UNIVERSIDADE FEDERAL DE ITAJUBÁ
Instituto de Física e Química - IFQ

SOLUÇÃO DAS EQUAÇÕES DE CAMPO DE EINSTEIN COM *MACHINE LEARNING*

TRABALHO FINAL DE GRADUAÇÃO
Agência financiadora: CNPq

Aluno: Daniel Augusto de Sousa Mendes
Matrícula: 2016000954
Curso: Física Bacharelado
Orientador: Prof. Dr. Alan Bendasoli Pavan

BRASIL
2022

Daniel Augusto de Sousa Mendes

Solução das Equações de Campo de Einstein com *Machine Learning*

Dissertação submetida ao Instituto de Física e Química com finalidade de aprovação no Trabalho de Conclusão de Curso.

Universidade Federal de Itajubá - UNIFEI

Instituto de Física e Química

Orientador: Prof. Dr. Alan Bendasoli Pavan

Brasil

2022

Dedico a Antonio e Jucileia, meus pais.

Agradecimentos

Principais agradecimentos ao meus pais que jamais me desencorajaram em seguir em busca de uma carreira científica, e que sem eles seria incapaz de realizar uma fração de meus estudos.

A minha namorada Gabriella, que por também ser física, compartilhou comigo todos momentos bons e ruins do caminho percorrido até o fim de minha graduação.

Aos meus colegas de curso pelos debates calorosos sobre tópicos da física, que acabaram por elucidar diversos conceitos e me confundir quase que permanentemente em outros. Também, aos meus antigos amigos que, apesar da distância, não se esqueceram de mim.

Aos meus orientadores e colegas de projetos que confiaram à mim trabalhos que acabaram por definir o rumo de minha carreira, principalmente a Alan Pavan por sua disponibilidade.

Aos docentes e colegas do projeto Epic Coders por meu tempo na maratona de programação e por meu primeiro "*Hello, world!*" em 2016.

Ao CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) pelo financiamento deste trabalho de conclusão de curso, e também pela bolsa de iniciação científica que garantiram meu primeiro estudo com os algoritmos aqui presentes.

"Eu gostaria que isso não precisasse ter acontecido na minha época", disse Frodo. "Eu também", disse Gandalf. "Como todos os que vivem nestes tempos. Mas a decisão não é nossa. Tudo que temos que decidir é o que fazer com o tempo que nos é dado."

(J. R. R. Tolkien. A Sociedade do Anel, Cap. II. 1954)

Resumo

A aplicação de técnicas de inteligência artificial na física, majoritariamente de redes neurais profundas, tem se popularizado nos últimos anos principalmente devido à propriedade destas redes neurais de atuarem como aproximadores universais de funções contínuas. Uma destas aplicações é conhecida como *Physics-Informed Neural Networks* (PINN) que realiza a tarefa de obter, numericamente, soluções para equações diferenciais parciais não-lineares provenientes de leis da física. O objetivo principal deste trabalho foi utilizar a PINN para serem estudadas soluções numéricas das Equações de Campo de Einstein e compará-las com as soluções exatas, quando disponíveis. Em especial, houve interesse em soluções do tipo buraco negro, onde foi avaliado que o formalismo é eficiente em obter soluções padrão de buracos negros como Schwarzschild, Schwarzschild de-Sitter e Schwarzschild Anti-de-Sitter. Também foi verificado que é capaz de obter soluções com conteúdos de matéria mais exóticos, como um buraco negro regular do tipo Phantom e um buraco de minhoca.

Palavras-chaves: Relatividade Geral; Inteligência Artificial; Equações Diferenciais Parciais; Deep Learning; Gravitação.

Abstract

The application of artificial intelligence techniques in physics, mostly of deep neural networks, has become popular in recent years, mainly due to the property of these neural networks to act as universal approximators of continuous functions. One of these applications is known as Physics-informed Neural Networks (PINN) wich performs the task of numerically obtaining solutions to nonlinear partial differential equations from laws of physics. The main objective of this work was to use PINN to study numerical solutions of Einstein's Field Equations and compare them with exact solutions, when available. In particular, there was interest in black hole solutions, where it was evaluated that the formalism is efficient in obtaining standard solutions for black holes such as Schwarzschild, Schwarzschild de-Sitter and Schwarzschild Anti-de-Sitter. It has also been found to be able to obtain solutions with more exotic matter contents, such as a regular black hole of type Phantom and a wormhole.

Key-words: General Relativity; Artificial Intelligence; Partial Differential Equations; Deep Learning; Gravitation.

Lista de ilustrações

Lista de abreviaturas e siglas

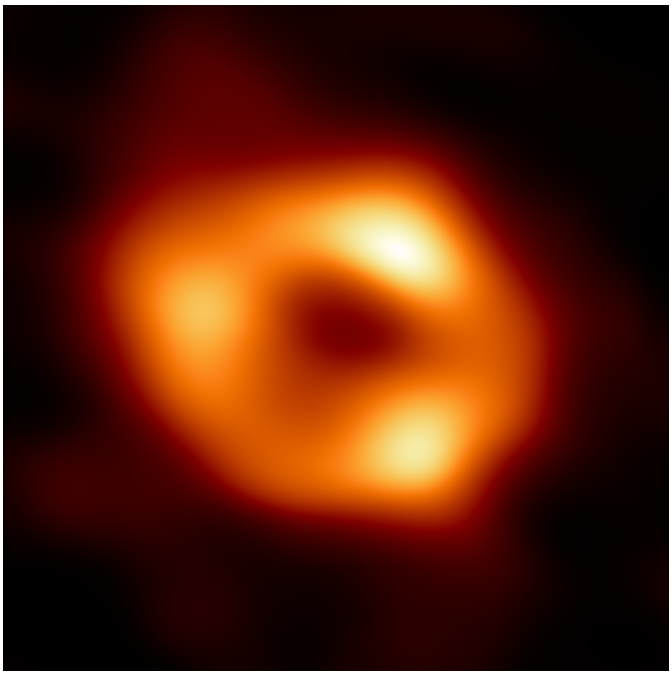
AI	Artificial Intelligence
GPU	Graphics Processing Unit
IA	Inteligência Artificial
MAE	Mean Absolute Error
MLP	Multilayer Perceptron
MSE	Mean Squared Error
NN	Neural Networks
PDE	Partial Differential Equation
PINN	Physics-informed Neural Networks
TPU	Tensor Processing Unit

Sumário

1 Introdução

Recentemente, os buracos negros tomaram conta dos debates científicos novamente graças às observações do Event Horizon Telescope que foi capaz, através de técnicas de interferometria, de produzir uma imagem do buraco negro supermassivo posicionado no centro de nossa galáxia (figura 1). Buracos negros são, de certa forma, regiões no espaço onde a gravidade é tão intensa que nem mesmo a luz pode escapar, podendo se originar através do colapso estelar[1]. Este tipo de objeto já tem sido estudado por gerações de físicos e ainda possui diversos problemas não resolvidos.

Figura 1 – Primeira imagem do buraco negro Sagittarius A*.



Fonte: EHT Collaboration (2022).

Os buracos negros são previsões da Teoria da Relatividade Geral proposta por Albert Einstein em 1915. Anteriormente, no século XVII, Isaac Newton inspirado pelos movimentos dos corpos celestes propôs uma lei de gravitação universal onde a interação gravitacional entre estes corpos surge na forma de uma força atrativa em função de suas massas. Einstein então propôs um aprimoramento na noção de gravidade como uma propriedade geométrica do espaço quadridimensional[2]. Esta nova interpretação conseguiu explicar fenômenos antes incompreensíveis como o desvio no periélio de Mercúrio e as lentes gravitacionais. Nesta noção geométrica, a disposição de matéria e energia no espaço acaba

[1] No caso de buracos negros de origem estelar.
[2] Três dimensões espaciais e uma temporal.

por curvá-lo e esta curvatura acaba por afetar o movimento dos objetos que percorrem trajetórias geodésicas em um espaço-tempo curvo.

Assim como as Equações de Maxwell governam a dinâmica de campos eletromagnéticos, na Relatividade Geral os campos gravitacionais também são governados por um conjunto de equações diferenciais parciais acopladas, chamadas de Equações de Campo de Einstein. As soluções destas equações podem ser interpretadas como descrições de diferentes tipos de objetos e sistemas gravitacionais como os buracos negros, cuja primeira solução foi encontrada em 1916 por Karl Schwarzschild (SCHWARZSCHILD, 1916). Neste contexto, o desenvolvimento de ferramentas matemáticas e computacionais que nos permitam solucionar equações diferenciais são fundamentais para elucidar as questões relativas a esse tema.

1.1 Inteligência artificial na física

É de conhecimento geral que estamos passando, na última década, por um período de interesse em aplicações de técnicas de inteligência artificial (IA), atingindo excelência em diversas tópicos como reconhecimento de imagens, reconhecimento de fala, robôs de conversação, direção autônoma, além de diversas outras. Devido a isto, a IA tem sido amplamente utilizada e estudada por diversas empresas de tecnologia como Google, IBM e Amazon.

Esta nova era de ânimo com relação à inteligência artificial não tem atingido apenas aplicações comerciais, mas também a pesquisa de base em áreas como Química, Matemática e Física. Por exemplo, as redes neurais (um modelo de *machine learning*) tem demonstrado uma útil capacidade de interpretar grande quantidade de dados com ruído, possuindo muito potencial em aplicações envolvendo dados científicos, como na descoberta de exoplanetas (VALIZADEGAN et al., 2021)

Outra propriedade interessante das redes neurais é que são aproximadores universais de funções. Esta propriedade tem gerado um "*boom*" nos últimos anos, onde pesquisadores tem se proposto a trabalhar em problemas sofisticados da Física como, por exemplo, na representação da correspondência AdS/CFT em teoria das cordas (HASHIMOTO et al., 2018) e na solução do problema de muitos corpos quântico (CARLEO; TROYER, 2016).

Como diversos fenômenos físicos são modelados por equações diferenciais, uma das aplicações mais interessantes de *machine learning* são aquelas relativas a análise e solução deste tipo de equação. Diversos autores têm proposto algoritmos que se utilizam de redes neurais (profundas ou rasas) na "descoberta" de conceitos físicos e equações diferenciais pela máquina (ITEN et al., 2020; RUDY et al., 2017), na solução das equações diferenciais (LIU; YANG; CAI, 2019; DOCKHORN, 2019; PISCOPO; SPANNOWSKY; WAITE, 2019) e até *frameworks* completos para auxiliar a solução de problemas físico-matemáticos que

envolvam este tipo de equação (RAISSI; PERDIKARIS; KARNIADAKIS, 2017a; RAISSI; PERDIKARIS; KARNIADAKIS, 2017b; RAISSI; PERDIKARIS; KARNIADAKIS, 2019; KORYAGIN; KHUDOROZKOV; TSIMFER, 2019).

Um dos trabalhos mais relevantes na solução de equações diferenciais foi proposto por Raissi e colaboradores (RAISSI; PERDIKARIS; KARNIADAKIS, 2019; RAISSI; PERDIKARIS; KARNIADAKIS, 2017). Nele os autores se dedicam a criar ferramentas computacionais para lidar com equações diferenciais parciais não-lineares, em especial, utilizando *deep learning*. Propostas em 2017, as *Physics-informed neural networks* (PINN) constituem uma ferramenta dedicada a construir uma solução numérica (RAISSI; PERDIKARIS; KARNIADAKIS, 2017a) ou realizar o ajuste de coeficientes presentes nas equações diferenciais parciais diferenciais não-lineares utilizando dados fornecidos (RAISSI; PERDIKARIS; KARNIADAKIS, 2017b).

1.2 Objetivos

Este trabalho está dividido em 5 capítulos e um apêndice: Uma breve introdução ao assunto abordado, um capítulo de fundamentação teórica onde serão explicitados tópicos essenciais da matemática, física e computação para a compreensão da metodologia, um capítulo de metodologia para se tornar compreensível o método de solução de uma equação diferencial com *machine learning*, um capítulo com os resultados obtidos para as equações de Einstein assim como uma breve análise dos mesmos, um capítulo de conclusões e discussões acerca do que foi desenvolvido durante o projeto e, por fim, um apêndice com informações a respeito da implementação dos códigos e *hardware* utilizado.

Neste trabalho foi tido como objetivo principal verificar se as *Physics-informed Neural Networks* são capazes de resolver numericamente as equações de campo de Einstein fornecendo as componentes da métrica que descrevem alguns objetos gravitacionais de interesse. Mais especificamente, pretendeu-se:

- Estudar as *Physics-informed Neural Networks* a fim de reproduzir a solução de algumas equações não-lineares.
- Implementar novas soluções para algumas equações diferenciais da Física como equação de propagação de calor, equação de onda e equação de um oscilador harmônico, comparando com suas soluções exatas para fins de avaliação do método numérico utilizado.
- Estudar as equações de campo de Einstein para que seja encontrada uma forma de utilizar inteligência artificial (PINN) para serem obtidos os termos da métrica para alguns objetos gravitacionais da Relatividade Geral, em especial, alguns tipos de buracos negros.

Ao fim do projeto, foi possível verificar que as *Physics-informed Neural Networks* são capazes de aproximar os termos da métrica para diversos tipos de buracos negros e outros objetos da relatividade geral.

2 Fundamentação Teórica

Neste capítulo serão apresentadas brevemente as principais ferramentas teóricas, matemáticas e computacionais utilizadas durante o desenvolvimento desta pesquisa e que serão necessárias para o entendimento da metodologia aplicada na resolução do problema almejado.

Como o objetivo deste trabalho foi resolver as equações de Einstein utilizando uma ferramenta de *Machine Learning*, é necessário compreender sucintamente o papel da IA neste processo, em específico, o do *Deep Learning*, um sub-campo do *Machine Learning*.

É preciso também entender os conceitos fundamentais das equações diferenciais na Física, seus tipos, sua dificuldade de solução exata e como ferramentas numéricas como as PINN podem nos ajudar nesta tarefa.

Por fim, se faz necessário ter um conhecimento elementar de Relatividade Geral/Gravitação para que seja possível compreender o problema em si, como abordá-lo da maneira pretendida e se os resultados obtidos fazem algum sentido físico.

2.1 Inteligência Artificial

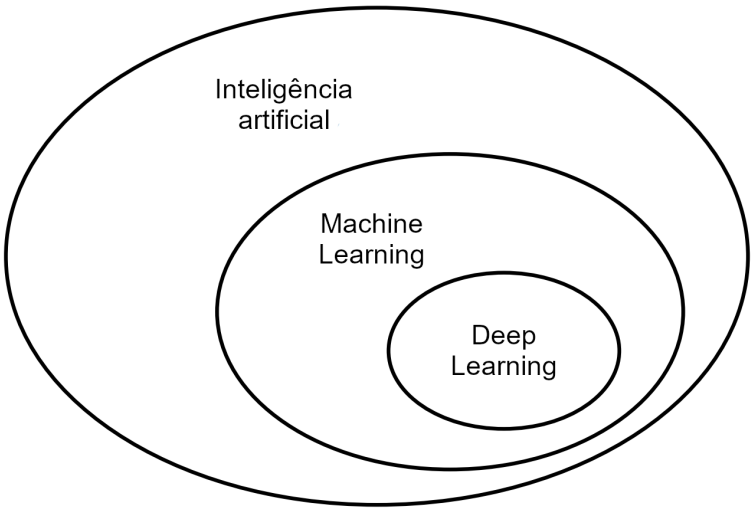
Surgida como um campo de pesquisa em meados de 1950, a IA nasce como uma tentativa de descrever matematicamente aspectos de aprendizado e inteligência, de forma que máquinas possam resolver problema reservados aos humanos como usar e simular linguagem, formular abstrações e conceitos além de aprimorar a si mesmas. É possível, então, descrever sucintamente IA como "o esforço em automatizar tarefas intelectuais normalmente realizadas por humanos" (CHOLLET, 2021).

Geralmente quando nos referimos à inteligência artificial, estamos na verdade citando uma enorme classe de algoritmos e ferramentas. Até meados de 1980 não haviam ainda implementações da característica de "aprendizado" na IA, as primeiras aplicações envolviam regras rígidas criadas por programadores para, por exemplo, fazer a máquina jogar xadrez. Este tipo de IA é conhecida como "Symbolic AI"[1].

Posteriormente, ao invés de termos um programador escrevendo regras explícitas para a máquina reproduzir, surgem implementações onde a máquina analisa "dados de entrada", observa as correspondentes "respostas" e através de diversos possíveis algoritmos é treinada para obter as respostas a partir dos dados de entrada. Surge então uma subclasse da inteligência artificial conhecida como *Machine Learning* (MITCHELL, 1997), e posteriormente outras como o *Deep Learning* (ver figura 2).

[1] Também conhecida como "*Good Old-Fashioned Artificial Intelligence*"

Figura 2 – Diagrama representando a relação entre inteligência artificial, *machine learning* e *deep learning*.



Fonte: Adaptado de Chollet (2021).

2.1.1 Deep Learning

O *Deep Learning* (LECUN; BENGIO; HINTON, 2015; GOODFELLOW; BENGIO; COURVILLE, 2016) é, atualmente, uma das mais famosas subáreas do *Machine Learning* baseadas no conceito de redes neurais, sendo responsável por inúmeras aplicações atuais em problemas que são naturais aos seres humanos como reconhecimento de fala, classificação de imagens, direção autônoma e criação de assistentes virtuais.

O termo *deep* (profundo) vem da ideia da criação de grafos constituídos de inúmeras camadas de processamento, sendo estes grafos as conhecidas redes neurais. Estas por sua vez são constituídas de vários neurônios, recebendo esses nomes devido à sua semelhança com um modelo simplificado do cérebro. Porém, apesar da semelhança não é intenção destes algoritmos simularem o cérebro humano. Podemos pensar nesta ferramenta como um "*framework* matemático para obtermos representações a partir de dados" (CHOLLET, 2017).

Este tipo de IA se tornou popular nos últimos anos já que oferece uma melhor performance em diversos problemas e por sua facilidade em lidar com dados "pouco lapidados".

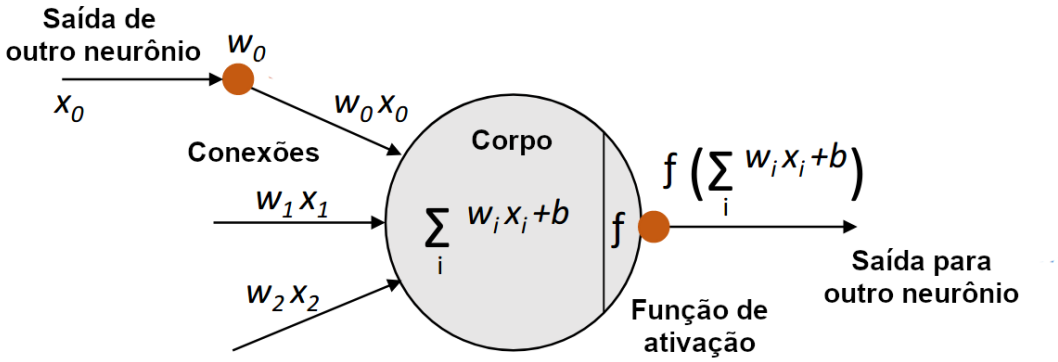
2.1.1.1 Neurônios

Conhecidos como neurônios artificiais[2], constituem uma das partes mais elementares das redes neurais. Lembrando vagamente neurônios reais, estas estruturas são capazes de receber sinais de outros neurônios, transformá-los e passar a informação modificada para os próximos neurônios.

[2] Também referenciados como "*units*".

Um neurônio funciona basicamente da seguinte forma: Sinais (números) são enviados por diversos neurônios através de conexões para um neurônio receptor, onde cada conexão possui um valor chamado de peso[3] (w_i). As informações enviadas chegam até o neurônio receptor e são multiplicadas por seus devidos pesos. Após isto, o neurônio receptor soma todos estes valores e adiciona um escalar b, conhecido como *bias*. Antes de passar o sinal adiante, o resultado destas somas é inserido em uma função não-linear, conhecida como função de ativação. Um esquema exemplificando este processo pode ser visualizado na figura 3.

Figura 3 – Esquema representando a estrutura de um neurônio.



Fonte: Adaptado de Bom; Faria (2021).

2.1.1.2 Redes neurais

Se pensarmos em uma rede neural como um grafo, os neurônios constituirão os vértices e as conexões entre os neurônios serão as arestas.

