Redes Neurais Fisicamente Guiadas (PINNs) na Aproximação de Soluções para a Equação de Benjamin-Bona-Mahony

Laboratório de Dinâmica dos Fluidos Computacional

Samuel Kutz Paranhos

Prof. Roberto Ribeiro-Jr

Universidade Federal do Paraná

14 de novembro de 2024

Sumário

A Equação de Benjamin-Bona-Mahony (BBM)

Equação de Benjamin-Bona-Mahony (BBM)

A BBM[1] é a *Equação Diferencial Parcial Não Linear*

$$u_t + u_x + uu_x - u_{xxt} = 0,$$

utilizada para estudar modelos de ondas longas e de baixa amplitude que propagam de forma unidirecional.

É uma alternativa ao modelo de Korteweg–De Vries (KdV), o qual é aplicado na modelagem da dinâmica de ondas aquáticas e diversas outras situações práticas[2].

[1] Thomas Brooke Benjamin, Jerry Lloyd Bona e John J Mahony (1972). "Model equations for long waves in nonlinear dispersive systems". Em: *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences* 272.1220, páginas 47–78.

[2] Crighton, 1995.

Equação de Benjamin-Bona-Mahony (BBM)

Estudaremos aproximações de soluções da BBM na forma:

$$\begin{aligned} u_t + u_x + uu_x - u_{xxt} &= 0, && (x,t) \in (-10, 20) \times (0,4), \\ u(x,0) &= u_0(x), && x \in (-10,20), \\ u(-10,t) &= g_1(t), && \\ u(20,t) &= g_2(t), && t \in (0,4), \end{aligned}$$

onde

- $u = u(x,t)$;
- u_0 é a **condição inicial** em $t = 0$;
- g_1, g_2 são **condições de contorno** que nos dizem como u atua na fronteira domínio de x.

Equação de Benjamin-Bona-Mahony (BBM)

Estudaremos aproximações de soluções da BBM na forma:

$$
\begin{aligned}
u_t + u_x + u u_x - u_{xxt} &= 0, && (x,t) \in (-10,20) \times (0,4),\\
u(x,0) &= u_0(x), && x \in (-10,20),\\
u(-10,t) &= g_1(t), && \\
u(20,t) &= g_2(t), && t \in (0,4),
\end{aligned}
$$

onde

- $u = u(x,t)$;
- u_0 é a **condição inicial** em $t = 0$;
- g_1, g_2 são **condições de contorno** que nos dizem como u atua na fronteira domínio de x.

Equação de Benjamin-Bona-Mahony (BBM)

Estudaremos aproximações de soluções da BBM na forma:

$$
\begin{aligned}
u_t + u_x + uu_x - u_{xxt} &= 0, && (x,t) \in (-10,20) \times (0,4),\\
u(x,0) &= u_0(x), && x \in (-10,20),\\
u(-10,t) &= g_1(t), && \\
u(20,t) &= g_2(t), && t \in (0,4),
\end{aligned}
$$

onde

- $u = u(x,t)$;
- u_0 é a **condição inicial** em $t = 0$;
- g_1, g_2 são **condições de contorno** que nos dizem como u atua na fronteira domínio de x.

Equação de Benjamin-Bona-Mahony (BBM)

A BBM possui **solução viajante** dada por

$$u(x,t) = A\,\mathrm{sech}^2(k(x-ct)),$$

onde

- $A > 0$ é dado e chamado de **amplitude**;
- $k = \sqrt{\dfrac{A}{12+4A}}$, chamado de **frequência**;
- $c = 1 + \dfrac{A}{3}$, chamado de **velocidade**.

Uma solução viajante de uma EDP pode ser entendida como uma onda de amplitude A que se propaga com velocidade constante c sem **mudar de forma**.

Equação de Benjamin-Bona-Mahony (BBM)

A BBM possui **solução viajante** dada por

$$u(x,t) = A\,\mathrm{sech}^2(k(x - ct)),$$

onde

- $A > 0$ é dado e chamado de **amplitude**;
- $k = \sqrt{\dfrac{A}{12 + 4A}}$, chamado de **frequência**;
- $c = 1 + \dfrac{A}{3}$, chamado de **velocidade**.

Uma solução viajante de uma EDP pode ser entendida como uma onda de amplitude A que se propaga com velocidade constante c sem **mudar de forma**.

