Proceeding Series of the Brazilian Society of Computational and Applied Mathematics, v. 11, n. 1, 2025.

Trabalho apresentado no XLIII CNMAC, Centro de Convenções do Armação Resort - Porto de Galinhas - PE, 2024

Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

Verificação do Método Pseudoespectral de Chebyshev com C-N para Reconstrução do Tensor de Condutividade Térmica usando a Heurística de Evolução Diferencial Melhorada

José Laércio Doricio[1], Milena A. L. Brandão[2], Vanda M. Luchesi[3]
ICENP/UFU, Ituiutaba, MG

Resumo. Este trabalho visa verificar numericamente a reconstrução do tensor de condutividade térmica em um modelo térmico transiente bidimensional com condições de contorno mistas, modelado analiticamente por equações diferenciais parciais resolvidas numericamente usando o método pseudoespectral de Chebyshev (MPC), para discretização das variáveis espaciais, em conjunto com o método de Crank-Nicolson (C-N), para discretização da variável temporal. A reconstrução da função de condutividade térmica é resolvida pelo método da Evolução Diferencial Melhorada em Paralelo (EDM). A EDM é uma variante do algoritmo de otimização Evolução Diferencial, ambos pertencem à classe de algoritmos evolucionários, que são técnicas de otimização baseadas em princípios inspirados no processo de evolução natural. Neste trabalho, EDM é empregado para obter a solução ótima para o tensor de condutividade térmica. Para verificar o código utilizamos o Método das Soluções Manufaturadas.

Palavras-chave. Método Espectral, Reconstrução de Parâmetros, Evolução Diferencial Melhorada.

1 Introdução

Neste artigo, mostramos os resultados da verificação de um método de solução numérica do problema de reconstrução do tensor de condutividade térmica em um modelo térmico transiente bidimensional com condições de contorno mistas. Para tal, tomamos um modelo de condução de calor bidimensional transiente com tensor de condutividade variável nas direções x e y, visando o estudo da condutividade de materiais isotrópicos e anisotrópicos. Sendo assim, consideraremos a seguinte EDP que descreve o problema no domínio $\Gamma \times (0,t]$, $t > 0$, com $\Gamma = (0,l_1) \times (0,l_2) \subseteq \mathbb{R}^2$.

$$C\frac{\partial T}{\partial t}(x,y,t) = \nabla(K(x,y)\nabla T(x,y,t)) + g(x,y,t) \quad \text{na região} \quad \Gamma \times (0,t), \tag{1}$$

onde

$$K(x,y) = \begin{bmatrix} k_{11}(x,y) & k_{12}(x,y) \\ k_{21}(x,y) & k_{22}(x,y) \end{bmatrix} \tag{2}$$

é o tensor de condutividade térmica, onde $k(x,y) = k_{11}(x,y) = k_{22}(x,y)$ e $k_{12}(x,y) = k_{21}(x,y) = 0$. Na fronteira, consideramos as condições de contorno do tipo Robin dadas por:

$$-k(0,y)\frac{\partial T}{\partial x}(0,y,t) + h_1(y)(T(0,y,t) - f_1(y,t)) = 0 \quad \text{para} \quad y \in (0,l_2), t \in (0,t), \tag{3}$$

[1]jldoricio@ufu.br
[2]milenabrandao@ufu.br
[3]vanda.luchesi@ufu.br

$$k(l_1, y)\frac{\partial T}{\partial x}(l_1, y, t) + h_2(y)(T(l_1, y, t) - f_2(y, t)) = 0 \quad \text{para} \quad y \in (0, l_2), t \in (0, t), \tag{4}$$

$$-k(x, 0)\frac{\partial T}{\partial y}(x, 0, t) + h_3(x)(T(x, 0, t) - f_3(x, t)) = 0 \quad \text{para} \quad x \in (0, l_1), t \in (0, t), \tag{5}$$

$$k(x, l_2)\frac{\partial T}{\partial y}(x, l_2, t) + h_4(x)(T(x, l_2, t) - f_4(x, t)) = 0 \quad \text{para} \quad x \in (0, l_1), t \in (0, t), \tag{6}$$