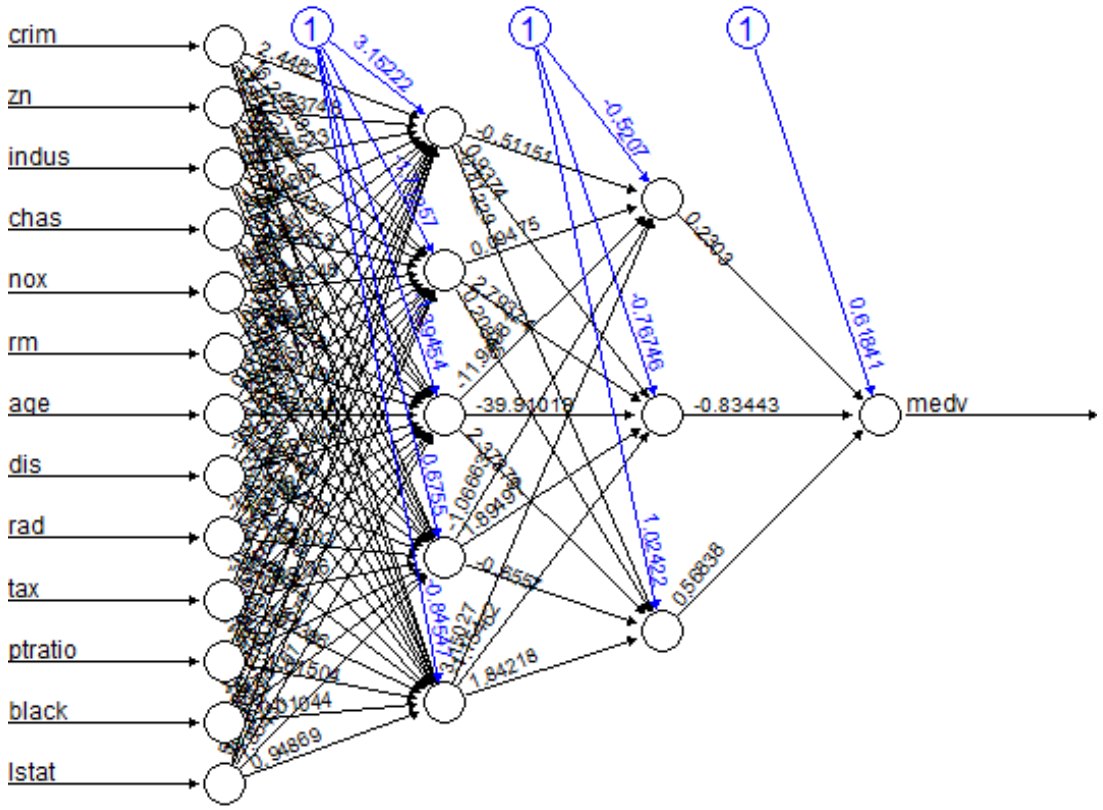
Existem diversas arquiteturas possíveis para uma rede neural, sendo a mais comum conhecida como *multilayer perceptron* (MLP). A MLP é uma arquitetura constituída de diversas camadas de neurônios, conectados com todos os neurônios da camada posterior e anterior (*fully connected layers*). Aqui, a informação adentra a estrutura por uma camada de *input*, atravessa uma série de camadas (*hidden layers*) e saem da rede transformadas em uma camada final *output*. Uma rede com este tipo de fluxo de informação é conhecido como *feedforward*. Classicamente, a MLP é aplicada à redes rasas (poucas camadas), já no *deep learning* temos diversas camadas, as redes neurais profundas.

Um exemplo de rede neural pode ser observado na figura 4. Esta rede é uma solução de um problema clássico conhecido como *boston house price prediction*, onde a intenção é prever o preço de uma casa baseado em alguns parâmetros como taxa de criminalidade por cabeça (crim) e número de quartos por habitação (rm). A informação referente a estes parâmetros é apresentada para a rede através de uma camada de entrada e é transformada por duas camadas escondidas até sair por uma camada de *output* constituída de um

[3] Também conhecido como *weight*.

neurônio, cuja saída indicará o valor da residência. Os neurônios são representados por círculos pretos, o *bias* de cada neurônio em números azuis e os *weights* de cada conexão em números pretos.

Figura 4 – Esquema representando uma rede neural.



Fonte: Alice (2015).

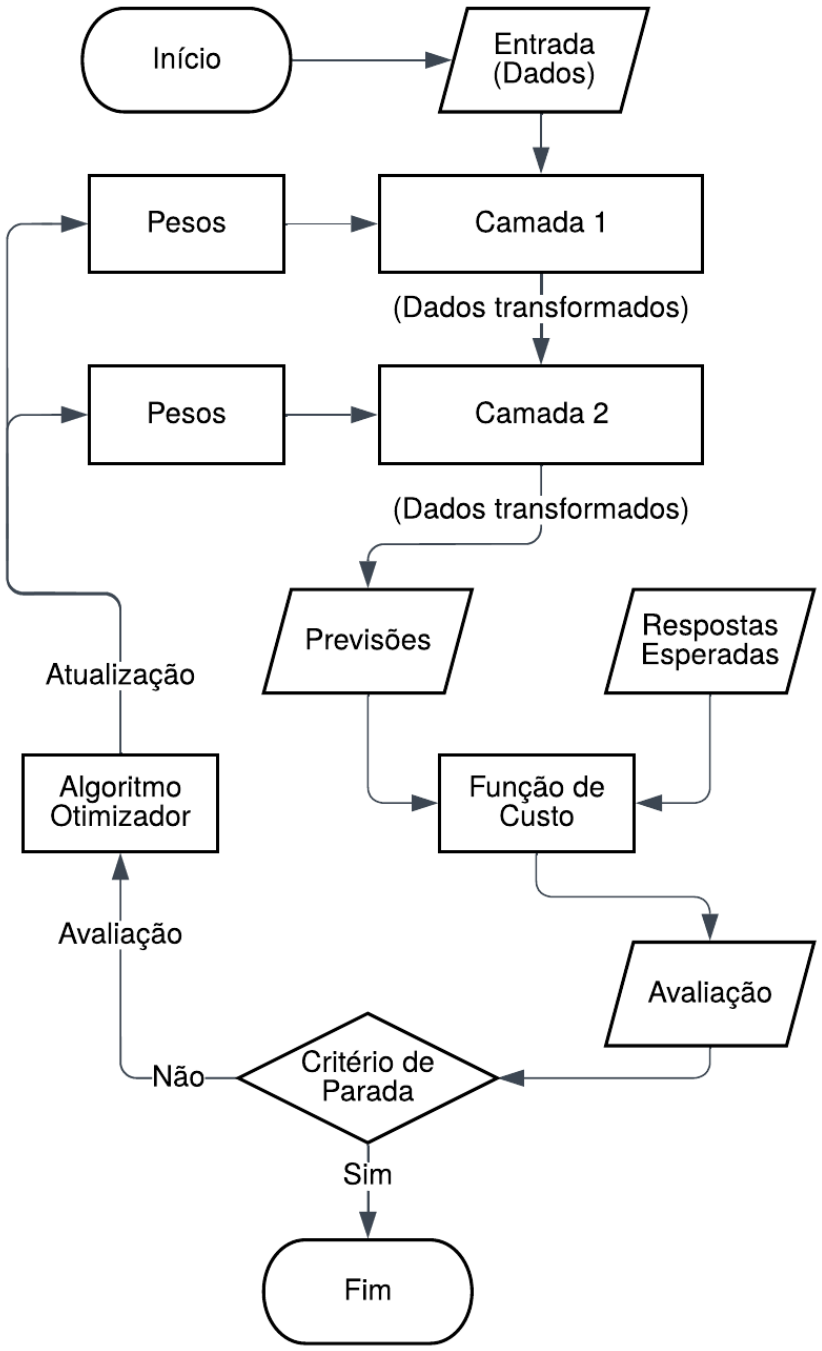
2.1.1.3 Treinamento da IA

O treinamento no *Deep Learning* consiste simplificadamente em alimentar uma série de camadas de redes neurais com dados e ajustar os parâmetros da rede para que ela consiga obter as respostas esperadas, as quais são fornecidas inicialmente, podendo assim, posteriormente, alimentar a rede com novos dados e novas respostas serem obtidas.

Isto é feito da seguinte forma: Os dados *input* são armazenados em tensores e utilizados como entradas na rede neural. Os parâmetros w_i e b da rede são inicialmente aleatórios e posteriormente serão otimizados para que ao final da rede se tenha uma saída que corresponde à "resposta" do problema a ser resolvido, o que deverá ser interpretada pelo usuário.

No final da rede neural, é obtida uma saída correspondente aos dados de entrada modificados pelos neurônios que serão comparados com resultados esperados para o conjunto de dados analisado, resultados aos quais também são fornecidos. O resultado

Figura 5 – Fluxograma simplificado de um algoritmo de *Deep Learning* com duas camadas de neurônios.



Fonte: Elaborado pelo autor.

proveniente da rede e o resultado esperado são analisados por uma função de custo que proporcionará uma avaliação do quão bem a rede neural conseguiu prever a resposta esperada. Esta avaliação será utilizada por um algoritmo otimizador para reajustar os *weights* e *bias* da rede a fim de que se melhore sua performance gradualmente, no processo chamado de treino. Esse processo está exemplificado simplificadamente com o fluxograma da figura 5. Este treino é repetido por um número pré-definido de vezes, as quais são chamadas de "épocas".

2.1.2 Teorema da aproximação universal

Uma rede neural consiste de uma cadeia de operações tensoriais dos dados de "*input*" capazes de resolver diversos problemas do mundo real. O que está por trás desta capacidade é uma suposta característica das redes neurais: São aproximadores universais de funções.

Os teoremas de aproximação universal são uma tentativa de se provar matematicamente que este tipo de ferramenta conseguiria aproximar "qualquer" função dado um determinado domínio.

Um dos resultados mais interessantes (ou mais conhecidos) são os obtidos por Kurt Hornik (HORNIK; STINCHCOMBRE; WHITE, 1989) em 1989. Kurt e colaboradores propuseram provar rigorosamente, através de uma variedade de definições teoremas e corolários, que redes neurais do tipo *feedforward* com pelo menos uma camada de neurônios podem aproximar qualquer função mensurável em um domínio finito com qualquer grau de precisão, desde que fossem cedidas uma quantidade necessária de neurônios. Desta forma, acabaram por firmar em seu trabalho que as redes neurais se portam como, ou são, uma classe de "aproximadores universais".

Esta ideia de redes neurais como aproximadores universais de funções é também o que garante o sucesso do *Deep Learning* em suas diversas aplicações nos últimos anos. Resultados mais recentes como de Lyn H. et al (LIN; TEGMARK; ROLNICK, 2017) reafirmam a eficiência de redes neurais mesmo que com poucas camadas[4]. Também mostram que se os dados a serem analisados pela rede neural são gerados por processos estatísticos, como os dos fenômenos físicos, então uma rede neural, mesmo com poucas camadas e neurônios, pode ser muito eficiente em lidar com estes problemas.

2.2 Equações Diferenciais

Segundo os autores Boyce e Diprina (2010), equações diferenciais são, simplesmente, equações que contém derivadas. São relações expressas em linguagem matemática envolvendo "taxas segundo a qual as coisas acontecem", as derivadas.

Muitos dos fenômenos físicos, princípios ou leis que descrevem o comportamento do mundo físico envolvem estas relações e as taxas de variação de seus parâmetros. Fazendo então com que grande parte destes processos físicos possuam modelos matemáticos compostos de equações diferenciais.

[4] O que é conhecido como *Cheap Learning*

2.2.1 Tipos

Equações diferenciais podem ser classificadas em diferentes tipos. É interessante que conheçamos a classificação das equações que almejamos resolver já que dependendo de sua classe, diferentes métodos serão eficazes, ou não, para cada determinado tipo.

Uma classificação importante é com relação a quantidade de variáveis independentes contidas na função que resolve a equação. Caso haja apenas uma variável, então é uma equação diferencial ordinária, caso haja mais, então é uma equação diferencial parcial, um tipo comum em muitos problemas físicos importantes.

Por exemplo, a equação 2.1 referente a um oscilador harmônico amortecido forçado é uma equação ordinária, e a equação 2.2 referente a equação de Burgers é uma equação parcial.

$$\alpha\left(\frac{\partial^2 u(t)}{\partial t^2}\right) + \beta\left(\frac{\partial u(t)}{\partial t}\right) + \gamma u(t) = F(t), \quad \alpha, \beta, \gamma = cte; \tag{2.1}$$

$$\frac{\partial u(t,x)}{\partial t} + u(t,x)\frac{\partial u(t,x)}{\partial x} - \alpha\frac{\partial^2 u(t,x)}{\partial x^2} = 0, \quad \alpha = cte. \tag{2.2}$$

Uma equação também pode ser classificada com relação a sua ordem. A ordem de uma equação diferencial se refere a ordem da derivada de maior ordem presente na expressão. A equação do oscilador harmônico 2.1 é de segunda ordem (no tempo) e a equação de Burgers 2.2 é de segunda ordem no espaço (mas de primeira ordem no tempo).

Alguns problemas físicos podem ser descritos por sistemas de equações diferenciais, quando temos mais de uma função desconhecida. Por exemplo, podemos escrever a equação do oscilador harmônico 2.1 na forma de um sistema a fim de reduzir sua ordem de 2 para 1:

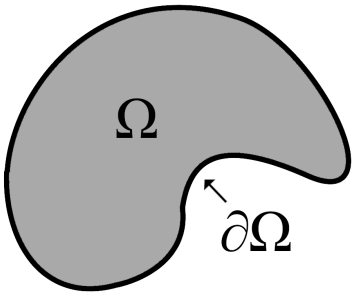
$$\begin{cases} v(t) = u_t(t), \\ \alpha v_t(t) + \beta v(t) + \gamma u(t) = F(t), \quad \alpha, \beta, \gamma = cte. \end{cases} \tag{2.3}$$

Ainda existem outros inúmeros tipos de classificações deste tipo de equação que não foram do escopo deste trabalho, como elípticas, hiperbólicas, parabólicas, integro-diferenciais, entre outras.

2.2.2 Condições Iniciais e Condições de contorno

Segundo Iório (2010) e Strauss (2008), sendo uma solução de equação diferencial $u(t,x)$, as condições iniciais são as condições que ditam o estado inicial do sistema em um tempo inicial t_0, que pode ser descrito por $u(t_0,x) = \phi(x)$. Já as condições de contorno se referem, em casos de intervalos finitos, às condições impostas nos limites do intervalo conhecido. No caso suposto, considerando um domínio espacial $\Omega \to [a,b]$, teríamos informação sobre o sistema em $u(t,a) = \phi_1(t)$ e $u(t,b) = \phi_2(t)$.

Figura 6 – Esquema representando um domínio Ω e um contorno $\partial\Omega$. A região cinza representa a área de interesse onde almejamos encontrar uma solução para a equação diferencial.



Fonte: Elaborado pelo autor.

Em um problema onde são conhecidas as condições iniciais tanto quanto as condições de contorno, é dado o nome de problema de condições mistas. Além disto, existem classificações para os tipos de condições iniciais e de contorno.

Recebe o nome de condições de Dirichlet quando é conhecido o valor da solução em determinados pontos ou intervalos, na forma da equação 2.4:

$$\alpha u(t,x) = f(t,x), \qquad t,\ x \in \partial\Omega, \qquad \alpha = const. \tag{2.4}$$

Onde $\partial\Omega$ é o contorno completo ou parcial de um domínio Ω.

As condições de Neumann se refere a situação em que é conhecido o valor da derivada em determinados pontos ou intervalos, na forma da equação 2.5, no caso de uma derivada com relação a alguma variável independente n do sistema[5].

$$\alpha\frac{\partial u(t,x)}{\partial n} = f(t,x), \quad t,\ x \in \partial\Omega \quad \alpha = const. \tag{2.5}$$

Em alguns problemas há o par condições de Dirichlet e Neumann, no sentido de que são conhecidos tanto 2.4 quanto 2.5. Este tipo de condições são conhecidas como condições de Cauchy, e constituíram a maioria dos casos estudados neste trabalho.

2.2.3 Linearidade e não-linearidade, e sua relação com a física

Segundo Strauss (2008) é possível criar um conceito de linearidade respeitando as relações 2.6:

$$\mathcal{N}(u+v) = \mathcal{N}u + \mathcal{N}v \quad e \quad \mathcal{N}(cu) = c\mathcal{N}u, \quad c = cte, \tag{2.6}$$

onde $\mathcal{N}$ é um operador diferencial e u e v são funções quaisquer. Se as equações podem ser escritas nos formatos

$$\mathcal{N}u = 0 \quad ou \quad \mathcal{N}u = g(t,\vec{x}), \tag{2.7}$$

[5] No caso da equação 2.5, n pode ser a coordenada t ou x

então estas equações são lineares, homogêneas ou não-homogêneas respectivamente. Nesse sentido, a equação do oscilador harmônico (2.1) é linear e não-homogênea e a equação de Burgers (2.2) é não-linear e homogênea.

Diversos fenômenos físicos não podem ser modelados satisfatoriamente por equações diferenciais lineares, por exemplo, a equação de movimento de um pêndulo simples já possui um termo não-linear. Este tipo de equação tem soluções exatas de mais difícil obtenção. Muitas vezes, não há solução conhecida ou é possível que sequer haja uma solução, assim, os métodos computacionais e numéricos se tornam extremamente úteis se queremos extrair a física destas equações.

2.3 Physics-informed Neural Networks

Como comentado anteriormente, equações diferenciais parciais não-lineares podem ser extremamente complicadas de se resolver analiticamente. Neste sentido, o uso de computadores na solução destas equações pode ser uma alternativa útil. O método numérico geralmente entrega ao usuário (cientista que está realizando os cálculos) uma tabela com números contendo valores das variáveis independentes (pontos) e os valores correspondentes à solução da equação nestes pontos. Estes valores da solução numérica são então analisados graficamente para que alguma informação (física) possa ser extraída da equação diferencial.

Já existem inúmeros algoritmos muito eficientes que são empenhados nesta tarefa, como método de diferenças finitas (ou elementos finitos), métodos de Runge-kutta, Método de Euler, etc (PRESS et al., 2007). Mesmo assim, dependendo do tipo da equação ainda é extremamente complicado resolver determinadas equações, principalmente as não-lineares. É neste sentido que, atualmente, surgem alternativas de algoritmos que se utilizam de técnicas de *machine learning*, como as PINN.

As PINN (RAISSI; PERDIKARIS; KARNIADAKIS, 2017a; RAISSI; PERDIKARIS; KARNIADAKIS, 2017b; RAISSI; PERDIKARIS; KARNIADAKIS, 2019) são uma classe de redes neurais (aproximadores universais) que são treinadas para representarem "leis da física" impostas por modelos contendo equações diferenciais não-lineares (ou lineares). Também podem ser vistas como um algoritmo ou metodologia em si.

Estas redes neurais foram propostas para poderem ser utilizadas em duas categorias principais de problemas que envolvem equações diferenciais: A solução numéricas das devidas equações (*data-driven solution*) e o ajuste de coeficientes da equação diferencial através de dados experimentais (*data-driven discovery*). Ambas estas aplicações são orientadas a dados (*data-driven*).

Foi proposto pelos autores, inicialmente, duas abordagens (ou algoritmos distintos) dependendo da natureza dos dados disponíveis: uma abordagem discreta (*discrete time*

models) e uma contínua (*continuous time models*). A segunda abordagem é descrita como "aproximadores de funções espaço-temporais *data-efficient*"[6], e foi a abordagem utilizada neste trabalho.

Na época de sua publicação, os algoritmos propostos por Raissi e colaboradores se demonstraram, em seus artigos iniciais, muito eficientes para problemas clássicos da mecânica dos fluidos, mecânica quântica, sistemas de reação-difusão, entre outros. Sendo então uma possível futura alternativa na solução de equações diferenciais parciais não-lineares.

