REVISTA INTERDISCIPLINAR DE PESQUISA EM ENGENHARIA

CILAMCE 2016

XXXVII IBERIAN LATIN AMERICAN CONGRESS ON COMPUTATIONAL METHODS IN ENGINEERING

BRASÍLIA - DF - BRAZIL

MÉTODO SEM MALHA COM DIFERENÇAS FINITAS GENERALIZADAS PARA SOLUÇÃO DE EQUAÇÕES DIFERENCIAIS PARCIAIS

Augusto César Albuquerque Ferreira

Paulo Marcelo Vieira Ribeiro

augusto.cesaralbuquerque@ufpe.br

paulo.vribeiro@ufpe.br

Universidade Federal de Pernambuco – UFPE

Centro de Tecnologia e Geociências

Departamento de Engenharia Civil

Rua Acadêmico Hélio Ramos, s/n, 50740-530, Pernambuco, Recife, Brasil

Resumo. *O desenvolvimento nas técnicas de soluções de EDPs vem apresentando enorme evolução nas últimas décadas, em parte com a popularização dos métodos numéricos, que são aplicados a problemas complexos, onde soluções analíticas não são possíveis. O Método dos Elementos Finitos (MEF) recebe destaque nessa evolução e é atualmente o método numérico mais difundido para solução de EDPs. Por outro lado, o MEF necessita de uma etapa prévia de pré-processamento, que consiste na elaboração da malha do domínio, em um procedimento conhecido com discretização. Dependendo da forma do domínio a ser solucionado, essa etapa pode demandar tempo considerável na preparação do problema numérico (pré-processamento). Pesquisas associadas a métodos sem malha vêm ganhando importância nas últimas décadas, pois evitam esta demora. O Método das Diferenças Finitas Generalizadas (MDFG) representa uma categoria interessante nesse aspecto, com uma extensão do procedimento tradicional (MDF) ao caso onde um grid ortogonal é desnecessário. Este aspecto traz em si a tendência de resoluções das EDPs com domínios cada vez mais complexos. Desta forma, o presente trabalho busca desenvolver códigos no MATLAB que busquem boa eficiência na resolução do problema em duas dimensões de Laplace para diferentes configurações de fronteira pelo MDFG.*

Palavras-chave: *Métodos sem malha, Diferenças Finitas Generalizadas, Elementos Finitos, Laplace*

1 INTRODUÇÃO

Atualmente o Método dos Elementos Finitos (MEF) é usado em larga escala. Nele, domínios complexos podem gerar grande desgaste computacional na preparação do problema numérico (pré-processamento), além de apresentar eventuais desvantagens em termos de resultados em função da qualidade da malha. Outra abordagem para a solução de EDPs esta no Método das Diferenças Finitas (MDF), que é utilizado com muito sucesso em problemas unidimensionais com malhas regulares ou irregulares. No entanto, para problemas bidimensionais é limitado a malhas regulares, causando assim problemas na ocorrência de fronteiras irregulares. Assim, métodos sem malha gradativamente ganham importância neste cenário de resoluções de EDPs. Um desses métodos é uma extensão do MDF, chamado de Método das Diferenças Finitas Generalizadas (MDFG), este possui a vantagem de não necessariamente ter que trabalhar com grids ortogonais como no MDF. Na Fig. 1, uma comparação da etapa de pré-processamento para o MEF e o MDFG.

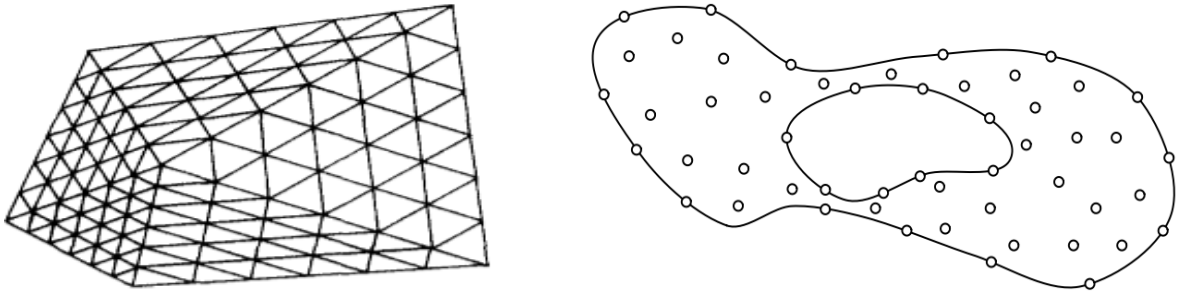

Figura 1. Exemplo de malha para o MEF e distribuição irregular dos pontos para o MDFG