$$T(x, y, 0) = T_0 \quad \text{na região} \quad \Gamma, \tag{7}$$

onde $T(x, y, t)$ é a temperatura, $C > 0$ é a constante de capacidade térmica, h_i com $i = 1, ..., 4$ são funções de transferência de calor e f_i com $i = 1, ..., 4$ são funções de fluxo de calor na fronteira. A discretização espacial da EDP na Eq. (1) é feita por meio do Método Pseudoespectral de Chebyshev (MPC) que tem precisão exponencial e para a discretização temporal aplicamos o método Crank Nicolson (C-N) devido à sua simplicidade, precisão de segunda ordem e principalmente por ser estável, veja mais detalhes em [1].

1.1 Método Pseudoespectral de Chebyshev (MPC)

Para aplicar o MPC, tomamos em $\Gamma = (0, l_1) \times (0, l_2) \subseteq \mathbb{R}^2$ uma malha composta por $(n+1) \times (n+1)$ pontos que são transformados em $(n+1)$ pontos de Chebyshev Gauss-Lobatto x_i e y_i nas direções horizontal e vertical, respectivamente. Para cada direção, seja $\hat{x}_i$ um ponto de colocação otimizada, denominado ponto Chebyshev Gauss-Lobatto:

$$\hat{x}_i = \frac{1}{2}\left[1 - cos(i\pi/n)\right], \; i = 0, 1, \cdots, n. \tag{8}$$

Desta forma, temos pontos não igualmente espaçados e vamos associá-los aos polinômios de Chebyshev $\phi(\hat{x}_i)$. Nos métodos espectrais aproximamos a função desejada, $u(x)$ (unidimensional), pela série de Chebyshev truncada. Para o método pseudo-espectral tomamos esses valores nos pontos de colocação $\hat{x}_i$. Assim

$$u_N(\hat{x}_i) \approx \sum_{i=0}^{N} c_i \phi(\hat{x}_i) \quad \text{com} \quad c_i = \frac{\langle u, \phi_i \rangle_w}{\langle \phi_i, \phi_i \rangle_w}, \tag{9}$$

onde a representação espectral é feita pelos coeficientes c_i e w é uma função peso que produza a ortogonalidade do polinômio de Chebyshev em relação ao produto interno. Para a diferenciação de Chebyshev seja $p(x)$ um polinômio interpolador e dado um conjunto de valores $u(\hat{x}_j) = p(\hat{x}_j)$, $j = 1, \cdots, N$, de uma função unidimensional $u(x)$, a derivada discreta em relação a x no ponto de colocação $\hat{x}_i$ pode ser aproximada por $u'(\hat{x}_i) \approx p'(\hat{x}_i)$. Como esta operação é linear podemos escrever $u'(\hat{x}_i) = e_j^T D u^T$ onde e_j é o $(j+1)$-ésimo vetor canônico de ordem N+1. A matriz $D = [d_{ij}]$, com $0 \leq i, j \leq N$, é chamada de matriz diferenciação. Seja D a matriz de diferenciação de Chebyshev $(n+1) \times (n+1)$ podemos construir a seguinte decomposição (por linha ou por coluna):

$$D = \begin{bmatrix} r_0^T \\ r_1^T \\ \vdots \\ r_n^T \end{bmatrix} = [d_0, d_1, \ldots, d_n], \quad r_i, d_i \in \mathbb{R}^{n+1}, \; i = 0, \ldots, n. \tag{10}$$

Aplicando o MPC no termo que possui a diferencial em relação a variável x da parte principal da EDP da Eq. (1) obtemos

$$\mathbf{T}_j^x := \begin{bmatrix} \dfrac{\partial}{\partial x}\left(k(x_0,y_j)\dfrac{\partial T(x_0,y_j,t)}{\partial x}\right) \\ \vdots \\ \dfrac{\partial}{\partial x}\left(k(x_n,y_j)\dfrac{\partial T(x_n,y_j,t)}{\partial x}\right) \end{bmatrix} \approx D(\mathbf{K}_j^x \dot{\mathbf{T}}_j), \quad j=0,\ldots,n, \tag{11}$$

onde $\mathbf{K}_j^x$ é a seguinte matriz diagonal

$$\mathbf{K}_j^x = \text{diag}(k(x_0,y_j),k(x_1,y_j),\ldots,k(x_n,y_j)), \quad j=0,1,\ldots,n. \tag{12}$$