2.3.1 PINN como solucionador de equações

O método consiste em criar redes neurais profundas e utilizar sua característica de aproximadores universais para aproximar a solução de equações diferenciais. Estas redes serão treinadas para representarem as informações que estão contidas na equação diferencial em si e em dados que serão cedidos pelo usuário para realização do processo de treinamento do modelo de *machine learning*. Estes dados serão basicamente compostos por pontos correspondentes as condições de contorno e condições iniciais do problema a ser resolvido. Caso seja possível e/ou necessário, pontos de uma solução exata podem ser cedidos para fins de comparação com a solução numérica ao final do treinamento. é possível tentar resolver uma equação diferencial da forma[7]:

$$u_t + \mathcal{N}[u] = 0, \quad x, t \in \Omega. \tag{2.8}$$

A equação 2.8 deve ser escrita na forma genérica (as vezes referenciada como "forma de resíduo"):

$$f := u_t + \mathcal{N}[u]. \tag{2.9}$$

Onde $u(t, x)$ representa a solução inferida pela máquina, x representa os graus de liberdade espaciais do sistema e t uma coordenada temporal, $\mathcal{N}$ é um operador diferencial não-linear (ou linear) e o domínio Ω é um subconjunto de $\Re^n$. As derivadas contidas na equação diferencial são calculadas por meio de "diferenciação automática" (BAYDIN et al., 2018), utilizando funções contidas em um *framework* de *machine learning* de escolha do programados, como o TensorFlow (ABADI et al., 2016).

O processo de otimização da rede e o que ela representará ao fim do treinamento são guiados por uma função de custo que deve ser muito bem definida. De forma genérica, para um modelo contínuo é possível definir a função de custo de acordo como o somatório

[6] No sentido de não necessitar de grande quantidade de dados.

[7] Nos artigos mais antigos, apenas estão presentes equações de primeira ordem no tempo, apresentando uma limitação do método. Porém trabalhos mais recentes tem encontrado formas de resolver equações de ordem maior.

dos seguintes erros quadráticos médios (MSE):

$$MSE = MSE_u + MSE_f. \tag{2.10}$$

Onde

$$MSE_u = \frac{1}{N_u}\sum_{i=1}^{N_u} |u(t_u^i, x_u^i) - u^i|^2, \tag{2.11}$$

e

$$MSE_f = \frac{1}{N_f}\sum_{i=1}^{N_f} |f(t_f^i, x_f^i)|^2. \tag{2.12}$$

Aqui $[t_u^i, x_u^i, u^i]_{i=1}^{N_u}$ representa as condições iniciais e de contorno de $u(t, x)$, e $[t_f^i, x_f^i]_{i=1}^{N_f}$ representam os "pontos de colocação" para $f(t, x)$. O erro MSE_u, erro referente as condições de fronteira, forçará a superfície (solução) gerada pela rede neural a corresponder as condições iniciais e de contorno dadas pelos dados cedidos pelo usuário, enquanto o erro MSE_f, erro referente à equação diferencial em si, fará com que a rede represente a equação diferencial escolhida.

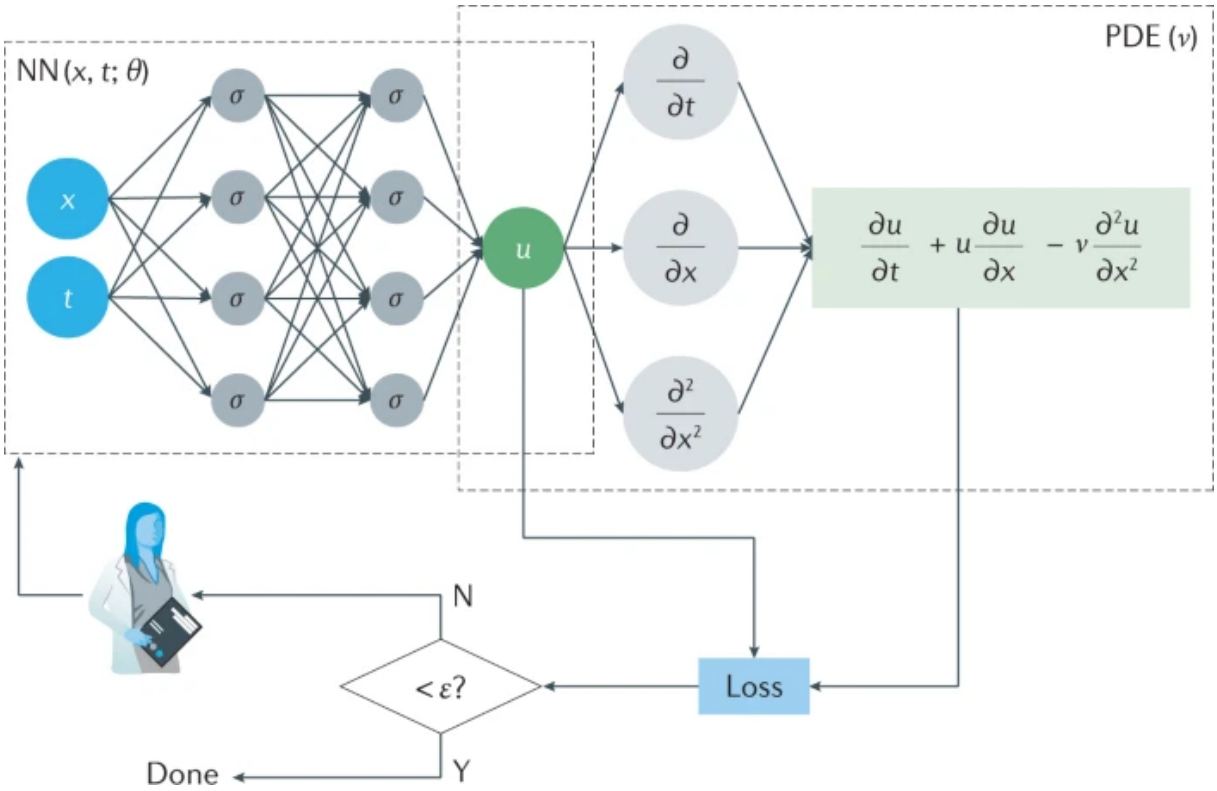
Um esquema representando o método pode ser visualizado na figura 7. Para resolvermos uma equação (PDE[8]), primeiramente é construída uma rede neural $NN(x, t; \theta)$, onde θ é o conjunto de *weights* e *biases*, x representa uma variável independente espacial, t representa a coordenada tempo e σ é uma ativação não-linear[9]. Assim, as entradas da rede neural serão as variáveis independentes do sistema e a saída será $u(x, t)$, fazendo com que a rede represente a solução da equação em si. Agora, utilizamos a rede neural para calcular as derivadas contidas na equação (aqui representada pela equação de Burgers) através de diferenciação automática para ser calculado o erro MSE_f. Também são gerados valores para comparar com os pontos conhecidos das condições de contorno para ser calculado o erro MSE_u. Após a rede ser treinada, com o custo (*loss*) é possível verificar se a rede cumpre um critério de convergência ϵ. Se sim, a rede então provavelmente aproxima numericamente a solução da equação diferencial e pode ser utilizada para os fins do usuário. Se não, então deve ser repensada a forma de treinar a rede, e todo o processo ser refeito.

2.3.2 Alguns exemplos reproduzidos da literatura

Aqui serão apresentados duas aplicações do algoritmo PINN atuando como *solver* em equações diferenciais parciais não-lineares. Estes dois exemplos foram reproduzidos da literatura (RAISSI; PERDIKARIS; KARNIADAKIS, 2019), seus algoritmos e implementações (códigos) serviram como base para as soluções dos problemas propostos no capítulo seguinte.

[8] *Partial Differential Equation.*
[9] Neste estudo, foi utilizada tangente hiperbólica.

Figura 7 – Esquema representando o método referente às *Physics-Informed Neural Networks*



Fonte: Karniadakis, et al 2021.

2.3.2.1 Equação de Burgers

A Equação de Burgers está presente em diversas áreas da matemática aplicada, como mecânica dos fluidos, acústica não-linear, dinâmica de gases e fluxo de tráfego. Para valores pequenos do parâmetro de viscosidade ν esta equação diferencial pode se tornar difícil de resolver pelos métodos numéricos mais conhecidos.

É interessante reproduzir este problema já que, apesar de ser uma equação não-linear, é um caso simples, com solução conhecida e contida na literatura, servindo de uma boa introdução para ser compreendida a metodologia e/ou capacidade da ferramenta (PINN).

Em uma dimensão espacial, a equação de Burgers com condições de contorno de Dirichlet pode ser escrita como:

$$\begin{aligned} &\frac{\partial u(t,x)}{\partial t} + u(t,x)\frac{\partial u(t,x)}{\partial x} - \left(\frac{0.01}{\pi}\right)\frac{\partial^2 u(t,x)}{\partial x^2} = 0, \\ &u(0,x) = -sin(\pi x), \qquad x \in [-1,1], \\ &u(t,-1) = u(t,1) = 0, \qquad t \in [0,1]; \end{aligned} \tag{2.13}$$

onde o parâmetro de viscosidade é $\nu = 0.01/\pi$ e $u(t,x)$ é a solução que almejamos aproximar com a PINN.

De acordo como exposto anteriormente, para o algoritmo ser aplicado deve ser escrita a eq. 2.13 no formato de resíduo (Eq. 2.9). Desta forma é obtido:

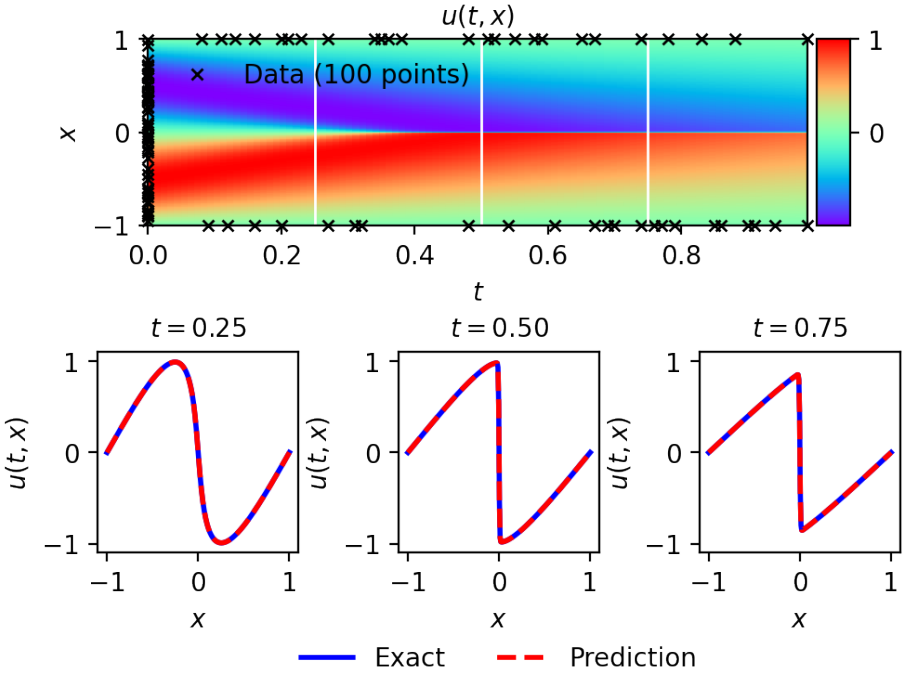
$$f := u_t + uu_x - \left(\frac{0.01}{\pi}\right) u_{xx}. \tag{2.14}$$

Um *script* com a solução numérica da Eq. 2.14 utilizando PINN foi disponibilizado por Raissi na plataforma GitHub (RAISSI, 2020). Assim é possível reproduzir a solução com certa facilidade.

Após definida a função de custo (2.10) e criado um conjunto de dados com as condições iniciais e de contorno, é necessário apenas definir parâmetros do algoritmo de *deep learning*. Neste caso específico, foi construída uma rede neural com 8 camadas (*hidden layers*) de 20 unidades cada (neurônios). São duas entradas referentes a valores para as coordenadas x e t e uma única saída com o valor da função numérica $u(x,t)$ aproximada pela rede neural.

Após treinada a rede neural por 50.000 épocas utilizando o otimizador L-BFGS-B em uma TPU (*tensor processing unit*) disponível em um serviço na nuvem[10], foi chegado ao custo final (*loss*) de 4.14×10^{-6} em apenas 567 segundos de treino. O resultado pode ser visualizado na Figura 8.

Figura 8 – Solução numérica para a Equação de Burgers utilizando PINN. A curva azul representa a solução exata e a curva tracejada vermelha representa a solução obtida usando PINN.



Fonte: Elaborado pelo autor.

[10] Foi utilizado o Google Colab.

Para as condições iniciais e de contorno para treino foram sorteados e utilizados 100 pontos dentre os disponíveis no conjunto de dados. Neste caso, como era conhecida a solução da equação também disponibilizada no conjunto de dados, foi possível comparar a solução obtida com a solução exata. Assim, o erro absoluto médio (MAE) foi de $1,48 \times 10^{-3}$.

2.3.2.2 Equação de Schrodinger não-linear

A Equação de Schrodinger não-linear é uma equação de campo utilizada para estudar sistemas quânticos, propagação de ondas em fibras ópticas, condensados de Bose-Einstein e ondas de plasma. Um exemplo desta equação, unidimensional, com condições de contorno periódicas pode ser descrita como

$$\begin{aligned} &i\frac{\partial h(t,x)}{\partial t} + 0.5\frac{\partial^2 h(t,x)}{\partial x^2} + h(t,x)|h^2(t,x)| = 0, \\ &h(0,x) = 2sech(x), \qquad x \in [-5,5], \\ &h(t,-5) = h(t,5), \qquad t \in [0,\pi/2], \\ &h_x(t,-5) = h_x(t,5). \end{aligned} \tag{2.15}$$

Aqui, $h(t,x) = u(t,x) + iv(t,x)$, desta forma, para resolvermos a Eq. 2.15 deveremos na verdade resolver um sistema de equações diferenciais que serão escritas no formato da Eq. 2.9:

$$\begin{cases} f_u := u_t + 0.5v_{xx} + (u^2 + v^2)v, \\ f_v := v_t - 0.5u_{xx} - (u^2 + v^2)u. \end{cases} \tag{2.16}$$

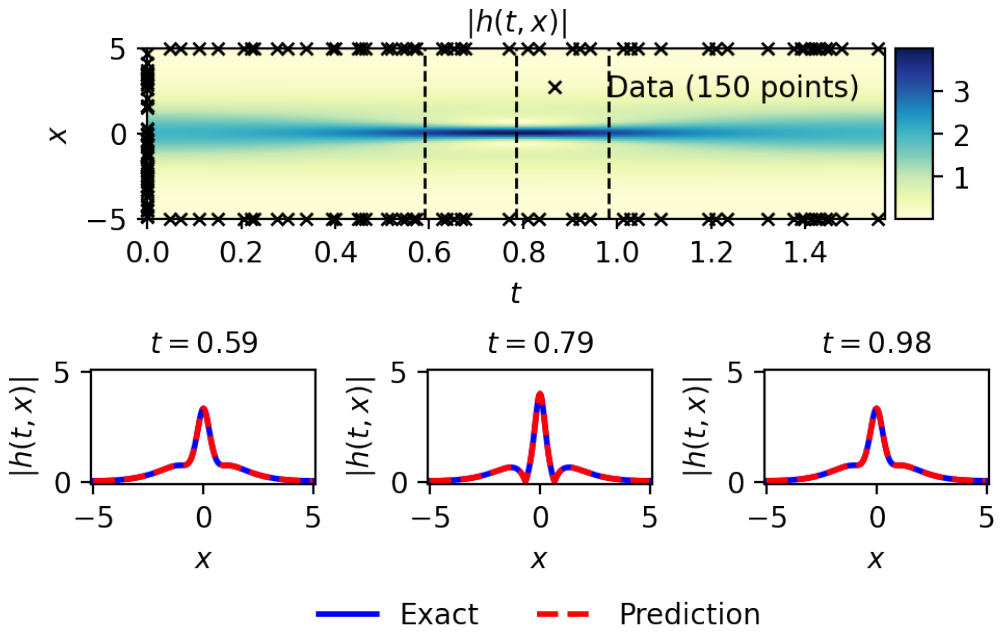
Computacionalmente a rede neural representará uma função complexa e terá um *output* duplo, de forma que a solução $h(t,x)$ seja representada pelas funções $u(x,t)$ e $v(t,x)$, ou seja, são uma tupla escrita como $h(t,x) = [u(t,x),\ v(t,x)]$.

O conjunto de dados para treino será composto de uma parte real e uma parte imaginária que servirão para otimização dos parâmetros $u(x,t)$ e $v(x,t)$ respectivamente. Compreender este caso é importante para que seja possível resolver problemas envolvendo sistemas de equações e posteriormente, de ordem mais alta que 1 no tempo.

Uma rede neural com 4 camadas de 100 unidades cada foi treinada uma vez com o otimizador ADAM por 50.000 épocas e uma segunda vez com o otimizador L-BFGS-B por mais 20.000 épocas. Na prática, o primeiro otimizador serve para iteradamente procurar mínimos na função de erro, e o segundo para refinar o resultado final obtido. Com 150 pontos para treino foi chegado ao resultado da Figura 9. Neste caso, o erro (MAE[11]) foi de $1,80 \times 10^{-3}$ para $u(t,x)$, $2,33 \times 10^{-3}$ para $v(t,x)$ e $1,40 \times 10^{-3}$ para $|h(t,x)|$.

[11] *Mean Absolute Error.*

Figura 9 – Solução numérica para a Equação do Schrodinger não-linear utilizando PINN.



Fonte: Elaborado pelo autor.

2.4 Relatividade Geral e Gravitação

A Teoria da Relatividade Geral foi uma teoria proposta por Albert Einstein nas primeiras décadas do século XX, trazendo uma nova descrição para a Gravitação de um ponto de vista geométrico.

Esta teoria entende a gravidade não como uma força, mas como uma deformação do espaço-tempo (quadridimensional). Esta deformação na geometria do espaço é causada pela presença de matéria e radiação, tendo seus fenômenos descritos por um conjunto de equações diferenciais conhecidas como Equações de Campo de Einstein.

Assim como a presença de carga elétrica no espaço altera o campo eletromagnético e os fenômenos associados são descritos por um conjunto de equações (Equações de Maxwell), a Relatividade Geral entende a massa como uma espécie de carga do campo gravitacional, e através das equações de Einstein é possível compreender diversos fenômenos, como ondas gravitacionais, buracos negros, diversos tipos de estrelas e buracos de minhoca.

2.4.1 Equações de campo

As equações de campo da gravitação são mais complicadas do que as do eletromagnetismo já que as equações de Maxwell são lineares devido ao campo em si não portar carga, enquanto campos gravitacionais carregam energia e momento, por consequência, interagem com sua própria fonte. Assim, as equações de campo da gravitação são equações diferenciais parciais não-lineares, onde a não linearidade representa o efeito da gravitação

sobre si mesma (WEINBERG, 1972).

Estas equações podem ser deduzidas de algumas formas e podem ser dadas no formato completo pela equação tensorial 2.17:

$$G_{\mu\nu} = \Lambda g_{\mu\nu} + \kappa T_{\mu\nu} \tag{2.17}$$

Onde μ e ν são índices dos tensores ($\mu, \nu = 0, 1, 2, 3$), κ é chamada de constante de acoplamento ($\kappa = 8\pi G/c^4$), G é a constante de gravitacional de Newton, $T_{\mu\nu}$ é o tensor de energia-momento, que descreve o fluxo e densidade de energia e momento no espaço-tempo, Λ é a constante cosmológica e $g_{\mu\nu}$ é a métrica que descreve a geometria e estrutura causal do espaço-tempo.

Também, na equação 2.18, o tensor de einstein ($G_{\mu\nu}$) é descrito por:

$$G_{\mu\nu} = R_{\mu\nu} - \frac{1}{2} R g_{\mu\nu} \tag{2.18}$$

Aqui, $R_{\mu\nu}$ é o tensor de curvatura de Ricci, que indica o quanto o tensor de uma métrica difere localmente da métrica do espaço euclidiano e R é o escalar de curvatura ou Escalar de Ricci, que contem a informação do traço do tensor de Ricci.