O MDFG traz a vantagem de dar continuidade na simplicidade das aproximações de derivadas parciais como no MDF e ao mesmo tempo de se utilizar nós no domínio sem relações entre si, dando grande liberdade na escolha dos nós. Esta ideia de se criar nuvens irregulares não é nova e já era usada no inicio da década de 70 com Jensen (1972), onde utilizava 06 pontos em grids irregulares e a aproximação das derivadas feita através da série de Taylor expandida em duas dimensões. Posteriormente percebeu-se a geração de singularidades frequentes neste tipo de análise, devido à possibilidade de formação de grids mal condicionados. Desta forma, foram propostas formas de se evitar tais singularidades. Perrone e Kao (1975) sugeriram um método chamado segmento de oito critérios, onde o grid é formado por um nó de cada octante em torno de um nó central e ocorre um processo de média para gerar os coeficientes das fórmulas em diferenças finitas. Liszka e Orkisz (1980) estabeleceram um critério de quatro quadrantes, buscando os dois nós mais próximos do nó central em cada quadrante. Posteriormente foi feito um estudo com equações elípticas cujas soluções exatas são conhecidas, com os resultados obtidos percebeu-se que o critério mais apropriado para colocação de nós no grid é o critério de quatro quadrantes (Benito et al.,2001). Mais tarde, são analisados resultados numéricos através da obtenção do resíduo em cada ponto escolhido, a qual deve se aproximar de zero, tais resíduos crescem quando ocorre o mal condicionamento do grid. Esse crescimento é combatido com o aumento do numero de pontos no grid (Gavete et al., 2003).

Dentro deste contexto, o presente trabalho busca criar um gerador automático de pontos que evite formações de singularidades, isso sendo feito através da adoção de uma distância

mínima entre os pontos do domínio. Assim é desenvolvido códigos no MATLAB que busquem boa eficiência para a resolução da equação diferencial parcial de Laplace com diferentes formas de domínio. Por fim, Comparam-se seus resultados e sua convergência com soluções exatas, quando possível, e com o MEF, estabelecendo um estudo sobre a consistência e estabilidade do método apresentado.

2 APRESENTAÇÃO DO PROBLEMA

O modelo matemático geral para o fluxo de calor (ou simplesmente equação do calor) em duas dimensões é dada pela equação (Nagle et al., 2012):

$$\frac{\partial u}{\partial t} = \beta \Delta u + P(x, y, t). \tag{1}$$

Onde β é a constante de difusividade do material, u representa a função temperatura ao longo do domínio considerado, P estar associado à presença de fontes internas ao sistema, e finalmente Δu é conhecido como laplaciano.

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}. \tag{2}$$

Quando a temperatura alcança um estado estacionário, ou seja, quando u não depende da temperatura, e não existem fontes internas, então $\frac{\partial u}{\partial t} = 0$ e a temperatura satisfaz a equação de Laplace:

$$0 = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}. \tag{3}$$

3 DESENVOLVIMENTO DO MÉTODO E PROGRAMA

3.1 Método das Diferenças Finitas Generalizadas

Qualquer função diferençável $f(x, y)$ para um dado domínio pode ser expandida através da série de Taylor em duas direções. A expansão da série sobre um ponto (x_o, y_o) truncada em seu sexto termo é dada por:

$$f(x, y) = f(x_o, y_o) + h\frac{\partial f_o}{\partial x} + k\frac{\partial f_o}{\partial y} + \frac{h^2}{2}\frac{\partial^2 f_o}{\partial x^2} + \frac{k^2}{2}\frac{\partial^2 f_o}{\partial y^2} + hk\frac{\partial^2 f_o}{\partial x \partial y}. \tag{4}$$

Onde $h = x - x_o$ e $k = y - y_o$

Em notação matricial:

$$\begin{bmatrix} h_1 & k_1 & \frac{h^2{}_1}{2} & \frac{k^2{}_1}{2} & h_1k_1 \\ h_2 & k_2 & \frac{h^2{}_2}{2} & \frac{k^2{}_2}{2} & h_2k_2 \\ h_3 & k_3 & \frac{h^2{}_3}{2} & \frac{k^2{}_3}{2} & h_3k_3 \\ h_4 & k_4 & \frac{h^2{}_4}{2} & \frac{k^2{}_4}{2} & h_4k_4 \\ h_5 & k_5 & \frac{h^2{}_5}{2} & \frac{k^2{}_1}{2} & h_5k_5 \end{bmatrix} \begin{bmatrix} \frac{\partial f_o}{\partial x} \\ \frac{\partial f_o}{\partial y} \\ \frac{\partial^2 f_o}{\partial x^2} \\ \frac{\partial^2 f_o}{\partial y^2} \\ \frac{\partial^2 f_o}{\partial x \partial y} \end{bmatrix} = \begin{bmatrix} f_1 - f_o \\ f_2 - f_o \\ f_3 - f_o \\ f_4 - f_o \\ f_5 - f_o \end{bmatrix}. \tag{5}$$

