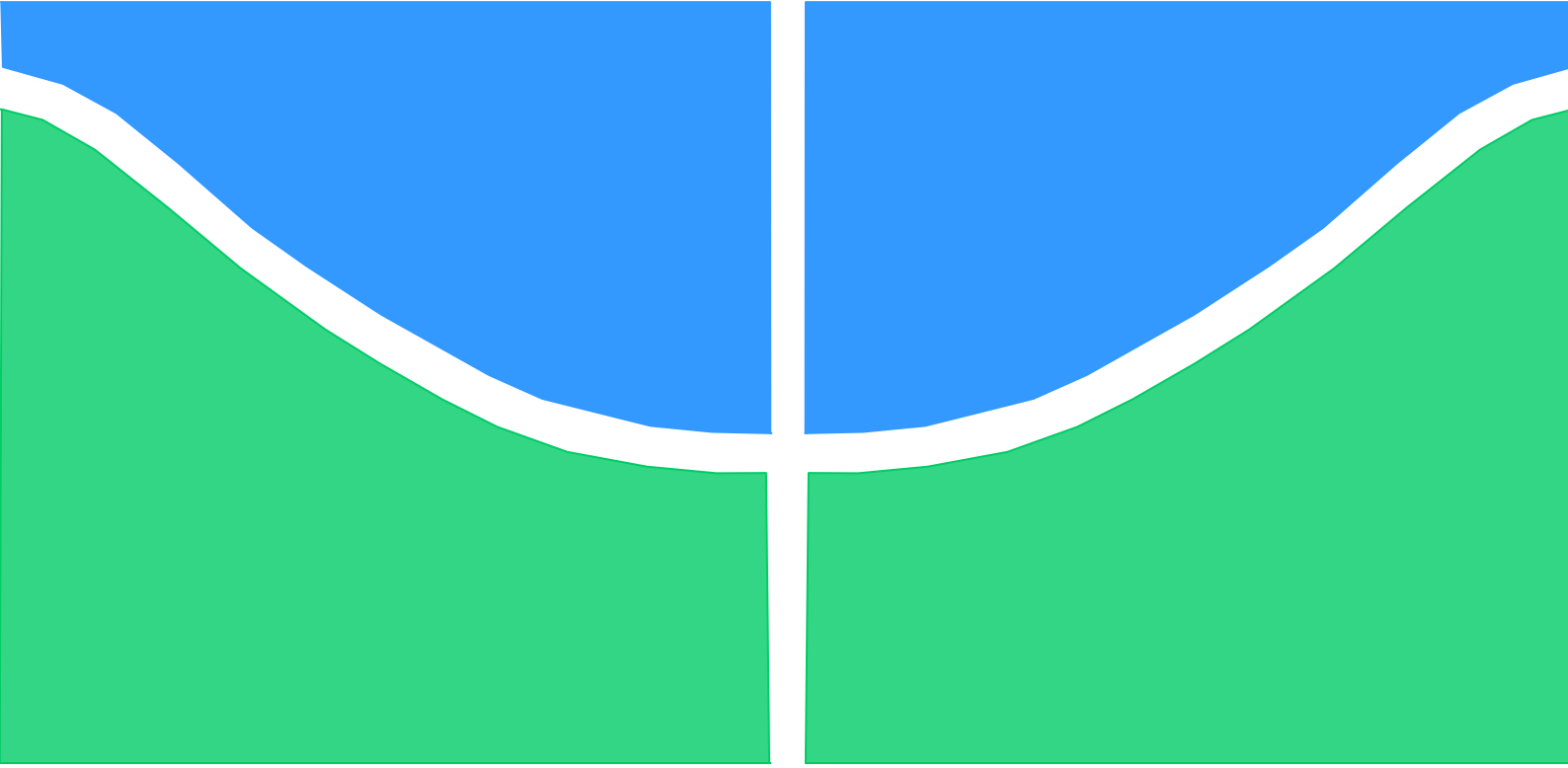

PROJETO DE GRADUAÇÃO

UMA ABORDAGEM DE APRENDIZADO DE MÁQUINA PARA MECÂNICA DOS FLUIDOS

Rogério Werneck Costa Rodrigues Filho

Brasília, 17 de maio de 2021

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

PROJETO DE GRADUAÇÃO

UMA ABORDAGEM DE APRENDIZADO DE MÁQUINA PARA MECÂNICA DOS FLUIDOS

Rogério Werneck Costa Rodrigues Filho

Relatório submetido ao Departamento de Engenharia Mecânica como requisito parcial para obtenção do grau de Bacharel em Engenharia Mecânica

Banca Examinadora

Prof. Adriano Possebon Rosa, ENM/UnB _____
Orientador

Prof. André von Borries Lopes, ENM/UnB _____

Prof. Braulio Gutierrez Pimenta, ENM/UnB _____

Agradecimentos

Primeiro gostaria de agradecer à Deus por ter conquistado mais esta etapa.

Em seguida, gostaria de agradecer à minha mãe, Adalgiza, por todo amor e dedicação. Ao meu pai, Rogério, por sempre me ensinar a ser cada vez melhor. À minha madrasta, Diana, por todo carinho. À minha irmã, Beatriz, que sempre me escuta.

Não posso deixar de agradecer aos vários amigos que tive o privilégio de ter ao meu lado ao longo desta graduação. Em especial aos que fundaram nossa atlética ao meu lado (Leo Arantes (veterano), Camis, Grande, Mihara, Chorão, Alicate, John, Trakinas, Cassebe e Takatsu). E claro, aos que em boa parte dos semestres estiveram ao meu lado, enlouquecendo com horas de estudo, noites não dormidas e chamadas infinitas no Google Meet, os amigos do Bruxaria (Bob, Dinha, Shun, Hansen, Enzin e Smile).

Agradeço ao professor Adriano Possebon por ter encarado este desafio comigo, além de outras ocasiões em que tive o prazer de trabalhar com ele.

E por fim, parafraseando Snopp Dogg ao receber sua estrela na calçada da fama, devo agradecer à mim. Por não ter desistido quando as coisas foram difíceis, e por ter me esforçado sempre que necessário.

Rogério Werneck Costa Rodrigues Filho

RESUMO

A crescente produção de dados tornou possível o uso de algoritmos de aprendizado de máquina no contexto de engenharia mecânica. A simulação numérica tradicional muitas vezes possui elevado custo computacional, de tal modo que realizar simulações em tempo real se torna inviável. Alguns programas de simulação numérica começaram a permitir que seus usuários utilizem computação em nuvem para realizar simulações mais rápidas, de tal forma que estas empresas começaram a desenvolver um banco de dados abundante com problemas resolvidos com a metodologia CAE (*Computer Aided Engineering*). Além disso, ambientes com elevado nível de monitoramento costumam possuir diversos sensores das condições ambientes, de tal forma que se cria um registro histórico das condições locais, como, por exemplo, a temperatura no interior de *data centers* ao longo do tempo. Por se tratar de um ambiente complexo, a modelagem computacional utilizando métodos numéricos pode ser inviável. Desta forma, o uso de algoritmos de aprendizado de máquina pode se tornar uma opção. Como algoritmos de aprendizado de máquina não necessitam de um modelo matemático que descreve as condições físicas do ambiente para se obter resultados válidos, o uso destas técnicas facilitam todo o processo de modelagem. Neste trabalho, são propostos modelos de aprendizado de máquina para solucionar o problema da equação do calor bidimensional e o problema da cavidade quadrada com tampa deslizante utilizando um Perceptron de múltiplas camadas, sendo a fonte de dados um conjunto de resultados obtidos com os métodos tradicionais de simulação numérica, como o método da projeção. O uso desta metodologia dispensa a aplicação de Navier-Stokes para produzir resultados confiáveis. Isto significa que o algoritmo consegue criar seu próprio modelo matemático capaz de resolver o problema tendo como base o banco de dados. Além disso, algoritmos de aprendizado de máquina não demandam elevado custo computacional após seu treinamento, de tal forma que simulações em tempo real começam a se tornar viáveis. Os resultados obtidos neste trabalho mostram que a precisão encontrada no resultado final não foi significativamente afetada de modo que são obtidos resultados de boa qualidade com o uso deste método.

ABSTRACT

The growth of data production has made it possible to use machine learning algorithms in the context of mechanical engineering. Traditional numerical simulation often has high computational cost, such that performing real-time simulations becomes unfeasible. Some numerical simulation programs have begun to allow their users to use cloud computing to perform faster simulations, such that these companies have begun to develop an abundant database of problems solved with CAE (Computer Aided Engineering) methodology.In addition, environments with a high level of monitoring usually have multiple sensors of the ambient conditions, such that a historical record of local conditions is created, such as the temperature inside data centers over time. As it is a complex environment, computational modeling using numerical methods may be impractical. Thus, the use of machine learning algorithms may become an option. Since machine learning algorithms do not need a mathematical model that describes the physical conditions of the environment to obtain valid results, the use of these techniques facilitates the entire modeling process. In this paper, machine learning models are proposed to solve the two-dimensional heat equation problem and the lid driven cavity flow problem lid using a multilayer Perceptron, with the data source being a set of results obtained with traditional numerical simulation methods, such as the projection method. The use of this methodology dispenses Navier-Stokes equation to produce reliable results. This means that the algorithm can create its own mathematical model capable of solving the problem based on data. In addition, machine learning algorithms do not require high computational cost after training, so that real-time simulations start to become feasible. The results obtained in this work show that the accuracy found in the final result was not significantly affected, so that good quality results are obtained using this method.

SUMÁRIO

1 Introdução 1

1.1 Mecânica dos Fluidos e Aprendizado de Máquina 1

1.2 Publicações Importantes 2

1.3 Aplicações 3

1.4 Objetivos 3

1.5 Estrutura do Trabalho 3

2 Fundamentação Teórica 5

2.1 Equações Governantes dos Fenômenos Físicos 5

2.1.1 Equação do Calor Bidimensional 5

2.1.2 Mecânica dos Fluidos 6

2.2 Problemas Propostos 6

2.2.1 Condução de Calor Bidimensional 6

2.2.2 Cavidade Quadrada com Parede Móvel 7

2.3 Metodologia Numérica 8

2.3.1 Geração e Convergência de Malha 8

2.3.2 Diferenças Finitas 9

2.3.3 Método da Projeção 10

2.4 Aprendizado de Máquina 10

2.4.1 Regressão e Classificação 11

2.4.2 Divisão do Conjunto de Dados em Teste e Treino 12

3 Introdução ao Método da Projeção 14

3.1 Método da Projeção 14

3.1.1 Malha Defasada 15

3.1.2 Critérios de Estabilidade e Convergência 15

4 Redes Neurais Artificiais 16

4.1 Perceptron 16

4.2 Perceptron de Múltiplas Camadas 17

4.3 Mecanismo de Aprendizagem 19

4.3.1 Função Perda e Retropropagação de Erro 19

4.3.2 Otimizador 20

4.3.3 Sobreajuste e Parada Antecipada 22
4.3.4 Função de Ativação 23

5 Resultados 26
5.1 Problema da Equação do Calor Bidimensional 26
5.1.1 Parâmetros utilizados 26
5.1.2 Função Perda 27
5.1.3 Comparativo com Resultados Numéricos 29
5.2 Problema da Cavidade 33
5.2.1 Parâmetros utilizados 33
5.2.2 Função Perda 34
5.2.3 Comparativo com Resultados Numéricos Existentes no Banco de Testes 37

6 Conclusões 42
6.1 Sugestão de Trabalhos Futuros 42

REFERÊNCIAS BIBLIOGRÁFICAS 43

Anexos 46

I Notebooks Utilizados 47

LISTA DE FIGURAS

2.1 Ilustração do problema de transferência de calor em placa bidimensional. As paredes laterais possuem temperatura fixa $T = 0$ e um quadrilátero com temperatura fixa $T = 1$ 6
2.2 Ilustração do problema da cavidade com tampa deslizante. As paredes laterais e inferior são fixas enquanto a parede superior se move com velocidade definida..... 7
2.3 Ilustração de uma malha quadrada. As intersecções representam os pontos que são efetivamente calculados..... 8
2.4 Malha discretizada com indicação dos índices 9
2.5 Diagrama ilustrativo de como são elaborados programas utilizando métodos tradicionais 11
2.6 Diagrama ilustrativo de como são elaborados programas utilizando aprendizado de máquina 11

3.1 Exemplo de malha defasada. Os círculos representam os pontos onde a pressão é calculada, os triângulos a componente u da velocidade e os quadrados a compontente v 15

4.1 Ilustração do funcionamento do Perceptron 17
4.2 Associação de Perceptrons 18
4.3 Perceptron de Múltiplas Camadas 18
4.4 Ilustração do avanço de sinal e retropropagação do erro para ajuste dos pesos sinápticos Adaptado de: Haykin (2008) 20
4.5 Ilustração do método do gradiente descente para uma função de uma variável. Adaptado de: Geron (2019) 20
4.6 Efeito de uma taxa de aprendizagem pequena para o funcionamento do algoritmo do gradiente descendente. Adaptado de Geron (2019) 21
4.7 Efeito de uma taxa de aprendizagem grande para o funcionamento do algoritmo do gradiente descendente. Adaptado de Geron (2019) 22
4.8 Exemplo do comportamento da curva Época x Perda para 4 diferentes valores de η. Adaptado de Geron (2019) 22
4.9 Comportamento da curva de perda para os dados de treino (preto) e teste (azul). Destaca-se o ponto de parada antecipada para evitar o sobreajuste do algoritmo. Adaptado de Haykin (2008) 23
4.10 Ilustração das funções de ativação utilizadas para modelos de classificação 24
4.11 Ilustração das funções de ativação utilizadas para modelos de regressão 25

5.1 Análise de convergência de malha para problema da placa aquecida..... 27
5.2 Comportamento da função perda para modelos (a) utilizando malha como entrada e 3 camadas, (b) utilizando malha como entrada e 4 camadas, (c) utilizando coordenadas como entrada e 3 camadas, (d) utilizando coordenadas como entrada e 4 camadas. São apresentados os resultados para diferentes taxas de aprendizado η. 28
5.3 Resultado numérico para $t = 0,00313$ 30
5.4 Resultado para $t = 0,00313$ para modelo (a) M_C_3_E_30 (b) M_C_4_E_30 (c) C_C_3_E_50 (d) C_C_4_E_50 30
5.5 Resultado numérico para $t = 0,09687$ 31
5.6 Resultado para $t = 0,09687$ para modelo (a) M_C_3_E_30 (b) M_C_4_E_30 (c) C_C_3_E_50 (d) C_C_4_E_50 31
5.7 Resultado numérico para $t = 0,16266$ 32
5.8 Resultado para $t = 0,16266$ para modelo (a) M_C_3_E_30 (b) M_C_4_E_30 (c) C_C_3_E_50 (d) C_C_4_E_50 32
5.9 Análise de convergência de malha para problema da cavidade..... 33
5.10 Função perda para modelo u com (a) 3 Camadas, (b) 4 Camadas e (c) com $\eta = 0,05$ 36
5.11 Função perda para modelo v com (a) 3 Camadas, (b) 4 Camadas e (c) com $\eta = 0,05$ 36
5.12 Função perda para modelo p com (a) 3 Camadas, (b) 4 Camadas e (c) com $\eta = 0,05$ 37
5.13 Resultado numérico para $t = 0,1$ 37
5.14 Resultado para $t = 0,1$ (a) resultado obtido pela rede (b) erro absoluto de p (c) erro absoluto de u (d) erro absoluto de v..... 38
5.15 Resultado numérico para $t = 2,0$ 38
5.16 Resultado para $t = 2,0$ (a) resultado obtido pela rede (b) erro absoluto de p (c) erro absoluto de u (d) erro absoluto de v..... 39
5.17 Resultado numérico para $t = 14,0$ 39
5.18 Resultado para $t = 14,0$ (a) resultado obtido pela rede (b) erro absoluto de p (c) erro absoluto de u (d) erro absoluto de v..... 40
5.19 Comparativo da linha central da cavidade para (a) pressão no eixo x (b) pressão no eixo y (c) velocidade vertical ao longo do eixo y (d) velocidade horizontal ao longo do eixo x 41

LISTA DE TABELAS

1.1 Artigos mais citados 2

5.1 Diferenciação da quantidade de neurônios nos principais tipos de modelo..... 27
5.2 EQM para cada modelo..... 28
5.3 Número de épocas para cada modelo..... 29
5.4 Tempo total de treinamento em segundos..... 29
5.5 Quantidade de neurônios nas camadas de entrada e saída..... 34
5.6 EQM de cada um dos modelos..... 34
5.7 Número de Épocas para cada modelo..... 35
5.8 Tempo total de treinamento em segundos 35

Capítulo 1

Introdução

1.1 Mecânica dos Fluidos e Aprendizado de Máquina

Fluidos estão presentes na natureza na forma líquida ou gasosa e possuem as mais diversas aplicações de engenharia, como a geração de energia. O estudo de fluidos em movimento, conhecido como dinâmica dos fluidos, e de fluidos em repouso, ou estática dos fluidos, em conjunto formam a Mecânica dos Fluidos (WHITE, 2019).

Experimentos e simulações voltadas para mecânica dos fluidos tradicionalmente lidam com um vasto volume de dados. O crescimento no volume de dados disponível nos dias de hoje permeia as mais diversas áreas científicas e também mercadológicas. Está ocorrendo uma grande confluência com a junção do vasto crescimento nos dados disponíveis, avanço nas tecnologias computacionais, redução dos custos computacionais, abundância de programas livres disponíveis e grande investimento na industria buscando soluções guiadas por dados. Com isso, o aprendizado de máquina começa a fazer rápidos avanços no contexto de mecânica dos fluidos (BRUNTON; NOACK; KOUMOUTSAKOS, 2020).

