428
```

***Figura 14**: Resultados usando Crank-Nicolson e $\delta = 0,0$*

```
EQUACAO DE ADVECCAO-DIFUSAO: CRANK-NICOLSON

JMAX= 21 TMAX= 1.00 T1= 1.00 T2= 0.00
BETA= 0.50000 DELTA= 0.50000
DELTA_T= 0.02500 DELTA_X= 0.20000
S= 0.06250 ALPHA= 0.10000
U= 1.00000 C= 0.12500

t= 0.000 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.559 0.062 0.004 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.025 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.611 0.124 0.014 0.001 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.050 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.657 0.184 0.031 0.004 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.075 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.697 0.242 0.052 0.008 0.001 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.100 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.733 0.297 0.078 0.015 0.002 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.125 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.764 0.350 0.107 0.024 0.004 0.001 0.000 0.000 0.000 0.000 0.000
t= 0.150 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.792 0.401 0.138 0.035 0.007 0.001 0.000 0.000 0.000 0.000 0.000
t= 0.175 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.816 0.448 0.172 0.050 0.011 0.002 0.000 0.000 0.000 0.000 0.000
t= 0.200 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.838 0.493 0.207 0.066 0.017 0.004 0.001 0.000 0.000 0.000 0.000
t= 0.225 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.857 0.534 0.243 0.085 0.024 0.006 0.001 0.000 0.000 0.000 0.000
t= 0.250 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.874 0.573 0.280 0.106 0.032 0.008 0.002 0.000 0.000 0.000 0.000
...
t= 0.500 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.964 0.833 0.613 0.378 0.198 0.088 0.034 0.012 0.004 0.001 0.000

t= 0.500 TE = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.987 0.943 0.829 0.624 0.376 0.171 0.057 0.013 0.002 0.000 0.000 0.000

EMQ= 0.0120
```

***Figura 15**: Resultados usando Crank-Nicolson e $\delta = 0,5$*

```
EQUACAO DE ADVECCAO-DIFUSAO: CRANK-NICOLSON

JMAX= 21 TMAX= 1.00 T1= 1.00 T2= 0.00
BETA= 0.50000 DELTA= 0.43750
DELTA_T= 0.02500 DELTA_X= 0.20000
S= 0.06250 ALPHA= 0.10000
U= 1.00000 C= 0.12500

t= 0.000 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.997 0.558 0.066 0.004 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.025 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.994 0.609 0.130 0.016 0.001 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.050 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.992 0.654 0.191 0.034 0.004 0.000 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.075 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.991 0.692 0.250 0.057 0.009 0.001 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.100 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.990 0.726 0.306 0.084 0.017 0.003 0.000 0.000 0.000 0.000 0.000 0.000
t= 0.125 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.989 0.756 0.358 0.114 0.027 0.005 0.001 0.000 0.000 0.000 0.000 0.000
t= 0.150 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.989 0.782 0.407 0.147 0.040 0.009 0.002 0.000 0.000 0.000 0.000 0.000
t= 0.175 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.989 0.805 0.454 0.181 0.055 0.013 0.003 0.000 0.000 0.000 0.000 0.000
t= 0.200 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.989 0.826 0.497 0.217 0.072 0.019 0.004 0.001 0.000 0.000 0.000 0.000
t= 0.225 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.989 0.844 0.537 0.253 0.092 0.027 0.007 0.001 0.000 0.000 0.000 0.000
t= 0.250 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.989 0.860 0.574 0.289 0.114 0.036 0.010 0.002 0.000 0.000 0.000 0.000
...
t= 0.500 TN = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.994 0.949 0.819 0.610 0.387 0.209 0.097 0.040 0.014 0.005 0.001 0.000

t= 0.500 TE = 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 0.998 0.987 0.943 0.829 0.624 0.376 0.171 0.057 0.013 0.002 0.000 0.000 0.000