$$\begin{bmatrix} \frac{\partial f_o}{\partial x} \\ \frac{\partial f_o}{\partial y} \\ \frac{\partial^2 f_o}{\partial x^2} \\ \frac{\partial^2 f_o}{\partial y^2} \\ \frac{\partial^2 f_o}{\partial x \partial y} \end{bmatrix} = \begin{bmatrix} h_1 & k_1 & \frac{h^2{}_1}{2} & \frac{k^2{}_1}{2} & h_1k_1 \\ h_2 & k_2 & \frac{h^2{}_2}{2} & \frac{k^2{}_2}{2} & h_2k_2 \\ h_3 & k_3 & \frac{h^2{}_3}{2} & \frac{k^2{}_3}{2} & h_3k_3 \\ h_4 & k_4 & \frac{h^2{}_4}{2} & \frac{k^2{}_4}{2} & h_4k_4 \\ h_5 & k_5 & \frac{h^2{}_5}{2} & \frac{k^2{}_1}{2} & h_5k_5 \end{bmatrix}^{-1} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{bmatrix} - \begin{bmatrix} h_1 & k_1 & \frac{h^2{}_1}{2} & \frac{k^2{}_1}{2} & h_1k_1 \\ h_2 & k_2 & \frac{h^2{}_2}{2} & \frac{k^2{}_2}{2} & h_2k_2 \\ h_3 & k_3 & \frac{h^2{}_3}{2} & \frac{k^2{}_3}{2} & h_3k_3 \\ h_4 & k_4 & \frac{h^2{}_4}{2} & \frac{k^2{}_4}{2} & h_4k_4 \\ h_5 & k_5 & \frac{h^2{}_5}{2} & \frac{k^2{}_1}{2} & h_5k_5 \end{bmatrix}^{-1} \begin{bmatrix} f_o \\ f_o \\ f_o \\ f_o \\ f_o \end{bmatrix}. \tag{6}$$

A ideia principal do método é substituir as derivadas parciais referentes a terceira e quarta linha da Eq. (6) na Eq. (3), isso para cada ponto interior ao domínio escolhido. Ao substituir geram-se coeficientes referentes aos pontos internos e de contorno, em forma matricial temos:

$$\begin{bmatrix} \alpha_{1,1} & \cdots & \alpha_{1,ni} \\ \vdots & \ddots & \vdots \\ \alpha_{ni,1} & \cdots & \alpha_{ni,ni} \end{bmatrix} \begin{bmatrix} f_1 \\ \vdots \\ f_{ni} \end{bmatrix} = - \begin{bmatrix} \gamma_1 \\ \vdots \\ \gamma_{ni} \end{bmatrix}. \tag{7}$$

$$[coef.dos\ pontos\ internos][f] = -[coef.dos\ pontos\ de\ contorno]$$

Obtendo-se um sistema de equações com solução única, cujas incógnitas são os valores de f e com ni (numero de pontos internos) equações.

3.2 Programa

Inicialmente o programa irá gerar pontos randômicos no contorno e na parte interior ao contorno. Cabe ao usuário escolher tais quantidades de pontos em ambas as regiões. Percebeu-se que a geração de pontos mal condicionados podem gerar problemas com singularidades, isso pode ocorrer com a formação de agrupamento de pontos muito próximos entre si ou vazios criados ao longo do domínio. A Fig. 2 ilustra o problema:

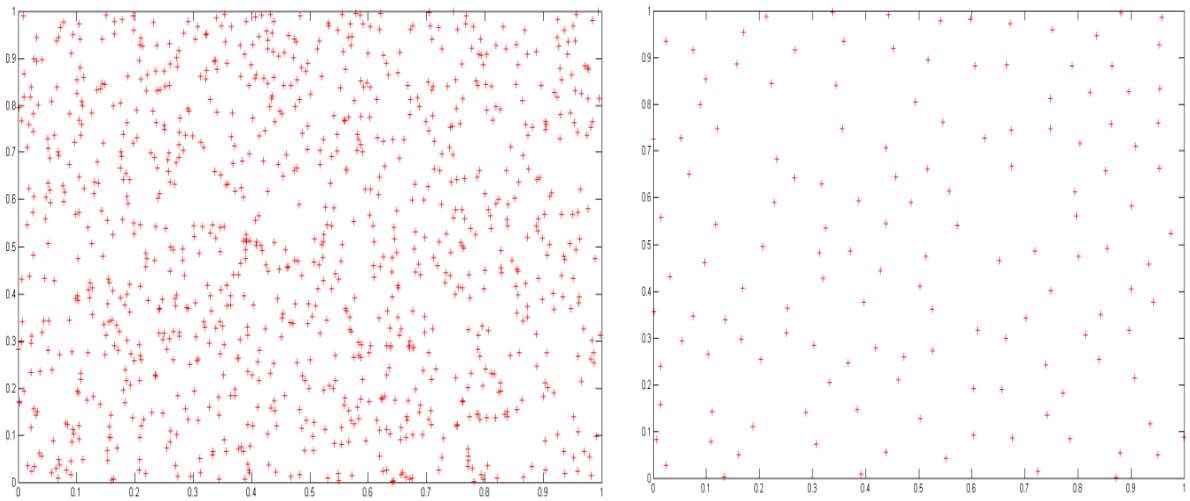

Figura 2. Exemplos de má formação dos pontos

Por isso foi criado uma função no MATLAB onde o usuário estabelece uma tolerância mínima de distancia entre os pontos, de forma que é possível ir ajustando tal distância a fim de se obter uma solução sem problemas. A função também estabelece uma boa distribuição dos pontos randômicos ao longo do domínio a fim de evitar vazios.