Técnicas de aprendizado de máquina podem ser utilizadas como uma forma ágil de solucionar problemas de mecânica dos fluidos, como o processamento de dados experimentais, otimização de formas, análise de turbulência (BRUNTON; NOACK; KOUMOUTSAKOS, 2020), escoamento de rios (IMRIE; DURUCAN; KORRE, 2000) dentre outras possíveis aplicações. Apesar do crescimento do uso de aprendizado de máquina na mecânica dos fluidos ser recente, Kolmogorov, já na década de 1940, vislumbrava o estudo da turbulência como um dos principais campos de aplicação do aprendizado de máquina através de métodos estatísticos (BRUNTON; NOACK; KOUMOUTSAKOS, 2020).

A inspiração humana para o desenvolvimento de máquinas muitas vezes parte da natureza, como as asas dos pássaros servem de inspiração para as asas dos aviões. No contexto de aprendizado de máquinas não poderia ser diferente. Tendo como inspiração a arquitetura cerebral e suas conexões, foram desenvolvidas as redes neurais artificiais (GERON, 2019).

A história das redes neurais data da primeira metade do século XX, com a introdução de teo-

rias de aprendizado biológico por McCulloch e Pitts (1943) e a implementação do primeiro modelo de aprendizado, como o Perceptron de Rosenblatt (1958), que possibilitou o treinamento de um algoritmo com único neurônio (GOODFELLOW, 2016). Redes neurais oferecem diversas funcionalidades e benefícios dada a forma que são organizadas. Dentre essas funcionalidades destacam-se (HAYKIN, 2008):

1. *Não linearidade*: um neurônio pode ser linear ou não linear, de tal forma que a rede como um todo pode possuir um caráter não linear. A não linearidade da rede é de extrema importância para a definição de problemas físicos complexos durante o treinamento do algoritmo.

2. *Mapeamento de entrada e saída*: a utilização de entradas e saídas bem conhecidas, durante o treinamento de uma rede neural é de extrema importância para a alteração dos pesos sinápticos buscando sempre minimizar as diferenças entre os parâmetros previstos e os valores reais.

3. *Adaptabilidade*: redes neurais tem a capacidade de adaptar seus pesos sinápticos seguindo as mudanças das condições de contorno. Em particular, uma rede treinada para um ambiente específico, pode ser facilmente retreinada para compreender pequenas alterações no ambiente.

No final da década de 1980, em um dos momentos de grande avanço dos modelos de aprendizado de máquina, o desenvolvimento de modelos de retropropagação de erro permitiu o avanço de redes neurais de múltiplas camadas (AGGARWAL, 2018; BRUNTON; NOACK; KOUMOUTSAKOS, 2020), o que elevou a possibilidade de usabilidade da ferramenta. Com isso, no início da década de 1990, surgiram diversas aplicações de redes neurais para mecânica dos fluidos, como a identificação e configuração de escoamentos multifásicos (BISHOP; JAMES, 1993).

Nos últimos anos têm ocorrido uma nova onda de avanços na área de aprendizado de máquina para mecânica dos fluidos. Isto ocorre devido aos grandes avanços na área de aprendizado profundo (*deep learning*) nos últimos anos (BRUNTON; NOACK; KOUMOUTSAKOS, 2020).

1.2 Publicações Importantes

Acessando a base de dados Scopus e utilizando como palavras chave *Machine Learning* e *Fluid Mechanics*, os artigos que possuem o maior número de citações se encontram na Tab. 1.1.

Tabela 1.1: Artigos mais citados

Trabalho	**Citações**
Brunton, Noack e Koumoutsakos (2020)	317
Imrie, Durucan e Korre (2000)	262
Gad-El-Hak (1996)	244
Raissi, Yazdani e Karniadakis (2020)	159
Gautier et al. (2015)	96
Zhu et al. (2019)	74

O trabalho de Brunton, Noack e Koumoutsakos (2020) faz uma revisão abrangente dos assuntos de maior interesse na sobreposição das áreas de aprendizado de máquina e mecânica dos fluidos, destacando os métodos mais interessantes e emergentes nas áreas. O trabalho de Imrie, Durucan e Korre (2000) tenta predizer o escoamento de rios do Reino Unido, extrapolando o banco de dados existente e identificando métodos que reduzam o erro nesta situação. Gad-El-Hak (1996) cita as redes neurais artificiais como uma alternativa para ser utilizada em mecanismos de controle, de forma que são uma ferramente conveniente, rápida e de fácil adaptação para algoritmos não lineares, conseguindo sem nenhuma informação prévia além dos dados, descrever os modelos matemáticos que governam so fenômenos. Visando identificar o comportamento de escoamentos através de imagens, Raissi, Yazdani e Karniadakis (2020) desenvolvem um algoritmo de aprendizado profundo capaz de traduzir imagens em números característicos de um escoamento, em especial visando aplicações médicas, onde a medição dos escoamentos não seria simples. Gautier et al. (2015) também buscam encontrar uma ferramenta de controle baseada em aprendizado de máquina, o objetivo de seu trabalho é reduzir a quantidade de vórtices gerados no clássico problema do degrau. Zhu et al. (2019) realizaram um estudo de escoamentos subsônicos em aerofólios, buscando um modelo para escoamentos turbulentos.

A diversidade de artigos entre os mais citados da área ilustra a abrangência de aplicações para este tipo de tecnologia e o alto nível de adaptabilidade para o algoritmo.

1.3 Aplicações

Além das aplicações citadas na seção 1.2, destacam-se algumas aplicações industriais como, o controle de temperatura em ambientes de missão crítica, como *data centers* (SONG; MURRAY; SAMMAKIA, 2014), predição da chegada prematura de fluidos injetados em reservatórios de petróleo (BAI; TAHMASEBI, 2020). Também podem ser realizadas aplicações médicas, visando por exemplo prever o crescimento de aneurismas (JIANG et al., 2020). Também, destacam-se aplicações científicas, como o estudo da condutividade térmica e viscosidade dinâmica de nanofluidos magnéticos (ESFE et al., 2015).

1.4 Objetivos

Este trabalho tem como objetivo treinar modelos de redes neurais que consigam predizer o comportamento de um fluido confinado e a transferência de calor em uma placa bidimensional. A rede neural utilizada possuirá como arquitetura o modelo do perceptron de multiplas camadas.

1.5 Estrutura do Trabalho

No capítulo 2 é feita uma breve introdução dos conceitos utilizados para a realização do trabalho. Uma breve introdução ao método numérico utilizado para gerar o banco de dados é apresentado

no 3. Em seguida, o capítulo 4 descreve a metodologia empregada no desenvolvimento do trabalho. Resultados são discutidos no capítulo 5, seguido das conclusões no capítulo 6. Os anexos contém os notebooks utilizados para o treinamento das redes neurais.

Capítulo 2

Fundamentação Teórica

2.1 Equações Governantes dos Fenômenos Físicos

O principal objetivo neste trabalho é estudar o comportamento de um fluido newtoniano, incompressível em uma cavidade, utilizando um algoritmo formado por redes neurais. Antes, porém, é apresentada a solução de um problema de condução de calor bidimensional devido à sua simplicidade. Desta forma, este problema será estudado como etapa inicial antes do avanço para o problema fluido.

2.1.1 Equação do Calor Bidimensional

A equação adimensional da difusão de calor transiente em uma placa bidimensional quadrada pode ser descrita como (ÇENGEL; GHAJAR, 2009)

$$\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}, \tag{2.1}$$

em que T representa a temperatura, t o tempo e x e y representam as direções do sistema de coordenadas. Para a adimensionalização, foi utilizado o comprimento da placa e α, que é a difusividade térmica do meio, como grandezas características.

Utilizando o método das diferenças finitas, a equação (2.1) pode ser discretizada como (ANDERSON, 1995):

$$\frac{T_{i,j}^{k+1} - T_{i,j}^{k}}{\Delta t} = \frac{T_{i+1,j}^{k} - 2T_{i,j}^{k} + T_{i-1,j}^{k}}{\Delta x^2} + \frac{T_{i,j+1}^{k} - 2T_{i,j}^{k} + T_{i,j-1}^{k}}{\Delta y^2}, \tag{2.2}$$

em que Δx é o tamanho do passo horizontal, Δy o tamanho do passo vertical, k representa o passo temporal, i o passo espacial ao longo do eixo x e j o passo espacial ao longo do eixo y.

2.1.2 Mecânica dos Fluidos

O movimento de um fluido newtoniano e incompressível é descrito pelo seguinte conjunto de equações (WHITE, 2019):

$$\nabla \cdot \mathbf{u} = 0 \tag{2.3}$$

e

$$\frac{\partial \mathbf{u}}{\partial t} = -\mathbf{u} \cdot \nabla \mathbf{u} - \nabla p + \frac{1}{Re} \nabla^2 \mathbf{u}. \tag{2.4}$$

A equação (2.3) é a equação da continuidade e a equação (2.4) é a equação de Navier-Stokes em sua forma adimensional. Nestas equações, ∇ é o operador nabla, $\mathbf{u}$ representa o vetor velocidade, t representa o tempo, p o campo de pressão e

$$Re = \frac{\rho U D}{\mu} \tag{2.5}$$

é o número de Reynolds, onde ρ é a massa específica do fluido, U representa a velocidade característica do escoamento, D o comprimento característico do escoamento e μ a viscosidade dinâmica.

2.2 Problemas Propostos

2.2.1 Condução de Calor Bidimensional

Um problema clássico envolve a solução da condução de calor em um domínio bidimensional. Neste trabalho, utilizando o método das diferenças finitas, o problema ilustrado na Figura 2.1 é resolvido.

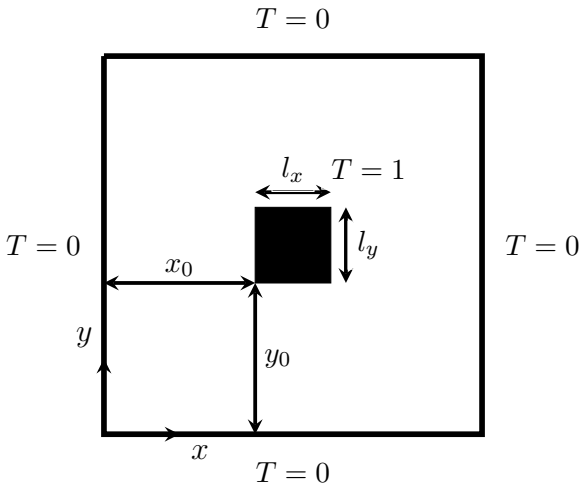

Figura 2.1: Ilustração do problema de transferência de calor em placa bidimensional. As paredes laterais possuem temperatura fixa $T = 0$ e um quadrilátero com temperatura fixa $T = 1$.

Neste problema, a posição do quadrilátero com temperatura fixa $T = 1$ é definida pelos valores

de x_0, l_x, y_0, l_y. São estudados diversos casos variando estas dimensões e o tempo, totalizando um banco de dado com mais de 35000 casos para que seja feito o treinamento da rede neural. Este banco fora gerado utilizando os métodos numéricos apresentados neste capitulo.

2.2.2 Cavidade Quadrada com Parede Móvel

Desde o início da década de 1960 (KAWAGUTI, 1961; BURGGRAF, 1966), o estudo do clássico problema da cavidade quadrada com parede móvel tem sido uma fonte de comparação para a solução numérica da equação de Navier-Stokes bidimensional (GHIA; GHIA; SHIN, 1982; ERTURK; CORKE; GöKÇöL, 2005; MARCHI; SUERO; ARAKI, 2009).

Trata-se de um problema teórico, no qual é proposto um escoamento forçado pela movimentação da face superior de um quadrado. Devido à condição de não deslizamento, o fluido adjacente à face deslizante inicia o movimento no interior da cavidade que, por causa de sua geometria, induz a formação de um vórtice central em seu interior (BURGGRAF, 1966).

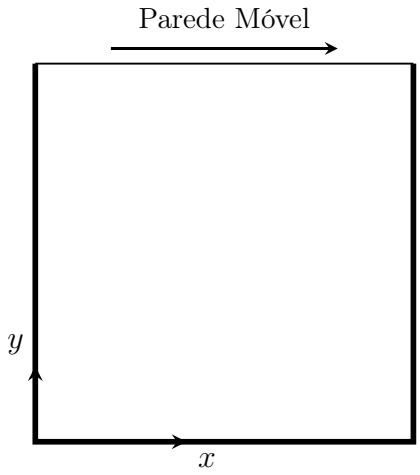

Figura 2.2: Ilustração do problema da cavidade com tampa deslizante. As paredes laterais e inferior são fixas enquanto a parede superior se move com velocidade definida.

Além disso, o problema pode modelar de forma simplificada alguns fenômenos reais, como o compartimento do trem de pouso de aviões (FAURE et al., 2006) e o interior de aneurismas intracranianos (HIRABAYASHI et al., 2003).

Neste problema, é feita uma análise do comportamento do fluido para diferentes números de Reynolds totalizando mais de 30000 casos onde foi feita a variação do número de Reynolds de 50 em 50, tendo início em $Re = 50$ e fim em $Re = 1000$, sendo cada um deles analisados para diferentes passos de tempo, variando de $t = 0$ até $t = 15$ com uso do método da projeção (BROWN; CORTEZ; MINION, 2001; CHORIN, 1968; BELL; COLELLA; GLAZ, 1989).

2.3 Metodologia Numérica

Para realizar o treinamento do algoritmo proposto necessita-se de um banco de dados com valores válidos e confiáveis dos problemas propostos. Desta forma, foram realizadas simulações numéricas e o armazenamento de seus resultados para a formação deste banco de dados.

2.3.1 Geração e Convergência de Malha

Para realizar a solução numérica de um problema, é necessário realizar uma discretização de sua malha. A discretização é o processo de definir pontos no espaço que terão as propriedades calculadas com o método numérico utilizado, em contraste ao método analítico que determina as propriedade de maneira contínua. Desta forma, soluções numéricas necessitam de uma malha de pontos (ANDERSON, 1995). Por exemplo, a Fig. 2.3 ilustra a discretização de um domínio quadrangular. O domínio foi discretizado em 10 pontos na direção x e 10 pontos na direção y, formando uma malha com 100 pontos igualmente distribuídos.

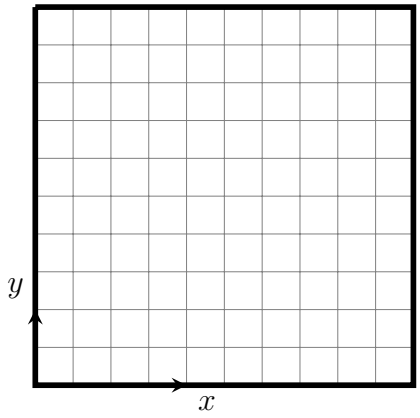

Figura 2.3: Ilustração de uma malha quadrada. As intersecções representam os pontos que são efetivamente calculados.

Durante o processo de discretização da malha, cada um dos pontos recebe um par de índices (i, j) que determinam a posição deste ponto na malha. O índice i representa a posição dos pontos ao longo do eixo das abscissas enquanto o índice j representa a posição dos pontos ao longo do eixo das ordenadas. Por exemplo, dado um ponto qualquer P com índices (i, j), o ponto imediatamente a a direita de P será o ponto $(i+1, j)$, o ponto imediatamente acima será o $(i, j+1)$ (ANDERSON, 1995).

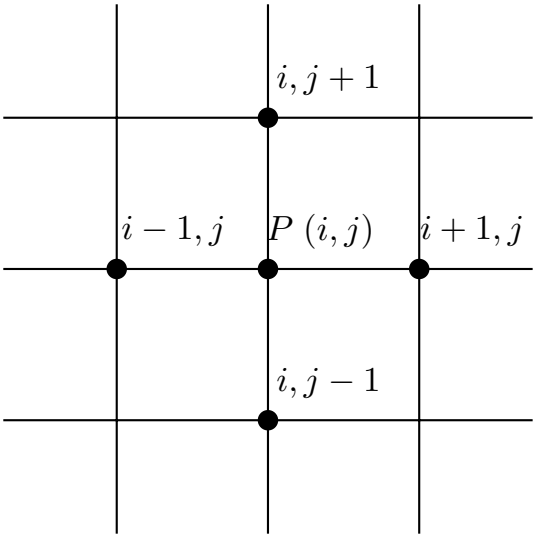

Figura 2.4: Malha discretizada com indicação dos índices

Para verificar se a malha utilizada na solução numérica possui um número de elementos adequado, deve ser realizado um estudo de convergência de malha. Este estudo pode ser realizado com a avaliação das alterações sofridas por uma propriedade em um determinado ponto, ou pela média desta propriedade em determinada região, com a variação do número de pontos existentes na malha. Para identificar a quantidade de elementos que devem ser utilizados me uma malha, deve ser feito um estudo de convergência de malha, monitorando a variação de uma propriedade com a variação da quantidade de elementos. Recomenda-se a utilização da menor malha que resulte em valores estáveis,ou seja que não vão variar significativamente com o aumento do número de elementos, evitando assim um custo computacional desnecessário.