Equação de Benjamin-Bona-Mahony (BBM)

A BBM possui **solução viajante** dada por

$$u(x,t) = A\,\mathrm{sech}^2(k(x-ct)),$$

onde

- $A > 0$ é dado e chamado de **amplitude**;
- $k = \sqrt{\dfrac{A}{12+4A}}$, chamado de **frequência**;
- $c = 1 + \dfrac{A}{3}$, chamado de **velocidade**.

Uma solução viajante de uma EDP pode ser entendida como uma onda de amplitude A que se propaga com velocidade constante c sem **mudar de forma**.

Equação de Benjamin-Bona-Mahony (BBM)

Substituindo a solução viajante na equação proposta, obtemos

$$
\begin{aligned}
u_t + u_x + uu_x - u_{xxt} &= 0, && (x,t) \in (-10,20) \times (0,4),\\
u(x,0) &= A\,\mathrm{sech}^2(kx), && x \in (-10,20),\\
u(-10,t) &= A\,\mathrm{sech}^2(k(-10-ct)), && \\
u(20,t) &= A\,\mathrm{sech}^2(k(20-ct)), && t \in (0,4),
\end{aligned}
$$

onde $A > 0,\ k = \sqrt{\dfrac{A}{12+4A}}$ e $c = 1 + \dfrac{A}{3}$.

- Como podemos resolver um problema assim tomando vantagem das ferramentas e avanços recentes de **Deep Learning**?

Physics Informed Neural Networks (PINNs)

Physics Informed Neural Networks (PINNs)

Apresentado por Raissi[3] em 2017, as Physics Informed Neural Networks (PINNs) são redes neurais adaptadas para aproximar soluções de EDPs, incorporando a física do problema.

- Muitas vezes métodos clássicos, como Diferenças Finitas, apresentam dificuldades ao lidar com problemas que envolvem muitas dimensões ou com choques/descontinuidades.

- PINNs são redes neurais contínuas, não há necessidade de interpolação dos resultados.

- Poderemos usufruir de todo ferramental moderno da área de **Deep Learning**.

[3] Maziar Raissi, Paris Perdikaris e George Em Karniadakis (2017). "Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations". Em: *arXiv preprint arXiv:1711.10561*.

Physics Informed Neural Networks (PINNs)

Apresentado por Raissi[3] em 2017, as Physics Informed Neural Networks (PINNs) são redes neurais adaptadas para aproximar soluções de EDPs, incorporando a física do problema.

- Muitas vezes métodos clássicos, como Diferenças Finitas, apresentam dificuldades ao lidar com problemas que envolvem muitas dimensões ou com choques/descontinuidades.

- PINNs são redes neurais contínuas, não há necessidade de interpolação dos resultados.

- Poderemos usufruir de todo ferramental moderno da área de **Deep Learning**.

[3] Maziar Raissi, Paris Perdikaris e George Em Karniadakis (2017). "Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations". Em: *arXiv preprint arXiv:1711.10561*.

Physics Informed Neural Networks (PINNs)

Apresentado por Raissi[3] em 2017, as Physics Informed Neural Networks (PINNs) são redes neurais adaptadas para aproximar soluções de EDPs, incorporando a física do problema.

- Muitas vezes métodos clássicos, como Diferenças Finitas, apresentam dificuldades ao lidar com problemas que envolvem muitas dimensões ou com choques/descontinuidades.
- PINNs são redes neurais contínuas, não há necessidade de interpolação dos resultados.
- Poderemos usufruir de todo ferramental moderno da área de **Deep Learning**.

[3] Maziar Raissi, Paris Perdikaris e George Em Karniadakis (2017). “Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations”. Em: *arXiv preprint arXiv:1711.10561*.

Physics Informed Neural Networks (PINNs)

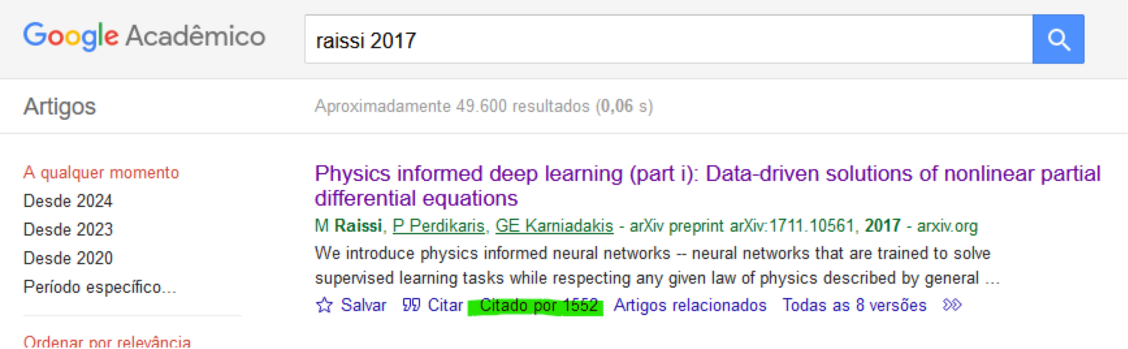


Figura: Artigo original Raissi, Perdikaris e Karniadakis, 2017, com destaque para o número de citações.