Após a etapa de pré-processamento, com os pontos devidamente ajustados, é feito a aproximação das derivadas usando a série de Taylor (6). A rotina criada busca para cada pivô os cinco pontos vizinhos mais próximos a ele (ver Fig. 3), melhorando assim os valores das derivadas aproximadas.

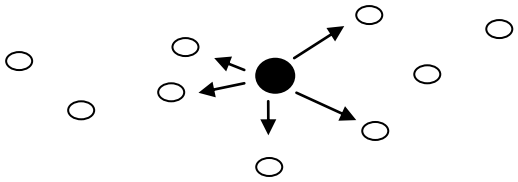

Figura 3. Pivô de preto buscando os 05 pontos vizinhos mais próximos.

Por fim são calculadas as temperaturas nos pontos escolhidos através da Eq. (7).

4 APLICAÇÕES E RESULTADOS

4.1 Comparação gráfica de região triangular

O exemplo a seguir é um triângulo de base e altura igual a 1,00 m com uma temperatura de 200 °C na base do modelo numérico e nos demais lados à temperatura é nula. As figuras a seguir mostram o pré-processamento e o comportamento das temperaturas em seu regime estacionário. A Fig. 4 foi rodada no código criado com uma tolerância mínima entre todos os pontos de 0,05 m, possuindo 143 pontos totais e 101 equações geradas (numero de pontos internos). Na Fig. 5 foi executada uma malha em elementos finitos produzida com o programa ANSYS 14.5. Foram empregados 78 elementos quadrilaterais do tipo PLANE 55 - Thermal solid.

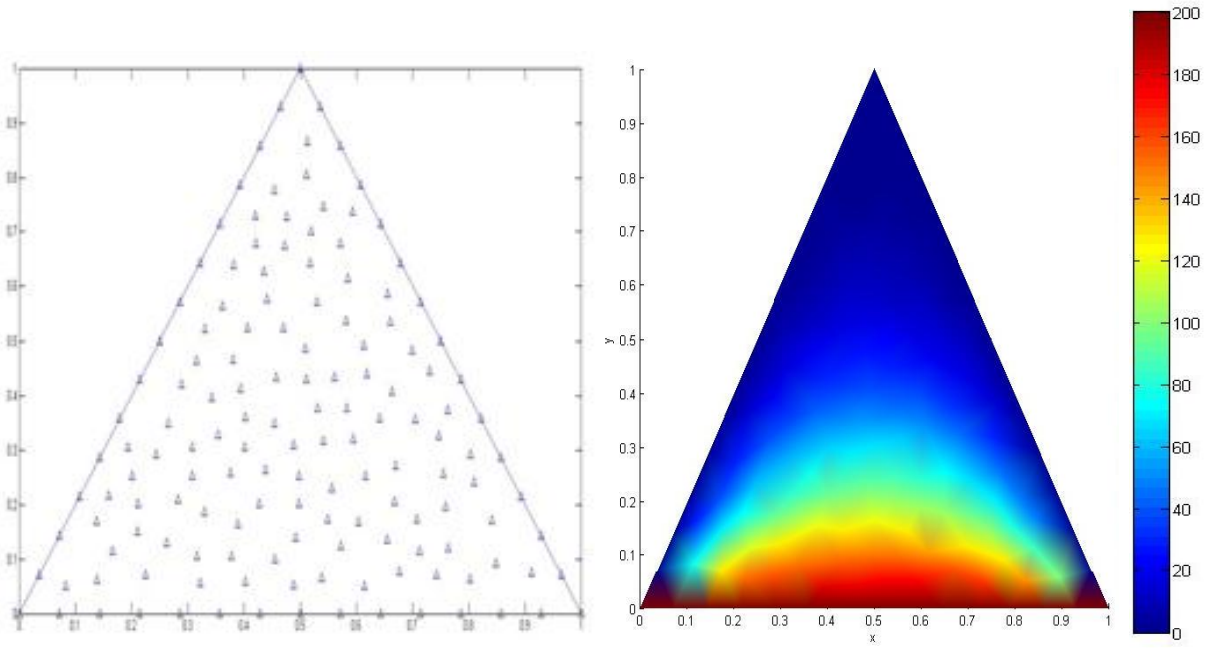

Figura 4. Distribuição dos pontos no MDFG e resultados das temperaturas (°C) na região triangular