O tensor métrico $g_{\mu\nu}$ pode ser descrito, de certa forma, como um função que nos diz como calcular o intervalo entre dois pontos no espaço-tempo. A métrica determina o quadrado de um elemento de linha infinitesimal ds, na forma:

$$ds^2 = g_{\mu\nu} dx^\mu dx^\nu, \tag{2.19}$$

onde dx^μ e dx^ν são elementos de linha infinitesimais de um quadrivetor de coordenadas. Por exemplo, no caso de um espaço-tempo plano, também conhecido como espaço-tempo de Minkowski, composto de quatro coordenadas (t, x, y, z), sua métrica pode ser descrita pela expressão 2.20:

$$ds^2 = g_{\mu\nu} dx^\mu dx^\nu = c^2 dt^2 - dx^2 - dy^2 - dz^2 \tag{2.20}$$

Escrevendo o tensor $g_{\mu\nu}$ em forma matricial, é tido:

$$g_{\mu\nu} = \begin{pmatrix} c^2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \tag{2.21}$$

Também é possível escrever a métrica do espaço plano em coordenadas esféricas, de acordo com as expressões 2.22 e 2.23.

$$ds^2 = c^2 dt^2 - dr^2 - r^2 d\theta^2 - r^2 sin^2\theta d\varphi = g_{\mu\nu} dx^\mu dx^\nu \tag{2.22}$$

$$g_{\mu\nu} = \begin{pmatrix} c^2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -r^2 & 0 \\ 0 & 0 & 0 & -r^2 sin^2\theta \end{pmatrix} \tag{2.23}$$

As equações de campo para o vácuo podem ser obtidas considerando que não há radiação presente ou outro tipo de campo, implicando que o tensor energia-momento $T_{\mu\nu}$ é nulo na equação 2.17. Assim, é tido:

$$G_{\mu\nu} = \Lambda g_{\mu\nu} \tag{2.24}$$

Agora é possível fazer a seguinte análise: desconsiderando o tensor energia-momento, como na equação 2.24, e igualando-a com a expressão para o tensor de Einstein em função do tensor de Ricci (2.18) é obtida a expressão

$$R_{\mu\nu} - \frac{1}{2} R g_{\mu\nu} = \Lambda g_{\mu\nu} \tag{2.25}$$

Contraindo os índices da equação 2.24 com $g^{\mu\nu}$ [12] uma expressão para o escalar de Ricci é obtida:

$$\begin{aligned} g^{\mu\nu} R_{\mu\nu} - \frac{1}{2} R g^{\mu\nu} g_{\mu\nu} &= \Lambda g^{\mu\nu} g_{\mu\nu} \ , \\ R - \frac{4}{2} R &= 4\Lambda \ , \\ R &= -4\Lambda. \end{aligned} \tag{2.26}$$

Substituindo este resultado na equação 2.25, pode ser chegado na expressão

$$R_{\mu\nu} = -\Lambda g_{\mu\nu} \tag{2.27}$$

Caso trabalhemos em um sistema gravitacional onde não há (ou não consideramos) os efeitos da constante cosmológica, é possível concluir que:

$$R_{\mu\nu} = 0 \tag{2.28}$$

As equações 2.27 e 2.28 são conhecidas como equações de Einstein no vácuo, com e sem constante cosmológica respectivamente.

No lado esquerdo destas equações temos o tensor de Ricci, que pode ser representado explicitamente pela seguinte equação:

$$R_{\mu\nu} = \frac{\partial \Gamma^{\lambda}_{\mu\lambda}}{\partial x^{\nu}} - \frac{\partial \Gamma^{\lambda}_{\mu\nu}}{\partial x^{\lambda}} + \Gamma^{\eta}_{\mu\lambda}\Gamma^{\lambda}_{\nu\eta} - \Gamma^{\eta}_{\mu\nu}\Gamma^{\lambda}_{\lambda\eta}, \tag{2.29}$$

onde $\Gamma^{\lambda}_{\mu\nu}$ é o símbolo de Christoffel, e pode ser escrito em termos das componentes da métrica como

$$\Gamma^{\lambda}_{\mu\nu} = \frac{1}{2} g^{\lambda\eta} \left(\frac{\partial g_{\eta\mu}}{\partial x^{\nu}} + \frac{\partial g_{\eta\nu}}{\partial x^{\mu}} - \frac{\partial g_{\mu\nu}}{\partial x^{\eta}} \right). \tag{2.30}$$

[12] $g^{\mu\nu} g_{\mu\nu} = 4$ no espaço-tempo 4-dimensional

2.4.2 Espaço-tempo estático e esfericamente simétrico

As equações diferenciais parciais correspondentes às equações de campo de Einstein podem ser resolvidas exatamente em alguns casos. As métricas correspondentes a estes casos são úteis para descrevermos e compreendermos sistemas gravitacionais como estrelas, buracos negros e buracos de minhoca, e inclusive ser descrita sua evolução.

Neste nosso estudo, foi considerado resolver exatamente estas equações na configuração mais simples: casos esfericamente simétricos e estáticos. Estes casos são importantes já que os testes da Relatividade Geral propostos por Einstein são conduzidos em espaço vazio e em campos gravitacionais que são aproximadamente estáticos e esfericamente simétricos.

Um espaço-tempo estático é aquele em que é possível encontrar um sistema de coordenadas $x^0 \equiv t$, x^1, x^2 e x^3 tal que a métrica ds^2 não dependa explicitamente do tempo t. Ele será esfericamente simétrico ou isotrópico se a métrica depender de x e dx apenas através de termos que são invariantes por rotação ($d\vec{x}^2$, $\vec{x} \cdot d\vec{x}$ e $\vec{x}^2$).

A forma mais geral de uma métrica estática e esfericamente simétrica (WEINBERG, 1972) pode ser dada por

$$ds^2 = F(r)dt^2 - 2E(r)dt\ \vec{x} \cdot d\vec{x} - D(r)(\vec{x} \cdot d\vec{x})^2 - C(r)d\vec{x}^2, \tag{2.31}$$

onde $F(r)$, $E(r)$, $D(r)$ e $C(r)$ são funções desconhecidas dependentes da variável radial $r \equiv \sqrt{\vec{x} \cdot \vec{x}}$, e $\vec{x} \cdot d\vec{x} = x^1dx^1 + x^2dx^2 + x^3dx^3$. Há uma forma de se deduzir esta expressão, mas é possível supor este formato como uma definição de métrica estática e isotrópica ou, simplesmente, um *ansatz* para tentarmos obter soluções para as equações de campo de Einstein.

É conveniente, substituir as coordenadas cartesianas por coordenadas esféricas r, θ e φ, definidas como:

$$\begin{aligned} x^1 &= r sin\theta cos\varphi \\ x^2 &= r sin\theta sin\varphi \\ x^3 &= r cos\theta \end{aligned} \tag{2.32}$$

Assim, se reescrita a métrica 2.31 teremos

$$ds^2 = F(r)dt^2 - 2rE(r)dtdr - r^2D(r)dr^2 - C(r)(dr^2 + r^2d\theta^2 + r^2sin^2\theta d\varphi^2). \tag{2.33}$$

É possível simplificar a forma da métrica redefinindo a coordenada temporal para eliminar o termo fora da diagonal principal da métrica reescrevendo-a como $t' \equiv t + \Phi(r)$, onde $\Phi(r)$ é uma função arbitrária de r tal que

$$\frac{d\Phi}{dr} = -\frac{rE(r)}{F(r)}. \tag{2.34}$$

O que faz a equação 2.33 se tornar

$$ds^2 = F(r)dt'^2 - G(r)dr^2 - C(r)(dr^2 + r^2 d\theta^2 + r^2 sin^2\theta d\varphi^2), \tag{2.35}$$

onde escrevemos um novo termo $G(r)$ definido como

$$G(r) \equiv r^2 \left(D(r) + \frac{E^2(r)}{F(r)} \right). \tag{2.36}$$

Também é possível reescrever a coordenada radial r se impormos mais uma condição dada pela relação: $r'^2 \equiv C(r)r^2$. Assim, é simplificada ainda mais a métrica escrevendo-a na sua forma padrão

$$ds^2 = B(r')dt'^2 - A(r')dr'^2 - r'^2 d\theta^2 - r'^2 sin^2\theta d\varphi^2, \tag{2.37}$$

onde:

$$A(r') = \left(1 + \frac{G(r)}{C(r)} \right) \left(1 + \frac{r}{2C(r)} \frac{dC(r)}{dr} \right)^{-2} \tag{2.38}$$

$$B(r') = F(r) \tag{2.39}$$

Assim o tensor $g_{\mu\nu}$ terá apenas componentes diagonais não-nulas dadas por[13]:

$$\begin{aligned} g_{tt} &= B(r), \\ g_{rr} &= -A(r), \\ g_{\theta\theta} &= -r^2, \\ g_{\varphi\varphi} &= -r^2 \sin^2\theta \end{aligned} \tag{2.40}$$

As funções $A(r)$ e $B(r)$ são os termos da métrica que devem ser obtidos através das soluções das equações de Einstein para que seja possível descrever sistemas gravitacionais.

Se calculados os termos do símbolo de Christoffel (equação 2.30) são obtidos os seguintes termos não-nulos:

$$\begin{aligned} \Gamma^r_{rr} = \frac{1}{2A(r)}\frac{dA(r)}{dr} \quad , \quad \Gamma^r_{\theta\theta} = -\frac{r}{A(r)} \quad , \\ \Gamma^r_{\varphi\varphi} = -\frac{r sin^2\theta}{A(r)} \quad , \quad \Gamma^r_{tt} = \frac{1}{2A(r)}\frac{dB(r)}{dr} \quad , \\ \Gamma^\theta_{r\theta} = \Gamma^\theta_{\theta r} = \frac{1}{r} \quad , \quad \Gamma^\theta_{\varphi\varphi} = -sin\theta cos\theta \quad , \\ \Gamma^\varphi_{\varphi r} = \Gamma^\varphi_{r\varphi} = \frac{1}{r} \quad , \quad \Gamma^\varphi_{\varphi\theta} = \Gamma^\varphi_{\theta\varphi} = cot\theta \quad , \\ \Gamma^t_{tr} = \Gamma^t_{rt} = \frac{1}{2B(r)}\frac{dB(r)}{dr} \end{aligned} \tag{2.41}$$

[13] A partir de agora, é denotado r' como r, e t' como t.

Introduzindo estes resultados na expressão para o tensor de Ricci (equação 2.29), são obtidos:

$$\begin{aligned}
R_{rr} &= \frac{B''(r)}{2B(r)} - \frac{1}{4}\left(\frac{B'(r)}{B(r)}\right)\left(\frac{A'(r)}{A(r)} + \frac{B'(r)}{B(r)}\right) - \frac{1}{r}\left(\frac{A'(r)}{A(r)}\right), \\
R_{\theta\theta} &= -1 + \frac{r}{2A(r)}\left(-\frac{A'(r)}{A(r)} + \frac{B'(r)}{B(r)}\right) + \frac{1}{A(r)}, \\
R_{\varphi\varphi} &= sin^2\theta R_{\theta\theta}, \\
R_{tt} &= -\frac{B''(r)}{2A(r)} + \frac{1}{4}\left(\frac{B'(r)}{A(r)}\right)\left(\frac{A'(r)}{A(r)} + \frac{B'(r)}{B(r)}\right) - \frac{1}{r}\left(\frac{B'(r)}{A(r)}\right), \\
R_{\mu\nu} &= 0 \quad , \quad para \quad \mu \neq \nu
\end{aligned} \tag{2.42}$$

onde o termo $'$ denota uma derivada em relação a coordenada radial r.

2.4.3 Soluções exatas das equações de Einstein

Como comentado anteriormente, para serem resolvidas as equações de Einstein em um caso esfericamente simétrico e estacionário é preciso encontrar $A(r)$ e $B(r)$. É possível começar a tentar encontrar soluções exatas fazendo mais um *ansatz* (D'INVERNO, 1998), escrevendo A e B tal que a métrica seja:

$$ds^2 = e^{\nu(r,t)}dt^2 - e^{\lambda(r,t)}dr^2 - r^2d\theta^2 - r^2sin^2\theta d\varphi^2, \tag{2.43}$$

onde ν e λ são novas funções. Aqui, as exponenciais são utilizadas para garantir que as funções envolvidas sempre sejam positivas, fazendo com que os termos da métrica não troquem de sinal.

Se contraídas 2.24 dessa vez com um tensor $g^{\nu\eta}$, são obtidas as equações de Einstein no formato tensorial misto

$$G^{\eta}_{\mu} = \Lambda\delta^{\eta}_{\mu}, \tag{2.44}$$

onde δ^{η}_{μ} é a função delta de Kronecker. Neste caso, se substituída a métrica 2.43 na equação 2.44 serão obtidas as seguintes componentes não-nulas:

$$\begin{aligned}
G^0_0 &= e^{-\lambda}\left(\frac{\lambda'}{r} - \frac{1}{r^2}\right) + \frac{1}{r^2} = \Lambda, \\
G^1_0 &= -e^{-\lambda}r^{-1}\dot{\lambda} = -e^{\lambda-\nu}G^0_1 = \Lambda\delta^1_0 = \Lambda\delta^0_1 = 0, \\
G^1_1 &= -e^{-\lambda}\left(\frac{\nu'}{r} + \frac{1}{r^2}\right) + \frac{1}{r^2} = \Lambda, \\
G^2_2 &= G^3_3 = \frac{1}{2}e^{-\lambda}\left(\frac{\nu'\lambda'}{2} + \frac{\lambda'}{r} - \frac{\nu'}{r} - \frac{\nu'^2}{2} - \nu''\right) + \frac{1}{2}e^{-\nu}\left(\ddot{\lambda} + \frac{\dot{\lambda}^2}{2} - \frac{\dot{\lambda}\dot{\nu}}{2}\right) = \Lambda.
\end{aligned} \tag{2.45}$$

Aqui $(')$ e $(\dot{})$ denotam derivadas em relação as coordenadas r e t respectivamente.

Se as equações anteriores para G_0^0, G_0^1, G_1^0 e G_1^1 forem satisfeitas, então é possível escrever as equações 2.46, já que a equação para $G_2^2 = G_3^3$ é combinação linear das outras.

$$\begin{aligned} e^{-\lambda}\left(\frac{\lambda'}{r} - \frac{1}{r^2}\right) + \frac{1}{r^2} &= \Lambda, \\ e^{-\lambda}\left(\frac{\nu'}{r} + \frac{1}{r^2}\right) - \frac{1}{r^2} &= -\Lambda, \\ \dot{\lambda} &= 0; \end{aligned} \tag{2.46}$$

Somando as duas primeiras expressões e integrando no tempo, é obtido:

$$\lambda' + \nu' = 0 \Rightarrow \lambda + \nu = h(t), \tag{2.47}$$

onde h é uma constante de integração. Também, graças à terceira equação de 2.46, é possível notar que as duas primeiras não dependem da variável temporal, ou seja, são equações ordinárias e podem ser resolvidas facilmente. Resolvendo para a primeira será obtido

$$\begin{aligned} e^{-\lambda}\left(\frac{\lambda'}{r} - \frac{1}{r^2}\right) + \frac{1}{r^2} &= \Lambda, \\ e^{-\lambda} = 1 + \frac{C}{r} - \frac{\Lambda r^2}{3} &= A^{-1}(r). \end{aligned} \tag{2.48}$$

Esta é a componente g_{rr} da métrica, onde C é uma constante de integração que agregaremos significado físico mais adiante. A componente g_{tt} pode ser obtida através da relação 2.47, e o termo exponencial pode ser eliminado redefinindo a coordenada temporal:

$$e^{\nu}dt^2 = e^{h(t)-\lambda}dt^2 = e^{-\lambda}e^{h(t)}dt^2 \Rightarrow e^{-\lambda}dt^2 = B(r)dt^2 \tag{2.49}$$

Percebendo que $A^{-1}(r) = B(r)$, é possível então denominar $B(r)$ por $F(r)$ a partir de agora. Esta função $F(r)$ é o termo da métrica advindo da solução das equações de Einstein que possibilita escreve a métrica, finalmente, como:

$$ds^2 = F(r)dt^2 - F^{-1}(r)dr^2 - r^2d\theta^2 - r^2sin^2\theta d\varphi^2 \tag{2.50}$$

Este elemento de linha é esfericamente simétrico e com ele podemos descrever alguns objetos gravitacionais de interesse, tais como buracos negros, impondo determinadas condições de contorno.

2.4.3.1 Buraco negro de Schwarzschild

Com os resultados obtidos anteriormente (2.48), se considerado que não há contribuição da constante cosmológica ($\Lambda = 0$) será obtida a conhecida solução de Schwarzschild:

$$F(r) = 1 - \frac{2m}{r} \tag{2.51}$$

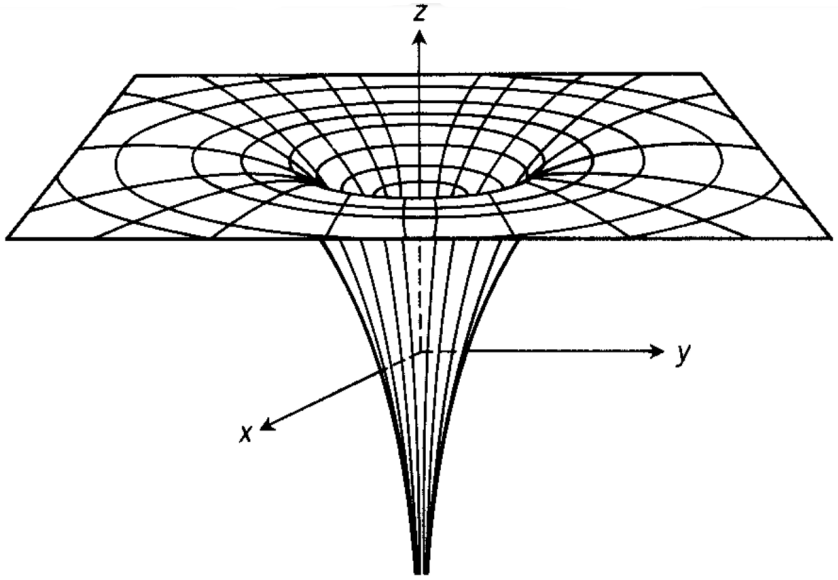
Aqui a constante C é substituída por $2m$, onde m é conhecida como a massa geométrica ($m = GM/c^2$). Esta constante pode ser obtida quando tomamos o limite de campo fraco ($r \to \infty$) e comparamos com a teoria Newtoniana de gravitação.

É possível interpretar este resultado como se uma partícula pontual estivesse situada na origem, sendo m sua massa em unidades relativísticas. Esta métrica pode ser utilizada para serem descritos buracos negros esfericamente simétricos e estáticos de massa m. Este tipo de objeto possui uma singularidade na origem ($r = 0$) e é assintoticamente plano, ou seja, no infinito a geometria do espaço-tempo se comporta como o espaço-tempo de Minkowski.

Mas como é possível interpretar fisicamente essas soluções? Objetos massivos geram campos gravitacionais que são interpretados como uma deformação do espaço e do tempo, modificando sua geometria. Obter o termo da métrica $F(r)$ garante que seja possível fazer medidas de distância e tempo entre dois eventos contidos neste espaço-tempo curvo.

A superfície $r = 2m$ é chamada horizonte de eventos e representa uma região limite do espaço-tempo do qual nem mesmo a luz consegue escapar, se comportando como uma membrana de caminho único. Essa superfície é obtida calculando os zeros da função $F(r)$. Uma representação do buraco negro de Schwarzschild pode ser observada na figura 10). Nela ficam explícitas as características da singularidade na origem e do horizonte de eventos.

Figura 10 – Projeção da solução de Schwarzschild para $t = const.$ e $\theta = \pi/2$



Fonte: D'Inverno (1998).