Usando a Eq. (11) e $\dot{\mathbf{T}}_j \approx D\mathbf{T}_j$, obtemos

$$\frac{\partial}{\partial x}\left(k(x_i,y_j)\frac{\partial T(x_i,y_j,t)}{\partial x}\right) \approx r_i^T(\mathbf{K}_j^x D\mathbf{T}_j), \quad i=1,\ldots,n-1,\ j=0,\ldots,n. \tag{13}$$

Aplicando o MCP nas condições de fronteira Eqs.(3)–(6), para os pontos (x_0,y_j), $j=0,\ldots,n$, e para (x_n,y_j), $j=0,\ldots,n$, com aproximação similar dada na Eq.(11) obtemos

$$\begin{aligned} \mathbf{T}_j^x &\approx \sum_{i=0}^{n} d_i k(x_i,y_j)\frac{\partial T(x_i,y_j,t)}{\partial x} \\ &= d_0 k(x_0,y_j)\frac{\partial T(x_0,y_j,t)}{\partial x} + \sum_{i=1}^{n-1} d_i k(x_i,y_j)\frac{\partial T(x_i,y_j,t)}{\partial x} + d_n k_{11}(x_n,y_j)\frac{\partial T(x_n,y_j,t)}{\partial x} \\ &= \left[h_1(y_j)d_0 e_1^T + D_1\check{\mathbf{K}}_j^x D_2 - h_2(y_j)d_n e_{n+1}^T\right]\mathbf{T}_j - d_0 h_1(y_j)f_1(y_j,t) + d_n h_2(y_j)f_2(y_j,t) \\ &\doteq \mathbf{A}_j\mathbf{T}_j + \mathbf{b}_j, \end{aligned} \tag{14}$$

onde

$$\mathbf{A}_j = h_1(y_j)d_0 e_1^T + D_1\check{\mathbf{K}}_j^x D_2 - h_2(y_j)d_n e_{n+1}^T, \quad \mathbf{b}_j = -d_0 h_1(y_j)f_1(y_j,t) + d_n h_2(y_j)f_2(y_j,t), \tag{15}$$

$$D_1 = [d_1,\ldots,d_{n-1}], \quad D_2 = \begin{bmatrix} r_1^T \\ \vdots \\ r_{n-1}^T \end{bmatrix}, \quad \check{\mathbf{K}}_j^x = \text{diag}(k(x_1,y_j),\ldots,k(x_{n-1},y_j)). \tag{16}$$

Analogamente, uma equação similar é obtida ao aplicar o MPC ao termo que possui a diferencial em relação a variável y da parte principal da EDP da Eq. (1).

2 Evolução Diferencial Melhorada

O algoritmo de Evolução Diferencial [7], [11] modificado usando o conceito de evolução com conjuntos embaralhados [5] é chamado de Evolução Diferencial Melhorada (EDM). A EDM é implementada em processamento paralelo da seguinte forma: (I) no processador mestre, uma população inicial é gerada aleatoriamente e depois dividida em k subconjuntos que são posteriormente distribuídos entre um máximo de k processadores, (II) o código de Evolução Diferencial é executado em cada subconjunto sequencialmente em processadores individuais até que um critério de parada predeterminado seja alcançado, (III) depois disso, as subpopulações são agregadas no processador mestre, passam por um processo de embaralhamento e são novamente divididas em novas subpopulações. O fluxograma que representa a implementação do método de otimização EDM em paralelo é ilustrado na Fig. 1. Mais detalhes sobre EDM podem ser vistos nos trabalhos de [3] e [6].