2.3.2 Diferenças Finitas

Soluções por diferenças finitas são amplamente utilizadas na mecânica dos fluidos computacional. O método apresenta as derivadas como expansão da série de Taylor (ANDERSON, 1995). Por exemplo, seja $u_{i,j}$ a componente na direção x da velocidade no ponto P, então a componente no ponto $(i+1, j)$ será descrita por:

$$u_{i+1,j} = u_{i,j} + \left(\frac{\partial u}{\partial x}\right)_{i,j} \Delta x + \left(\frac{\partial^2 u}{\partial x^2}\right)_{i,j} \frac{(\Delta x)^2}{2} + \left(\frac{\partial^3 u}{\partial x^3}\right)_{i,j} \frac{(\Delta x)^3}{6} + ... \tag{2.6}$$

A equação (2.6) é exata caso o número de termos seja infinito e a série convirja ou para Δx tendendo a zero (ANDERSON, 1995).

Resolvendo a equação (2.6) para $\frac{\partial u}{\partial x}$:

$$\left(\frac{\partial u}{\partial x}\right)_{i,j} = \frac{u_{i+1,j} - u_{i,j}}{\Delta x} - \left(\frac{\partial^2 u}{\partial x^2}\right)_{i,j} \frac{\Delta x}{2} - \left(\frac{\partial^3 u}{\partial x^3}\right)_{i,j} \frac{(\Delta x)^2}{6} + ... \tag{2.7}$$

De forma que, o primeiro termo do lado direito da equação (2.7) ($\frac{u_{i+1,j}-u_{i,j}}{\Delta x}$) é a representação por diferenças finitas da derivada parcial de u em relação a x e os outros termos são os erros por truncamento (ANDERSON, 1995). Ou seja, uma aproximação do lado esquerdo da equação (2.7) é dada por:

$$\left(\frac{\partial u}{\partial x}\right)_{i,j} \approx \frac{u_{i+1,j} - u_{i,j}}{\Delta x} \tag{2.8}$$

O erro por truncamento, na equação (2.7) indica o que está sendo negligenciado durante a aproximação por diferenças finitas. Observando o termo de menor ordem para Δx, nota-se que se trata de uma aproximação de primeira ordem (ANDERSON, 1995), ou seja:

$$\left(\frac{\partial u}{\partial x}\right)_{i,j} = \frac{u_{i+1,j} - u_{i,j}}{\Delta x} + \mathcal{O}(\Delta x). \tag{2.9}$$

Para o caso da derivada de segunda ordem, temos a seguinte aproximação (ANDERSON, 1995):

$$\left(\frac{\partial^2 u}{\partial x^2}\right)_{i,j} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{(\Delta x)^2} + \mathcal{O}(\Delta x)^2. \tag{2.10}$$

2.3.3 Método da Projeção

Existem diversos métodos para solucionar problemas físicos utilizando a base de diferenças finitas. Neste trabalho, o banco de dados utilizado para o treinamento das redes neurais para solucionar o problema da cavidade com tampa deslizante, foi obtido através do método da projeção de primeira ordem. Este método foi desenvolvido por Chorin (1968) e Temam (1969) e será mais detalhado no Capítulo 3.

2.4 Aprendizado de Máquina

Nesta seção serão abordados apenas conceitos básicos a respeito de aprendizado de máquina, mais detalhes sobre o método utilizado são explorados no capítulo 4.

Os métodos de aprendizado de máquina (*machine learning*, em inglês) são métodos computacionais que, através de um conjunto de dados, identificam quais algoritmos governam o fenômeno analisado. A programação tradicional, por outro lado, demanda um especialista no fenômeno analisado, de forma que seja possível realizar uma modelagem matemática específica para as condições do problema.

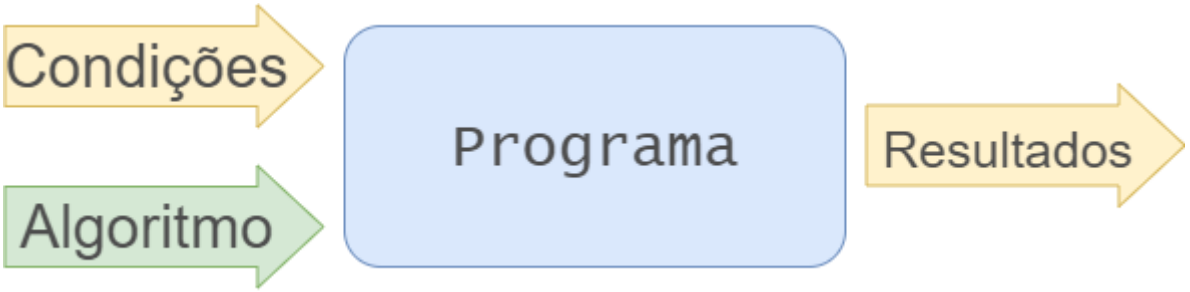

Figura 2.5: Diagrama ilustrativo de como são elaborados programas utilizando métodos tradicionais

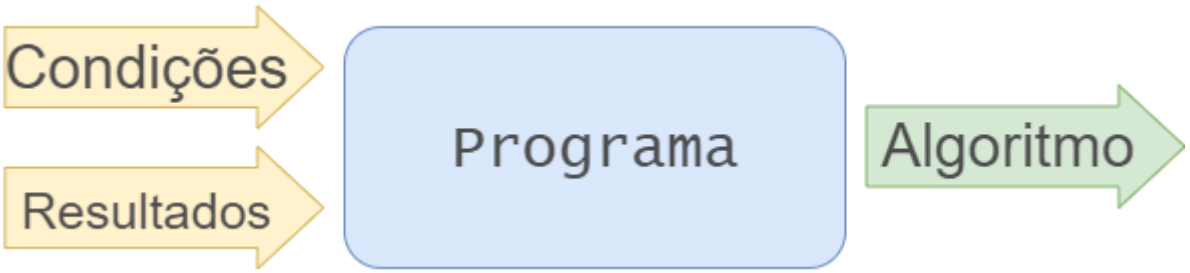

Figura 2.6: Diagrama ilustrativo de como são elaborados programas utilizando aprendizado de máquina

Realizando um comparativo entre os dois métodos, através dos diagramas presentes nas Figuras 2.5 e 2.6 podemos identificar que ambos os métodos trabalham com três conjuntos:

- Condições: As condições que definem o problema analisado;
- Algoritmo: Modelo matemático que descreve o fenômeno;
- Resultados: Conjunto de dados que resulta da aplicação das condições do problema no modelo matemático estudado.

Desta maneira, diversos fenômenos que possuem um amplo conjunto de dados já armazenados podem ser analisados através do uso de aprendizado de máquina, como por exemplo identificar o valor de imóveis em determinada região (PARK; BAE, 2015) ou ainda identificar as condições atmosféricas nos arredores de aeroportos (ALMEIDA; FRANÇA; VELHO, 2020).

2.4.1 Regressão e Classificação

É possível realizar uma divisão dos diferentes tipos de aprendizado de máquina quanto ao resultado final que é entregue pelo algoritmo. Caso o problema tenha como resposta final a classificação dos dados, trata-se de um algoritmo de Classificação. Por exemplo, analisando a massa específica de um fluido, comprimento característico do escoamento, velocidade e viscosidade dinâmica do

fluido, é possível treinar um algoritmo que classifique o escoamento como laminar, turbulento ou ainda como de transição (ÇENGEL, 2015; GERON, 2019).

Por outro lado, problemas que possuem um valor numérico não binário como resposta, são chamados de algoritmos de Regressão. Por exemplo, com os mesmos parâmetros indicados no exemplo de classificação, é possível treinar um algoritmo que indique o valor numérico do número de Reynolds do escoamento (GERON, 2019).

Ou seja, se a saída do algoritmo é um conjunto binário, trata-se de um problema de Classificação. Se a saída do algoritmo é um conjunto não binário, trata-se de um problema de Regressão. Neste trabalho, o problema a ser resolvido é um problema de Regressão.

2.4.2 Divisão do Conjunto de Dados em Teste e Treino

Como indicado na figura 2.6, um conjunto de dados com os resultados esperados é necessário para realizar o treinamento de algoritmos de aprendizado de máquina. Este treinamento necessita de um conjunto de dados de validação para que seja verificado o comportamento do algoritmo com um conjunto de dados até então desconhecido. Desta forma, o processo de aprendizagem de máquina demanda dois conjuntos de dados distintos (GERON, 2019):

- Conjunto de dados de treino: conjunto utilizado pelo programa para aprender qual é o algoritmo relacionado ao problema.
- Conjunto de dados de teste: conjunto utilizado para identificar se o programa está enviesado e consegue prever apenas condições já conhecidas. Este conjunto é o responsável pela validação do modelo proposto.

Para que a análise seja reproduzível, neste trabalho será utilizada a função train_test_split da biblioteca scikit-learn compatível com a linguagem de programação *Python*. Nesta função, são indicados os conjuntos de dados de entrada (X), os valores dos resultados reais (y), o tamanho do conjunto de teste em relação ao conjunto total (test_size) e a maneira que os dados serão divididos (random_state). A saída desta função será os conjuntos de entrada para treino, teste e os resultados reais de treino e teste (GERON, 2019).

```
1 from sklearn.model_selection import train_test_split
2
3 X_treino, X_teste, y_treino, y_teste = train_test_split(X, y, test_size=0.33,
      random_state=101)
```

Listing 2.1: Exemplo de uso da função train_test_split

Com isso, fica clara a possibilidade de resolver problemas de mecânica dos fluidos utilizando ferramentas de aprendizado de máquina, uma vez que as ferramentas para geração de um banco de dados confiável estão bem definidas para problemas mais simples e, para a resolver problemas mais complexos, ainda existe certa dificuldade em se definir os algoritmos necessários, ou ainda demandam de elevado custo computacional. Para a solução dos problemas propostos por meio

do aprendizado de máquinas, será utilizada a biblioteca *Tensor Flow* disponível para a linguagem *Python*.

Capítulo 3

Introdução ao Método da Projeção

3.1 Método da Projeção

Desenvolvido por Chorin (1968) e Temam (1969), o método da projeção visa resolver a Equação de Navier-Stokes. O método possui três etapas básicas, sendo:

- Etapa 1: A restrição de incompressibilidade é ignorada, e utiliza-se uma velocidade intermediária $\boldsymbol{u}^*$ calculada através da equação de Navier-Stokes, desconsiderando o termo da pressão. Como a restrição da incompressibilidade está sendo ignorada, $\boldsymbol{u}^*$ pode não ser solenoidal;
- Etapa 2: Uma nova pressão é calculada seguindo as condições de contorno de Neumann, impondo a condição da velocidade $\boldsymbol{u}$ ser solenoidal;
- Etapa 3: Por fim, deve ser calculada a nova velocidade $\boldsymbol{u}$.

As etapas podem ser descritas matematicamente da seguinte forma:

$$\text{Etapa 1} \begin{cases} \frac{\boldsymbol{u^*}-\boldsymbol{u^k}}{\Delta t} + \boldsymbol{u^k} \cdot \nabla \boldsymbol{u^k} = \frac{1}{Re} \nabla^2 \boldsymbol{u^k} \\ \boldsymbol{u^*} = \boldsymbol{u}_b \quad \text{em} \ \ \partial\Omega \end{cases} , \tag{3.1}$$

em seguida,

$$\text{Etapa 2} \begin{cases} \nabla^2 p^{k+1} = \frac{1}{\Delta t} \nabla \cdot \boldsymbol{u^*} \\ \nabla p^{k+1} \cdot \hat{\boldsymbol{n}} = 0 \ \ \textbf{em} \ \ \partial\Omega \end{cases} , \tag{3.2}$$

e por fim,

$$\text{Etapa 3} \begin{cases} \boldsymbol{u}^{k+1} = \boldsymbol{u^*} - \Delta t \nabla p^{k+1} \end{cases} , \tag{3.3}$$

onde k representa o passo temporal, $\hat{\boldsymbol{n}}$ é um vetor normal, $\partial\Omega$ é a fronteira do domínio Ω definido por $(0,1) \times (0,1)$.

3.1.1 Malha Defasada

As Fig.2.4 ilustra uma malha tradicional onde a propriedade é avaliada nos nós. Para a implantação do método da projeção, utilizar uma malha tradicional gera uma oscilação da pressão, desta forma, utiliza-se uma malha defasada onde a pressão, a componente horizontal da velocidade e a componente vertical da velocidade são calculados em pontos distintos na malha, conforme ilustrado na Fig. 3.1 (VERSTEEG, 2007):

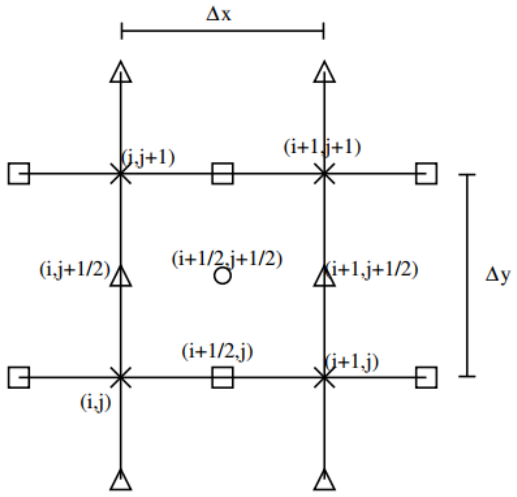

Figura 3.1: Exemplo de malha defasada. Os círculos representam os pontos onde a pressão é calculada, os triângulos a componente u da velocidade e os quadrados a compontente v

3.1.2 Critérios de Estabilidade e Convergência

Os parâmetros selecionados durante uma simulação devem respeitar alguns critérios para que os resultados obtidos através do algoritmo tenha significado físico e sejam condizente com a realidade. No caso do método da projeção de primeira ordem, deve-se respeitar as seguintes condições para o passo de tempo Δt e incremento espacial Δx (HINCH, 2020):

$$\Delta t < \Delta x \tag{3.4}$$

sendo

$$\Delta x < \frac{1}{\sqrt{Re}} \tag{3.5}$$

e ainda

$$\Delta t < \frac{1}{4} Re \Delta x^2. \tag{3.6}$$

Capítulo 4

Redes Neurais Artificiais

4.1 Perceptron

O Perceptron, também conhecido como Perceptron de Rosenblatt (1958), foi a primeira forma e também a mais simples de um algoritmo de redes neurais. Sua função, na época de sua concepção, era de classificar dois diferentes grupos. Sua proposta consistia em um modelo que realiza a combinação linear dos parâmetros de entrada e, em seguida, classificava o resultado em um determinado grupo, utilizando o resultado desta combinação linear como critério de separação.

Sua concepção consiste em um único neurônio com pesos sinápticos w ajustáveis. Os parâmetros de entrada são linearmente somados e adicionados de um viés b.

O funcionamento do Perceptron pode ser formulado como

$$z = \sum_{i=1}^{m} w_i x_i + b, \tag{4.1}$$

onde m indica o número de neurônios existentes na camada, x_i são os valores de entrada, w_i são os pesos sinápticos, b é o viés e z o resultado da combinação linear que é o valor de entrada de uma função de ativação (SKANSI, 2018; AGGARWAL, 2018).

Seja, por definição, $x_0 = 1$, de tal forma que é possível reescrever a Eq. 4.1:

$$z = \sum_{i=0}^{m} w_i x_i \tag{4.2}$$

onde o primeiro termo do somatório, $x_0 w_0$ é o próprio viés b.

A definição dos pesos sinápticos ocorre através de um processo iterativo chamado de treinamento. O primeiro valor utilizado pela rede neural é definido de maneira arbitrária e, durante o treinamento, é realizada uma análise com o auxilio de um otimizador (mais detalhes na seção 4.3.2) para que se encontre o melhor valor para os pesos sinápticos.

Após a combinação linear dos pesos sinápticos e os valores de entrada, o valor obtido é utilizado

como parâmetro de entrada de uma função de ativação φ. No caso do Perceptron, a função de ativação é um classificador binário conhecido como função sinal (SKANSI, 2018; AGGARWAL, 2018):

$$y = \varphi(z) = sgn(z) = \begin{cases} +1 & , z \geq 0 \\ 0 & , z < 0 \end{cases} \tag{4.3}$$

Existem outros tipos de função de ativação, alguns exemplos serão detalhados em outras seções deste capítulo.

Um Perceptron com duas entradas pode ser ilustrado pelo diagrama presente na Fig. 4.1.

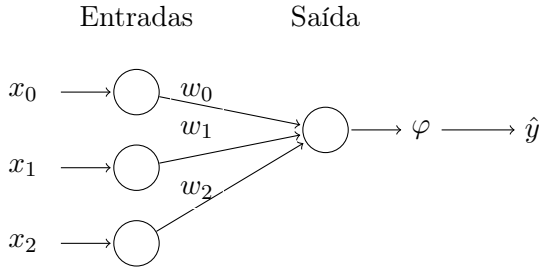

Figura 4.1: Ilustração do funcionamento do Perceptron

Cada uma das circunferências representa um neurônio utilizado pela rede neural. No neurônio presente na camada de saída, o valor encontrado é utilizado como entrada de uma função de ativação que indica o valor predito pela rede neural, neste caso, $\hat{y} \in (0, 1)$. Contudo, não há nenhuma obrigatoriedade quanto a existência de uma função de ativação nos neurônios.

4.2 Perceptron de Múltiplas Camadas

Dada a formulação e funcionamento do Perceptron, é possível expandir o conceito para formar uma associação de perceptrons, de modo que passa a ser possível obter mais de um resultado como saída. A Fig. 4.2 ilustra uma rede neural com 4 neurônios de entrada e 6 neurônios de saída.