2.4.3.2 Buraco negro de Schwarzschild-de Sitter

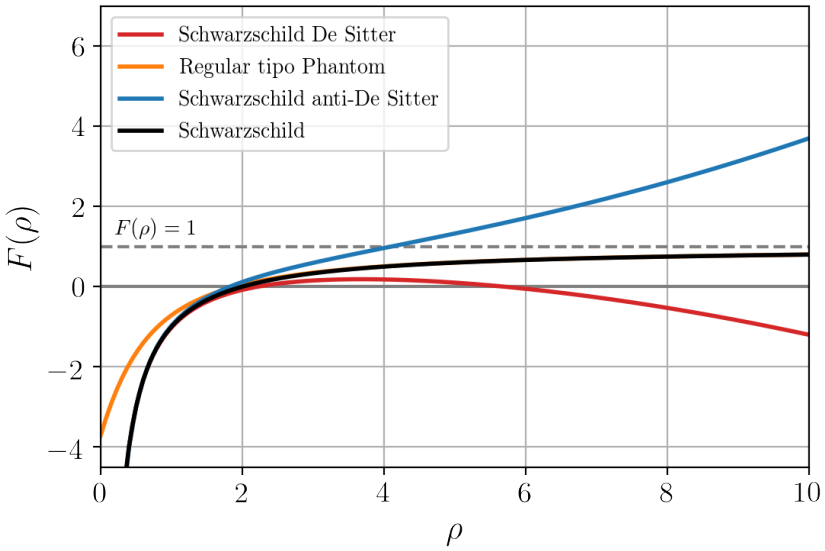
Consideremos agora a contribuição da constante cosmológica na equação (2.48). A constante cosmológica foi primeiramente adicionada às equações de campo por Einstein

para contrabalancear atração gravitacional e forçar uma ideia de universo estático. Porém, atualmente a adição da constante cosmológica aos modelos surge no contexto de uma "força" repulsiva ou atrativa na expansão do universo. Neste sentido, sendo a constante cosmológica positiva a métrica pode representar um buraco negro em um universo em expansão acelerada. Neste caso, sendo $\Lambda > 0$, obteremos a solução conhecida como Schwarzschild-de Sitter, tal que a componente da métrica seja dada por:

$$F(r) = 1 - \frac{2m}{r} - \frac{\Lambda r^2}{3} \tag{2.52}$$

Nesse buraco negro é possível identificar a presença de dois horizontes. O horizonte de eventos $F(r_{he}) = 0$ e o horizonte cosmológico $F(r_{hc}) = 0$ tal que $r_{he} < r_{hc}$. Um comparativo entre as métricas pode ser visualizado na figura 11.

Figura 11 – Exemplos de soluções para algumas métricas.



Fonte: Elaborado pelo autor (2022).

2.4.3.3 Buraco negro de Schwarzschild-Anti-de Sitter

Caso considerada uma constante cosmológica negativa $\Lambda < 0$, é possível definir a solução de Schwarzschild-Anti-de Sitter:

$$F(r) = 1 - \frac{2m}{r} + \frac{|\Lambda| r^2}{3} \tag{2.53}$$

Nesse buraco negro é possível identificar a presença de apenas um horizonte de eventos $F(r_{he}) = 0$.

Este caso é similar ao anterior, porém não há interpretação física, no sentido de um objeto astrofísico. Mesmo assim, este tipo de solução é interessante em estudos de teorias quânticas de campos, na correspondência AdS/CFT e na teoria das cordas.

2.4.3.4 Classe de soluções para um campo escalar tipo Phantom

Nos casos anteriores foram considerados buracos negros singulares e cujo tensor energia-momento era nulo. Contudo, podemos obter soluções do tipo buraco negro na presença de conteúdo material. Um tipo de matéria simples o bastante para gerar soluções exatas seria um campo escalar ϕ com um potencial efetivo $V(\phi)$.

Neste caso, obtendo as componentes das equações de Einstein e equação de Klein-Gordon e impondo certas condições de contorno específicas (BRONNIKOV; FABRIS, 2006), é possível escrever um sistema de equações acopladas da forma:

$$(F(\rho)r^2\phi')' = \varepsilon r^2 \frac{dV}{d\phi}, \tag{2.54}$$

$$(F'(\rho)r^2)' = -2r^2V, \tag{2.55}$$

$$2\frac{r''}{r} = -\varepsilon\phi'^2, \tag{2.56}$$

$$F(\rho)(r^2)'' - r^2F''(\rho) = 2. \tag{2.57}$$

Aqui, ε é uma constante tal que $\varepsilon = +1$ indica um campo escalar usual, com energia cinética positiva e $\varepsilon = -1$ indica um campo conhecido como Phantom, com energia cinética negativa. O sinal ($'$) denota a derivada em relação à coordenada ρ que é dada pela expressão

$$r = \sqrt{\rho^2 + b^2} \quad , \quad b = const. > 0. \tag{2.58}$$

Da equação 2.57, é possível integrar e escrever:

$$D' \equiv \left(\frac{F(\rho)}{r^2}\right)' = \frac{2(\rho_0 - \rho)}{r^4}, \tag{2.59}$$

onde $D(\rho) = F(\rho)/r^2$ e ρ_0 é uma constante de integração.

Utilizando as equações anteriores e impondo $\varepsilon = -1$, a equação 2.59 permite obter:

$$\begin{aligned} D(\rho) &= \frac{F(\rho)}{r^2(\rho)}, \\ &= \frac{c}{b^2} + \frac{1}{b^2+\rho^2} + \frac{\rho_0}{b^3}\left(\frac{b\rho}{b^2+\rho^2} + tan^{-1}\frac{\rho}{b}\right), \end{aligned} \tag{2.60}$$

onde c é uma constante de integração. Esta função $F(\rho)$ representa a componente g_{tt} da métrica. Essa métrica pode ser interpretada como um buraco negro regular, um buraco de minhoca ou um universo de Kantowski-Sachs dependendo dos valores atribuídos a ρ_0 e c. Aqui o termo "regular" indica que é um buraco negro sem singularidade na origem.

Para o caso do buraco negro regular é possível ter um ou dois horizontes presentes. No caso de um buraco de minhoca é possível identificar a garganta dele, isto é, a região que conecta dois universos, fazendo $\rho \to 0$, o que implica em $r = b$.

Especial atenção deve ser dada à equação 2.57, já que todas as métricas obtidas até agora, devem respeitá-la. Assim, se possível resolvê-la numericamente, seria possível tentar obter todos os casos apresentados até o momento. Este foi o objetivo deste trabalho, utilizar as PINN para este fim.

3 Metodologia

Este capítulo tem como objeto explicitar sucintamente como foram abordados novos problemas, partindo de um exemplo mais simples (equação do calor) e gradativamente trabalharmos em problemas mais sofisticados, até abordarmos um caso que, contra-intuitivamente, é mais difícil (oscilador harmônico amortecido forçado). O modelo desta última solução será modificado para, finalmente, obtermos as métricas através da solução das equações de Einstein.

3.1 Implementação e Hardware

Quando se trata das ferramentas as quais utilizamos para criar nossos modelos com as PINN, são diversas as opções. Na implementação dos *scripts* geralmente são utilizadas bibliotecas como o TensorFlow e Keras para programarmos os códigos na linguagem Python. Porém, já existem bibliotecas em desenvolvimento especificamente para trabalharmos com as PINN (LU et al., 2020). Para treinarmos os modelos são utilizadas *workstations* ou ferramentas de computação na nuvem com GPUs e TPUs.

Como o foco deste trabalho não é dissertar sobre a implementação em si ou sobre as máquinas utilizadas (*hardware*), uma breve discussão está contida no Apêndice A. Em resumo, os códigos foram implementados utilizando a linguagem Python e o *framework* TensorFlow. Os modelos foram treinados utilizando GPUs e TPUs na plataforma Google Colab.

3.2 Equação do Calor

A equação de condução do calor é uma equação diferencial parcial utilizada como modelo matemático para descrever processos de difusão de calor em sólidos, geralmente isotrópicos e homogêneos. A devida equação em três dimensões espaciais cartesianas e com uma fonte de calor pode ser escrita como:

$$\frac{\partial u}{\partial t} = \alpha \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \right) + f(\vec{x}, t) = \alpha \nabla^2 u + f(\vec{x}, t) \tag{3.1}$$

Aqui, $u(x, y, z, t)$ é o campo de temperaturas, $f(\vec{x}, t)$ é uma função que representa uma fonte de calor e α é o coeficiente de difusão térmica.

Seja analisado um caso simples, como o de uma barra uniforme homogênea unidimensional com condições iniciais senoidais e sem fonte de calor. Este problema pode ser

descrito pela equação e condições de contorno:

$$\begin{aligned} &\frac{\partial u(x,t)}{\partial t} - 0,10\frac{\partial^2 u(x,t)}{\partial x^2} = 0, \\ &u(0,x) = -\sin(n\pi x), \quad n = cte, \quad x \in [-1,1], \\ &u(t,-1) = u(t,1) = 0, \quad t \in [0,1]. \end{aligned} \tag{3.2}$$

A solução exata pode ser obtida por separação de variáveis (HANCOCK, 2006) e é dada pela eq. 3.3.

$$u(x,t) = C_1 \sin\left(\frac{n\pi}{L}x\right)\exp\left(-\alpha\frac{n^2\pi^2}{L^2}t\right) \tag{3.3}$$

Onde C_1 é uma constante que diz respeito à amplitude da senoide utilizada como condições iniciais, n é um número inteiro positivo que diz respeito ao modo de vibração da senoide e L é o comprimento da barra.

Para este problema ser resolvido é preciso criar um conjunto de dados com as condições de contorno. Como é conhecida a sua solução exata, é possível agregá-la ao conjunto de dados para, posteriormente, ser comparada com a solução numérica. Assim, foram sorteados 100 pontos na fronteira do domínio para serem utilizados como treino, uma quantidade que foi suficiente para serem observados os resultados esperados.

Como esta equação também possui derivada primeira no tempo, é possível utilizar o mesmo método de solução usado na solução da equação de Burgers. Assim, foi construída uma rede com 8 camadas de 20 neurônios cada e treinada por 10.000 épocas utilizando L-BFGS-B. Dois resultados obtidos estão expostos nas figuras 12 e 13, com respectivos erros (MAE[1]) $7,86 \times 10^{-4}$ e $4,16 \times 10^{-3}$. Como este é um caso simples, não é necessariamente preciso utilizar TPUs ou GPUs, é possível treinar a rede em menos de 200 segundos em uma CPU.

3.3 Equação de Onda

A equação de onda é uma equação diferencial parcial de segunda ordem utilizada em dinâmica dos fluidos, acústica e eletromagnetismo, para representarmos ondas clássicas como as ondas mecânicas e ondas de luz. Para um caso espacialmente tridimensional é possível escrever a equação 3.4:

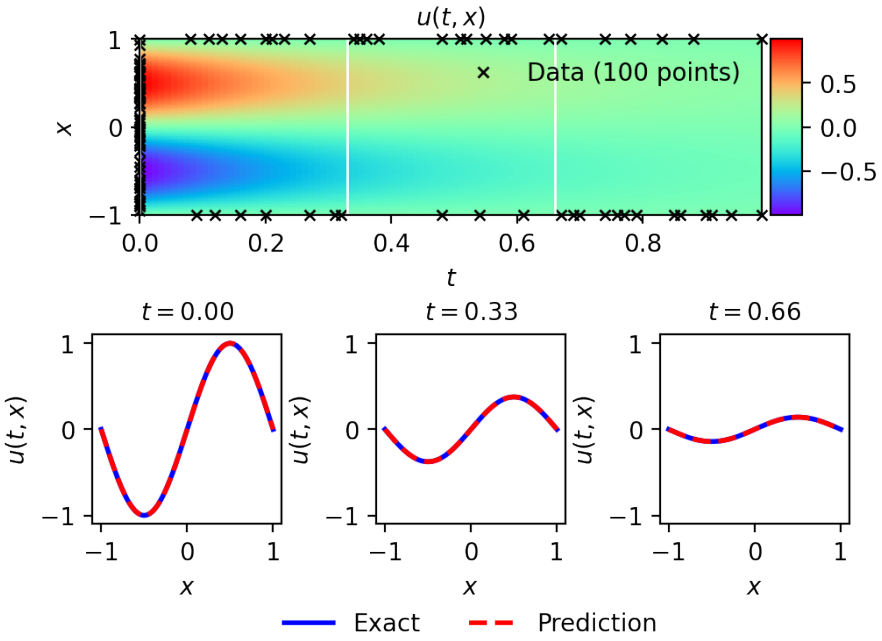
$$\frac{\partial^2 h}{\partial t^2} = c^2\left(\frac{\partial^2 h}{\partial x^2} + \frac{\partial^2 h}{\partial y^2} + \frac{\partial^2 h}{\partial z^2}\right) = c^2\nabla^2 h, \tag{3.4}$$

onde $h(t,x,y,x)$ é uma função escalar e c é a velocidade de propagação da onda. Percebamos que a dada equação é de segunda ordem no tempo e o algoritmo PINN,

[1] Aqui, o erro MAE foi escolhido para se realizar a comparação da superfície referente à solução numérica com a proveniente da solução exata após efetuado o treinamento da rede neural. Não confundir com MSE, que é o erro utilizado para se calcular a *loss*.

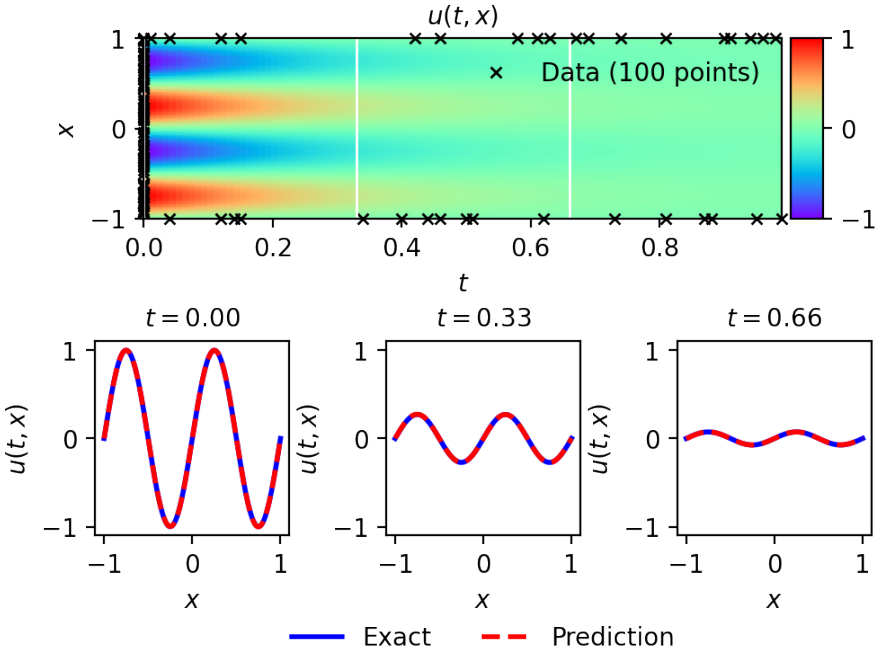
supostamente, não engloba soluções deste tipo de equação. Porém, como foi visto no caso da equação de Schrodinger, é possível resolver um sistema de equações diferenciais.

Figura 12 – Solução numérica para a equação do calor em uma barra com $n = 1$ utilizando PINN.



Fonte: Elaborado pelo autor.

Figura 13 – Solução numérica para a equação do calor em uma barra com $n = 2$ utilizando PINN.



Fonte: Elaborado pelo autor.

Desta forma, para este problema ser abordado é possível reescrever a equação 3.4 (unidimensional) no formato de um sistema de equações de primeira ordem:

$$\begin{cases} f_u := u_t - v, \\ f_v := v_t - c^2 u_{xx}. \end{cases} \tag{3.5}$$

Seja considerado um caso de uma corda fina unidimensional homogênea vibrante com condições de contorno de Cauchy e velocidade de acordo com a equação 3.6.

$$\begin{aligned} &\frac{\partial h^2(x,t)}{\partial t^2} - c^2 \frac{\partial^2 h(x,t)}{\partial x^2} = 0, \quad c = cte, \\ &u(0,x) = -\sin(n\pi x), \quad n = cte, \quad x \in [-5,5], \\ &u(t,-1) = u(t,1) = 0, \quad t \in [0,1.5], \\ &u_t(0,x) = u_t(t,-1) = u_t(t,1) = 0. \end{aligned} \tag{3.6}$$

A solução exata pode ser obtida por separação de variáveis (LEVITUS, 2020) e é dada pela equação 3.7.

$$u(x,t) = C_1 \sin\left(\frac{n\pi}{L}x\right)\left[C_2 \sin\left(\frac{n\pi c}{L}t\right) + C_3 \cos\left(\frac{n\pi c}{L}t\right)\right] \tag{3.7}$$

Onde C_1 é uma constante que diz respeito à amplitude da senoide utilizada como condições iniciais, C_2 e C_3 são constantes que devem ser obtidas para se adequar a solução às condições de contorno, v é a velocidade de propagação da onda no meio, n é um inteiro positivo que diz respeito ao "modo de vibração" da senoide e L é o comprimento da corda.

É possível chegar a resultados como os das figuras 14 e 15 utilizando 4 camadas de 100 neurônios[2], 50.000 épocas para o ADAM e 10.000 para o L-BFGS-B. Os erros foram respectivamente $2,63 \times 10^{-3}$ e $2,40 \times 10^{-2}$.

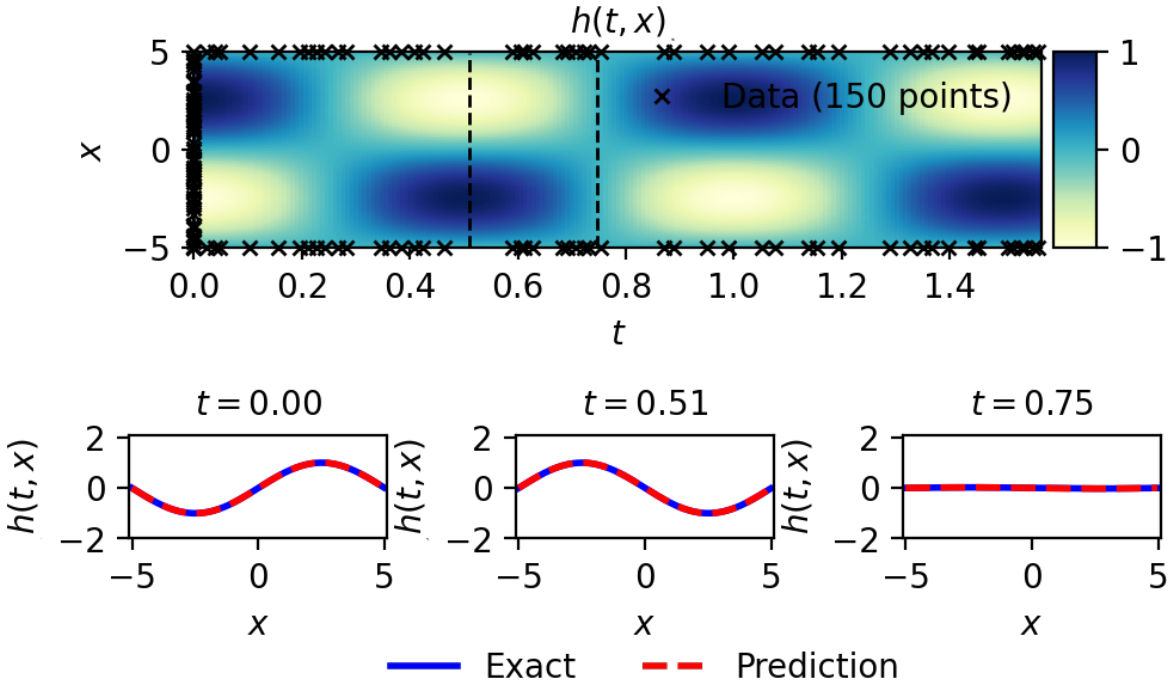
Também é possível resolver a equação utilizando apenas pontos das condições iniciais, sem as condições de contorno, esta estratégia é interessante para serem resolvidos os problemas posteriores. A Figura 16 representa uma solução para a equação de onda com condições iniciais $h(0,x) = 2\text{sech(x)}$, neste caso, a onda se propaga para o infinito.

3.4 Oscilador Harmônico Amortecido Forçado

Equações de osciladores harmônicos representam, na mecânica clássica, sistemas que quando dispostos a determinadas condições iniciais sofrem uma determinada força restauradora proporcional a sua posição, gerando uma movimentação característica. Estas equações podem representar uma grande variedade de sistemas físicos desde pêndulos e sistemas massa-mola até fenômenos quânticos como vibrações moleculares.