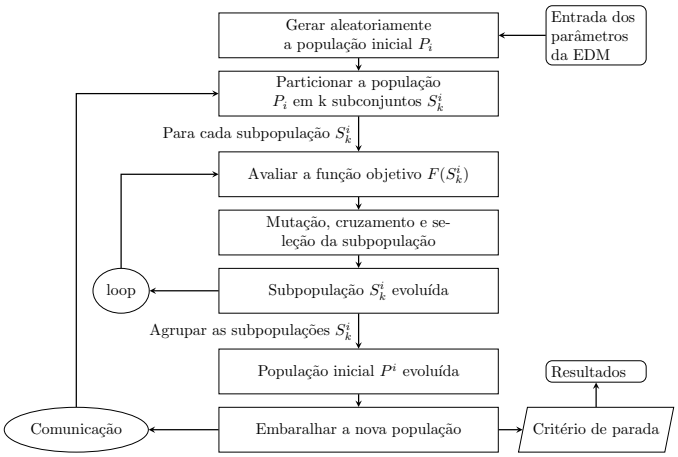

Figura 1: Fluxograma do Método de Evolução Diferencial Melhorada. Fonte: dos autores.

3 Verificação de Código

O Método da Solução Manufaturada (MSM) desenvolvido por [10], [8] e estudado por [4] e [9] modifica as equações diferenciais governantes conduzindo a solução para uma solução manufaturada pré-estabelecida. Considere a EDP escrita como $\mathscr{L}(Q,\xi,t) = \mathscr{F}(\xi,t)$, com condições de contorno dadas por $\mathscr{B}(Q,\xi,t) = \mathscr{G}(\xi,t)$, onde Q é a solução da EDP, ξ é o domínio computacional e, $\mathscr{F}(\xi,t)$ e $\mathscr{G}(\xi,t)$ são termos fonte que dependem do domínio computacional e do tempo, mas não dependem da solução Q. Cada variável do escoamento, representada em Q, é atribuída à alguma função contínua, por exemplo trigonométrica, exponencial, polinômial ou combinações dessas funções. Esse conjunto de soluções é representado por $Q^e(x,y)$. Estas soluções manufaturadas são substituídas na equação diferencial parcial contínua para gerar os termos fonte:

$$\mathscr{F}(\xi,t) = \mathscr{L}(Q^e,\xi,t) \ , \quad \mathscr{G}(\xi,t) = \mathscr{B}(Q^e,\xi,t) \ . \tag{17}$$

Os termos fonte $\mathscr{F}$ e $\mathscr{G}$ modificam as equações governantes para:

$$\mathscr{L}(Q,\xi,t) = \mathscr{L}(Q^e,\xi,t) \ , \quad \mathscr{B}(Q,\xi,t) = \mathscr{B}(Q^e,\xi,t) \ . \tag{18}$$

Devido à introdução dos termos fonte, a solução numérica da Eq. (18) deve convergir para a solução exata Q^e. Isto ocorrerá se a implementação e a discretização das equações governantes estiverem corretas. Além disso, a ordem de convergência de malha deve ser verificada comparando a diferença entre a solução convergida Q^c com a solução exata Q^e utilizando uma norma qualquer. O erro $\mathscr{E}(\ Q^c, Q^e\ ) = \parallel Q^c - Q^e \parallel$ é utilizado para calcular a ordem de convergência de malha do método numérico, através da Eq. (19). Se a ordem numérica coincidir com a ordem teórica, então pode-se dizer que o código está verificado e livre de erros de implementação. As soluções manufaturadas podem ser quaisquer funções diferenciáveis de classe C^n, com n grande o suficiente de tal forma que os termos de erro de truncamento da expansão em séries de Taylor, das variáveis discretizadas, não sejam nulos. Para calcular a ordem de convergência de malha do método numérico, considere uma variável de interesse $\mathscr{I}$ representando, por exemplo, a temperatura, velocidade, etc. Se a solução exata $\mathscr{I}^e$ é conhecida (gerada pela solução manufaturada), então, para um epaçamento de malha h, a variável $\mathscr{I}$ pode ser expandida em séries de Taylor gerando a seguinte série infinita: $\mathscr{I}(h) = \mathscr{I}^e + A_1h^1 + A_2h^2 + ... + A_nh^n + ...$ Se a implementação de um código numérico for de ordem 2 por exemplo, então $A_1 = 0$ e $A_2 \neq 0$. Truncando a série de Taylor a partir do

