ESTUDO COMPARATIVO DE SOLUÇÕES NUMÉRICAS PARA O PROBLEMA DE DIRICHLET UTILIZANDO O MÉTODO DAS DIFERENÇAS FINITAS E REDES NEURAIS

João Pedro Silva Leite[1] (IC), João Paulo Roquim Romanelli (PQ)[1]
[1]*Universidade Federal de Itajubá*

Palavras-chave: Equações Diferenciais Parciais. Equações Elípticas. Diferenças Finitas. Redes Neurais.

Introdução

O problema de Dirichlet estudado neste trabalho é dado por

$$u_{xx} + u_{yy} = f(x,y), (x,y) \in \Omega$$
$$u(x,y) = g(x,y), (x,y) \in \partial\Omega$$

onde f e g são funções conhecidas.

O objetivo básico associado ao problema de Dirichlet é encontrar uma função que satisfaça a equação e a condição de contorno.

Muitas vezes, a solução analítica de uma EDP elíptica é muito custosa. Logo, faz-se necessário a utilização de métodos numéricos, como o método das diferenças finitas, [DF] e o método das redes neurais [LLF].

Neste trabalho foram obtidas soluções numéricas para o problema de Dirichlet utilizando o Método das Diferenças Finitas e o Método Baseado em Redes Neurais.

Metodologia

Nesta seção serão apresentados os aspectos básicos dos métodos de Diferenças Finitas e o Método baseado em redes neurais.

Método das Diferenças Finitas

O método das diferenças finitas (MDF) é um método amplamente conhecido e difundido. Em poucas palavras, o método consiste na aproximação das derivadas presentes na EDP, pelas diferenças finitas [DF, AN]. A ideia central do método, neste trabalho, é transformar o problema de Dirichlet em um sistema linear, cuja solução é uma aproximação para a solução exata.

A fim de se obter o sistema linear, considera-se uma malha/discretização do domínio Ω e de sua fronteira $\partial\Omega$. Assim obtem-se o problema de Dirichlet discreto

$$u_{xx}(x_i, y_j) + u_{yy}(x_i, y_j) = f(x_i, y_j) \quad (1)$$

$$u(x_i, y_j) = g(x_i, y_j), \quad (2)$$

onde é um conjunto (x_i, y_j) de pontos de uma malha associados ao domínio Ω e ao bordo $\partial\Omega$.

Em cada ponto de (x_i, y_j) associado a discretização de Ω aproxima-se as derivadas parciais de segunda ordem pelas diferenças finitas, isto é,

$$u_{xx}(x_i, y_j) \simeq \frac{u(x_i+h_x, y_j) - 2u(x_i, y_j) + u(x_i+h_x, y_j)}{h_x^2} \quad (3)$$

e

$$u_{yy}(x_i, y_j) \simeq \frac{u(x_i, y_j-h_y) - 2u(x_i, y_j) + u(x_i, y_j+h_y)}{h_y^2}, \quad (4)$$

com h_x, h_y suficientemente pequenos, de forma a garantir uma boa aproximação. Análise mais detalhada do erro pode ser encontrada em [DF].

As aproximações dadas pelas expressões (3) e (4) são substituídas na equação (1), com isso obtém-se o seguinte conjunto de equações

$$\frac{1}{h_x^2}[u(x_i + h_x, y_j) - 2u(x_i, y_j) + u(x_i + h_x, y_j)] +$$

$$\frac{1}{h_y^2}[u(x_i, y_j - h_y) - 2u(x_i, y_j) + u(x_i, y_j + h_y)] =$$

$$f(x_i, y_j)$$

juntamente com as equações associadas às condições de contorno $u(x_i, y_j) = g(x_i, y_j)$. Essas equações, cujas variáveis são $u(x_i, y_j) := u_{ij}$ formam o sistema linear

$$Au = b,$$

onde A é uma matriz simétrica, u vetor de incógnitas e b vetor constante obtido a partir da função f e g. O desenvolvimento detalhado pode ser encontrado em [DF].