Physics Informed Neural Networks (PINNs)

A ideia geral da técnica de PINNs é transformar a EDP em um **problema de otimização**.

$$
\begin{aligned}
u_t + u_x + uu_x - u_{xxt} &= 0, && (x,t) \in (-10,20) \times (0,4),\\
u(x,0) &= u_0(x), && x \in (-10,20),\\
u(-10,t) &= g_1(t), && \\
u(20,t) &= g_2(t), && t \in (0,4).
\end{aligned}
$$

Definimos os resíduos da seguinte maneira:

- $\min |\mathcal{D}(x,t)| \quad$ s.a. $\quad (x,t) \in (-10,20) \times (0,4)$,

  onde $\mathcal{D} := u_t + u_x + u\,u_x - \,u_{xxt}$ representa o resíduo da EDP;

Physics Informed Neural Networks (PINNs)

A ideia geral da técnica de PINNs é transformar a EDP em um **problema de otimização**.

$$
\begin{aligned}
u_t + u_x + uu_x - u_{xxt} &= 0, && (x,t) \in (-10,20) \times (0,4),\\
u(x,0) &= u_0(x), && x \in (-10,20),\\
u(-10,t) &= g_1(t), && \\
u(20,t) &= g_2(t), && t \in (0,4).
\end{aligned}
$$

- $\min |u(x,0) - u_0(x)| \quad \text{s.a.} \quad x \in (-10,20),$

  é o resíduo da condição inicial;

- $\min\big(|u(-10,t) - g_1(t)| + |u(20,t) - g_2(t)|\big) \quad \text{s.a.} \quad t \in (0,4),$

  é o resíduo das condições de contorno.

Physics Informed Neural Networks (PINNs)

Assim, para uma dada $\hat{u}$ ser solução aproximada da EDP, deve satisfazer os problemas

- $\min |\mathcal{D}(x,t)| = \min |\hat{u}_t + \hat{u}_x + \hat{u}\,\hat{u}_x - \hat{u}_{xxt}| \quad \text{s.a.} \quad (x,t) \in (-10,20) \times (0,4),$
- $\min |\hat{u}(x,0) - u_0(x)| \quad \text{s.a.} \quad x \in (-10,20),$
- $\min\big(|\hat{u}(-10,t) - g_1(t)| + |\hat{u}(20,t) - g_2(t)|\big) \quad \text{s.a.} \quad t \in (0,4).$

A partir disso, formulamos uma função **Erro Quadrático Médio** (EQM), que incorpora os três problemas.

Physics Informed Neural Networks (PINNs)

Minimize o **Erro Quadrático Médio** dado por

$$\mathrm{EMQ} = \mathrm{EMQ}_{\mathcal{D}} + \mathrm{EMQ}_0 + \mathrm{EMQ}_b,$$

onde

$$\mathrm{EMQ}_{\mathcal{D}} = \frac{1}{N_{\mathcal{D}}} \sum_{i=1}^{N_{\mathcal{D}}} \big[\mathcal{D}(x_i, t_i)\big]^2,$$

- (x_i, t_i), $i = 1, \cdots, N_{\mathcal{D}}$, $N_{\mathcal{D}}$ representa pontos de colocação escolhidos aleatoriamente no domínio $(-10, 20) \times (0, 4)$

Physics Informed Neural Networks (PINNs)

Minimize o **Erro Quadrático Médio** dado por

$$\mathrm{EMQ} = \mathrm{EMQ}_{\mathcal{D}} + \mathrm{EMQ}_0 + \mathrm{EMQ}_b,$$

onde

$$\mathrm{EMQ}_0 = \frac{1}{N_0}\sum_{i=1}^{N_0}\big(u(x_i,0) - u_0(x_i)\big)^2,$$