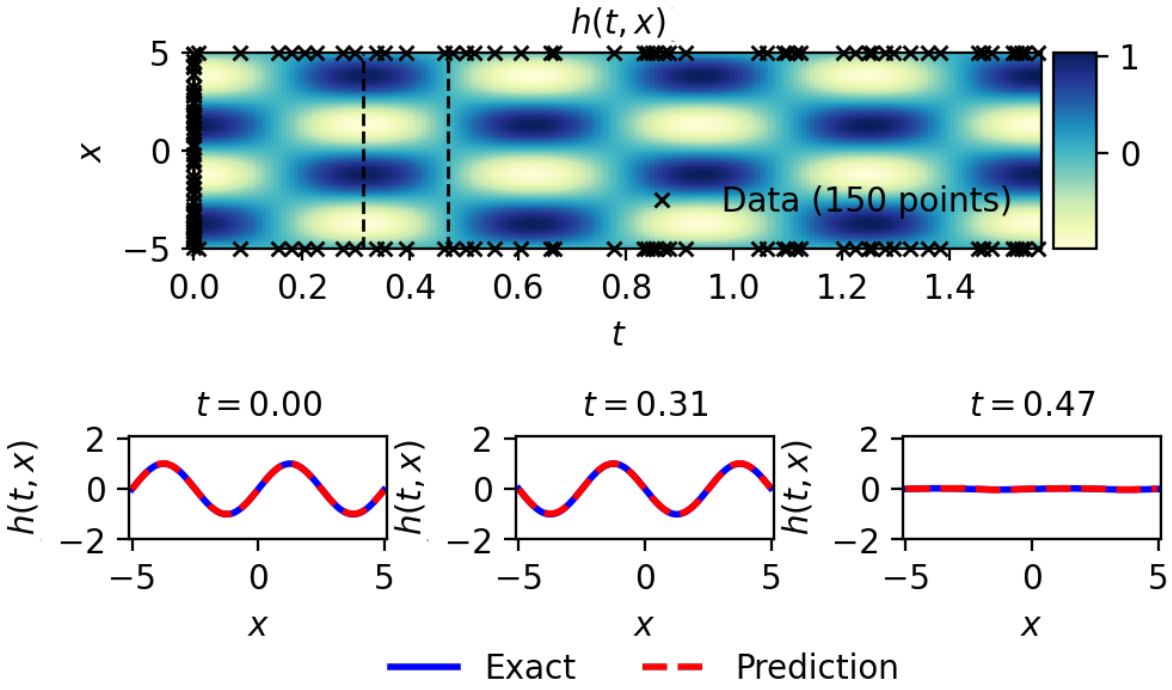
[2] Mesma arquitetura do caso da equação de Schrodinger não-linear.

Figura 14 – Solução numérica para o 2º harmônico da equação de onda ($n = 2$ e $c = 10$) utilizando PINN.



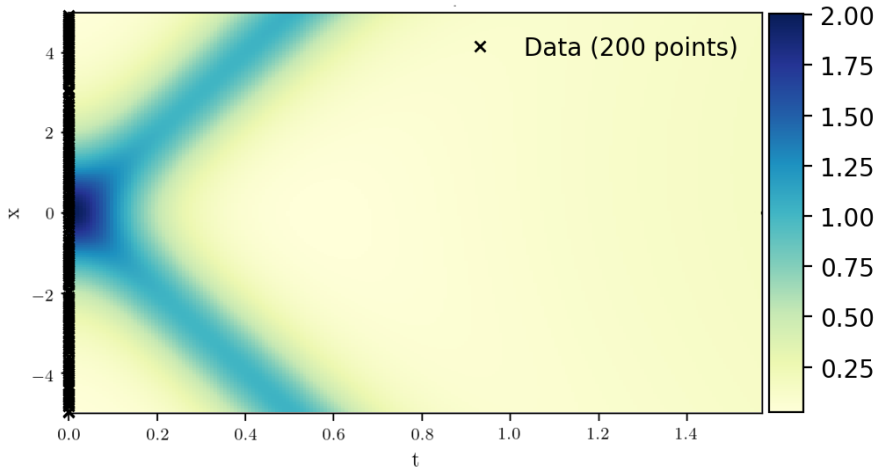
Fonte: Elaborado pelo autor.

Figura 15 – Solução numérica para o 4º harmônico da equação de onda ($n = 4$ e $c = 8$) utilizando PINN.



Fonte: Elaborado pelo autor.

Figura 16 – Solução numérica para equação de onda unidimensional sem condições de contorno utilizando PINN.



Fonte: Elaborado pelo autor.

A equação de movimento do sistema massa-mola com uma força de atrito proporcional à velocidade e sujeito a uma força externa pode ser descrita de forma genérica por:

$$m\left(\frac{\partial^2 s(t)}{\partial t^2}\right) + \gamma\left(\frac{\partial s(t)}{\partial t}\right) + k\ s(t) = F(t), \quad m, \gamma, k = cte. \tag{3.8}$$

Onde m é a massa, γ é o coeficiente de atrito, k é a constante elástica e $F(t)$ é a força externa atuando no sistema.

Apesar desta expressão não ser uma equação diferencial parcial, ela ainda é uma equação de segunda ordem (no tempo) e, consequentemente, deverá ser resolvida por meio de sua expressão na forma de um sistema de equações diferenciais de primeira ordem. Além disso, a natureza quase sempre periódica da solução torna este caso simples o mais computacionalmente custoso abordado até o momento, o motivo será melhor discutido na conclusão. O sistema de equações é dado por:

$$\begin{cases} f_u := u_t - v, \\ f_v := mv_t + \gamma u_t + ku - F(t). \end{cases} \tag{3.9}$$

Seja resolvido um sistema modelado pela equação de um oscilador harmônico amortecido com uma força externa do tipo $F(t) = F_0 \cos(\omega t)$, dada pela equação 3.10.

$$\begin{aligned} &6\left(\frac{\partial^2 s(t)}{\partial t^2}\right) + 0.5\left(\frac{\partial s(t)}{\partial t}\right) + 0.5 s(t) = 0.1\cos(0.8t), \\ &u(0) = 1, \quad u_t(0) = 0, \quad t \in [0, 200]. \end{aligned} \tag{3.10}$$

A solução exata da equação não-homogênea correspondente a este oscilador harmônico pode ser obtida pelo método dos coeficientes indeterminados (VALLE, 2016) e é dada

no formato da equação 3.11.

$$u_{geral}(t) = u_h(t) + u_p(t); \tag{3.11}$$

Onde $u_h(t)$ representa a solução geral da equação homogênea e $u_p(t)$ uma solução particular. Estas soluções são dadas por 3.12 e 3.13, respectivamente.

$$u_h(t) = \exp\left(-\frac{\gamma}{2m}t\right)\left[C_1 \sin\left(\frac{\sqrt{2\frac{k}{m} - \left(\frac{\gamma}{m}\right)^2}}{2}t\right) + C_2 \cos\left(\frac{\sqrt{2\frac{k}{m} - \left(\frac{\gamma}{m}\right)^2}}{2}t\right)\right] \tag{3.12}$$

$$u_p(t) = \frac{F_0}{-m\omega^2 + \gamma\omega + k}\cos(\omega t) \tag{3.13}$$

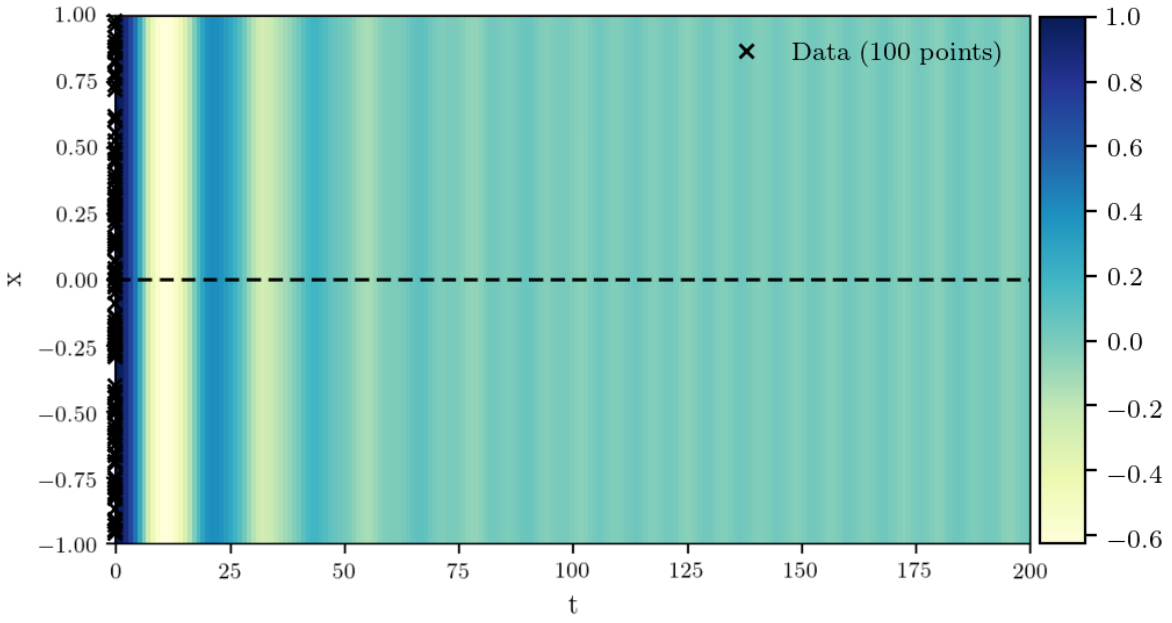
Os *scripts* que foram usados para resolver os problemas anteriores foram concebidos para resolvermos equações diferenciais parciais. Porém, ainda é possível utilizar a mesma metodologia para resolver este caso. É preciso apenas gerar um conjunto de dados com as condições iniciais para $s(0)$ e $s_t(0)$ e resolver o problema havendo uma dimensão "fictícia" (referente à coordenada x dos outros casos) que não interfere nos resultados. Os pontos utilizados para treino serão similares aos utilizados para o caso da equação de onda sem condições de contorno, onde apenas foram utilizadas as condições iniciais.

Como a solução deste caso constitui uma superfície periódica e de alta frequência, o custo computacional é alto utilizando as PINN. Com 8 camadas de 20 neurônios é possível chegar em um bom resultado intercalando os otimizadores (ADAM e L-BFGS-B) em ciclo até atingirmos uma *loss* baixa o suficiente ($3,00 \times 10^{-6}$).

Os resultados para $u = s(t)$ e $v = s_t(t)$ podem ser conferidos nos gráficos de densidade das figuras 17 e 18. Para a solução do oscilador harmônico ser visualizada como de usual é necessário apenas fazer um recorte por exemplo, em $x = 0$. O que pode ser visualizado nas figuras 19 e 20. O erro neste caso foi de $7,05 \times 10^{-3}$.

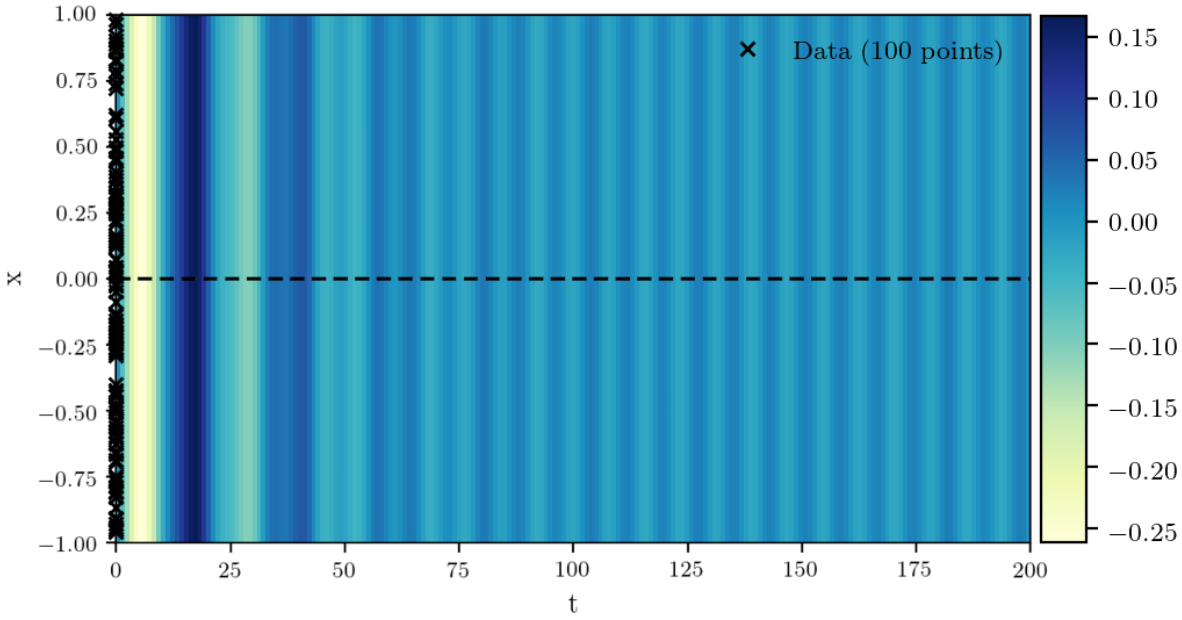
A metodologia utilizada na solução deste problema foi exatamente a mesma aplicada na solução das equações de campo de Einstein para obtenção das métricas. O que será apresentado no capítulo seguinte.

Figura 17 – Solução numérica ($u = s(t)$) para equação do oscilador harmônico amortecido forçado utilizando PINN.



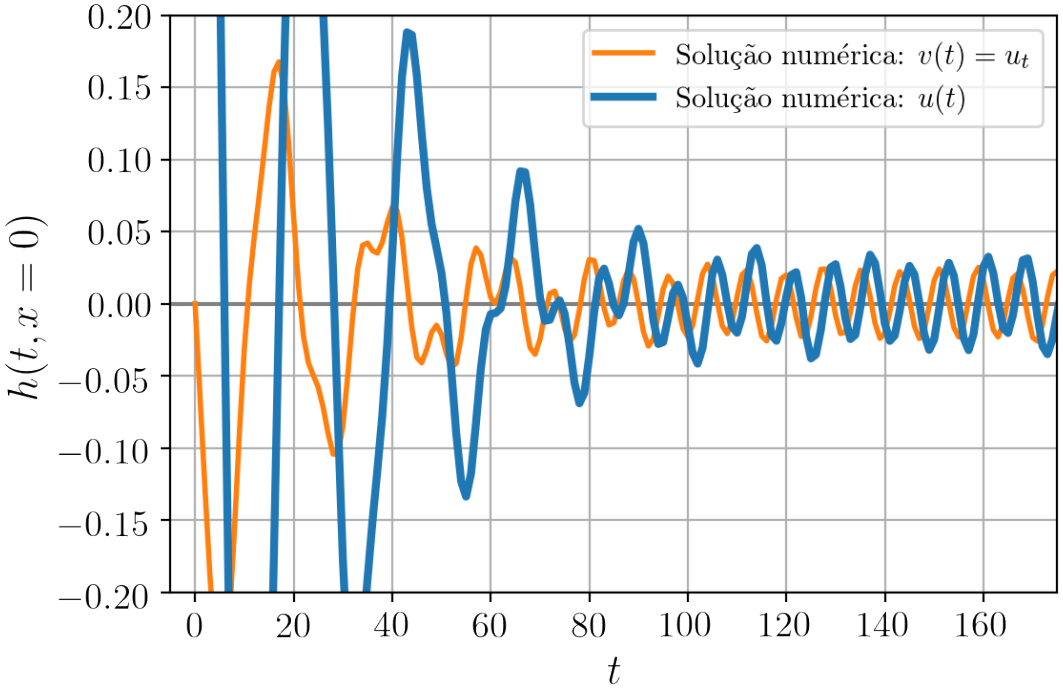
Fonte: Elaborado pelo autor.

Figura 18 – Solução numérica ($v = s_t(t)$) para equação do oscilador harmônico amortecido forçado utilizando PINN.



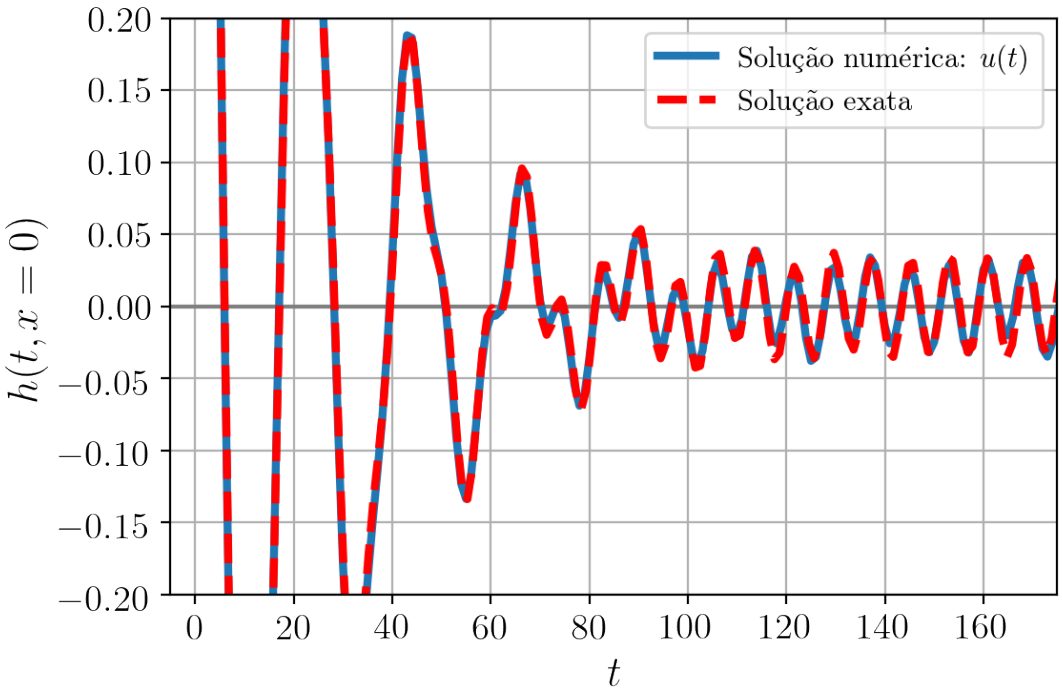
Fonte: Elaborado pelo autor.

Figura 19 – Recorte da solução numérica ($x = 0$) para equação oscilador harmônico amortecido forçado utilizando PINN.



Fonte: Elaborado pelo autor.

Figura 20 – Comparação da solução numérica com a solução exata ($x = 0$) da equação oscilador harmônico amortecido forçado utilizando PINN.



Fonte: Elaborado pelo autor.

4 Resultados e Análise

O objetivo deste trabalho foi verificar se é possível obter numericamente as componentes da métrica para alguns objetos gravitacionais através de uma ferramenta conhecida como PINN. É possível fazer isso resolvendo a equação 2.57. Lembrando que foi definida uma nova coordenada radial dada por 2.58, é possível escrever a equação que é preciso resolver como:

$$-(\rho^2 + b^2) \cdot \frac{\partial^2 F(\rho)}{\partial \rho^2} + 2F(\rho) - 2 = 0. \tag{4.1}$$

Para o método das PINN ser aplicado foi utilizada uma metodologia similar à utilizada no caso do oscilador harmônico, sendo que cada objeto possuía determinados critérios a serem impostos em forma de condições de contorno. Assim, como nos casos anteriores, deve-se escrever a equação de interesse no formato de resíduo, o que resulta no seguinte sistema:

$$\begin{cases} f_v = u_\rho - v, \\ f_u = -(\rho^2 + b^2) \cdot v_\rho + 2u - 2; \end{cases} \tag{4.2}$$

Em todos os casos foi construída uma rede neural com 5 camadas de 200 neurônios cada. O treinamento foi feito intercalando os otimizadores L-BFGS-B e ADAM até que um erro pequeno o suficiente fosse alcançado.