EMQ= 0.0145
```

***Figura 16**: Resultados usando Crank-Nicolson e $\delta = 0,5(1 - C)$*

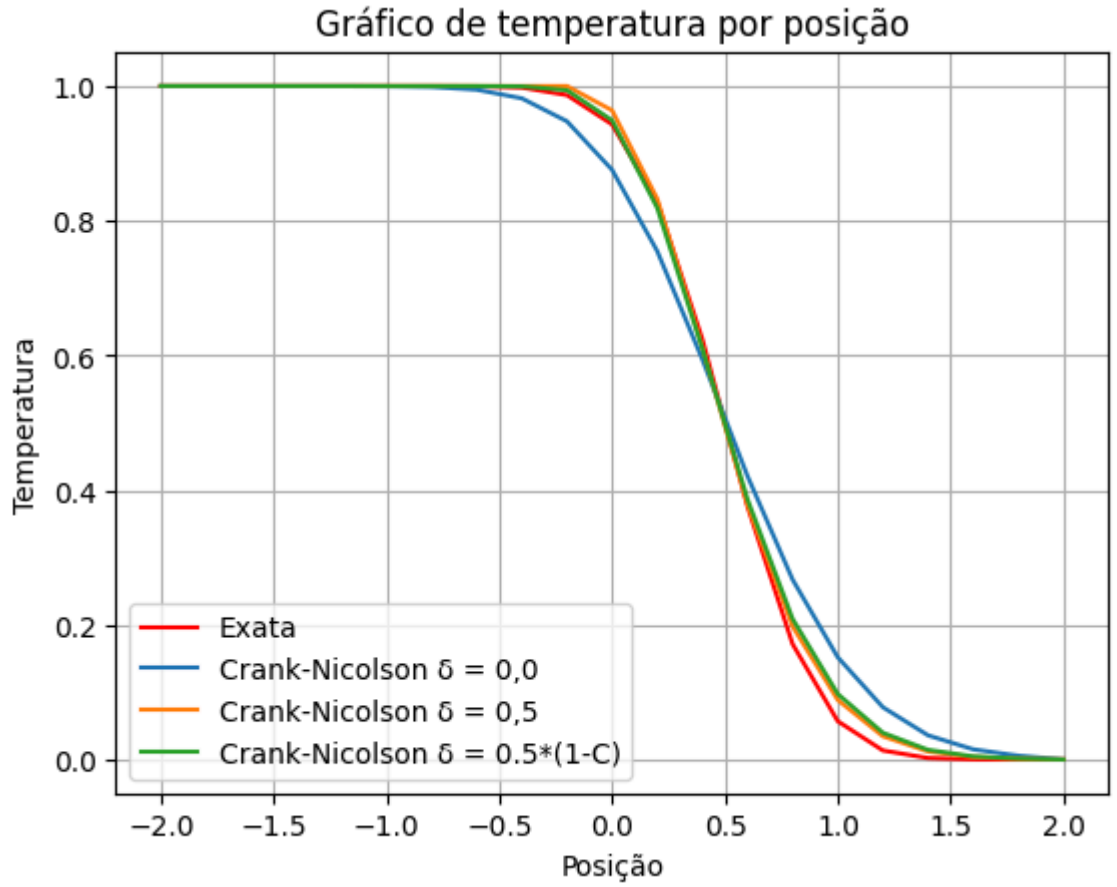

***Figura 17**: Gráfico com todos os resultados do caso 4*

Novamente, o esquema mais acurado foi com $\delta = 0,5$. O esquema com $\delta = 0,0$ foi o pior devido a difusividade numérica artificial já explicada em casos anteriores. Por fim, o esquema com $\delta = 0,5(1 - C)$ também ficou com uma acurácia intermediária.

3. Conclusão

Em síntese, a aplicação dos métodos numéricos revelou-se confiável na predição do comportamento da solução do problema estudado. Os métodos implícitos, especialmente aqueles com $\delta = 0,5$, apresentaram excelentes valores de Erro Médio Quadrático (EMQ), destacando-se por sua robustez, mesmo sem a necessidade de atender a condições de estabilidade. Por outro lado, os métodos explícitos, embora limitados por suas restrições de estabilidade em algumas configurações físicas, demonstraram uma notável precisão nas previsões quando aplicáveis. Esses achados demonstram a eficácia dos métodos numéricos no tratamento de uma variedade de cenários, cada um com suas vantagens particulares.

4. Código