- x_i, $i = 1, \cdots, N_0$ representa N_0 pontos de colocação no intervalo $(-10, 20)$

Physics Informed Neural Networks (PINNs)

Minimize o **Erro Quadrático Médio** dado por

$$\text{EMQ} = \text{EMQ}_{\mathcal{D}} + \text{EMQ}_0 + \text{EMQ}_b,$$

onde

$$\text{EMQ}_b = \frac{1}{N_b}\sum_{i=1}^{N_b}\Big(|\hat{u}(-10,t) - g_1(t)| + |\hat{u}(20,t) - g_2(t)|\Big)^2,$$

- $t_i,\ i = 1, \cdots, N_b$, pontos de colocação no intervalo em $(-10, 20) \times (0, 4)$.

E assim, treinamos uma rede com a função de perda EMQ sobre os pontos de colocação como nossos dados.

Physics Informed Neural Networks (PINNs)

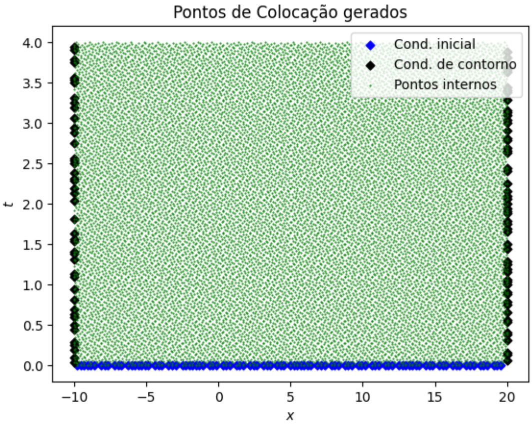


Figura: Pontos de colocação gerados a partir da *seed* 0.
$N_{\mathcal{D}} = 15000$, $N_b = 100$, $N_0 = 100$.

Physics Informed Neural Networks (PINNs)

No nosso caso, a arquitetura da rede ficou decidida em:

Parâmetro	**Valor**
Amplitude inicial (A)	5
Domínio espacial (x)	$(-10, 20)$
Domínio temporal (t)	$(0, 4)$
Pontos de colocação no domínio ($N_{\mathcal{D}}$)	15 000
Pontos de colocação na fronteira (N_b)	100
Pontos de colocação no valor inicial (N_0)	100
Neurônios por camada	50
Camadas intermediárias	3
Iterações do otimizador	15 000
Taxa de aprendizado	10^{-3}
Função de ativação	tanh

Objetivo

Objetivo

Ao procurarmos na literatura, não encontramos nada (até o momento) sobre o estudo das PINNs na aproximação de soluções da BBM, portanto, nossos objetivos são:

- Compreender o funcionamento da técnica;
- Aplicar o método e verificar sua eficácia em comparação com a solução exata;
- Investigar o desempenho de diferentes otimizadores em uma arquitetura fixa.

Metodologia

Metodologia

Para implementar da técnica de PINNs, foram utilizadas as seguintes tecnologias:

- A linguagem de programação Python;
- Google Colab para acesso a GPUs;
- Biblioteca **DeepXDE**[4] para implementação PINNs;

Essa biblioteca funciona como uma interface para diversos *back-ends*, como *TensorFlow* e *PyTorch*, facilitando o processo de definição das EDPs e a criação das redes.

[4] Lu Lu et al. (2021). "DeepXDE: A deep learning library for solving differential equations". Em: *SIAM review* 63.1, páginas 208–228.

Principais Otimizadores

Principais Otimizadores

De forma similar ao artigo[5], realizaremos uma *corrida de otimizadores*, ou seja, investigaremos qual otimizador é mais eficaz para encontrar parâmetros da rede que aproxima a BBM.

Para o estudo numérico, realizamos experimentos com os seguintes otimizadores disponibilizados pelo *back-end Tensorflow* da biblioteca DeepXDE:

- **ADAM** (Adaptive Moment Estimation)
- **NADAM** (ADAM com *Nesterov Momentum*)
- **SGD** (Stochastic Gradient Descent)
- **L-BFGS** (Limited memory *Broyden–Fletcher–Goldfarb–Shanno*)

[5] John Taylor et al. (2022). "Optimizing the optimizer for data driven deep neural networks and physics informed neural networks". Em: *arXiv preprint arXiv:2205.07430*.