4.1 Buraco negro regular tipo Phantom

NA equação 4.1, caso haja um campo escalar que varia no espaço, b será maior que 0. Seja considerado, então, um caso com $b = 1$. Também sabemos que este tipo de buraco negro é regular, ou seja, em $\rho = 0$ a solução $F(\rho)$ possui um valor finito. Foi criado então um conjunto de dados com as seguintes condições iniciais:

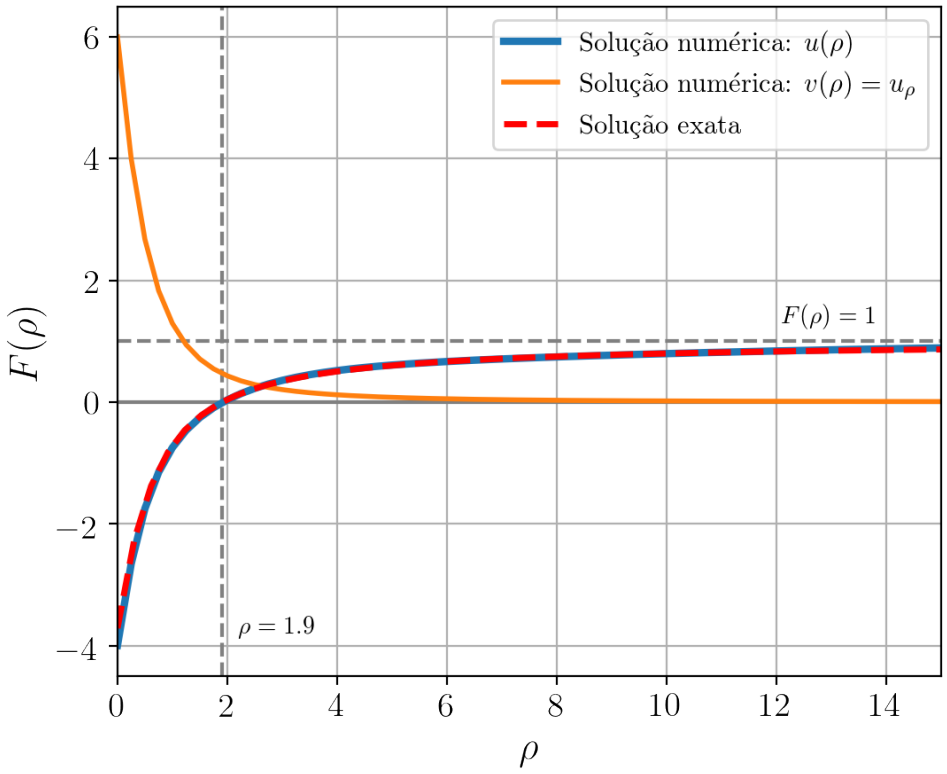
$$F(0) = -4; \quad F'(0) = 6; \quad \rho \in [0; 50]. \tag{4.3}$$

Esta solução é equivalente a um buraco negro com parâmetros $c = -\pi\rho_0/2b$ e $\rho_0 = 10$, podendo ser escrita como:

$$F(\rho) = -\frac{3\pi}{2}(1 + \rho^2) + \frac{1 + 3\rho + \rho^2 + 3\rho^3}{1 + \rho^2} + 3(1 + \rho^2) \cdot tan^{-1}(\rho). \tag{4.4}$$

Seria possível, também, impor que $F(+\infty) \to 1$, já que com este tipo de buraco negro a métrica volta a ser a usual (Minkowski) quando nos distanciamos o suficiente da fonte. Porém, para ser obtido um bom resultado não foi necessário impor esta condição.

Figura 21 – Solução numérica referente ao termo da métrica $F(\rho)$ para um buraco negro regular do tipo Phantom utilizando PINN.



Fonte: Elaborado pelo autor.

O resultado numérico obtido utilizando PINN pode ser visualizado no gráfico contido na figura 21.

Neste buraco negro temos um horizonte de eventos em $\rho = 1,900$ e a máquina conseguiu obter um horizonte em $\rho = 1,897$, ou seja, com erro percentual de 0,16%. O erro MAE[1] da curva foi de $2,89 \times 10^{-2}$, o que é um resultado suficiente para ser observado o comportamento observado.

4.2 Buraco de minhoca

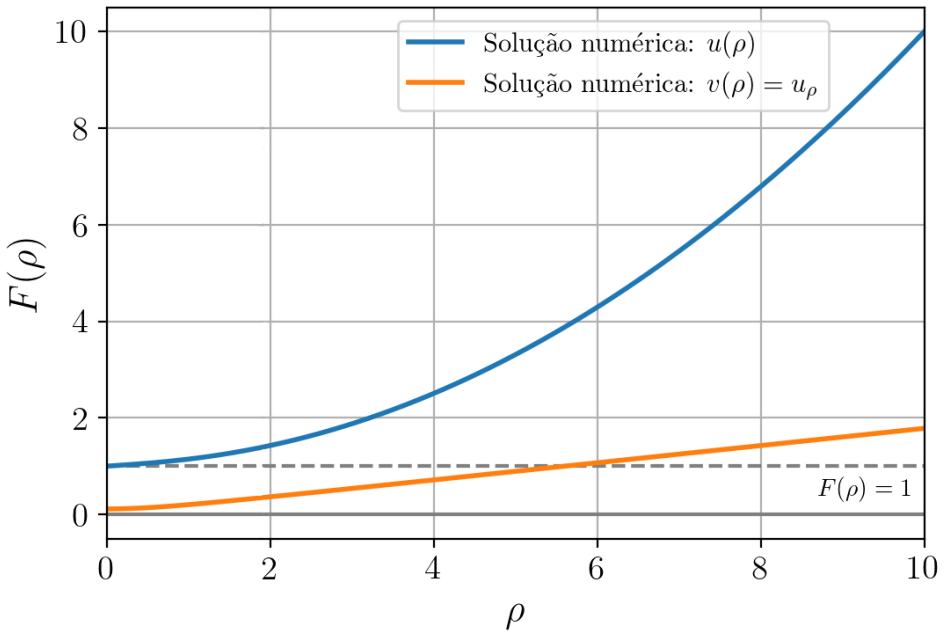
Para haver um buraco de minhoca é preciso que sua garganta (b) seja não-nula. Aqui foi escolhido $b = 1$. Assim, a função $F(0) > 0$ de modo que não existam horizontes de eventos presentes. O comportamento no infinito espacial será escolhido como sendo ($F(+\infty) = +\infty$). Desta vez foi possível tentar uma estratégia diferente, escolhendo arbitrariamente determinadas condições e então verificando se o método PINN pode esboçar uma solução que apresente o comportamento esperado para um objeto deste tipo. Então, foram impostas as seguintes condições iniciais e finais:

$$F(0) = 1; \quad F(10) = 10; \quad \rho \in [0; 10]. \tag{4.5}$$

[1] Estes erros foram calculados apenas dentro do domínio estabelecido.

Na figura 22 é apresentado o resultado numérico obtido utilizando a PINN.

Figura 22 – Solução numérica referente ao termo da métrica $F(\rho)$ para um buraco de minhoca utilizando PINN.



Fonte: Elaborado pelo autor.

Como estavamos procurando uma solução e não tentando verificar algum caso já conhecido com solução exata, não foi possível obter o erro MAE. Porém, é possível verificar o quão bem estão sendo obedecidas a equação e as condições de contorno pelo valor final da *loss*, que neste caso foi de 2.35×10^{-6}, sendo um excelente resultado se comparado com a *loss* dos outros casos, que tiveram a mesma ordem de grandeza.

4.3 Buraco negro de Schwarzschild

Como foi argumentado anteriormente, a solução de Schwarzschild é uma solução de vácuo. Assim, não havendo um campo escalar, a constante b será nula na equação 4.1, o que faz com que se retorne à coordenada radial usual $r = \sqrt{\rho^2 + b^2}\Big|_{b\to 0} = \rho$. É sabido que este tipo de buraco negro faz com que a geometria volte a ser plana quando nos distanciamos da fonte ($F(+\infty) \to 1$) e que, como é um buraco negro com singularidade, em $\rho = 0$ a solução diverge para $-\infty$. Foi criado, então, um conjunto de dados com as seguintes condições iniciais[2]:

$$F(0,4) = -4; \quad F(5) = 0,6; \quad F'(5) = 0,08; \quad \rho \in [0,4;5]. \tag{4.6}$$

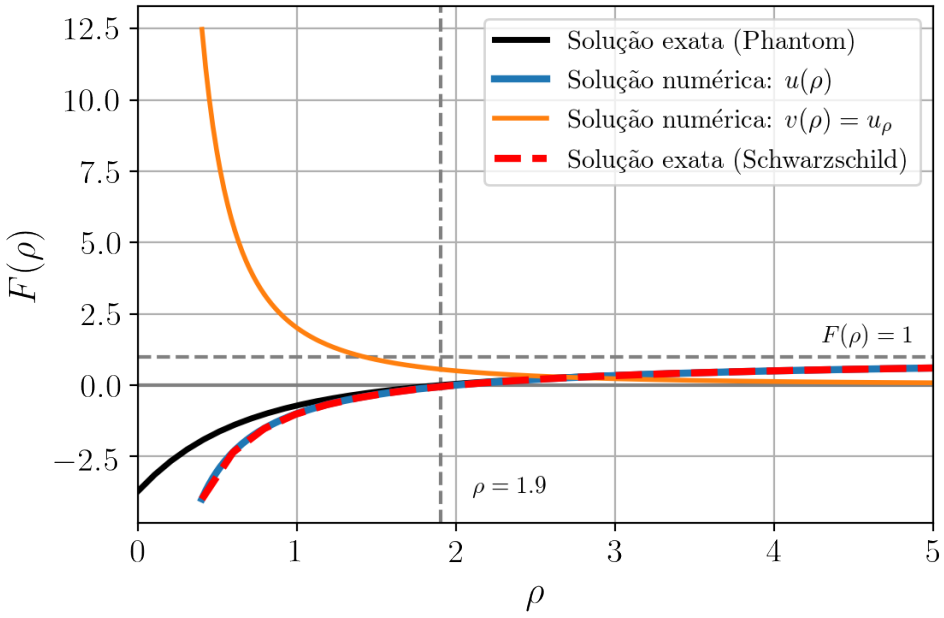
[2] Estas condições e domínio para que o algoritmo retorne um bom resultado foram obtidas através de testes.

Este caso é equivalente a um buraco negro com massa geométrica $m = 1$, cuja solução exata é dada por 4.7.

$$F(\rho) = 1 - \frac{2}{\rho}; \tag{4.7}$$

Na figura 23 é apresentado o resultado numérico obtido utilizando a PINN para este caso

Figura 23 – Solução numérica referente ao termo da métrica $F(\rho)$ para um buraco negro de Schwarzschild utilizando PINN.



Fonte: Elaborado pelo autor.

Para este buraco negro, temos um horizonte de eventos em $\rho = 2,0000$ e o resultado numérico obteve um horizonte de eventos em $\rho = 2,0006$, ou seja, com erro de 0,03%. O erro MAE da curva foi de $8,32 \times 10^{-4}$, o que é um resultado muito bom.

4.4 Buraco negro de Schwarzschild-de Sitter

Assim como a solução de Schwarzschild, deve-se considerar $b = 0$. Diferente dos casos anteriores a métrica deste tipo de buraco negro diverge quando nos distanciamos da fonte ($F(+\infty) \to -\infty$), no entanto ainda é um buraco negro com singularidade, então em $\rho = 0$ a solução também diverge para $-\infty$. Foram gerados dois conjuntos de dados com as distintas condições iniciais[3]:

$$F(0,4) = -7; \quad F'(0,4) = 32; \quad F(5) = -0,125; \quad F'(5) = 0,21; \quad \rho \in [0,4;5]; \text{ e} \tag{4.8}$$

$$F(0,4) = -4,032; \quad F'(0,4) = 12,34; \quad F(5) = -4,4; \quad F'(5) = 1,92; \quad \rho \in [0,4;5]. \tag{4.9}$$

[3] É bem provável que nestes casos poderiam ser utilizadas menos condições, mas isto não foi testado.

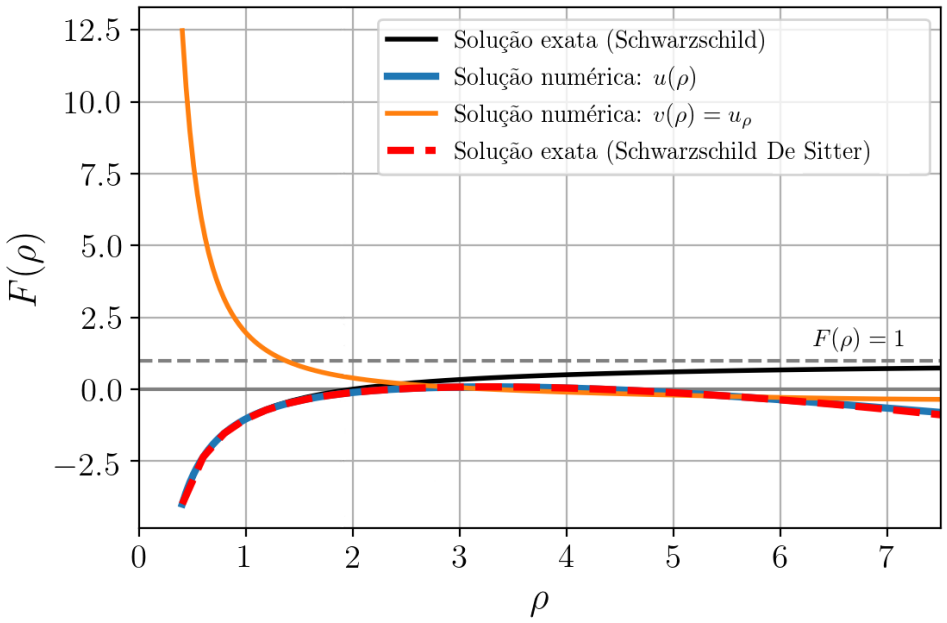
Estas condições são referentes a um caso com dois horizontes (um de eventos e um cosmológico) e um caso com singularidade nua, ou seja, quando a função $F(\rho)$ não cruza o eixo ρ, não havendo assim horizonte de eventos para cobrir a singularidade. As soluções exatas para ambos os casos são dadas respectivamente por 4.10 e 4.11.

$$F(\rho) = 1 - \frac{2}{\rho} - 0.029\rho^2 \tag{4.10}$$

$$F(\rho) = 1 - \frac{2}{\rho} - 0.2\rho^2 \tag{4.11}$$

Os resultados numéricos obtidos utilizando PINN podem ser visualizados nos gráficos contidos nas figuras 24 e 25. Para estes objetos foram obtidos os erros MAE de $2,08 \times 10^{-4}$ e $2,35 \times 10^{-3}$ respectivamente, o que podem ser considerados resultados bons.

Figura 24 – Solução numérica referente ao termo da métrica $F(\rho)$ para um buraco negro do tipo Schwarzschild-De Sitter com dois horizontes utilizando PINN.



Fonte: Elaborado pelo autor.

4.5 Buraco negro de Schwarzschild-Anti-de Sitter

Este caso é similar ao do buraco negro de Schwarzschild-de Sitter, porém a métrica agora diverge positivamente quando nos distanciamos da fonte ($F(+\infty) \to +\infty$). Foi criado, então, um conjunto de dados com as condições:

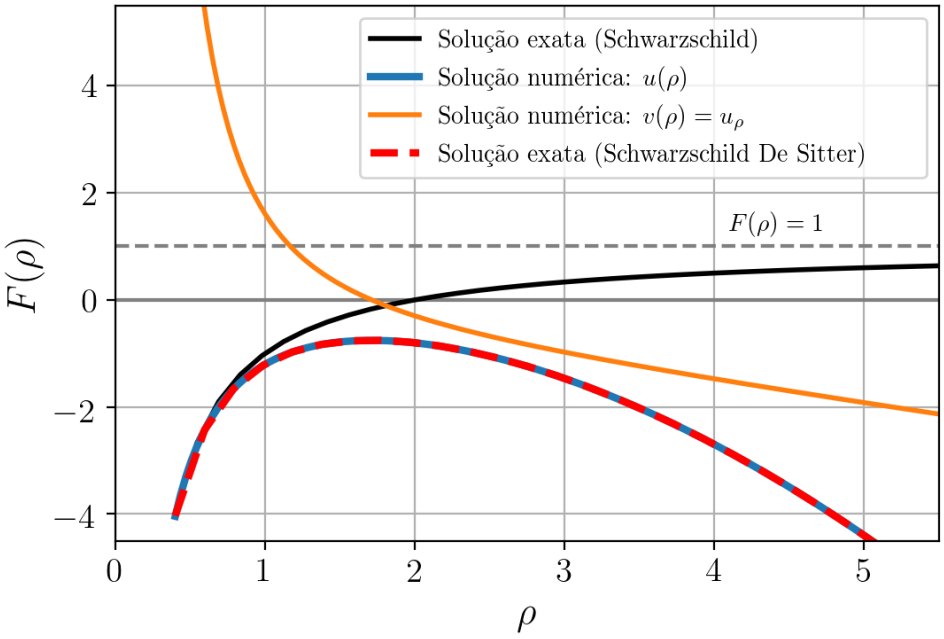
$$F(0,4) = -7; \quad F'(0,4) = 12,5232; \quad F(5) = 1,325; \quad F'(5) = 0,37; \quad \rho \in [0,4;5]. \tag{4.12}$$

A solução exata é dada por 4.13.

$$F(\rho) = 1 - \frac{2}{\rho} + 0.029\rho^2 \tag{4.13}$$

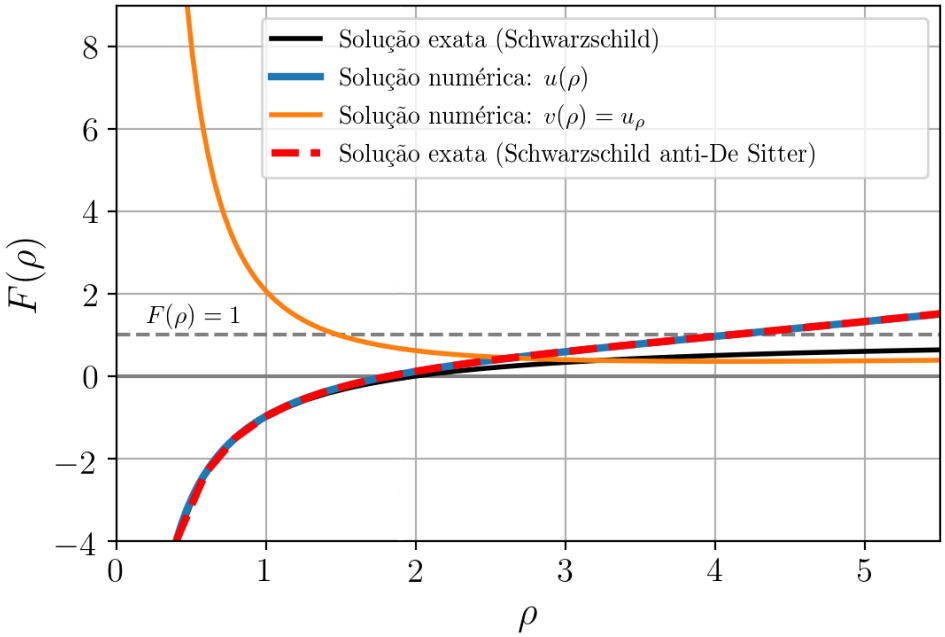
O resultado numérico obtido utilizando PINN pode ser visualizado no gráfico contido na imagem 26. Para este objeto, há um horizonte de eventos em $\rho = 1,824$ e a máquina conseguiu obter um horizonte em $\rho = 1,827$, ou seja, com erro de 0,16%. O erro MAE da curva foi de $1,46 \times 10^{-3}$, o que é um bom resultado.

Figura 25 – Solução numérica referente ao termo da métrica $F(\rho)$ para um buraco negro do tipo Schwarzschild-De Sitter com singularidade nua utilizando PINN.



Fonte: Elaborado pelo autor.

Figura 26 – Solução numérica referente ao termo da métrica $F(\rho)$ para um buraco negro do tipo Schwarzschild-Anti-De Sitter utilizando PINN.



Fonte: Elaborado pelo autor.

5 Conclusões e Discussões

As PINN constituem uma ferramenta numérica extremamente recente se comparado com outros métodos já consagrados para solução de equações diferenciais da física, portanto ainda possuem muito potencial a ser explorado assim como problemas a serem resolvidos.

Um dos problemas mais consideráveis se compararmos com outros métodos é o alto custo computacional. Para resolvermos um problema simples como o da equação de calor iremos gastar alguns minutos em uma boa GPU, enquanto em outros métodos poderíamos até resolver em segundos em uma CPU.

Também é difícil aproximar "funções de alta frequência". Por exemplo, quando se tenta resolver a equação de onda para os primeiros harmônicos é fácil obter uma boa aproximação, porém, quando se tenta aproximar harmônicos superiores (a partir do 5º) o tempo de treinamento e/ou tamanho da rede necessário se tornam impraticáveis. Este problema é conhecido na literatura como "F-principle" ou "spectral bias" (KARNIADAKIS et al., 2021). Por este motivo é muito mais custoso resolver com PINN o caso do oscilador harmônico do que uma equação não-linear como a de Burgers.