```
"""
  Trabalho 2 - Métodos Númericos para Equações Diferenciais
  Aluno: Victor Luis Teixeira Reis - 202110049511
"""

import numpy as np
import matplotlib.pyplot as plt

def initial_condition(x):
  """Implementação da condição inicial do problema"""
  if x < 0.0:
    return T_1
  elif x > 0.0:
    return T_2
  else:
    return (T_1 + T_2)/2

def analytic_solution (x, t):
  """Calcula a solução analítica do problema em uma posição x e instante
t"""
  serie_length = 10 # Números maiores do que esse tamanho não resultaram
em grandes diferenças na aproximação
  partial_sum = 0.0

  for k in range(1, serie_length+1):
    partial_sum +=
(1/(2*k-1))*np.exp((-alpha*((2*k-1)**2)*(np.pi**2)*t)/(L**2))*np.sin((2*
k-1)*((np.pi*(x-u*t))/L))

  return 0.5 - (2/np.pi)*partial_sum

def plot_solution (x_grid, aprox_solution, exact_solution):
  """Plota um gráfico com uma linha representando a solução analítica e
pontos representando a solução numérica"""
```

```
  plt.plot(x_grid, exact_solution, label="Exata", color="red")
  plt.scatter(x_grid, aprox_solution, label="Numérico", marker="*")

  plt.xlabel('Posição')
  plt.ylabel('Temperatura')
  plt.title('Gráfico de temperatura por posição')
  plt.grid(True)
  plt.legend()
  plt.show()


def plot_all_solution (x_grid, exact_solution, solutions):
  """Plota um gráfico com várias linhas representando cada tipo de
solução"""
  plt.plot(x_grid, exact_solution, label="Exata", color="red")

  for solution in solutions:
    plt.plot(x_grid, solution["array"], label=solution["label"])

  plt.xlabel('Posição')
  plt.ylabel('Temperatura')
  plt.title('Gráfico de temperatura por posição')
  plt.grid(True)
  plt.legend()
  plt.show()

def print_temperature (time, temperature, solution_type="TN"):
  """Função para printar as temperaturas em cada posição da malha em um
instante de tempo"""
  temp_string = ""

  for each_temp in temperature:
    temp_string += f'{each_temp:.3f} '

  print(f't= {time:.3f} {solution_type} = {temp_string}')

def EMQ (aprox_solution, exact_solution, size):
  """Calcula o erro médio quadrado"""
  emq_result = 0.0

  for j in range(size):
    emq_result += (aprox_solution[j] - exact_solution[j])**2

  emq_result = emq_result / size
  emq_result = np.sqrt(emq_result)
```

```
  return emq_result


def tdma(a, b, c, temperature, size, beta, delta):
  """Algoritmo de Thomas"""
  d = np.zeros(size)
  new_temperature = np.zeros(size)
  p = np.zeros(size)
  q = np.zeros(size)

  d[0] = temperature[0]
  d[size-1] = temperature[size-1]

  for j in range(1, size-1):
    d[j] = (1-beta)*((1-delta)*C+S)*temperature[j-1] +
(1-(1-beta)*((1-2*delta)*C+2*S))*temperature[j] +
(1-beta)*(-delta*C+S)*temperature[j+1]

  # Condições de contorno
  p[0] = 0.0
  q[0] = T_1
  p[size-1] = 0.0
  q[size-1] = T_2

  for j in range(1, size-1):
    p[j] = b[j] / (a[j] - c[j]*p[j-1])
    q[j] = (d[j] + c[j]*q[j-1]) / (a[j] - c[j]*p[j-1])

  new_temperature[size-1] = q[size-1]

  for j in range(size-2, -1, -1):
    new_temperature[j] = p[j]*new_temperature[j+1] + q[j]

  return new_temperature

def solve_partial_equation(temperature, final_t, size, beta, delta):
  """Função principal para a resolução do problema"""

  if beta == 0.5:
    print(f'EQUACAO DE ADVECCAO-DIFUSAO: CRANK-NICOLSON\n')
  elif beta == 1.0:
    print(f'EQUACAO DE ADVECCAO-DIFUSAO: TOTALMENTE IMPLÍCITO\n')
  elif beta == 0 and delta == 0.5:
    print(f'EQUACAO DE ADVECCAO-DIFUSAO: FTCS (EXPLÍCITO)\n')

    if 0 > C**2 or C**2 > 2*S or 2*S > 1:
```

```
      return print("Condição de estabilidade não satisfeita (0 <= C² <= 2s <= 1)!")