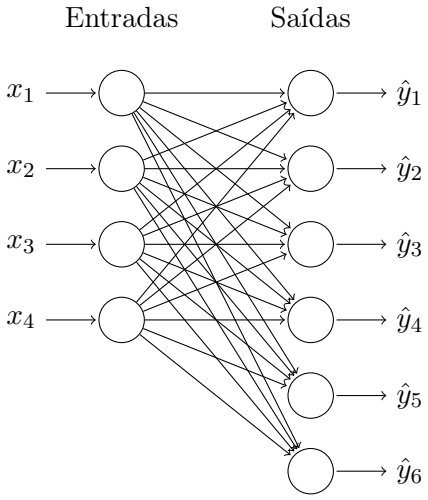

Figura 4.2: Associação de Perceptrons

Além da associação de Perceptrons aumentando a quantidade de neurônios em cada camada, é possível realizar a associação de camadas, formando os Perceptrons de Múltiplas Camadas, sigla em língua inglesa MLP. Alguns problemas podem possuir entradas que não estão linearmente relacionadas com os valores de saída, sendo necessário acrescentar camadas ocultas para a obtenção de predições mais próximas dos valores reais.

A Fig. 4.3 ilustra um MLP totalmente conectado com uma camada oculta. Ser totalmente conectado significa que dado um neurônio de uma camada, ele estará conectado com todos os neurônios da camada anterior e também da próxima camada.

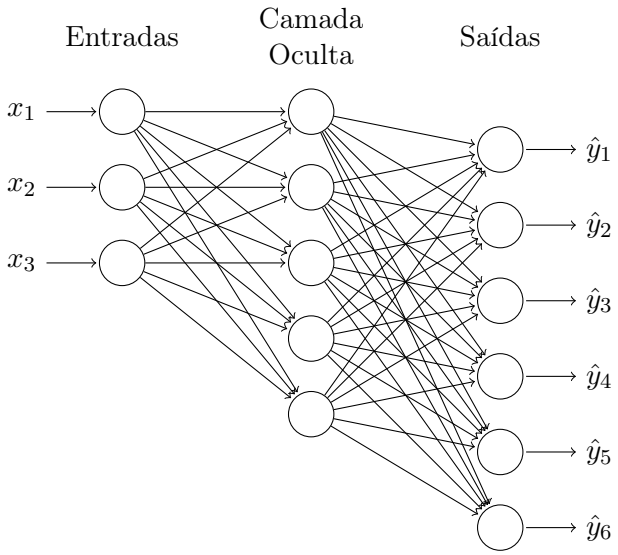

Figura 4.3: Perceptron de Múltiplas Camadas

Caso a arquitetura da rede tenha duas ou mais camadas ocultas, utiliza-se o termo Aprendizado Profundo, em inglês *Deep Learning* para este tipo de arquitetura de rede.

4.3 Mecanismo de Aprendizagem

É preciso compreender como cada camada de uma rede neural se comporta para que se estruture a melhor rede possível para a solução dos problemas propostos. Desta forma, deve-se entender os mecanismos existentes durante o processo, destacam-se:

- Função Perda;
- Retropropagação de Erro;
- Otimizadores;
- Sobreajuste;
- Parada Antecipada;
- Função de Ativação.

4.3.1 Função Perda e Retropropagação de Erro

Em uma rede neural, o sinal da função avança da esquerda para a direita, isto é, os dados de entrada são tratados ao avançar por cada uma das camadas, desde a camada de entrada até a camada de saída. O treinamento de uma rede neural começa com valores arbitrários para os pesos sinápticos, sendo feita um passo de treinamento (Época) com estes valores. Ao final da primeira época, é realizada uma análise através de uma função de perda (GOODFELLOW, 2016).

As funções de perda visam identificar a discrepância entre o valor previsto pelo algoritmo $\hat{y}$ e o valor real y. Para problemas de regressão, costuma-se utilizar a função de Erro Quadrático Médio (EQM), definida como

$$EQM = \frac{1}{t}\sum_{i=1}^{t}(\hat{y}_i - y_i)^2, \tag{4.4}$$

onde t é o total de amostras analisadas (GOODFELLOW, 2016; AGGARWAL, 2018; GERON, 2019).

Após o cálculo da função perda, se inicia a propagação do sinal de erro, que ocorre da direita para a esquerda.A Fig. 4.4 ilustra o funcionamento do sinal da função e do sinal de erro.

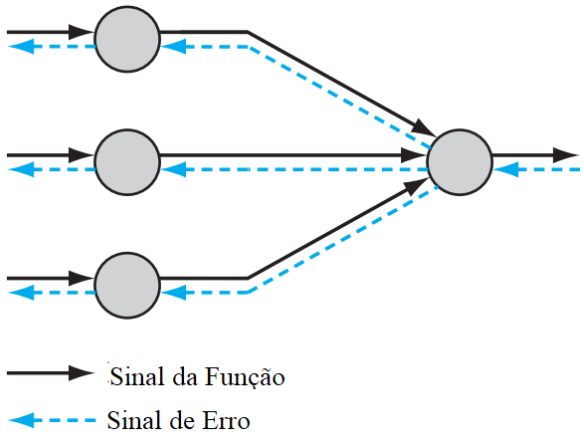

Figura 4.4: Ilustração do avanço de sinal e retropropagação do erro para ajuste dos pesos sinápticos Adaptado de: Haykin (2008)

4.3.2 Otimizador

Quase todos algoritmos de redes neurais envolvem algum tipo de otimização (GOODFELLOW, 2016). Otimização é a tarefa de minimizar ou maximizar uma função. No contexto de redes neurais, o foco é minimizar a função perda, descrita na seção 4.3.1.

Com o uso de cálculo diferencial, é possível identificar para qual direção deve-se mover o valor de entrada da função buscando minimizar os valores obtidos, até que se encontre o valor mínimo da função. A Fig. 4.5 ilustra o método do gradiente descendente para uma função de uma única variável. Note que o ponto escolhido gradativamente se move para a direita, dado que a derivada no ponto é negativa, caso o ponto estivesse localizado a direita da região de mínimo, a derivada no ponto seria positiva e, por isso, o ponto deveria ser deslocado para a esquerda (GERON, 2019).

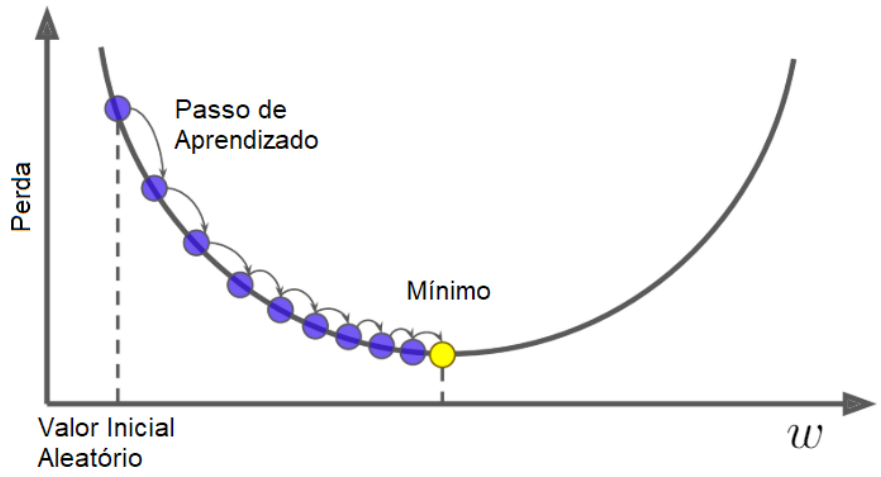

Figura 4.5: Ilustração do método do gradiente descente para uma função de uma variável. Adaptado de: Geron (2019)

No caso de funções de múltiplas variáveis, é necessário utilizar o conceito de derivada parcial,

dado que a derivada parcial indicará a direção que se deve mover o ponto em relação a cada um dos parâmetros. Desta forma, o gradiente da função será utilizado para atualização do ponto que será utilizado no próximo passo de aprendizado.

Existem diversos tipos de otimizadores comumente utilizados para o treinamento de redes neurais artificiais. Neste trabalho será abordado apenas o método do Gradiente Descendente.

O método do gradiente descendente consiste na aplicação dos conceitos ilustrado pela Fig. 4.5 para uma função de múltiplas variáveis, de tal forma que são obtidos novos pesos sinápticos que otimizem a função, o método pode ser descrito por (GERON, 2019; GOODFELLOW, 2016; HAYKIN, 2008):

$$\boldsymbol{w}_{\text{novo}} = \boldsymbol{w} - \eta \nabla_w f(\boldsymbol{w}), \tag{4.5}$$

$\boldsymbol{w}$ representa o vetor formado pelos pesos sinápticos, η é a taxa de aprendizagem, ∇ é o operador nabla e f é a função de perda utilizada.

A taxa de aprendizagem determina a velocidade de convergência do algoritmo. O valor escolhido deve ser grande o suficiente para que o algoritmo tenha um tempo de convergência aceitável e pequeno o suficiente para que o valor consiga convergir para o mínimo da função. As Figs. 4.6 e 4.7 ilustram a influência da taxa de aprendizagem na atualização dos pesos sinápticos.

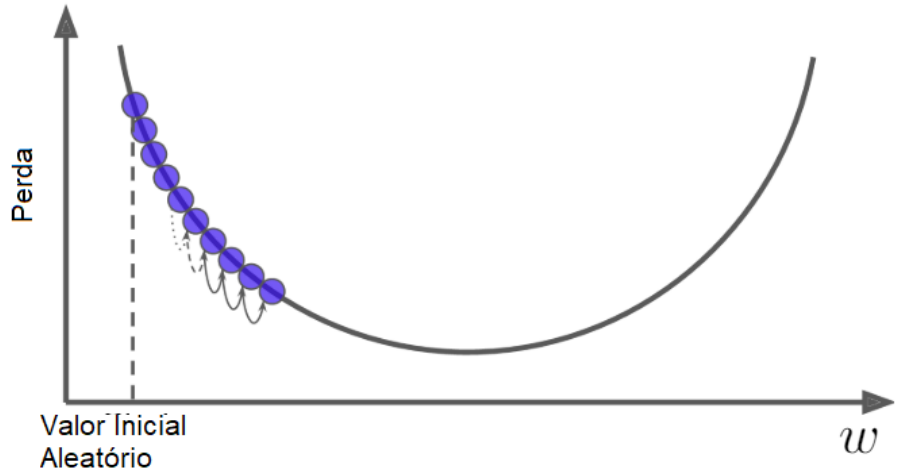

Figura 4.6: Efeito de uma taxa de aprendizagem pequena para o funcionamento do algoritmo do gradiente descendente. Adaptado de Geron (2019)

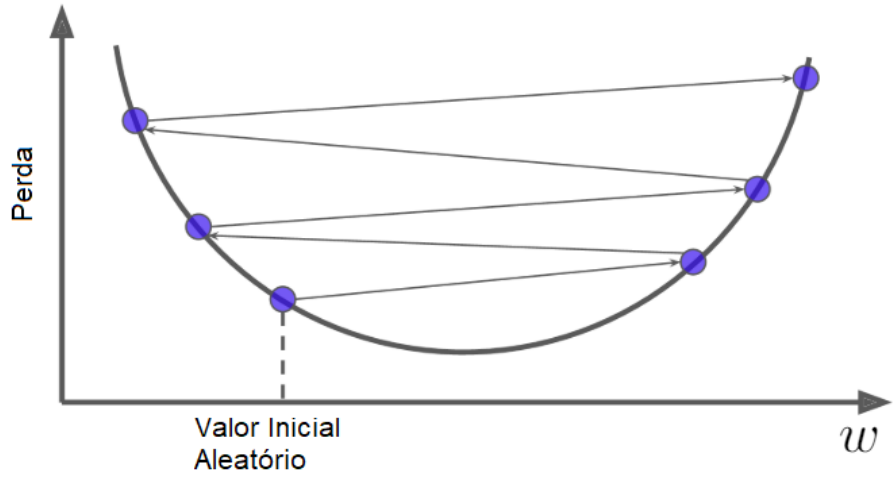

Figura 4.7: Efeito de uma taxa de aprendizagem grande para o funcionamento do algoritmo do gradiente descendente. Adaptado de Geron (2019)

Para identificar durante o treinamento se o valor de taxa de aprendizagem escolhido está adequado, observa-se o comportamento da curva época por perda, onde época indica quantos ciclos de treinamento já foram realizados. A Fig. 4.8 ilustra o comportamento desta curva para diferentes valores de η. Note que, só há prejuízo significativo para valores elevados de eta, uma vez que valores pequenos impactam apenas no tempo de aprendizagem.

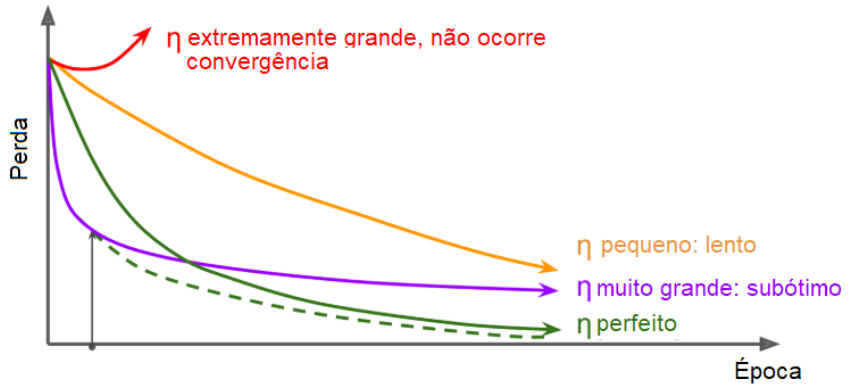

Figura 4.8: Exemplo do comportamento da curva Época x Perda para 4 diferentes valores de η. Adaptado de Geron (2019)

4.3.3 Sobreajuste e Parada Antecipada

Um problema comum presente durante o treinamento de redes neurais é a de se obter um algoritmo sobreajustado, ou seja, o algoritmo consegue obter boas predições para os dados do banco de teste, mas não apresenta bons resultados para novos dados. Para evitar este problema o número de ciclos de treinamento (épocas) deve ser limitado. O número de épocas deve ser grande o suficiente para que o ajuste do gradiente descendente seja eficiente, mas pequeno o suficiente

para evitar o sobreajuste (GERON, 2019; AGGARWAL, 2018; HAYKIN, 2008).

Observando o comportamento da curva época × perda para o conjunto de dados de treino e, acoplando o mesmo mecanismo de análise para o conjunto de dados de teste, percebe-se que após alguns ciclos de treinamento, a função perda do conjunto de dados de teste tende a aumentar. Isso ocorre no momento que o algoritmo começa a ficar sobreajustado, ou seja, passa a obter bons resultados apenas para o conjunto de treino. Desta forma, é usual realizar o monitoramento de ambas as curvas e, quando a função de perda referente ao conjunto de teste atinge seu mínimo, encerrar o processo de treinamento. Este comportamento, bem como o ponto de parada, são ilustrados na Fig. 4.9 (GERON, 2019; AGGARWAL, 2018; HAYKIN, 2008).

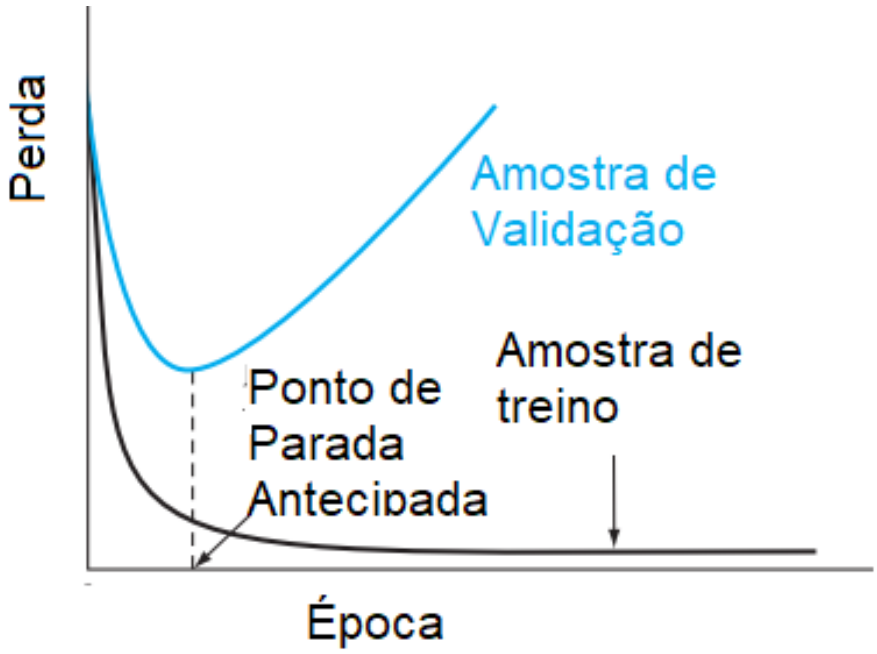

Figura 4.9: Comportamento da curva de perda para os dados de treino (preto) e teste (azul). Destaca-se o ponto de parada antecipada para evitar o sobreajuste do algoritmo. Adaptado de Haykin (2008)

Ressalta-se que, durante o processo de treinamento da rede neural, os únicos valores utilizados para a atualização dos pesos sinápticos são aqueles presentes no banco de dados de treino. O banco de dados de teste é monitorado apenas para indicar o ponto de parada do treinamento buscando evitar o sobreajuste (GERON, 2019; AGGARWAL, 2018; HAYKIN, 2008).