Método Baseado em Redes Neurais

Uma rede neural multilayer perceptron com m camadas é simplesmente uma função $f: \Re^n \to \Re$ dada por:

$$f_\theta(x) = (G_m \circ \sigma \circ G_{m-1} \circ \ldots \circ \sigma \circ G_1)(x)$$

ode G_1 é chamada de camada de entrada, G_m é chamada de camada de saída, $G_i(z) = w_i z + b_i$, $i = 2, \ldots, m-1$ são chamadas de camadas internas, σ são chamadas de funções de ativação e $\theta = \{w_i, b_i\}$, m, $i = 1$ o conjunto de parâmetros que serão obtidos após treinamento da rede.

Para resolver uma EDP, a rede neural é treinada para se aproximar da função de solução. A função perda utilizada é a própria EDP juntamente com sua condição de contorno. O procedimento básico é descrito a seguir.

Seja A uma amostra de pontos do domínio Ω e $|A|$ a quantidade de pontos em A, seja $\hat{u}(x, y, w, b)$ a rede neural com parâmetros w e b. A dimensão dos parâmetros é definida conforme a estrutura da rede. Para facilitar a notação, omite-se a dependência dos parâmetros e será escrito apenas $\hat{u}(x, t)$.

A primeira parcela da função perda está associada à equação de Poisson e é escrita como

$$L_1(\hat{u}) = \frac{1}{|A|}\left\{\sum \left[\hat{u}_{xx}(x_i, y_j) + u_{yy}(x_i, y_j) - f(x, y)\right]^2\right\}$$

onde (x_i, y_j) pertence ao conjunto A. Observe que o índices i e j indicam todos os pontos do conjunto amostra.

A condição de contorno, $u(x, y) = g(x, y)$, corresponde à segunda parcela dada por

$$L_2(\hat{u}(x_i, y_j)) = \frac{1}{|\partial A|}\left\{\sum \left[\hat{u}(x_i, y_j) - g(x_i, y_j)\right]^2\right\}$$

onde os pontos ∂A é uma amostra de ponto no bordo de Ω e $|\partial A|$ a respectiva quantidade de pontos.

Então a função perda utilizada no problema é dada por

$$L(\hat{u}(x_i, t_i)) = L_1(\hat{u}(x_i, y_j)) + L_2(\hat{u}(x_i, y_j))$$

A solução numérica é então obtida a partir de métodos de otimização baseados no método do Gradiente Descendente aplicado a função perda L. O resultado do processo de otimização nos fornece os parâmetros b e w ótimos.

A ideia central é que se $L \to 0$, então $\hat{u} \to u$, isto é, se o valor da função perda converge para 0, a rede neural converge para a solução. A discussão de convergência de redes neurais pode ser encontrada em [KH] onde se discute a aproximação de funções por redes neurais e em [SS] e [LYZD], por exemplo, mostra-se a convergência para a solução de seus respectivos problemas.

Resultados e discussão

Nesta seção serão apresentados e discutidos os resultados obtidos neste trabalho.

O problema de Dirichlet considerado neste trabalho é dado por

$$u_{xx} + u_{yy} = 2x^2 + 2y^2 - 2x - 2y, \ (x, y) \in \Omega$$

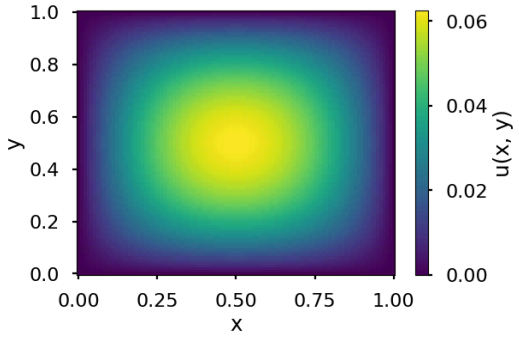
$$u(x, y) = 0, \ (x, y) \in \partial\Omega$$

cuja solução analítica é dada por

$$u(x, y) = x(1 - x)y(1 - y)$$

e o mapa de calor da solução analítica está na figura (1).

Figura 1: Mapa de Calor Solução Analítica.