Resultados

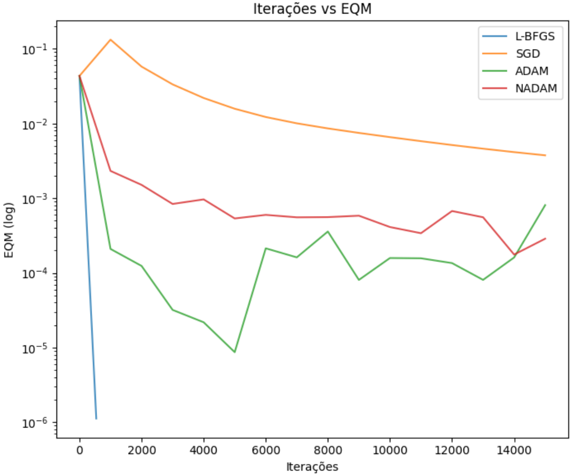
Resultados

GIF

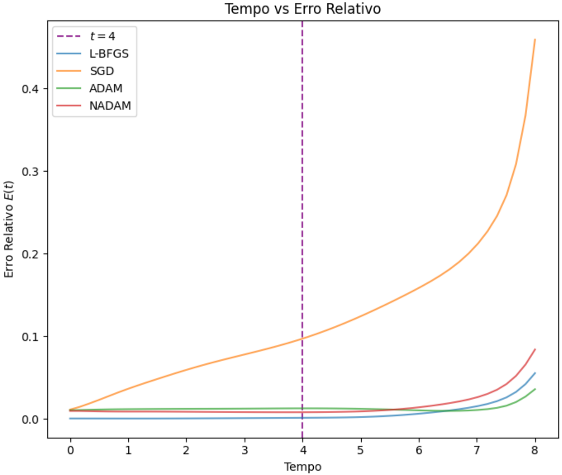
Resultado obtido com a arquitetura fixada:

Parâmetro	**Valor**
Amplitude inicial (A)	5
Domínio espacial (x)	$(-10, 20)$
Domínio temporal (t)	$(0, 4)$
Pontos de colocação no domínio ($N_{\mathcal{D}}$)	$15\,000$
Pontos de colocação na fronteira (N_b)	100
Pontos de colocação no valor inicial (N_0)	100
Neurônios por camada	50
Camadas intermediárias	3
Iterações do otimizador	$15\,000$
Taxa de aprendizado	10^{-3}
Função de ativação	tanh

Resultados



Iterações vs EQM
L-BFGS
SGD
ADAM
NADAM
EQM (log)
Iterações
0
2000
4000
6000
8000
10000
12000
14000
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}



Tempo vs Erro Relativo
$t = 4$
L-BFGS
SGD
ADAM
NADAM
Erro Relativo $E(t)$
Tempo
0.0
0.1
0.2
0.3
0.4
0
1
2
3
4
5
6
7
8

Conclusão

Conclusões

- Pouca dificuldade para chegar em aproximações satisfatórias em relação às outras equações;
- Ao contrário de outras equações, o **L-BFGS** se destacou como o otimizador mais eficaz;
- Dentro do domínio temporal escolhido, a solução aproximada pela PINN apresenta erro satisfatório;
- Existe a possibilidade de que haja uma conexão entre os otimizadores utilizados em PINNs e as características da EDP escolhida;
- Em trabalhos futuros, investigar se classes de EDPs (hiperbólicas, parabólicas ou elípticas) possuem otimizadores mais adequados que outros.

Muito obrigado!

Referências I

Thomas Brooke Benjamin, Jerry Lloyd Bona e John J Mahony (1972). "Model equations for long waves in nonlinear dispersive systems". Em: *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences* 272.1220, páginas 47–78.

DG Crighton (1995). "Applications of kdv". Em: *KdV'95: Proceedings of the International Symposium held in Amsterdam, The Netherlands, April 23–26, 1995, to commemorate the centennial of the publication of the equation by and named after Korteweg and de Vries*. Springer, páginas 39–67.

Lu Lu, Xuhui Meng, Zhiping Mao e George Em Karniadakis (2021). "DeepXDE: A deep learning library for solving differential equations". Em: *SIAM review* 63.1, páginas 208–228.

Referências II

Maziar Raissi, Paris Perdikaris e George Em Karniadakis (2017). "Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations". Em: *arXiv preprint arXiv:1711.10561*.

John Taylor, Wenyi Wang, Biswajit Bala e Tomasz Bednarz (2022). "Optimizing the optimizer for data driven deep neural networks and physics informed neural networks". Em: *arXiv preprint arXiv:2205.07430*.