  elif beta == 0 and delta == 0:
    print(f'EQUACAO DE ADVECCAO-DIFUSAO: UPWIND (EXPLÍCITO)\n')

    if C + 2*S > 1:
      return print("Condição de estabilidade não satisfeita (C + 2s <= 1)!")

  elif beta == 0 and delta == 0.5*(1-C):
    print(f'EQUACAO DE ADVECCAO-DIFUSAO: LAX-WENDROFF (EXPLÍCITO)\n')

    if (C**2)+ 2*S > 1:
      return print("Condição de estabilidade não satisfeita (C² + 2s <= 1)!")

  print(f'JMAX= {size:.0f} TMAX= {T_1:.2f} T1= {T_1:.2f} T2= {T_2:.2f}')
  print(f'BETA= {beta:.5f} DELTA= {delta:.5f}')
  print(f'DELTA_T= {delta_t:.5f} DELTA_X= {delta_x:.5f}')
  print(f'S= {S:.5f} ALPHA= {alpha:.5f}')
  print(f'U= {u:.5f} C= {C:.5f}\n')

  # Expressões que calculam os elementos aj, bj, cj,
  a_j = 1 + beta*((1-2*delta)*C + 2*S)
  b_j = beta*(-delta*C + S)
  c_j = beta*((1-delta)*C + S)
  iteration = 0

  a = np.array([a_j for _ in range(size)])
  b = np.array([b_j for _ in range(size)])
  c = np.array([c_j for _ in range(size)])

  for j in np.arange(0.0, final_t+delta_t, delta_t):
    temperature = tdma(a, b, c, temperature, size, beta, delta)

    # Lógica para pular algumas linhas de impressão quando o resultado é muito extenso
    if j == final_t and iteration >= max_print_size:
      print("...")
      print_temperature(j, temperature)

    if iteration < max_print_size:
      print_temperature(j, temperature)

    iteration += 1
```

```
  print() # Só para pular uma linha
  print_temperature(final_t, exact_final_temperature,
solution_type="TE")

  quadratic_error = EMQ(temperature, exact_final_temperature, size)

  print(f'\nEMQ= {quadratic_error:.4f}\n')

  plot_solution(x_grid, temperature, exact_final_temperature)

  return temperature

########################################################################
#  Variáveis Gerais
########################################################################

u = 1.0 # Velocidade de advecção
alpha = 0.1 # Difusividade térmica
delta_x, delta_t = 0.2, 0.025
final_t = 0.5

C = (u*delta_t)/delta_x
S = (alpha*delta_t)/(delta_x**2)

T_1 = 1.0
T_2 = 0.0

X_1 = -2.0
X_2 = 2.0

L = X_2 - X_1

size = int((L/delta_x) + 1)
x_grid = np.linspace(X_1, X_2, size)
max_print_size = 11 # Número máximo de linhas que serão printadas antes
de pular as linhas restantes

initial_temperature = np.array([initial_condition(x) for x in x_grid])
exact_final_temperature = np.array([analytic_solution(x, final_t) for x
in x_grid])

########################################################################
#  1 ° Caso (beta=0.0, u=0.0, delta_x=0.2449, delta_t=0.1, final_t=1.0)
########################################################################
```

```
# ftcs_solution = solve_partial_equation(initial_temperature, final_t,
size, 0.0, 0.5)
# upwind_solution = solve_partial_equation(initial_temperature, final_t,
size, 0.0, 0.0)
# lax_wendroff_solution = solve_partial_equation(initial_temperature,
final_t, size, 0.0, 0.5*(1-C))