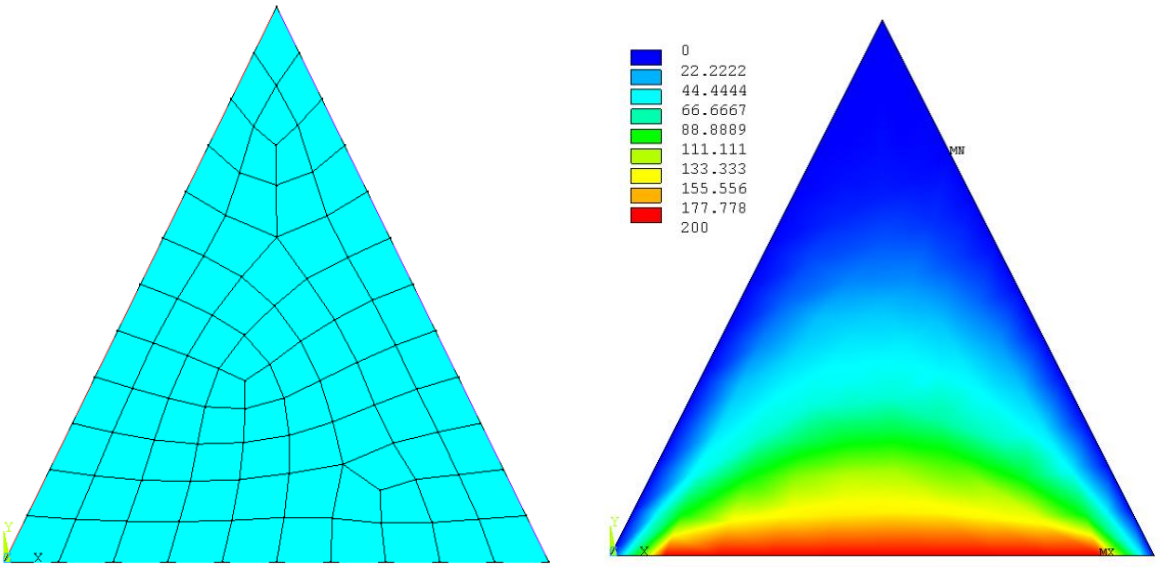

Figura 5. Discretização em MEF e resultados das temperaturas (°C) na região triangular

Na discretização em elementos finitos é necessário gerar os pontos na região escolhida e posteriormente criar as conectividades (construção dos elementos). No MDFG, a segunda etapa mencionada de pré-processamento é eliminada, tornando o desgaste computacional menor. A distribuição de pontos pelo MDFG foi gerada com um tempo de 1,52 segundos.

4.2 Comparação numérica de região circular

A Fig 6. Ilustra o problema com círculo de raio igual a 5,00 m, temperatura de 10 °C ao longo do contorno do semi circulo inferior e nula ao longo do semi circulo superior. Na Fig. 7

mostra-se a distribuição dos pontos usando uma tolerância mínima de 0,40 m (356 pontos) e posteriormente uma tolerância mínima de 0,20 m (1039 pontos) entre todos os pontos.