4.3.4 Função de Ativação

A escolha da função de ativação é parte crítica durante o desenho da arquitetura de uma rede neural (AGGARWAL, 2018). No caso do Perceptron, a escolha da função sinal é motivada pela necessidade de se classificar o resultado em dois conjuntos. No entanto, existem outros tipos de função que podem ser mais adequadas para cada caso, sendo necessária uma revisão de estudos anteriores do problema em questão.

Para problemas de classificação, destacam-se algumas funções de ativação, como a função sinal, descrita na Eq. (4.3), e também as funções sigmoide,

$$\varphi(z) = \frac{1}{1 + \exp(-z)} \tag{4.6}$$

e a função tangente hiperbólica,

$$\varphi(z) = \tanh(z) = \frac{\exp(2z) - 1}{\exp(2z) + 1}. \tag{4.7}$$

A Fig. 4.10 ilustra o comportamento dessas funções.

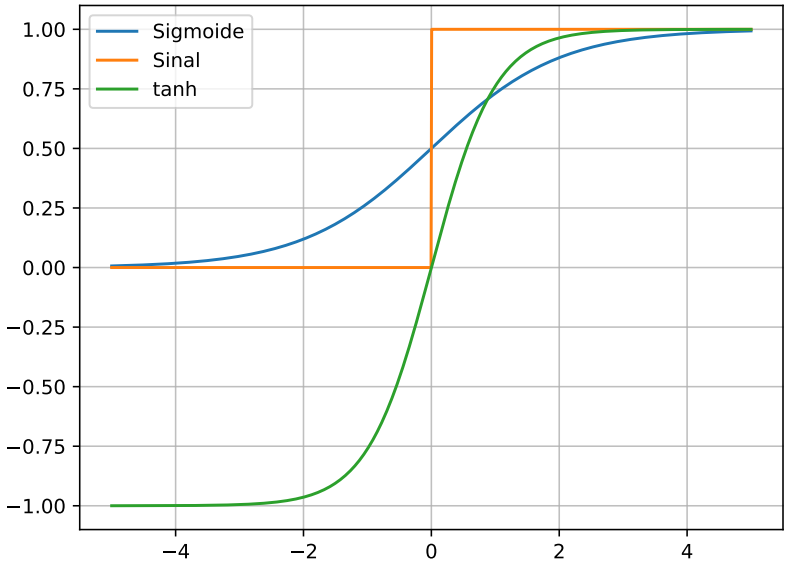

Figura 4.10: Ilustração das funções de ativação utilizadas para modelos de classificação

No caso de problemas de regressão, destaca-se a função *ReLU*, em inglês *Rectified Linear Unit*, descrita por

$$\varphi(z) = \max(z, 0), \tag{4.8}$$

e suas variantes como a *Leaky ReLU*

$$\varphi(z) = \begin{cases} z & , z \geq 0 \\ 0,01z & , z < 0 \end{cases}. \tag{4.9}$$

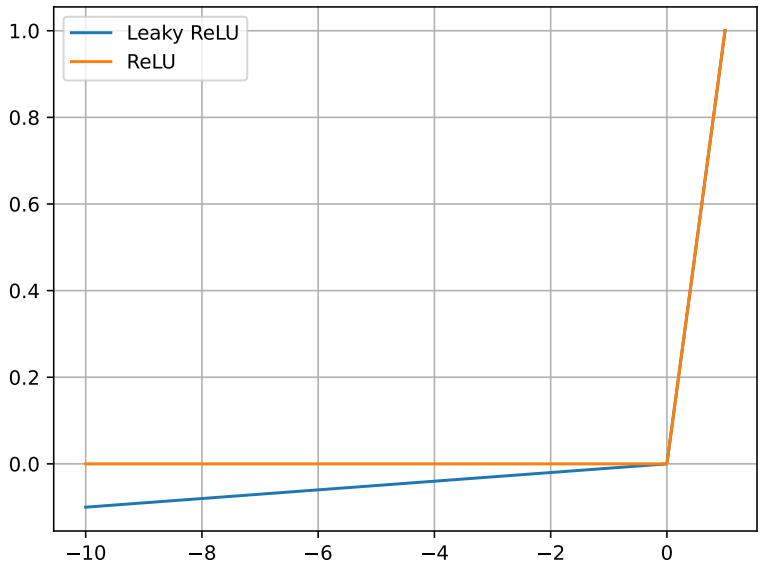

Figura 4.11: Ilustração das funções de ativação utilizadas para modelos de regressão

Note, porém, que, apesar das características das funções de ativação, não existem impeditivos de utilizar funções com resultados não binários nas camadas ocultas das redes neurais utilizadas como classificadores, basta que a função de ativação escolhida para a camada de saída seja um classificador binário. Analogamente, não há a proibição do uso de classificadores binários nas camadas ocultas de redes neurais utilizadas em problemas de regressão. Para estes problemas, recomenda-se que a última camada não possua função de ativação, sendo composta apenas pela combinação linear dos parâmetros de saída da camada anterior.

Capítulo 5

Resultados

5.1 Problema da Equação do Calor Bidimensional

Para a solução do problema da placa bidimensional aquecida, necessita-se de uma rede neural que solucione os valores de temperatura T em cada ponto da malha. Para possibilitar o treinamento da rede, foi gerado um banco de dados com mais de 700 possíveis posições para a região aquecida com $T = 1$, salvando ao longo dos passos de tempo, totalizando mais de 35000 casos distintos. Para gerar este banco, foi utilizada de uma máquina virtual hospedada no *Google Cloud Platform* com *Ubuntu* 20.04, 16 Gb de RAM e SSD de 60 Gb. O tempo necessário para realizar as simulações e armazenar os dados foi de 28 horas.

5.1.1 Parâmetros utilizados

Dois modelos básicos de rede foram utilizados, sendo a principal mudança o tipo de entrada utilizada. O primeiro modelo (Malha) utiliza uma malha indicando a condição de contorno de cada um dos pontos além do passo de tempo. O segundo modelo (Coordenadas) utiliza as coordenadas da região aquecida e o passo de tempo como valores de entrada.

A quantidade de neurônios na saída é definida pelo tamanho da malha utilizada no estudo. Para determinar a quantidade de pontos necessários em uma malha, é necessário realizar uma análise de convergência de malha. Neste caso, para a análise, foi realizado o monitoramento da temperatura média ao longo da malha para diferentes números de pontos, conforme ilustrado na Fig. 5.1.

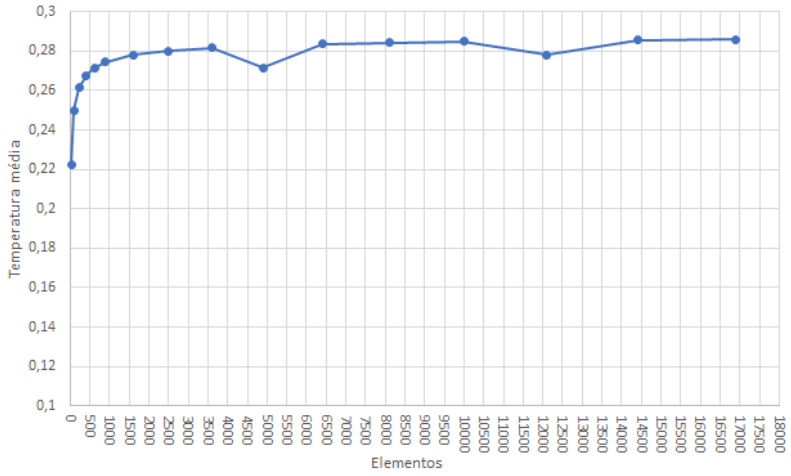

Figura 5.1: Análise de convergência de malha para problema da placa aquecida

Desta forma, uma malha com 1681 elementos foi selecionada para realizar as simulações presentes no banco de dados deste trabalho.

Tabela 5.1: Diferenciação da quantidade de neurônios nos principais tipos de modelo

Tipo de Entrada	**Número de Neurônios de Entrada**	**Número de Neurônios de Saída**
Coordenadas	5	1681
Malha	1682	1681

O número de neurônios das camadas ocultas é automaticamente definido de tal forma que as camadas sempre estejam linearmente espaçadas.

5.1.2 Função Perda

O monitoramento da função perda é de extrema importância para identificar se o treinamento está sendo eficiente. A figura 5.2 ilustra o comportamento da função para os modelos ao longo do treinamento. Observe que todas as taxas de aprendizado se encontram dentro dos valores aceitáveis, dado que em todos os casos ocorreu convergência.

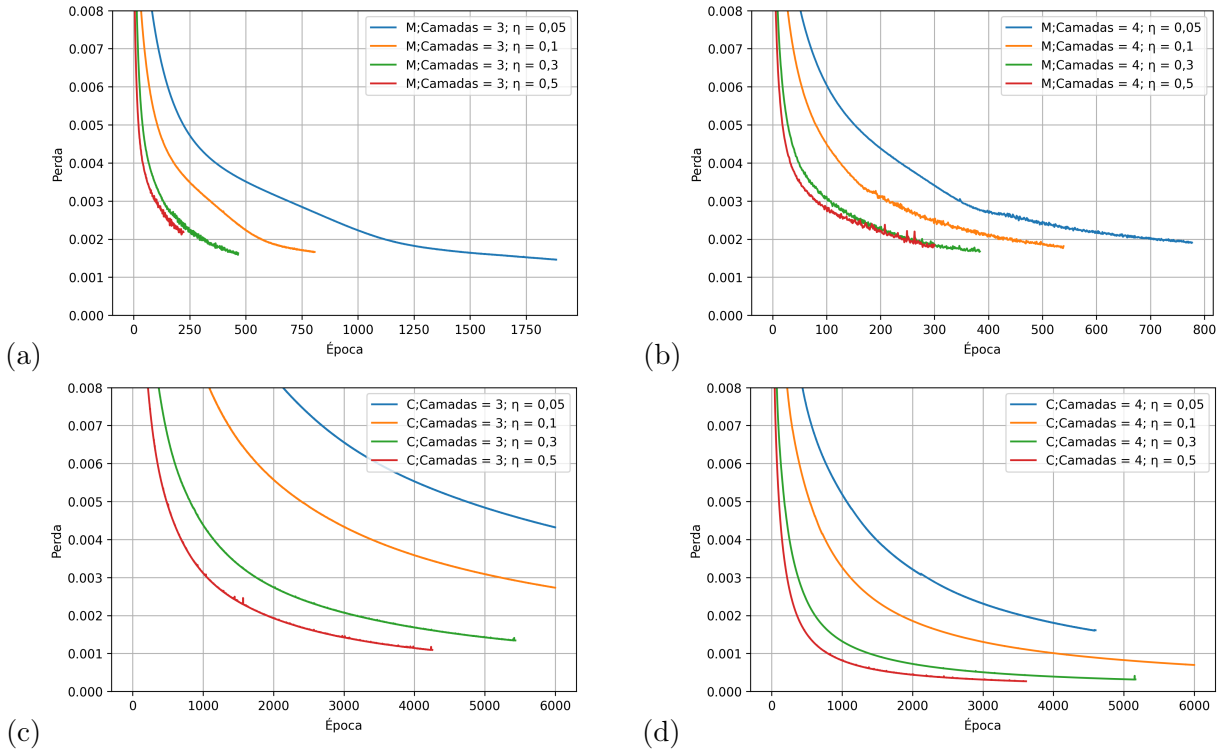

Figura 5.2: Comportamento da função perda para modelos (a) utilizando malha como entrada e 3 camadas, (b) utilizando malha como entrada e 4 camadas, (c) utilizando coordenadas como entrada e 3 camadas, (d) utilizando coordenadas como entrada e 4 camadas. São apresentados os resultados para diferentes taxas de aprendizado η.

Os valores mínimos obtidos para o EQM em cada um dos casos são apresentados na Tab. 5.2. Note que com o aumento da taxa de aprendizado, ocorre a necessidade de um maior número de épocas para que ocorra a convergência dos pesos sinápticos. Além disso, para um mesmo valor de taxa de aprendizado, com o aumento do número de camadas, ocorre uma redução no número de épocas.

Tabela 5.2: EQM para cada modelo

Camadas	**Taxa de Aprendizado**	**Malha**	**Coordenadas**
3	0,05	0,001463	0,004312
3	0,1	0,001624	0,002736
3	0,3	0,001439	0,001334
3	0,5	0,001828	0,001084
4	0,05	0,001845	0,001612
4	0,1	0,001674	0,000707
4	0,3	0,001417	0,000327
4	0,5	0,001450	0,000282

O tempo médio necessário para o treinamento de cada época foi de aproximadamente 2 segundos em todos os modelos treinados, os tempos totais e o número de épocas são indicados nas Tabs. 5.3 e 5.4.

Tabela 5.3: Número de épocas para cada modelo

Camadas	Taxa de Aprendizado	Malha	Coordenada
3	0,05	1884	6000
3	0,1	807	6000
3	0,3	466	5434
3	0,5	222	4255
4	0,05	777	4599
4	0,1	539	6000
4	0,3	384	5165
4	0,5	300	3612

Tabela 5.4: Tempo total de treinamento em segundos

Camadas	Taxa de Aprendizado	Malha	Coordenada
3	0,05	3768	12000
3	0,1	1614	12000
3	0,3	932	10868
3	0,5	444	8510
4	0,05	1554	9198
4	0,1	1078	12000
4	0,3	768	10330
4	0,5	600	7224

5.1.3 Comparativo com Resultados Numéricos

Para ilustrar os resultados obtidos, um valor arbitrário presente no banco de dados de teste foi escolhido para realizar a comparação entre o resultado obtido numericamente e através da rede neural. Foram escolhidos 4 modelos:

- Malha, 3 Camadas, $\eta = 0,3$, EQM = 0,001439 (M_C_3_E_30);
- Malha, 4 Camadas, $\eta = 0,3$, EQM = 0,001417 (M_C_4_E_30);
- Coordenadas, 3 Camadas, $\eta = 0,5$, EQM=0,001084 (C_C_3_E_50);
- Coordenadas, 4 Camadas, $\eta = 0,5$, EQM= 0,000282 (C_C_4_E_50).

Os modelos escolhidos são os melhores de cada um dos 4 diferentes grupos analisados. Para realizar a visualização do avanço temporal, o caso escolhido possui $x_0 = 0,2$, $l_x = 0,3$, $y_0 = 0,1$, $l_y = 0,2$. Para o caso descrito, o banco de dados possui 53 resultados ao longo do tempo. Sendo o primeiro tempo armazenado $t = 0,00313$ e avançando até $t = 0,16266$ com um passo de tempo de $0,00313$. Foram selecionados 3 passos de tempo, sendo eles: $t = \{0,00313; 0,09687; 0,16266\}$. Para o último tempo escolhido a temperatura já está em regime permanente. Para a obtenção dos resultados selecionados, o tempo de simulação foi, em média, de 0,2 segundos o que já é uma vantagem significativa na obtenção de resultados rápidos.

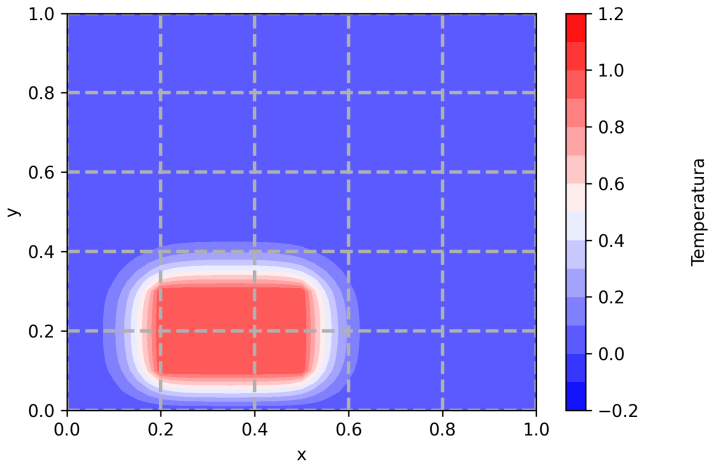

Figura 5.3: Resultado numérico para $t = 0,00313$

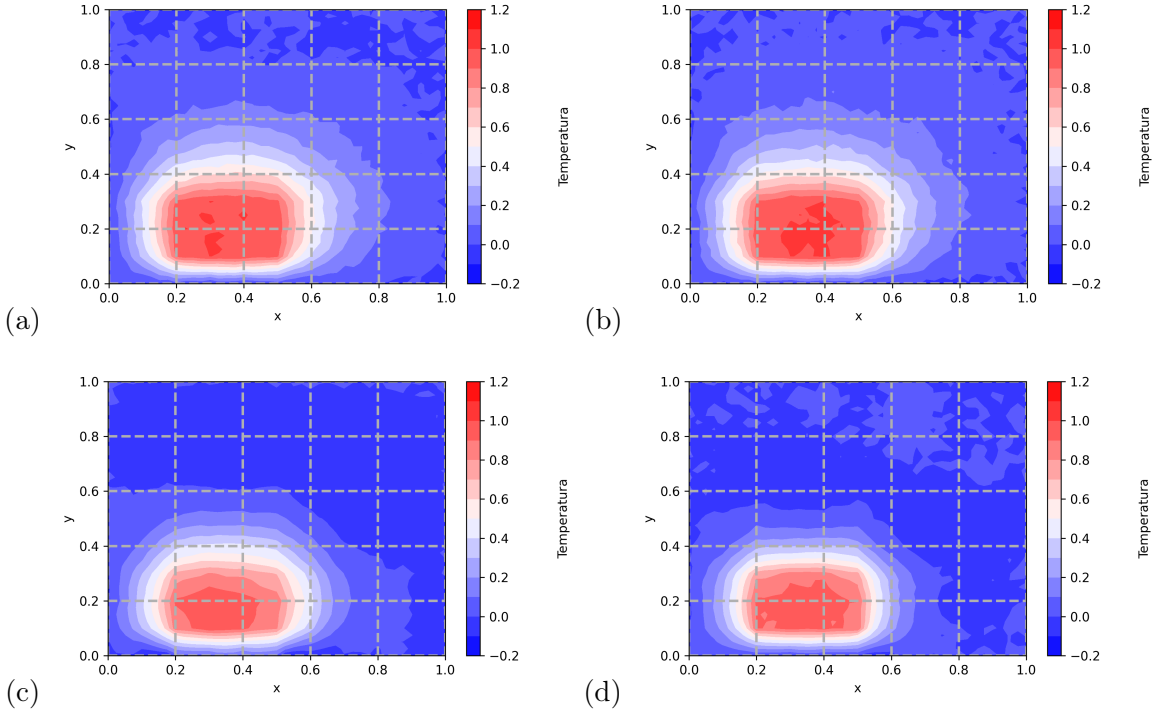

Figura 5.4: Resultado para $t = 0,00313$ para modelo (a) M_C_3_E_30 (b) M_C_4_E_30 (c) C_C_3_E_50 (d) C_C_4_E_50

Note que no momento inicial, as Figs. 5.4 (a) e (b) apresentam franjas maiores quando comparadas com as figuras (c) e (d). O resultado esperado, ilustrado pela Fig. 5.3 apresenta franjas bem estreitas. Em contra partida, a região de temperatura $T = 0$, apresenta ruídos mais significativos nos resultados obtidos através dos inputs de coordenadas.