DOI: 10.5540/03.2025.011.01.0511

termo m tem-se: $\mathscr{I}(h) = \mathscr{I}^e + Ah^m$, onde A e m são incógnitas. Se a implementação do código for de ordem 2, então o coeficiente $m \to 2$ quando a malha é refinada. Se uma solução exata $\mathscr{I}^e$ é conhecida então exitem duas incógnitas nessa equação. Essas incógnitas são os coeficientes A e m. Logo, são necessárias soluções em duas malhas distintas para determinar o valor dessas incógnitas, ou seja, $\mathscr{I}(h_1) = \mathscr{I}^e + Ah_1^m$, $\mathscr{I}(h_2) = \mathscr{I}^e + Ah_2^m$. A ordem de precisão é obtida dividindo uma equação pela outra e, em seguida, isolando m nessa equação:

$$\left(\frac{h_2}{h_1}\right)^m = \frac{\|\mathscr{I}(h_2) - \mathscr{I}^e\|}{\|\mathscr{I}(h_1) - \mathscr{I}^e\|} \quad \longrightarrow \quad m = \frac{\log \frac{\|\mathscr{I}(h_2) - \mathscr{I}^e\|}{\|\mathscr{I}(h_1) - \mathscr{I}^e\|}}{\log \frac{h_2}{h_1}} \ . \tag{19}$$

A solução manufaturada usada para verificação código desenvolvido nesse trabalho é dada por:

$$\begin{array}{ll}
T(x,y,t) = e^{-t/2}(sen(xy) + cos(xy)), & K(x,y) = 20sen(xy) + 20cos(xy), \\
g(x,y,t) = (x-1/2)^2 + (y-1/2)^2, & h1(y) = 3sen(y), \\
h2(y) = 3cos(y), & h3(x) = 3sen(x), \\
h4(x) = 3cos(x), & f1(y,t) = -20ye^{-t/2} + 3sen(y)e^{-t/2}, \\
\multicolumn{2}{l}{f2(y,t) = 20ye^{-t/2}(cos^2 y - sen^2 y)) + 3cos(y)e^{-t/2}(sen(y) + cos(y)),} \\
\multicolumn{2}{l}{f3(x,t) = -20xe^{-t/2} + 3sen(x)e^{-t/2},} \\
\multicolumn{2}{l}{f4(x,t) = 20xe^{-t/2}(cos^2 x - sen^2 x) + 3cos(x)e^{-t/2}(sen(x) + cos(x)),} \\
\multicolumn{2}{l}{s(x,y,t) = e^{-t/2}(-(1/2)(sen(xy) - cos(xy)) + 80(cos(xy)y^2 sen(xy)} \\
\multicolumn{2}{r}{+cos(xy)x^2 sen(xy))) - (x-1/2)^2 - (y-1/2)^2,}
\end{array} \tag{20}$$

onde $s(x,y,t)$ é o termo fonte da solução manufaturada que é adicionado na Eq. (1) após g(x,y,t). O teste de convergência de malha é feito usando malhas com tamanhos $h \times h$ com $h = 4, 5, 6, ..., 16$ e plotado em escala logarítmica, mostrado pela Fig. 2. Verifica-se que o método numérico implementado com discretização espacial feita com o MPC e com discretização temporal usando C-N possui ordem de decaimento super-geométrica do erro e está entre as ordens de referência 12 a 15, chegando a 16, com padrão descrito por [2]. Há um limitante para o decaimento do erro em torno de 10^{-9}, que está ligado aos erros numéricos do método para resolução do sistema linear gerado pelo C-N. O resultado permite avaliar que o código está livre de erros de implementação. A próxima etapa de verificação é resolver o problema inverso para determinar $K(x,y)$, onde a solução manufaturada gerada, com $s(x,y,t)$ incorporada à $g(x,y,t)$, será utilizada para testar o método de busca da $K(x,y)$ usando a EDM.