Resultado Método Diferenças Finitas

O método das diferenças finitas foi implementado em uma malha de tamanho 100 x 100 pontos. Isso diz que os valores h_y e h_x são da ordem de 10^{-4}. A figura 2 mostra o mapa de calor da solução numérica obtida utilizando o método das diferenças finitas neste caso. A figura 3 mostra o mapa de calor do erro relativo utilizando a solução analítica.

Figura 2: Solução Numérica: método das diferenças finitas

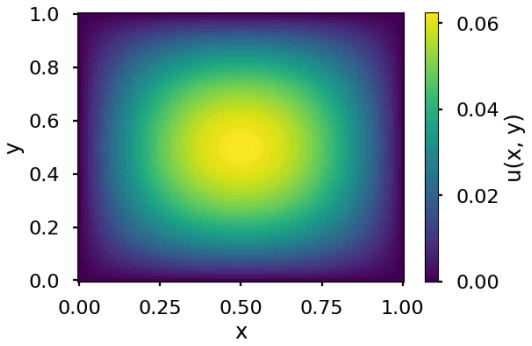
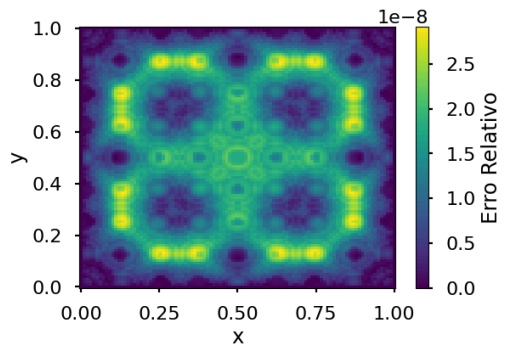


Figura 3: Erro relativo: Solução Analítica x MDF



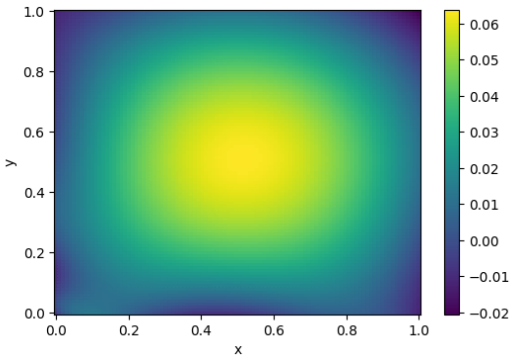
Os valores máximos de erro ficaram distribuídos em regiões centrais do domínio. Observe que o erro máximo está em torno de 2.5×10^{-8}.

Resultado Método baseado em redes neurais

A rede neural implementada neste trabalho possui duas camadas internas com 20 neurônios em cada camada. O treinamento foi realizado em um conjunto com 100 pontos aleatórios do domínio que foram projetados na fronteira de forma a se obter a função perda. Foram realizadas 10 iterações com 2000 épocas cada. A taxa de aprendizado utilizada foi de 0.001.

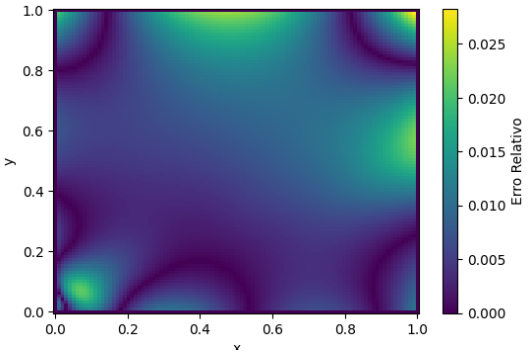
A figura 4 mostra a solução numérica obtida a partir da rede neural calculada nos mesmos pontos da malha da utilizada no método das diferenças finitas. Pode-se observar que, qualitativamente, a solução se comporta como a solução analítica apresentada anteriormente.

Figura 4: Solução Numérica: Redes Neurais



Os erros relativos calculados foram maiores que os erros obtidos pelo métodos das diferenças finitas. A figura 5 mostra o mapa de calor dos erros relativos no domínio, cujos os maiores foram da ordem de 10^{-2}.