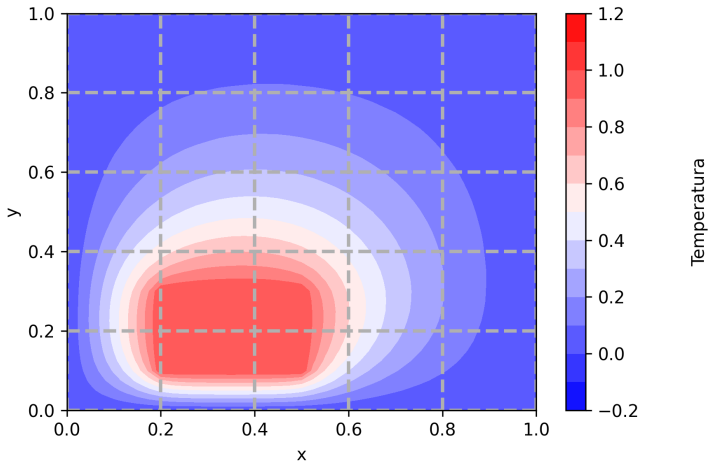

Figura 5.5: Resultado numérico para $t = 0,09687$

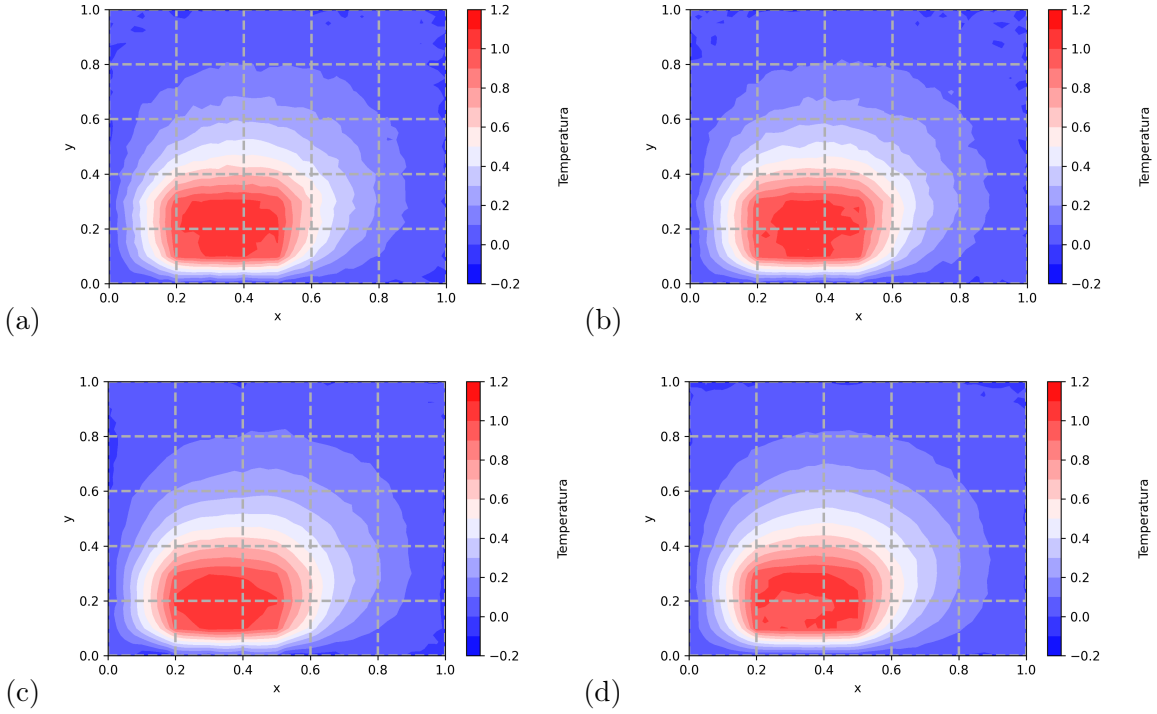

Figura 5.6: Resultado para $t = 0,09687$ para modelo (a) M_C_3_E_30 (b) M_C_4_E_30 (c) C_C_3_E_50 (d) C_C_4_E_50

Com o avançar do tempo, os resultados gerados através das redes neurais que possuem como entrada as coordenadas da região aquecida continuam a apresentar franjas mais próximas do resultado numérico, enquanto as redes obtidas através da malha possuem uma região serrilhada em todas as regiões de transição entre as franjas. Isso pode ser observado nas Figs. 5.6 (a), (b), (c) e (d).

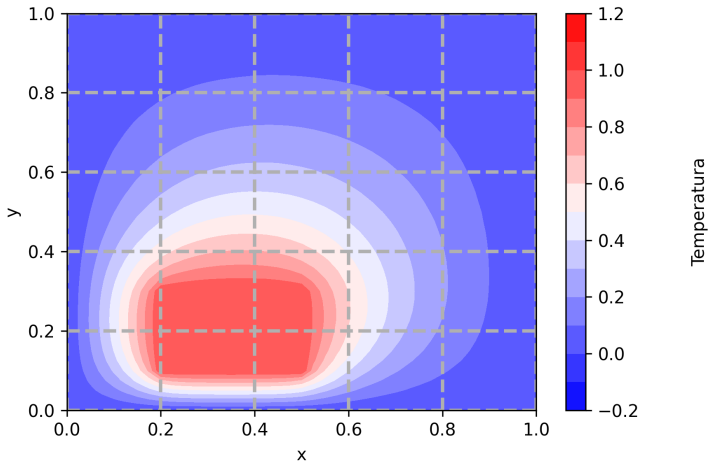

Figura 5.7: Resultado numérico para $t = 0,16266$

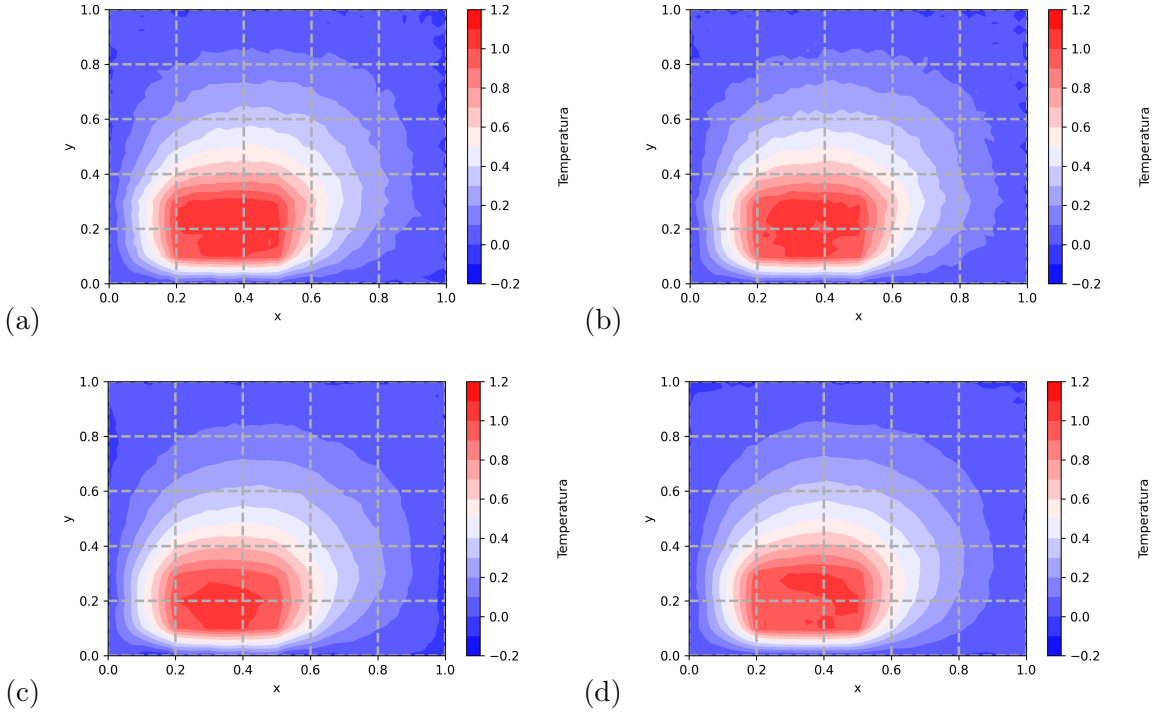

Figura 5.8: Resultado para $t = 0,16266$ para modelo (a) M_C_3_E_30 (b) M_C_4_E_30 (c) C_C_3_E_50 (d) C_C_4_E_50

No caso do regime permanente, nota-se, pelas Figs 5.8 (a), (b), (c) e (d) que o padrão se preserva. As franjas possuem uma definição mais suave nas redes obtidas através de coordenadas da região aquecida.

Com isso, a ferramenta consegue reproduzir com precisão os resultados obtidos numericamente para a condução de calor em uma placa aquecida. Sendo os modelos obtidos através das coordenadas os mais precisos e com menos ruídos nas regiões de transição.

5.2 Problema da Cavidade

Para a solução do problema da cavidade, são necessárias três diferentes redes neurais, sendo uma para solucionar a velocidade horizontal u, a segunda para a velocidade vertical v e por fim, a terceira, para solucionar a pressão p. Para realizar o treinamento, foi gerado um banco de dados com mais de 30000 casos distintos, variando o número de Reynolds e o passo de tempo. O número de Reynolds nos casos analisados se encontra no intervalo entre 50 e 1000 variando de 50 em 50, ou seja, $Re \in \{50, 100, \cdots, 900, 950, 1000\}$. Para gerar este banco, foi utilizada uma máquina virtual hospedada no *Google Cloud Platform* com *Ubuntu* 20.04, 16 Gb de RAM e SSD de 60 Gb. O tempo necessário para realizar as simulações e armazenar os dados foi de 114 horas.

5.2.1 Parâmetros utilizados

A quantidade de neurônios na saída é definida pelo tamanho da malha utilizada no estudo. Para determinar a quantidade de pontos necessários em uma malha, é necessário realizar uma análise de convergência de malha. Neste caso, para a análise, foi realizado o monitoramento da pressão média ao longo da malha para diferentes números de pontos, conforme ilustrado na Fig. 5.9.

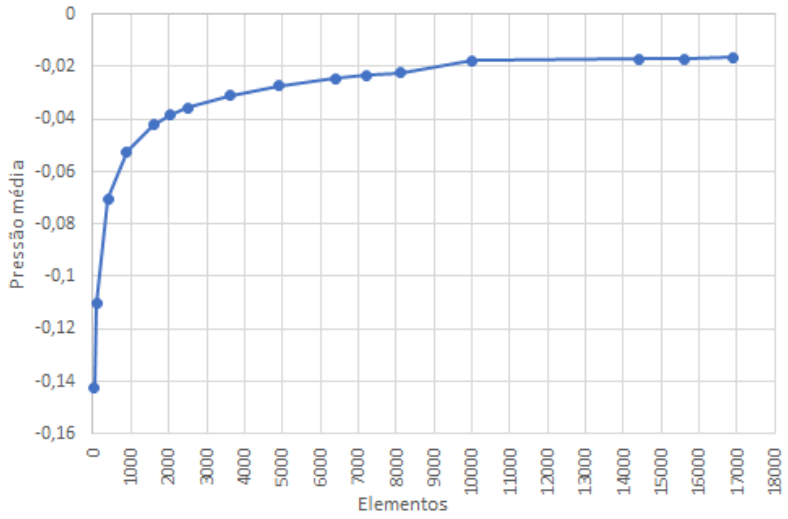

Figura 5.9: Análise de convergência de malha para problema da cavidade

Note que a convergência ocorre para uma malha com aproximadamente 10000 elementos, contudo uma malha com 5000 elementos já apresenta resultados satisfatórios para a realização da análise. Para o presente estudo, utilizou-se uma malha para u com (71×72) elementos, v com (72×71) elementos e p com (72×72) elementos. A diferença entre as malhas se da pela implementação da malha defasada. Desta forma, é possível ter como número base de neurônios nas camadas de entrada e saída os presentes na Tab. 5.5.

Tabela 5.5: Quantidade de neurônios nas camadas de entrada e saída

Modelo	**Número de Neurônios de Entrada**	**Número de Neurônios de Saída**
u	2	5112
v	2	5112
p	2	5184

Para os diferentes números de camadas propostos, utiliza-se um número de neurônios em cada camada de tal forma que as camadas fiquem com uma quantidade linearmente espaçada de neurônios.

5.2.2 Função Perda

Assim como no problema anterior, o monitoramento da função perda é de extrema importância para identificar se o treinamento está sendo eficiente. A Tab. 5.6 exibe os valores finais de EQM obtidos em cada um dos modelos. Alguns dos treinamentos não obtiveram a convergência, neste caso o valor apresentado é "-". A não convergência ocorre devido à uma taxa de aprendizado muito elevado. Dos modelos estudados, nota-se na Tab. 5.6 que o que possui menor EQM é o de 5 camadas com taxa de aprendizado $0,05$.

Tabela 5.6: EQM de cada um dos modelos

Camadas	**Taxa de Aprendizado**	u	v	p
3	0,01	0,002792	0,001258	0,000860
3	0,05	0,000741	0,000937	0,000492
3	0,08	0,001434	0,001046	0,000579
3	0,1	0,001504	0,001045	0,000597
3	0,5	0,002349	0,001515	0,001122
4	0,01	0,000741	0,000510	0,000040
4	0,05	0,001448	0,001008	0,000632
4	0,08	-	-	-
4	0,1	-	-	-
4	0,5	-	-	-
5	0,05	0,000366	0,000338	0,000035
5	0,08	-	-	-
5	0,1	-	-	-
5	0,5	-	-	-

O tempo médio necessário para o treinamento de cada época foi de aproximadamente 3 segundos em todos os modelos treinados com 5 camadas e de 2 segundos para os demais modelos, os tempos totais e o número de épocas são indicados nas Tabs. 5.7 e 5.8.

Tabela 5.7: Número de Épocas para cada modelo

Camadas	**Taxa de Aprendizado**	u	v	p
3	0,01	949	1117	907
3	0,05	663	750	440
3	0,08	412	405	282
3	0,1	312	376	269
3	0,5	91	96	70
4	0,01	706	1292	6000
4	0,05	482	414	296
4	0,08	-	-	-
4	0,1	-	-	-
4	0,5	-	-	-
5	0,05	3352	1905	1627
5	0,08	-	-	-
5	0,1	-	-	-
5	0,5	-	-	-

Tabela 5.8: Tempo total de treinamento em segundos

Camadas	**Taxa de Aprendizado**	u	v	p
3	0,01	1900	2234	1814
3	0,05	1326	1500	880
3	0,08	824	810	564
3	0,1	624	852	560
3	0,5	91	192	140
4	0,01	1412	2600	12000
4	0,05	964	818	588
4	0,08	60	60	60
4	0,1	60	60	60
4	0,5	60	60	60
5	0,05	10056	5715	4881
5	0,08	90	90	90
5	0,1	90	90	90
5	0,5	90	90	90

Perceba, na Tab. 5.8, que os modelos que não obtiveram convergência ainda assim tiveram pelo menos 60 segundos de treinamento. Isso ocorre devido ao critério de parada antecipada. Este critério estava programado para continuar o treinamento por 30 épocas após o mínimo da função perda do banco de dados de validação. Sendo assim, como cada época demanda de 2 a 3 segundos de treinamento, foram gastos entre 60 e 90 segundos nos casos que divergiram.

Observando a Tab. 5.7 e as Figs. 5.10, 5.11 e 5.12, existe um crescimento significativo do custo computacional ao reduzirmos a taxa de aprendizado. Além disso, esperava-se que com o aumento do número de camadas implicasse em uma redução do EQM, o que não ocorreu. Isso pode ser explicado pelo critério de parada antecipada, note, em especial nas Figs5.10 (c) , 5.11 (c) e 5.12 (c), que existem alguns movimentos abruptos de subida do EQM em todos os casos para o modelo de 4 camadas, de tal forma que, possivelmente um aumento no número de épocas no critério de

parada antecipada poderia permitir a obtenção de novos pontos de mínimo.