4 Reconstrução de $K(x,y)$ usando o Método Heurístico EDM

Para determinar $K(x,y)$ usando a EDM deve-se ter $T(x,y,0)$ e $T(x,y,t_f)$ que podem ser determinados analiticamente ou medidos experimentalmente em laboratório. Como conhecemos o valor de $T(x,y,t)$, dado pela solução manufaturada, temos esses dois valores. Então o processo de resolução do problema para determinar $K(x,y)$ é um processo de minimização modelado por:

$$Min\ E = \|T(x,y,t_f) - \tilde{T}(x,y,t_f)\|, \quad s.a \ \ K_{min} < K(x,y) < K_{max} \ . \tag{21}$$

onde $\tilde{T}(x,y,t_f)$ é a solução numérica da EDP, $T(x,y,t_f)$ é a solução analítica da EDP. O processo de busca de $K(x,y)$ é feito pela EDM em Paralelo. O problema descrito pela Eq. (1) foi resolvido na malha $h \times h$ com $h = 9$ pontos totalizando um domínio de busca com 81 pontos para a EDM. Os gráficos na Fig. 3 mostram a comparação entre $T(x,y,1)$ e $\tilde{T}(x,y,1)$ com $x = 0, ..., 1$ e $y = 0, ..., 1$ e entre $K(x,y)$ e $\tilde{K}(x,y)$ no mesmo domínio em t = 1. O gráfico na Fig. 2 à direita mostra a magnitude do erro entre $T(x,y,1)$ e $\tilde{T}(x,y,1)$. Já os gráficos da Fig. 4 mostram a magnitude do erro entre $K(x,y)$ e $\tilde{K}(x,y)$ em t = 1, onde o gráfico da direita nessa figura mostra o erro no interior

DOI: 10.5540/03.2025.011.01.0511

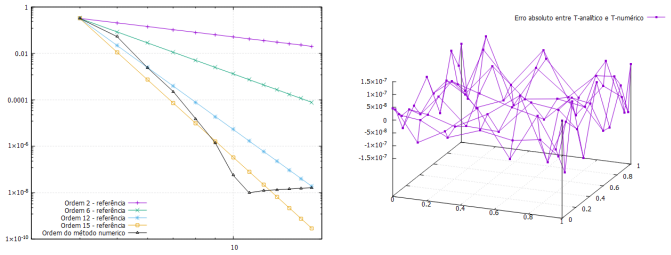

Figura 2: À esquerda: Ordem de convergência de malha. À direita: magnitude do erro entre T analítico e T numérico em t=1. Fonte: dos autores.

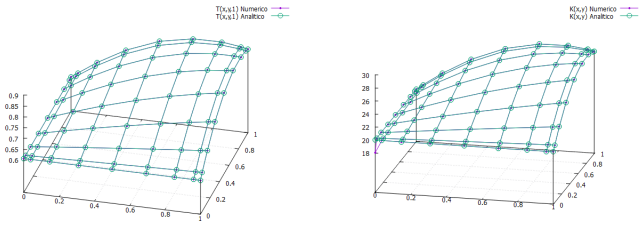

Figura 3: Comparação entre a solução anal. e num. para T e K em t=1. Fonte: dos autores.

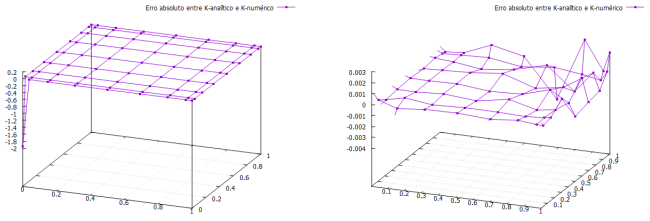

Figura 4: Diferença entre K analítico e K numérico em t=1. Fonte: dos autores.

do domínio. O tempo de execução do algoritmo para reconstruir $K(x, y)$ usando EDM em Paralelo foi de 55,34 horas usando 10 processadores, sendo 6 físicos e 4 virtuais em uma arquitetura de memória compartilhada. A diferença entre T analítico e T numérico gerou erro pontual máximo de $1,5 \times 10^{-7}$ que está próximo ao valor esperado conforme indicado pelo gráfico de ordem de convergência de malha apresentado na Fig. 2. O erro cometido ao reconstruir a condutividade térmica K no interior do domínio foi pontualmente de no máximo $3,0 \times 10^{-3}$, no entanto no ponto $(x, y) = (0, 0)$, que é um ponto de quina, um erro pontual de magnitude $1,9$ foi obtido. Apesar do valor enorme do erro nesse ponto, a temperatura T está muito precisa. Esse fenômeno ocorre porque no ponto $(0, 0)$ as derivadas parciais de T em relação a x e a y valem zero para a solução manufaturada construída. Se considerarmos as condições de contorno dadas pelas Eqs. (3) - (6), vemos que K multiplica a derivada parcial de T e portanto, nessa condição, qualquer valor numérico para K é válido. Isto significa fisicamente que se não houver variação térmica em Γ, aquecimento ou resfriamento, fica impossível determinar ou reconstruir o valor de K usando a modelagem dada pela Eq. (1). O tempo computacional para execução do código está muito alto, porém esse problema foi resolvido com uma técnica de interpolação com funções radiais RBF