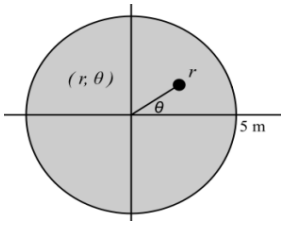

Figura 6. Ilustração do círculo em coordenadas polares

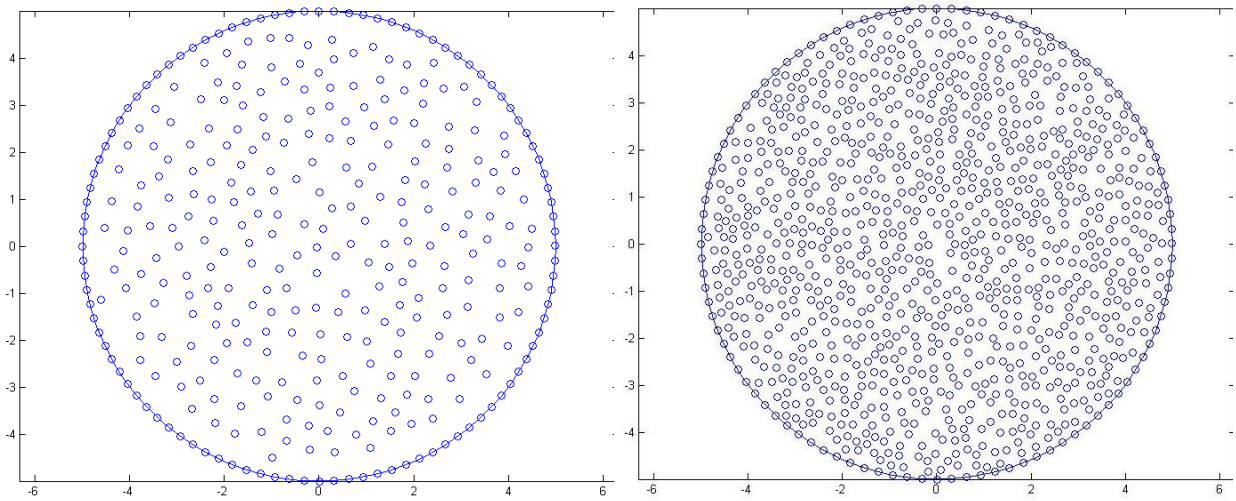

Figura 7. Distribuição dos pontos em MDFG para 356 e 1039 pontos

Após a definição da tolerância mínima e do numero de pontos a ser lançado no círculo, o tempo de pré-processamento para o primeiro caso (356 pontos) foi de 1,39 segundos e para o segundo caso (1039 pontos) 2,57 segundos. A Tabela 1 apresenta valores de temperatura ao longo do domínio pelo MDFG, comparado com a solução exata pelo método da separação de variáveis, com o aumento da nuvem randômica de pontos percebe-se a convergência para a solução exata. No primeiro caso foram geradas 255 equações e no segundo 938 equações.

Tabela 1. Comparação de resultados das temperaturas da região circular e erros associados

r (m)	θ (rad)	Solução exata (°C)	MDFG (°C) com 356 pontos [erro]	MDFG (°C) com 1039 pontos [erro]
2,61	0,98	2,22	2,83 [27,48%]	2,27 [2,25%]
4,75	3,00	9,34	9,74 [4,28%]	9,38 [0,04%]
3,99	1,49	0,71	0,89 [25,35%]	0,73 [2,82%]

Na Fig. 8, percebe-se uma definição mais clara de como se distribui a temperatura no regime estacionário quando se aumenta a quantidade de pontos no domínio.

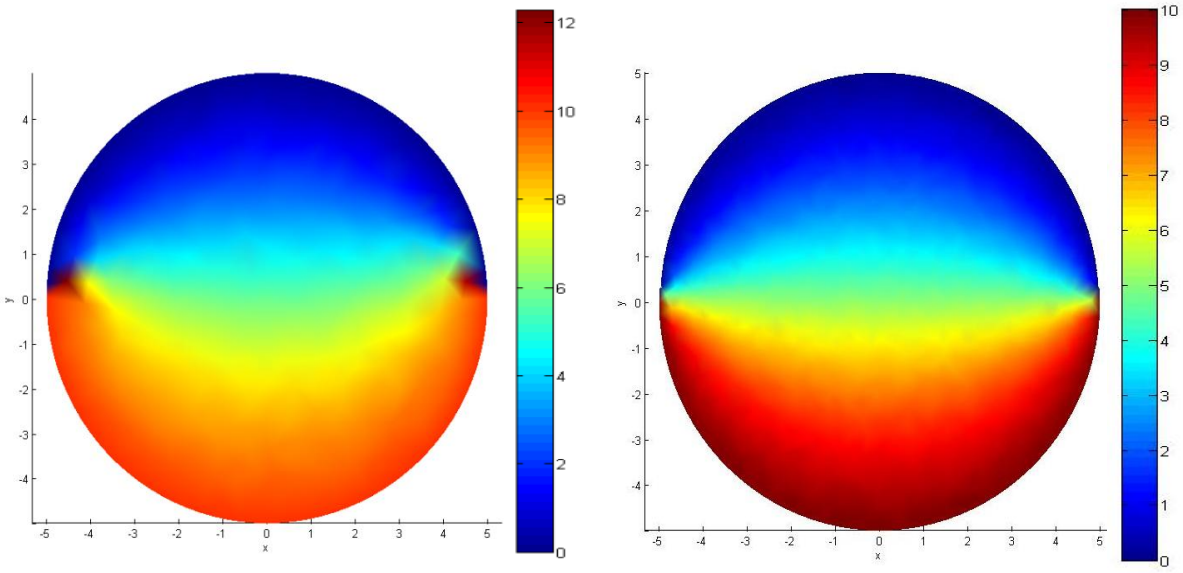

Figura 8. Distribuição da temperatura (°C) para 356 e 1039 pontos no domínio

4.3 Ameba

A grande liberdade na escolha dos pontos do domínio traz em si a vantagem de se usar o método em contornos ainda mais complexos que os apresentados até então. Assim foi criada uma função no MATLAB capaz de construir uma espécie de "ameba". Tal construção foi feita pela junção de duas funções seno com duas funções circulares, como mostrado na Fig. 9 e Fig. 10:

```
1   function[x,y,xc,yc,xv,yv]=gerador_ameba1(rmin,nit)
2
3   % Ameba
4
5   %% Contorno
6
7   % seno de cima
8   xpf1 = linspace(2.5*pi,pi/2,15);
9   ypf1 = 0.5*sin(xpf1)+1;
10
11  % circulo a esquerda
12  r = 1.5;
13  xc = pi/2;
14  yc = 0;
15
16  theta = linspace(pi/2,1.5*pi,15);
17  xcf1 = r*cos(theta) + xc;
18  ycf1 = r*sin(theta) + yc;
19
```