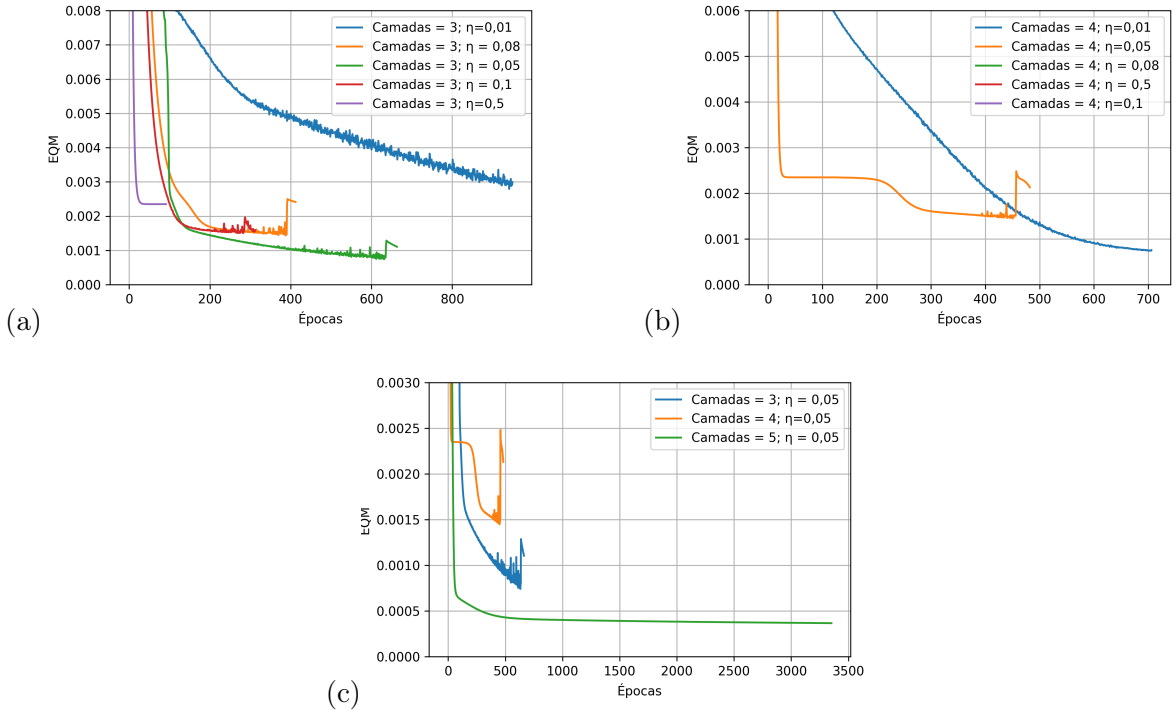

Figura 5.10: Função perda para modelo u com (a) 3 Camadas, (b) 4 Camadas e (c) com $\eta = 0,05$

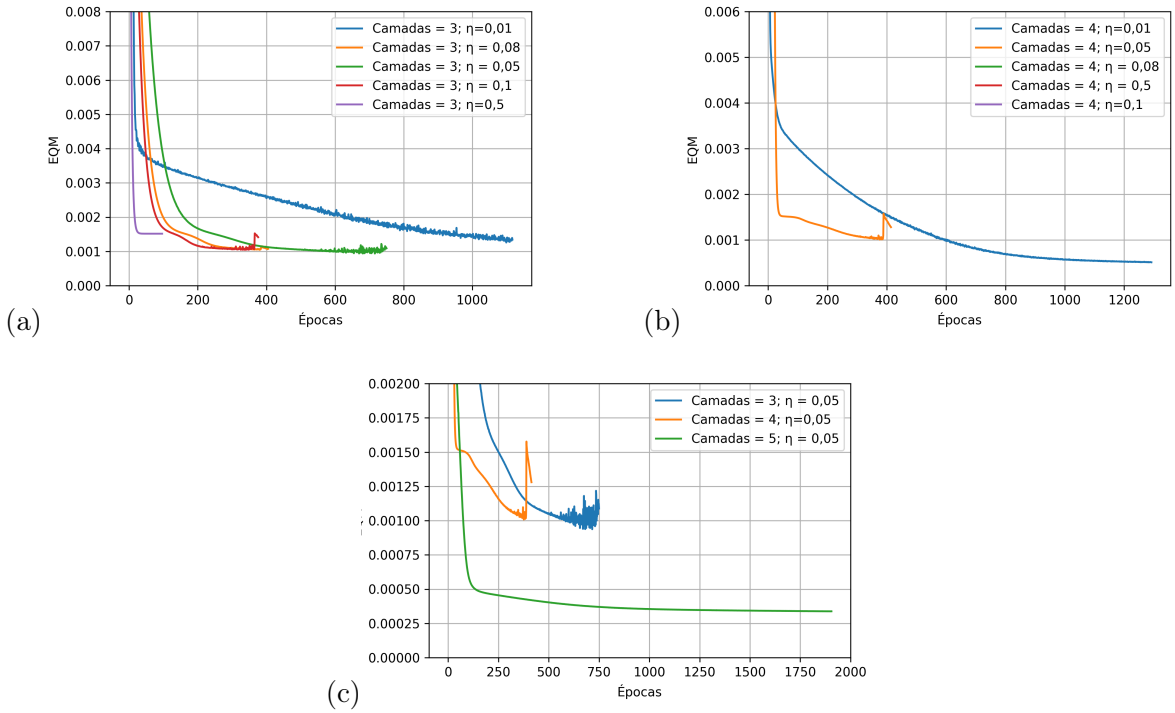

Figura 5.11: Função perda para modelo v com (a) 3 Camadas, (b) 4 Camadas e (c) com $\eta = 0,05$

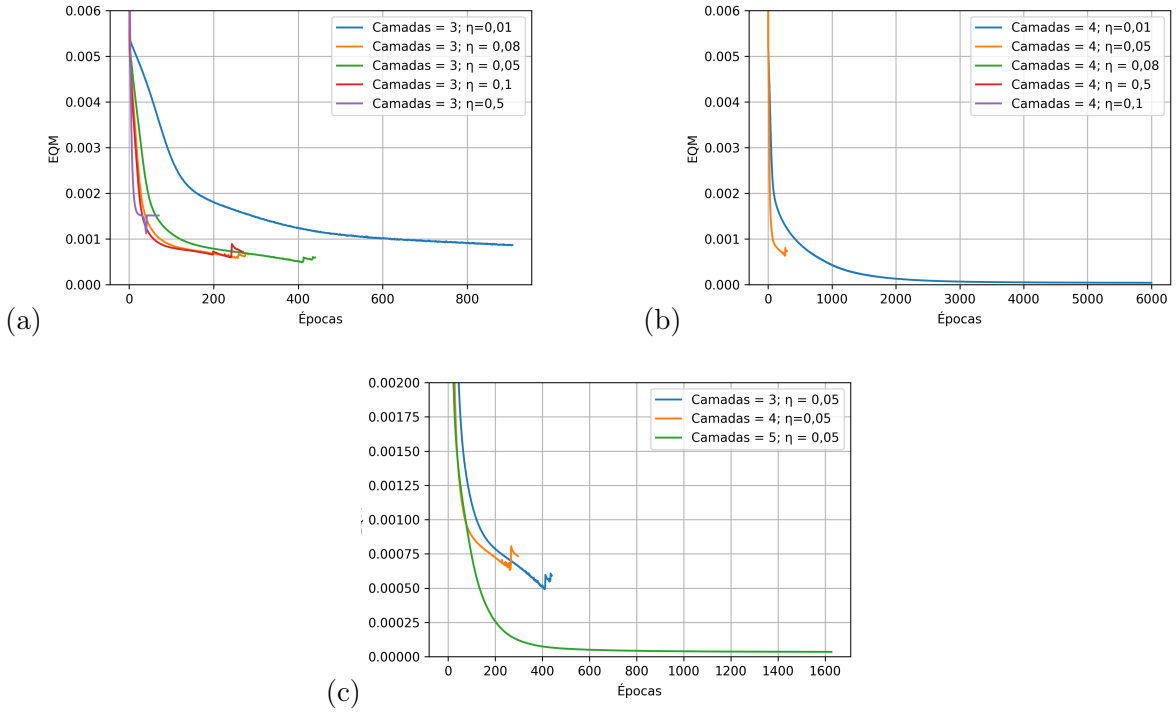

Figura 5.12: Função perda para modelo p com (a) 3 Camadas, (b) 4 Camadas e (c) com $\eta = 0,05$

5.2.3 Comparativo com Resultados Numéricos Existentes no Banco de Testes

Observando os valores da Tab. 5.6 nota-se que para os três modelos o caso que apresentou o melhor desempenho foi o de 5 camadas com taxa de aprendizado 0,05. Desta forma, será realizada uma análise comparativa dos resultados obtidos através desta rede com o banco de dados existente. Ao utilizar a rede treinada, os resultados demoram 0,2 segundos para serem obtidos.

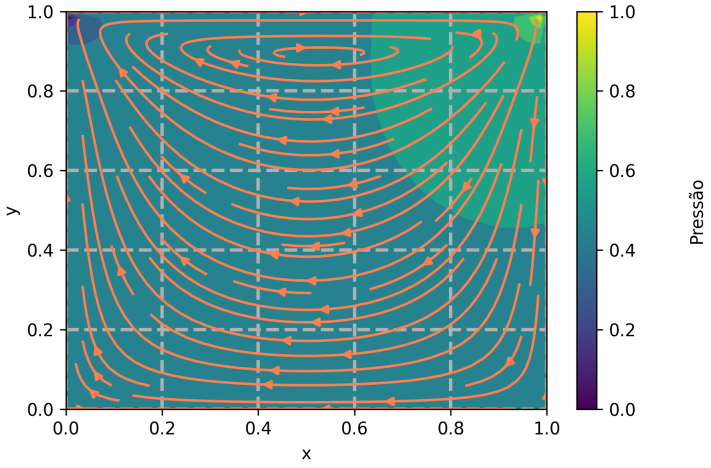

Figura 5.13: Resultado numérico para $t = 0,1$

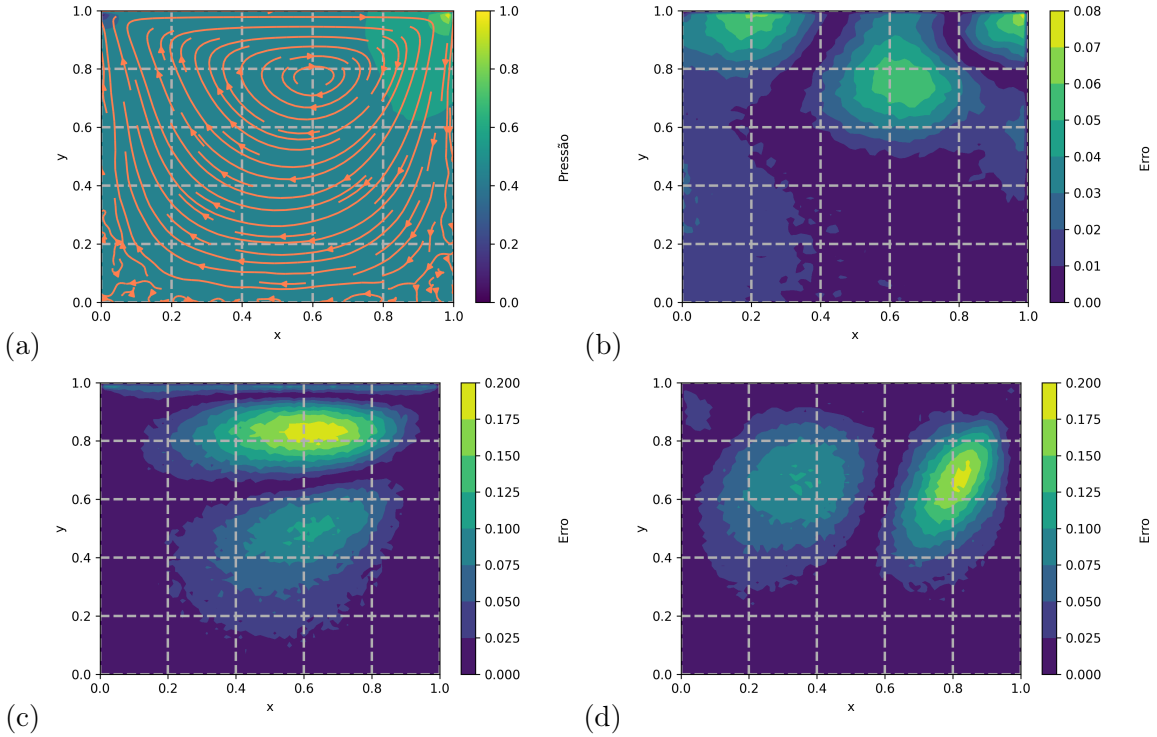

Figura 5.14: Resultado para $t = 0,1$ (a) resultado obtido pela rede (b) erro absoluto de p (c) erro absoluto de u (d) erro absoluto de v

Note que, apesar do baixo valor encontrado para EQM em todos os três modelos,e apesar de existir uma região de recirculação bem definida na Fig. 5.14 o resultado obtido via redes neurais, ilustrado na Fig. 5.13 diverge bastante do resultado numérico, conforme ilustrado pelos erros absolutos nas Figs. 5.14 (b), (c) e (d).

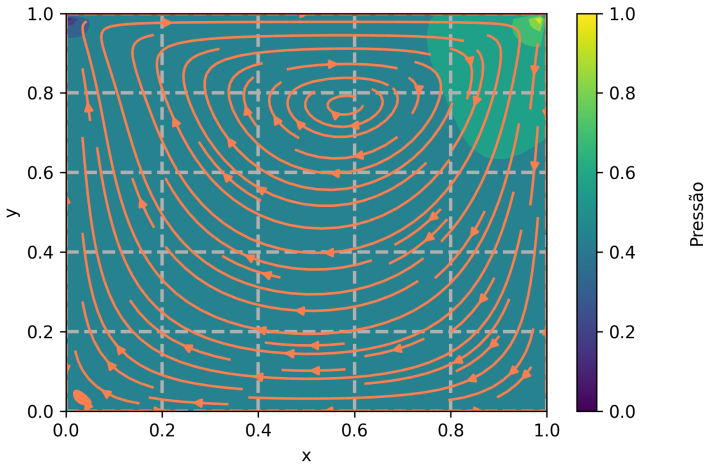

Figura 5.15: Resultado numérico para $t = 2,0$

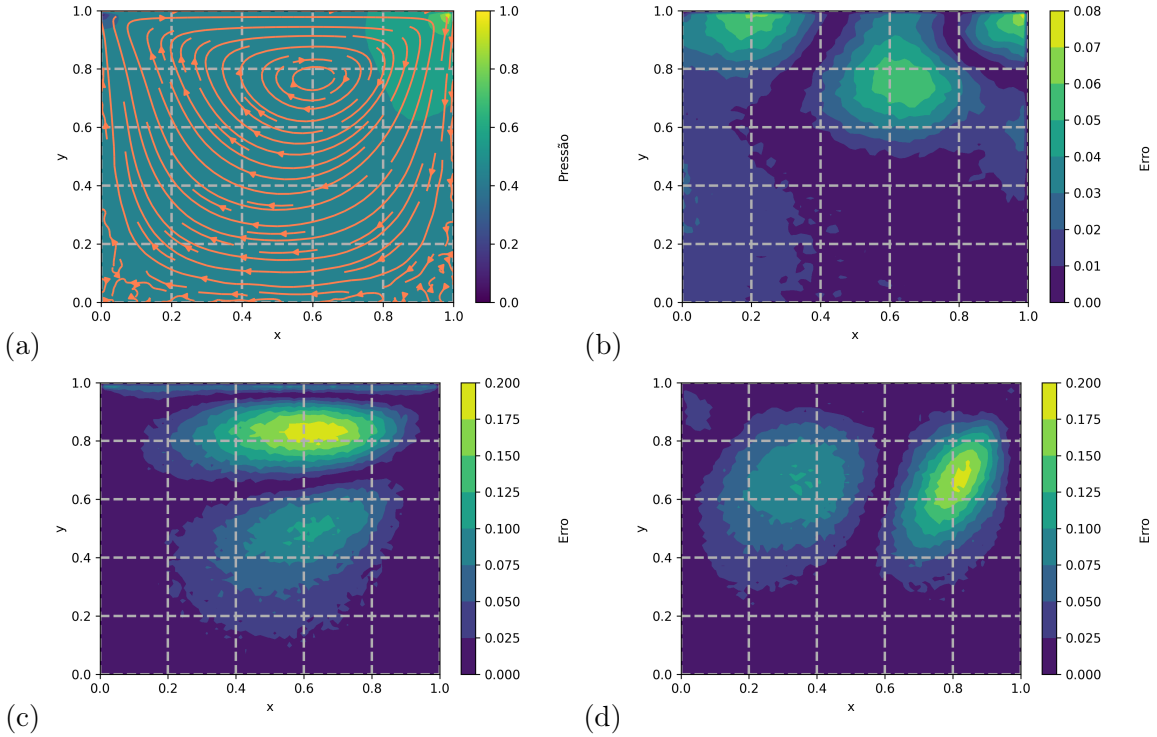

Figura 5.16: Resultado para $t = 2,0$ (a) resultado obtido pela rede (b) erro absoluto de p (c) erro absoluto de u (d) erro absoluto de v

Com o avançar do tempo, conforme ilustrado nas Figs. 5.15 e 5.14 (a),(b),(c) e (d), nota-se a mesma tendência, apesar do EQM ser baixo, ainda é possível perceber grandes divergências quando comparamos os resultados numéricos com o obtido via redes neurais.