DOI: 10.5540/03.2025.011.01.0511

discutido em mais detalhes em outro trabalho, onde conseguimos reduzir o tempo computacional para 8,4 minutos e além disso também amenizou o problema de reconstrução da K em pontos onde as derivadas parciais de T se anulam em Γ.

5 Considerações Finais

A verificação do algoritmo pode ser realizada por meio do Método das Soluções Manufaturadas. O procedimento indicou que o método de evolução temporal C-N em conjunto com a discretização espacial feita por MPC, para a Equação Diferencial Parcial que modela a condução térmica transiente bidimensional com condições de contorno mistas dadas pelas Eqs. (1) e (3) - (6) possui uma excelente ordem de convergência de malha, do tipo hiper-geométrica. A reconstrução do tensor de condutividade $K(x, y)$ foi obtido com boa precisão exceto em pontos onde as derivadas parciais de $T(x, y, t)$ se anulam em Γ. O tempo computacional de execução do algoritmo não foi bom, mas já era esperado porque a quantidade de variáveis de busca é muito grande, forçando muito a heurística de busca, no entanto conseguimos reduzir muito esse tempo, para minutos, usando uma técnica descrita em outro trabalho.

Referências

[1] E. Boos, V. M. Luchesi e F. S. V. Bazán. "Thermal conductivity reconstruction method with application in a face milling operation". Em: **International Journal of Numerical Methods for Heat & Fluid Flow** 29.5 (2021), pp. 681–711.

[2] J. P. Boyd. **Chebyshev and Fourier Spectral Methods**. DOVER Publications, Inc., 2000.

[3] M. A. L. Brandão, J. L. Doricio e S. de F. P. Saramago. **Otimização Avançada: Evolução Diferencial Melhorada em Paralelo**. Vol. 99. Notas em Matemática Aplicada. São Carlos: SBMAC, 2024. ISBN: 9786586388282.

[4] C. O. E. Burg e V. K. Murali. "Efficient Code Verification using the Residual Formulation of the Method of Manufactured Solutions". Em: 34th AIAA Fluid Dynamics Conference. 2004, pp. 1–13.

[5] Q. A. Duan, V. K. Gupta e S. Sorooshian. "A parallel differential evolution algorithm". Em: **Journal of Optimization Theory and Applications** 76.3 (1993).

[6] S. de F. P. Saramago, J. L. Doricio e M. A. L. Brandão. "Optimum design of 3r robot manipulador by using improved differential evolution implemented in parallel computation". Em: **Brazilian Electronic Journal of Mathematics** 1.2 (2020), pp. 83–103.

[7] K. V. Price, R. M. Storn e J. A. Lampinen. **Differential Evolution: A Practical Approach to Global Optimization**. Springer, 2005.

[8] P. J. Roache. "Verification of Codes and Calculations". Em: **AIAA Journal** 36 (1998), pp. 696–702.

[9] H. G. da Silva, M. A. F. Medeiros e L. F. de Souza. "A Verification Test for a Direct Numerical Simulation Code that uses a High Order Discretization Scheme". Em: 18th International Congress of Mechanical Engineering. 2005.

[10] S. Steinberg e P. J. Roache. "Symbolic Manipulation and Computational Fluid Dynamics". Em: **AIAA Journal** 22 (1984), pp. 1390–1394.

[11] R. Storn e K. Price. "Differential evolution: a simple and efficient heuristic for global optimization over continuous spaces". Em: **Journal of Global Optimization** 11(4) (1997), pp. 341–359.