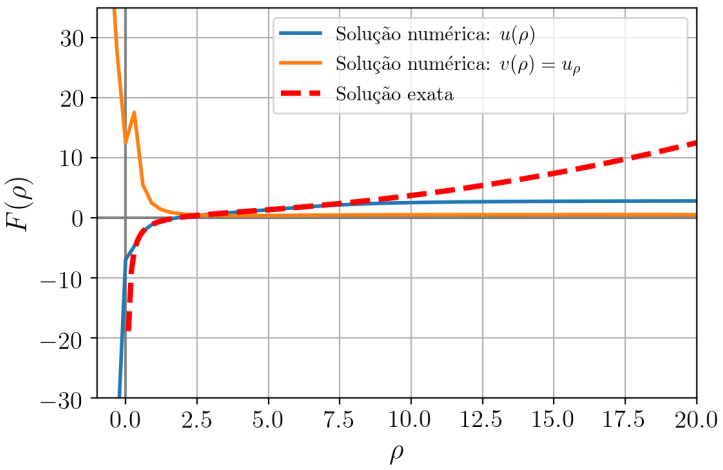
Um tópico que também merece atenção é como a função de custo é muitas vezes definida de forma complicada, envolvendo diversos termos. Isto faz os termos competirem entre si. Por exemplo, podemos ter uma *loss* muito baixa, porém quando verificamos os resultados podemos ver que a rede está representando muito bem as condições de contorno enquanto representa a equação diferencial muito fracamente. Um termo da função de custo compensa o outro e entrega uma boa *loss* enquanto a solução da equação está exageradamente incorreta.

É muito arriscado confiar em soluções das equações fora do domínio em que a rede foi treinada. Quando se tenta utilizar a rede neural pra prever o termo da métrica fora do domínio em que a equação foi solucionado vemos que a solução numérica difere fortemente da solução exata, comportamento que pode ser observado no gráfico da figura 27. Apesar de algumas extrapolações parecerem razoáveis (imagem 28[1]) a extrapolação em geral não retorna bons resultados.

É muito complicado resolver casos onde há singularidades envolvidas. No caso do buraco negro de Schwarzschild foram feitos extensivos testes até que determinadas condições de contorno retornassem um bom resultado. Mesmo assim, é possível ver (imagem 29) que a rede neural entrega uma solução contínua para este caso, o que também já era esperado.

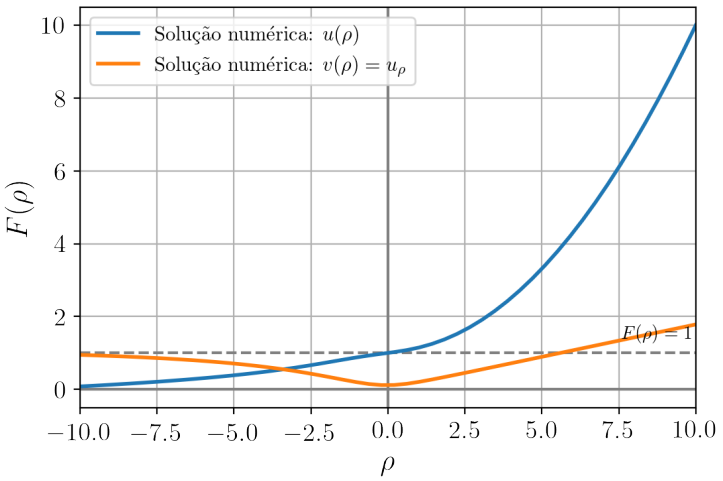
[1] Neste caso, estudos posteriores precisam ser realizados.

Figura 27 – Extrapolação da solução para o buraco negro de Schwarzschild-Anti-de Sitter.



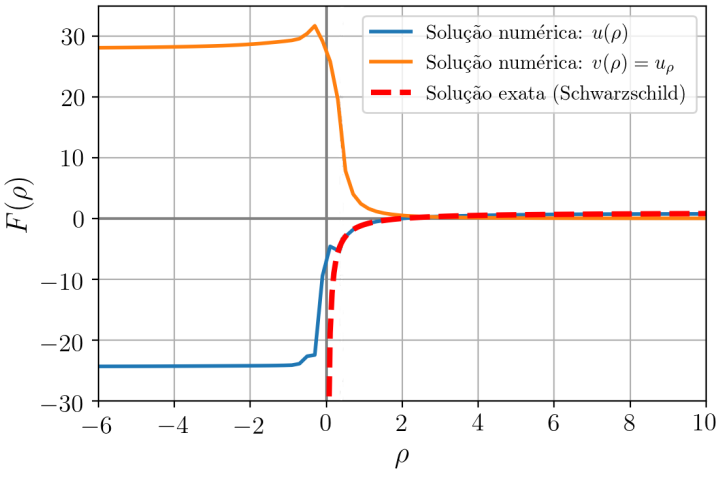
Fonte: Elaborado pelo autor.

Figura 28 – Extrapolação da solução para o buraco de minhoca.



Fonte: Elaborado pelo autor.

Figura 29 – Extrapolação da solução para o buraco negro de Schwarzschild.



Fonte: Elaborado pelo autor.

Mesmo sendo um algoritmo recente, diversas implementações tem surgido na intenção de incrementar o método base PINN. A equação de onda pode ser resolvida com o algoritmo original implementando algumas modificações (MOSELEY; MARKHAM; NISSEN-MEYER, 2020). Casos onde o domínio é muito extenso, resultando em um grande custo computacional, podem ter seu tempo de treino reduzido utilizando PPINN (*Parareal Physics-informed Neural Network*), uma variante que reparte em frações, dividindo um problema custoso em problemas menores e de mais fácil solução (MENG et al., 2020). Uma variante muito interessante é a *Multi-scale Physics Informed Neural Networks*, onde os autores se propõem a forçar a formação de soluções de alta frequência incrementando o método com séries de Fourier, além de resolver o problema da função de custo atribuindo pesos aos termos para evitar que um termo compense outro (WANG; WANG; PERDIKARIS, 2021)[2]. A extrapolação das soluções para fora do domínio talvez possa ser melhorada com a utilização de redes neurais recorrentes, ou similares.

Apesar dos problemas ainda não resolvidos, este tipo de ferramenta tem demonstrado ótimos resultados em alguns tópicos, em especial, na aproximação de operadores não-lineares (LU et al., 2021). Sendo possível verificar que o custo computacional é menor do que outros algoritmos neste caso.

PINN e similares geralmente são implementados utilizando TensorFlow, o que pode não ser uma ferramenta muito amigável para alguns programadores. Porém, já existem bibliotecas em desenvolvimento para a utilização desses algoritmos, em especial a DeepXDE (LU et al., 2020), que possui código aberto no GitHub[3], sendo continuamente aprimorado por diversos contribuidores.

Em conclusão, as *Physics-informed Neural Networks* se demonstraram capazes de aproximar satisfatoriamente as componentes das métricas para diversos tipos de objetos gravitacionais da Relatividade Geral, com um erro bom o suficiente param serem observados os comportamentos esperados para as soluções. É possível que um erro menor possa ser alcançado já que não foi realizada uma análise muito rigorosa dos hiperparâmetros da rede neural. As PINN constituem uma ferramenta extremamente recente e portanto ainda carecem de soluções para alguns problemas, principalmente o alto custo computacional, porém sua habilidade de lidar com poucos dados de treino, dados com ruído e casos de não-linearidade, garantem que ainda tenha muito potencial para a resolução de problemas da Física e outras áreas.

Este trabalho de conclusão de curso teve seus resultados preliminares apresentados oralmente no XLI Encontro Nacional de Física de Partículas e Campos e possuiu financiamento do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) em forma de bolsa de Iniciação Tecnológica em TICs.

[2] Também se propõem a resolver a equação de onda.
[3] <https://github.com/lululxvi/deepxde>.

Referências

ABADI, M. et al. *TensorFlow: A system for large-scale machine learning*. 2016. 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16). Disponível em: <https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf>. Acesso em: 18 mar 2022. Citado 2 vezes nas páginas 26 e 66.

ALICE, M. *Fitting a Neural Network in R; neuralnet package*. 2015. Datascience+. Disponível em: <https://datascienceplus.com/fitting-neural-network-in-r/>. Acesso em: 09 maio 2022. Citado na página 20.

BAYDIN, A. et al. Automatic differentiation in machine learning: a survey. *Journal of Machine Learning Research*, v. 18, p. 1–43, 2018. Citado na página 26.

BOM, C. D.; FARIA, E. *Deep Learning Lectures at XIII CBPF School version (2021), Lecture 1*. 2021. Clear nights are the Best! Disponível em: <https://clearnightsrthebest.com/deep-learning-minicourse/>. Acesso em: 09 fev 2022. Citado na página 19.

BOYCE, W.; DIPRIMA, R. *Equações Diferenciais Elementares e Problemas de Valores de Contorno*. 9. ed. Rio de Janeiro, RJ: LTC - Livros Técnicos e Científicos Editora Ltda, 2010. Citado na página 22.

BRONNIKOV, K.; FABRIS, J. Regular phantom black holes. *Physical Review Letters*, v. 96, n. 25-30, p. 1–4, 2006. Citado na página 40.

CARLEO, G.; TROYER, M. Solving the quantum many-body problem with artificial neural networks. *Science*, v. 355, p. 602–606, 2016. Citado na página 14.

CARNEIRO, T. et al. Performance analysis of google colaboratory as a tool for accelerating deep learning applications. *IEEE Access*, v. 6, p. 61677–61685, 2018. Citado na página 67.

CHOLLET, F. *Deep Learning with Python*. 1. ed. Shelter Island, NY: Manning Publications Co., 2017. Citado 2 vezes nas páginas 18 e 66.

CHOLLET, F. *Deep Learning with Python*. 2. ed. Shelter Island, NY: Manning Publications Co., 2021. Citado 3 vezes nas páginas 17, 18 e 66.

D'INVERNO, R. *Introducing Einstein's Relativity*. 1. ed. New York: Oxford University Press, 1998. Citado 2 vezes nas páginas 36 e 38.

DOCKHORN, T. A discussion on solving partial differential equations using neural networks. *ArXiv*, v1, n. 1907.07200, p. 1–9, 2019. Citado na página 14.

EHT, C. *Astronomers Reveal First Image of the Black Hole at the Heart of Our Galaxy*. 2022. Event Horizon Telescope. Disponível em: <https://eventhorizontelescope.org/blog/astronomers-reveal-first-image-black-hole-heart-our-galaxy>. Acesso em: 30 maio 2022. Citado na página 13.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. 1. ed. [S.l.]: MIT Press, 2016. Citado na página 18.

HANCOCK, M. *Linear Partial Differential Equations Lecture Notes, The 1D Heat Equation*. 2006. MIT Open Courseware. Disponível em: <https://ocw.mit.edu/courses/mathematics/18-303-linear-partial-differential-equations-fall-2006/lecture-notes/heateqni.pdf>. Acesso em: 03 mar 2022. Citado na página 43.

HASHIMOTO, K. et al. Deep learning and ads/cft. *Physical Review D*, v. 98, p. 4–15, 2018. Citado na página 14.

HORNIK, K.; STINCHCOMBRE, M.; WHITE, H. Multilayer feed-forward networks are universal approximators. *Neural Networks*, v. 2, p. 359–366, 1989. Citado na página 22.

ITEN, R. et al. Discovering physical concepts with neural networks. *Physical Review Letters*, v. 124, n. 010508, p. 1–10, 2020. Citado na página 14.

IóRIO, V. *EDP: Um Curso de Graduação*. Rio de Janeiro: Coleção Matemática Universitária - IMPA, 2010. Citado na página 23.

KARNIADAKIS, G. E. et al. Physics-informed machine learning. *Nature Review Physics*, v. 3, p. 422–440, 2021. Citado 2 vezes nas páginas 28 e 58.

KORYAGIN, A.; KHUDOROZKOV, R.; TSIMFER, S. Pydens: a python framework for solving differential equations with neural networks. *ArXiv*, v1, n. 1909.11544, p. 1–8, 2019. Citado na página 15.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *Nature*, v. 521, p. 436–444, 2015. Citado na página 18.

LEVITUS, M. *The Wave Equation in One Dimension*. 2020. Chemistry Libretexts. Disponível em: <https://chem.libretexts.org/>. Acesso em: 03 mar 2022. Citado na página 45.

LIN, H.; TEGMARK, M.; ROLNICK, D. Why does deep and cheap learning work so well? *Journal of Statistical Physics*, v. 168, p. 1223–1247, 2017. Citado na página 22.

LIU, Z.; YANG, Y.; CAI, Q. Neural network as a function approximator and its application in solving differential equations. *Applied Mathematics and Mechanics volume*, v. 40, p. 237–248, 2019. Citado na página 14.

LU, L. et al. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, n. 3, p. 218–229, 2021. Citado na página 60.

LU, L. et al. Deepxde: A deep learning library for solving differential equations. *ArXiv*, v1, n. 1907.04502, p. 1–21, 2020. Citado 2 vezes nas páginas 42 e 60.

MENG, X. et al. Ppinn: Parareal physics-informed neural network for time-dependent pdes. *Computer methods in applied mechanics and engineering*, n. 307, 2020. Citado na página 60.

MITCHELL, T. *Machine Learning*. 1. ed. [S.l.]: McGraw-Hill, 1997. Citado na página 17.

MOSELEY, B.; MARKHAM, A.; NISSEN-MEYER, T. Solving the wave equation with physics-informed deep learning. *ArXiv*, v1, n. 2006.11894, p. 1–13, 2020. Citado na página 60.

PISCOPO, M.; SPANNOWSKY, M.; WAITE, P. Solving differential equations with neural networks: Applications to the calculation of cosmological phase transitions. *Physical Review D*, v. 100, n. 016002, p. 1–12, 2019. Citado na página 14.

PRESS, W. et al. *Numerical Recipes: The art of Scientific Computing*. 3. ed. [S.l.]: Cambridge University Press, 2007. Citado na página 25.

RAISSI, M. *Physics Informed Neural Networks*. 2020. GitHub. Disponível em: <https://github.com/maziarraissi/PINNs>. Acesso em: 08 abr 2022. Citado na página 29.

RAISSI, M.; PERDIKARIS, P.; KARNIADAKIS, G. Inferring solutions of differential equations using noisy multi-fidelity data. *Journal of Computational Physics*, v. 335, p. 736–746, 2017. Citado na página 15.

RAISSI, M.; PERDIKARIS, P.; KARNIADAKIS, G. e. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *ArXiv*, v1, n. 1711.10561, p. 1–22, 2017. Citado 2 vezes nas páginas 15 e 25.

RAISSI, M.; PERDIKARIS, P.; KARNIADAKIS, G. e. Physics informed deep learning (part ii): Data-driven discovery of nonlinear partial differential equations. *ArXiv*, v1, n. 1711.10566, p. 1–19, 2017. Citado 2 vezes nas páginas 15 e 25.

RAISSI, M.; PERDIKARIS, P.; KARNIADAKIS, G. e. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, v. 378, p. 686–707, 2019. Citado 3 vezes nas páginas 15, 25 e 27.

RUDY, S. et al. Data-driven discovery of partial differential equations. *Science Advances*, v. 3, n. 4, p. 1–6, 2017. Citado na página 14.

SCHWARZSCHILD, K. Über das gravitationsfeld eines massenpunktes nach der einsteinschen theorie. *Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften (Berlin)*, v. 6, p. 186–196, 1916. Citado na página 14.

STRAUSS, W. *Partial Differential Equations: An Introduction*. New York: Wiley, 2008. Citado 2 vezes nas páginas 23 e 24.

VALIZADEGAN, H. et al. Exominer: A highly accurate and explainable deep learning classifier that validates 200+ new exoplanets. *Bulletin of the AAS*, v. 56, n. 6, p. 218–229, 2021. Citado na página 14.

VALLE, M. *Aula 7, Equações Não-Homogêneas com Coeficientes Constantes e o Método dos Coeficientes a Determinar*. 2016. IME, Unicamp, MA311 - Cálculo III. Disponível em: <https://www.ime.unicamp.br/~valle/Teaching/2016/MA311/Aula7.pdf>. Acesso em: 03 mar 2022. Citado na página 47.

WANG, s.; WANG, H.; PERDIKARIS, P. On the eigenvector bias of fourier feature networks: From regression to solving multi-scale pdes with physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, v. 384, 2021. Citado na página 60.

WEINBERG, S. *Gravitation and Cosmology: Principles and Applications of the General Theory of Relativity*. 1. ed. New York, London, Sydney, Toronto: John Wiley Sons, Inc., 1972. Citado 2 vezes nas páginas 32 e 34.

Apêndices

APÊNDICE A – Implementação e Hardware

A.1 TensorFlow

Em relação a implementação, não só devemos nos preocupar com o *framework* para solução de nossas equações diferenciais (PINN, ou outro algoritmo similar), mas também com o *framework* sobre o qual criaremos nosso modelo de *machine learning*, ou seja, qual conjunto de ferramentas usaremos para programar nosso algoritmo de *Deep Learning*.

O TensorFlow (ABADI et al., 2016) é um sistema criado por pesquisadores e outros funcionários da Google para implementação e desenvolvimento de modelos de *machine learning*, em especial, para auxiliar a implementação de modelos de redes neurais profundas. A biblioteca é criada para funcionar em diversas plataformas, sistemas operacionais e diferentes estruturas de unidades de processamento, em especial, auxiliar o uso de *clusters* de GPUs.

A ferramenta de Maziar Raissi (PINNs) foi originalmente desenvolvida utilizando uma implementação na linguagem Python 3.5-3.6 e TensorFlow versão 1.15.X. É possível encontrar implementações independentes na plataforma GitHub que transpõem as PINN para versões mais recentes de suas APIs estruturantes, como TensorFlow 2.X, que promete ser mais eficiente, além de utilizar o *framework* Keras (ver (CHOLLET, 2017; CHOLLET, 2021), livro do criador do Keras) para construir as redes neurais, o que deixam os *scripts* mais simples e amigáveis.

Porém, optamos por inicialmente utilizar as ferramentas como concebidas e implementadas por Raissi e colaboradores, em Python 3.6 e TensorFlow 1.15.2, o que se seguiu até o fim deste projeto.

A.2 Workstation e Colab

Como comentado anteriormente, uma nova onda de IA acontece já que possuímos, atualmente, *hardwares* poderosos o suficiente para que consigamos treinar modelos de *machine learning* extremamente robustos, em especial, modelos de *Deep Learning*.

A ferramenta concebida por Raissi e colaboradores não escapa da extensiva necessidade de processamento. Para algumas equações diferenciais mais simples, como uma equação de propagação do calor unidimensional, podemos utilizar uma rede neural com relativamente poucos neurônios e camadas na rede, o que é totalmente praticável de ter seu modelo treinado em uma CPU em no máximo algumas dezenas de minutos para atingirmos uma precisão desejável.

Para outros problemas, como uma equação de onda com condições de Dirichlet, precisaremos de mais poder computacional para que possamos treinar nosso modelo em um tempo praticável. Desta forma, é essencial que pensemos no *hardware* que utilizaremos para treinar nossos modelos, já que, com os computadores que utilizamos em nosso cotidiano, utilizar tais métodos é impraticável.

Para o projeto estava disponível, no LAC (Laboratório de Astrofísica Computacional) da UNIFEI, uma *workstation* com aproximadamente 25Gb de RAM e uma GPU NVIDIA. O que foi suficiente para o treinamento dos modelos nos momentos iniciais do projeto.

Porém, devido a pandemia de COVID-19 no início de 2020 as atividades letivas no Campus Itajubá foram suspensas e o acesso ao LAC, e consequentemente à *workstation*, foi reduzido consideravelmente. Desta forma, tivemos que buscar outras formas de treinar os modelos. Há alternativas para a utilização de *hardware* na nuvem para treinamento de modelos de ML, como a AWS da Amazon, e serviços grátis providos pela Kaggle e Google.

O serviço da Google conhecido como Google Colaboratory, ou simplesmente Colab, cede aos usuários o uso de CPUs, GPUs e até TPUs. Uma avaliação prática conduzida por Caneiro et al (CARNEIRO et al., 2018) chegou a conclusão de que, para aceleração de treinos de *Deep Learning*, os recursos disponibilizados pelo Colab tem performance semelhante a *hardware* local se ponderados os recursos alocados. Em alguns casos, o Colab se demonstra até mais eficiente.

Porém, Carneiro e colaboradores salientam que para tratarmos de casos do "mundo real", que um pesquisador encararia, os recursos grátis do Google Colaboratory estão longe do suficiente.

Como previsto, durante o desenvolvimento deste projeto ficamos limitados a *runs* de menos de 12 horas (limite do Colab para uso de GPUs e TPUs) para treinamento de nossos modelos, o que impactou na precisão das soluções das equações mais sofisticadas. Porém, os resultados finais foram satisfatórios como um todo.