Figura 5: Erro relativo: Solução Analítica x Rede Neural



Analises

A partir dos resultados apresentados

anteriormente, o método baseado em redes neurais é capaz de aproximar de maneira adequada as soluções de equações diferenciais elípticas associadas a problemas de Dirichlet.

As redes neurais apresentaram duas desvantagens com relação ao método das diferenças finitas. O primeiro foi a qualidade da aproximação. O método MDF se mostrou bastante superior. Entretanto, outros trabalhos, por exemplo [HF], obtém-se erros médios da ordem de 10^{-7}.

A melhora dos resultados pode ser obtida a partir de novas arquiteturas e ajustes no treinamento da rede tais como ajustes na função perda, no número de épocas, otimizadores entre outros.

O segundo ponto desfavorável as redes neurais foi o tempo de computação, nesse caso, a solução obtida pelo métodos das diferenças finitas, na malha analisada levou ~30 segundos para ser calculada enquanto a rede neural utilizada levou ~ 2 minutos.

Entretanto, o método baseado em redes neurais apresenta vantagens. A primeira delas é a respeito da implementação facilitada pelas bibliotecas disponíveis. Há dezenas de bibliotecas em Python que tornam o processo de implementação do código bastante simples. Algumas bibliotecas como, por exemplo, a biblioteca PyDens, [CPD], se equipara em tempo de processamento ao método das diferenças finitas.

Uma outra vantagem no método das redes neurais é o uso de malhas livres, o que também facilita a implementação. Em dimensões maiores que 2, pode se tornar bastante complicada quando se usa o método das diferenças finitas

Conclusões

Neste trabalho vimos que o método das redes neurais é capaz de aproximar de maneira adequada as soluções de equações diferenciais de problemas de Dirichlet. O primeiro algoritmo implementado acabou se mostrando um pouco lento, mas outras bibliotecas podem ser tão eficientes quanto o método das diferenças finitas.

O método das diferenças finitas também funcionou de maneira satisfatória, o que era esperado já que é um método conhecido e utilizado. A vantagem é a melhora da precisão que pode ser sempre obtida tomando os valores dos incrementos cada vez menores. O problema, neste caso, é o refinamento da malha, que cresce exponencialmente conforme se divide o tamanho da malha.

Nesse sentido, as redes neurais passam a ser uma alternativa interessante para problemas em dimensões maiores devido a facilidade das bibliotecas implementadas em Python, como a PyDEns.

Agradecimentos

Agradeço a Unifei e a FAPEMIG pela bolsa de estudos.

Referências

[AN] Burden, R. L; Faires, D. J; Burden, A. M. Análise Numérica, Editora Cengage Learning. 2016.

[DF] Cuminato, J. A.; Meneguette Junior, M. Disctretização de Equações Diferenciais Parciais: Técnicas de Diferenças Finitas. Editora SBM. 2013

[HF] Ferreira, Hudson O; Romanelli, João P. R. **Análise de parâmetros de redes neurais com aplicações à equação de Laplace.** Trabalho de Conclusão de Curso - Engenharia da Computação, Unifei, campus Itabira, 2023.

[KH] Hornik, Kurt; **Approximation Capabilities of Multilayer Feedforward Networks.** Neural Networks, 1991

[PD] Koryagin, A. Khudorozkov, R. Tsimfer, S. **PyDEns: a Python Framework for Solving Differential Equations with Neural Networks.** Disponível em https://arxiv.org/abs/1909.11544, 2019.

[LLF] Lagaris, Isaac E.; Likas, Aristidis; Fotiadis, Dimitrios I. **Artificial neural networks for solving ordinary and partial differential equations.** Disponível em https://arxiv.org/abs/physics/9705023, 1997.

[LYZD] Li, Jian; Yue Jing; Zhang Wen; Duan Wansuo. **The Deep Learning Galerkin Method for the General Stokes Equations.** Disponível em https://arxiv.org/abs/2009.11701 , 2020.

[SS] Sirignano, Justin; Spiliopoulos, Konstantinos; **DGM: A deep learning algorithm for solving partial differential equations**. Journal of Computational Physics, 375, 2018.