Figura 9. Função criada no código para construção de ameba

```
ycf1 = r*sin(theta) + yc;

xcf1(1)=[];
ycf1(1)=[];

% seno abaixo
xpf2 = linspace(pi/2,2.5*pi,15);
ypf2 = (0.5*sin(xpf2)-2);


xpf2(1)=[];
ypf2(1)=[];


% circulo a direita
r = 1.5;
xc = pi/2+2*pi;
yc = 0;

theta = linspace(-pi/2,pi/2,15);
xcf2 = r*cos(theta) + xc;
ycf2 = r*sin(theta) + yc;

xcf2(1)=[];
ycf2(1)=[];

% coordenadas de todo o contorno

xv=[xpf1 xcf1 xpf2 xcf2 ];
yv=[ypf1 ycf1 ypf2 ycf2 ];

```

Figura 10. Continuação da função criada no código para construção de ameba

Na Fig. 11 formação do pré-processamento com tolerância mínima entre os pontos de 0,30 m:

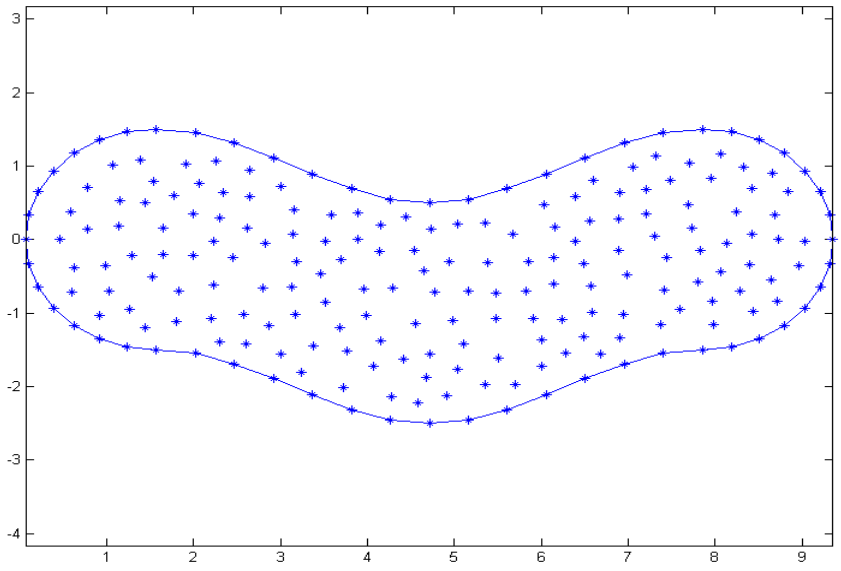

Figura 11. Pré-processamento da ameba

Colocando uma temperatura de 100 °C na função seno superior e nula no contorno restante, cria-se um gráfico (ver Fig. 12) onde o tamanho do círculo cresce com o aumento da temperatura ao longo do domínio. Em tal solução foram geradas 83 equações.

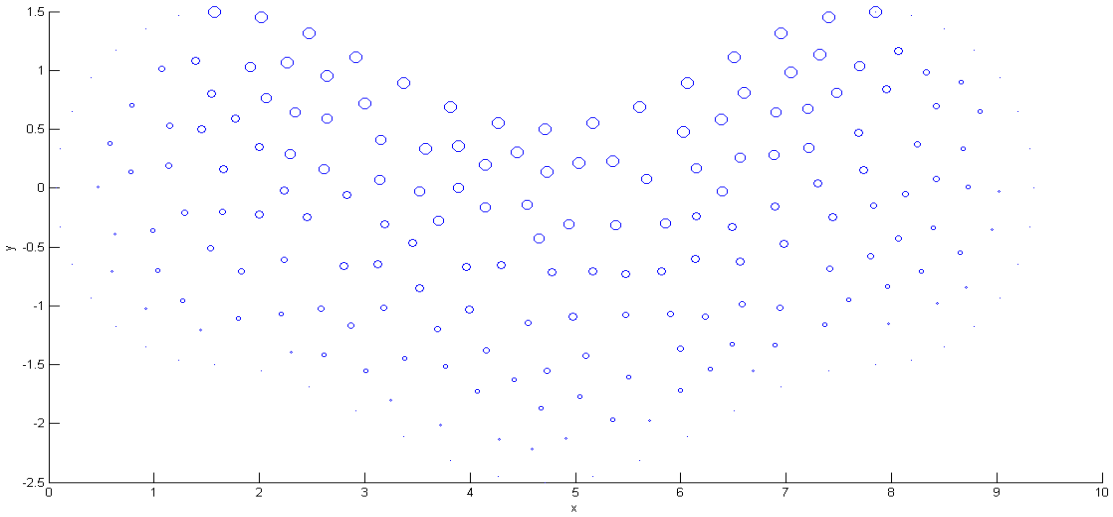

Figura 12. Gráfico de dispersão da temperatura (°C) com círculos

É perceptível o aumento dos diâmetros dos círculos à medida que se aproxima da parte superior da ameba. A simetria existente da figura dá consistência aos resultados obtidos. A Fig. 13 esclarece ainda mais os resultados, mas em termos de cores.