# solutions = [
#   {
#     "label": "FTCS",
#     "array": ftcs_solution
#   },
#   {
#     "label": "Upwind",
#     "array": upwind_solution
#   },
#   {
#     "label": "Lax-Wendroff",
#     "array": lax_wendroff_solution
#   }
# ]
# plot_all_solution(x_grid, exact_final_temperature, solutions)

########################################################################
#  2 ° Caso (beta=0.0, u=0.5, delta_x=0.2, delta_t=0.05, final_t=1.0)
########################################################################

# ftcs_solution = solve_partial_equation(initial_temperature, final_t,
size, 0.0, 0.5)
# upwind_solution = solve_partial_equation(initial_temperature, final_t,
size, 0.0, 0.0)
# lax_wendroff_solution = solve_partial_equation(initial_temperature,
final_t, size, 0.0, 0.5*(1-C))

# solutions = [
#   {
#     "label": "FTCS",
#     "array": ftcs_solution
#   },
#   {
#     "label": "Upwind",
#     "array": upwind_solution
#   },
#   {
#     "label": "Lax-Wendroff",
#     "array": lax_wendroff_solution
```

```
#   }
# ]
# plot_all_solution(x_grid, exact_final_temperature, solutions)


#########################################################################
#  3 ° Caso (beta=1.0, u=1.0, delta_x=0.2, delta_t=0.025, final_t=0.5)
#########################################################################
# total_implicit_solution_1 =
solve_partial_equation(initial_temperature, final_t, size, 1.0, 0.0)
# total_implicit_solution_2 =
solve_partial_equation(initial_temperature, final_t, size, 1.0, 0.5)
# total_implicit_solution_3 =
solve_partial_equation(initial_temperature, final_t, size, 1.0,
0.5*(1-C))

# solutions = [
#   {
#     "label": "Totalmente Implícito δ = 0,0",
#     "array": total_implicit_solution_1
#   },
#   {
#     "label": "Totalmente Implícito δ = 0,5",
#     "array": total_implicit_solution_2
#   },
#   {
#     "label": "Totalmente Implícito δ = 0.5*(1-C)",
#     "array": total_implicit_solution_3
#   }
# ]
# plot_all_solution(x_grid, exact_final_temperature, solutions)

#########################################################################
#  4 ° Caso (beta= 0.5, u=1.0, delta_x=0.2, delta_t=0.025, final_t=0.5)
#########################################################################
crank_solution_1 = solve_partial_equation(initial_temperature, final_t,
size, 0.5, 0.0)
crank_solution_2 = solve_partial_equation(initial_temperature, final_t,
size, 0.5, 0.5)
crank_solution_3 = solve_partial_equation(initial_temperature, final_t,
size, 0.5, 0.5*(1-C))

solutions = [
  {
    "label": "Crank-Nicolson δ = 0,0",
    "array": crank_solution_1
```

```
  },
  {
    "label": "Crank-Nicolson δ = 0,5",
    "array": crank_solution_2
  },
  {
    "label": "Crank-Nicolson δ = 0.5*(1-C)",
    "array": crank_solution_3
  }
]

plot_all_solution(x_grid, exact_final_temperature, solutions)
```

5. Referências

1. **Notas de aula de Métodos Numéricos de Equações Diferenciais do professor Helio Pedro Amaral Souto.** Disponível em: <https://ead.iprj.uerj.br/moodle/pluginfile.php/8304/mod_resource/content/21/Notas_de_Aula.pdf>. Acesso em: 16 jun. 2024
2. **C. A. J. Fletcher. Computational Techniques for Fluid Dynamics, Volume 1. SpringerVerlag, 1991.**
3. **M. Necati Özisik. Heat Conduction. John Wiley & Sons, 1980.**
4. **Segundo trabalho computacional de Métodos Numéricos de Equações Diferenciais do professor Helio Pedro Amaral Souto.** Disponível em: <https://ead.iprj.uerj.br/moodle/mod/assign/view.php?id=22499>. Acesso em: 16 jun. 2024