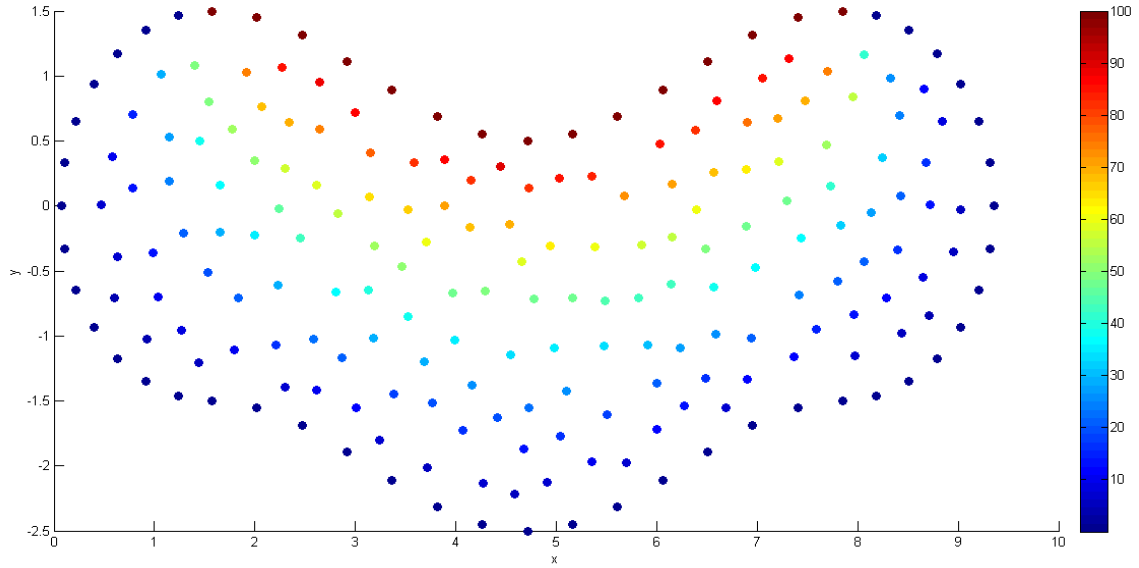

Figura 13. Gráfico de dispersão da temperatura (°C) com cores

Para a solução da ameba foram gerados 143 pontos totais, levando um tempo de 2,48 segundos para a formação dos pontos. Este tempo indica que, apesar do aumento da complexidade do contorno gerado, o tempo de pré-processamento não foi distante dos exemplos anteriores. Problemas de singularidades são gerados quando os espaçamentos entre os pontos de contorno diferem consideravelmente dos espaçamentos entre os pontos internos.

5 CONCLUSÕES

Os exemplos validados confirmam o funcionamento do método, seja qual for o domínio apresentado. Os problemas frequentes de singularidades característica do método foram eliminados, isto ocorria devido à formação de pontos mal distribuídos. Isto foi solucionado pela adoção de um limite mínimo de distância entre eles e com a exclusão de possíveis vazios no domínio. A minimização do tempo de pré-processamento em relação ao MEF merece destaque, já que não ocorrem preocupações com a construção de uma malha de elementos. Dispensando um possível uso de softwares especializados em gerações de malhas.

O presente trabalho demonstra o bom funcionamento do MDFG para uma equação diferencial parcial de segunda ordem, equações diferenciais de ordem superior não foram abordadas. Porém, o trabalho contribui para a expansão dos métodos sem malhas, apresentando sua dificuldade (geração de singularidades) e criando uma alternativa de solução para tal problema, bem como explicitando suas vantagens (extensão do MDF para domínios irregulares e menor tempo de pré-processamento em relação ao MEF).

6 REFERÊNCIAS

Benito, J. J., Urena, F., & Gavete, L., 2001. Influence of several factors in the generalized finite difference method. *Applied Mathematical Modelling*, vol. 25, pp. 1039-1053.

Gavete, L., Gavete, M. L., & Benito, J. J ., 2003. Improvements of generalized finite difference method and comparison with other meshless method. *Applied Mathematical Modelling*, vol. 27, pp. 831-847.

Jensen, P.S., 1972. Finite diference techniques for variables grids, *Computers & Structures*, vol. 2, pp. 17-29.

Liszka, T ., & Orkisz, J., 1980. The finite difference method at arbitrary irregular grids and its application in applied mechanics. *Computers & Structures*, vol. 2, pp. 83-95.

Nagle, R. K., Saff, E. B., & Snider, A. D., 2012. *Equações Diferenciais*. Pearson Education do Brasil.

Perrone, N., & Kao, R., 1975. A general finite difference method for arbitrary meshes, Catholic University Of America, *Computers & Structures*, vol. 5, pp. 45-58.