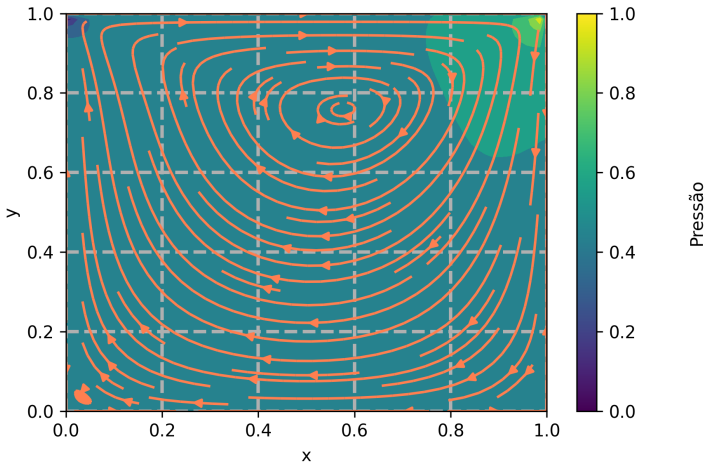

Figura 5.17: Resultado numérico para $t = 14,0$

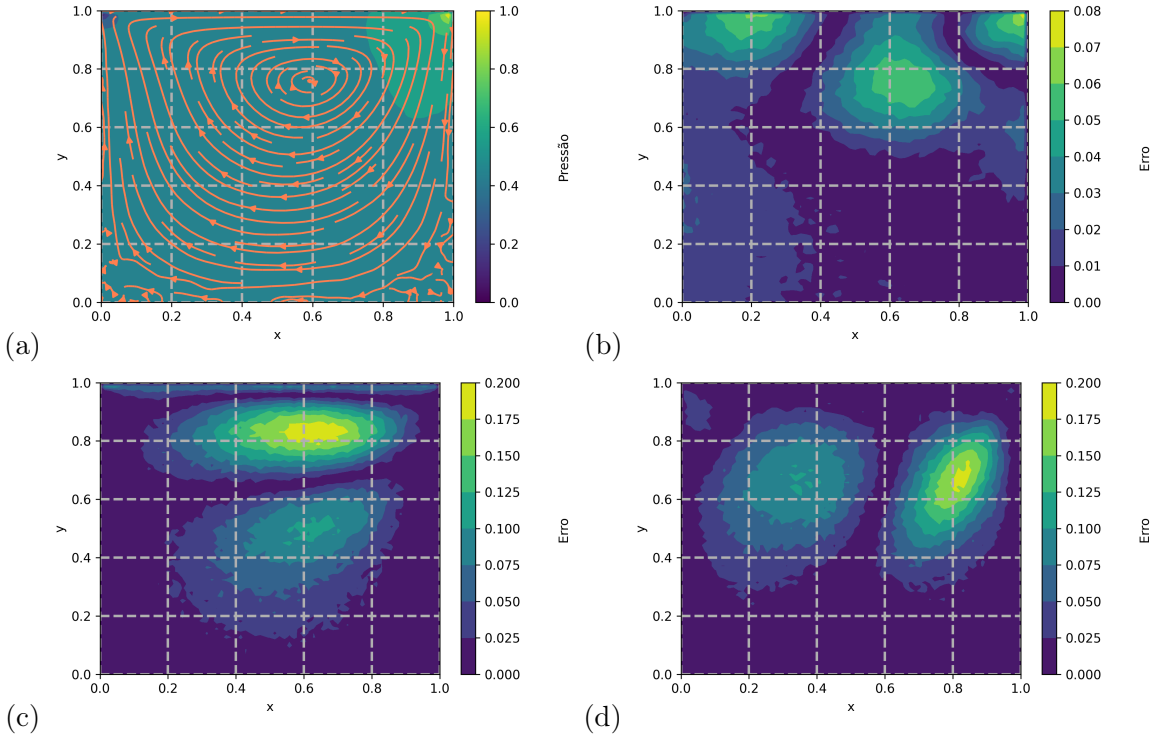

Figura 5.18: Resultado para $t = 14,0$ (a) resultado obtido pela rede (b) erro absoluto de p (c) erro absoluto de u (d) erro absoluto de v

Chegando ao regime permanente, as regiões com maiores divergências permanecem as mesmas, conforme indicado nas Figs. 5.18 (b), (c) e (d).

Observe também que, das Figs. 5.13, 5.15 e 5.17 (b),(c) e (d) boa parte do domínio possui erro baixo. Neste caso, isso ocorre nas regiões mais afastadas da região central do vórtice presente na cavidade.

Observando a linha central da cavidade e, analisando as propriedades em relação ao eixo, temos

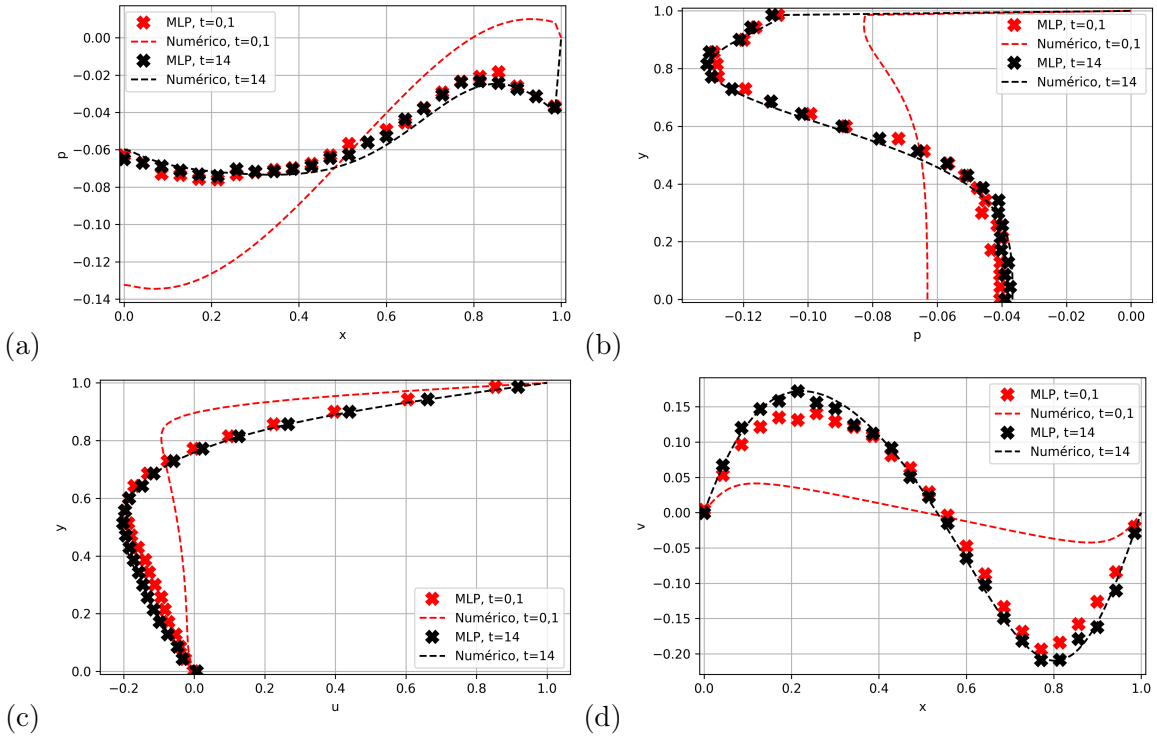

Figura 5.19: Comparativo da linha central da cavidade para (a) pressão no eixo x (b) pressão no eixo y (c) velocidade vertical ao longo do eixo y (d) velocidade horizontal ao longo do eixo x

Note que os resultados obtidos através do MLP no regime permanente são muito próximos dos obtidos numericamente, enquanto no regime transiente existe um grande desacoplamento.

Com isso, esta ferramenta não é capaz de predizer com exatidão o comportamento mecânico de um fluido confinado durante o período transiente.

Capítulo 6

Conclusões

O uso de redes neurais sem dúvidas reduz o tempo necessário para a obtenção de resultados preliminares. Quando comparados os centésimos de segundo necessários para a obtenção de um resultado ao utilizar uma rede neural treinada com os minutos necessários para se obter o mesmo resultado através do método numérico.

Assim, associada à precisão obtida para o estudo do problema de condução de calor em uma placa aquecida, o modelo MLP pode ser utilizado para realização de estudos de condução de calor ao longo de uma placa. Também foi possível notar que não há ganhos em se utilizar como parâmetro de entrada os pontos da malha, sendo mais vantajoso utilizar as coordenadas da região aquecida como parâmetro de entrada.

No entanto, no caso do problema da cavidade, os resultados não foram satisfatórios, apesar do baixo EQM. Em especial, nas regiões de recirculação os resultados obtidos diferiram dos resultados numéricos. Isso foi notado tanto para os modelos que visavam predizer os valores das velocidade u e v quanto para o modelo que visava predizer o campo de pressão p.

6.1 Sugestão de Trabalhos Futuros

Para continuar o estudo e melhorar a percepção a cerca de redes neurais, sugere-se que sejam realizados os seguintes trabalhos:

- Aumentar o número de camadas ocultas;
- Realizar o treinamento de redes neurais LTSM (*Long Short Term Memory* , conforme indicação de Brunton, Noack e Koumoutsakos (2020);
- Realizar o estudo para o problema da cavidade com obstáculo;
- Realizar o estudo utilizando a formulação de vorticidade e linhas de corrente.

REFERÊNCIAS BIBLIOGRÁFICAS

AGGARWAL, C. C. *Neural Networks and Deep Learning*. Springer International Publishing, 2018. Disponível em: <https://doi.org/10.1007/978-3-319-94463-0>.

ALMEIDA, V. A. de; FRANÇA, G. B.; VELHO, H. F. de C. Short-range forecasting system for meteorological convective events in Rio de Janeiro using remote sensing of atmospheric discharges. *International Journal of Remote Sensing*, Informa UK Limited, v. 41, n. 11, p. 4372–4388, fev. 2020. Disponível em: <https://doi.org/10.1080/01431161.2020.1717669>.

ANDERSON, J. D. *Computational Fluid Dynamics*. McGraw-Hill Education, 1995. ISBN 0070016852. Disponível em: <https://www.xarg.org/ref/a/0070016852/>.

BAI, T.; TAHMASEBI, P. Efficient and data-driven prediction of water breakthrough in subsurface systems using deep long short-term memory machine learning. *Computational Geosciences*, Springer Science and Business Media LLC, v. 25, n. 1, p. 285–297, set. 2020. Disponível em: <https://doi.org/10.1007/s10596-020-10005-2>.

BELL, J. B.; COLELLA, P.; GLAZ, H. M. A second-order projection method for the incompressible Navier-Stokes equations. *Journal of Computational Physics*, v. 85, n. 2, p. 257–283, 1989. ISSN 0021-9991. Disponível em: <https://www.sciencedirect.com/science/article/pii/0021999189901514>.

BISHOP, C.; JAMES, G. Analysis of multiphase flows using dual-energy gamma densitometry and neural networks. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, Elsevier BV, v. 327, n. 2-3, p. 580–593, abr. 1993. Disponível em: <https://doi.org/10.1016/0168-9002(93)90728-z>.

BROWN, D. L.; CORTEZ, R.; MINION, M. L. Accurate projection methods for the incompressible Navier–Stokes equations. *Journal of Computational Physics*, v. 168, n. 2, p. 464–499, 2001. ISSN 0021-9991. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0021999101967154>.

BRUNTON, S. L.; NOACK, B. R.; KOUMOUTSAKOS, P. Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics*, Annual Reviews, v. 52, n. 1, p. 477–508, jan. 2020. Disponível em: <https://doi.org/10.1146/annurev-fluid-010719-060214>.

BURGGRAF, O. R. Analytical and numerical studies of the structure of steady separated flows. *Journal of Fluid Mechanics*, Cambridge University Press (CUP), v. 24, n. 1, p. 113–151, jan. 1966. Disponível em: <https://doi.org/10.1017/s0022112066000545>.

ÇENGEL, Y. *Mecânica dos Fluidos: Fundamentos e Aplicações*. New York, NY: McGraw-Hill Education, 2015. ISBN 978-8580554908.

ÇENGEL, Y. A.; GHAJAR, A. J. *Tranferência de Calor e Massa*. New York, NY: Amgh Editora, 2009.

CHORIN, A. J. Numerical solution of the Navier-Stokes equations. *Mathematics of Computation*, American Mathematical Society (AMS), v. 22, n. 104, p. 745–762, 1968.

ERTURK, E.; CORKE, T. C.; GöKÇöL, C. Numerical solutions of 2-d steady incompressible driven cavity flow at high Reynolds numbers. *International Journal for Numerical Methods in Fluids*, Wiley, v. 48, n. 7, p. 747–774, 2005. Disponível em: <https://doi.org/10.1002/fld.953>.

ESFE, M. H. et al. Designing an artificial neural network to predict thermal conductivity and dynamic viscosity of ferromagnetic nanofluid. *International Communications in Heat and Mass Transfer*, Elsevier BV, v. 68, p. 50–57, nov. 2015. Disponível em: <https://doi.org/10.1016/j.icheatmasstransfer.2015.06.013>.

FAURE, T. M. et al. Visualizations of the flow inside an open cavity at medium range Reynolds numbers. *Experiments in Fluids*, Springer Science and Business Media LLC, v. 42, n. 2, p. 169–184, nov. 2006. Disponível em: <https://doi.org/10.1007/s00348-006-0188-8>.

GAD-EL-HAK, M. Modern developments in flow control. *Applied Mechanics Reviews*, v. 49, n. 7, p. 365–469, 1996.

GAUTIER, N. et al. Closed-loop separation control using machine learning. *Journal of Fluid Mechanics*, Cambridge University Press, v. 770, p. 442–457, 2015.

GERON, A. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, 2019. ISBN 1492032646. Disponível em: <https://www.xarg.org/ref/a/1492032646/>.

GHIA, U.; GHIA, K.; SHIN, C. High-re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method. *Journal of Computational Physics*, Elsevier BV, v. 48, n. 3, p. 387–411, dez. 1982. Disponível em: <https://doi.org/10.1016/0021-9991(82)90058-4>.

GOODFELLOW, I. *Deep Learning (Adaptive Computation and Machine Learning series)*. The MIT Press, 2016. ISBN 0262035618. Disponível em: <https://www.xarg.org/ref/a/0262035618/>.

HAYKIN, S. O. *Neural Networks and Learning Machines (3rd Edition)*. Pearson, 2008. ISBN 0131471392. Disponível em: <https://www.xarg.org/ref/a/0131471392/>.

HINCH, E. J. *Think before you compute : a prelude to computational fluid dynamics*. New York: Cambridge University Press, 2020. ISBN 978-1108789998.

HIRABAYASHI, M. et al. Characterization of flow reduction properties in an aneurysm due to a stent. *Physical Review E*, American Physical Society (APS), v. 68, n. 2, ago. 2003. Disponível em: <https://doi.org/10.1103/physreve.68.021918>.

IMRIE, C.; DURUCAN, S.; KORRE, A. River flow prediction using artificial neural networks: Generalisation beyond the calibration range. *Journal of Hydrology*, v. 233, n. 1-4, p. 138–153, 2000.

JIANG, Z. et al. A deep learning approach to predict abdominal aortic aneurysm expansion using longitudinal data. *Frontiers in Physics*, Frontiers Media SA, v. 7, jan. 2020. Disponível em: <https://doi.org/10.3389/fphy.2019.00235>.

KAWAGUTI, M. Numerical solution of the Navier-Stokes equations for the flow in a two-dimensional cavity. *Journal of the Physical Society of Japan*, Physical Society of Japan, v. 16, n. 11, p. 2307–2315, nov. 1961. Disponível em: <https://doi.org/10.1143/jpsj.16.2307>.

MARCHI, C. H.; SUERO, R.; ARAKI, L. K. The lid-driven square cavity flow: numerical solution with a 1024 x 1024 grid. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, scielo, v. 31, p. 186 – 198, 09 2009. ISSN 1678-5878. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S1678-58782009000300004&nrm=iso>.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, Springer Science and Business Media LLC, v. 5, n. 4, p. 115–133, dez. 1943. Disponível em: <https://doi.org/10.1007/bf02478259>.

PARK, B.; BAE, J. K. Using machine learning algorithms for housing price prediction: The case of fairfax county, virginia housing data. *Expert Systems with Applications*, Elsevier BV, v. 42, n. 6, p. 2928–2934, abr. 2015. Disponível em: <https://doi.org/10.1016/j.eswa.2014.11.040>.

RAISSI, M.; YAZDANI, A.; KARNIADAKIS, G. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, American Association for the Advancement of Science, v. 367, n. 6481, p. 1026–1030, 2020.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, American Psychological Association (APA), v. 65, n. 6, p. 386–408, 1958. Disponível em: <https://doi.org/10.1037/h0042519>.

SKANSI, S. *Introduction to Deep Learning: From Logical Calculus to Artificial Intelligence (Undergraduate Topics in Computer Science)*. Springer, 2018. ISBN 3319730037. Disponível em: <https://www.xarg.org/ref/a/3319730037/>.

SONG, Z.; MURRAY, B. T.; SAMMAKIA, B. A dynamic compact thermal model for data center analysis and control using the zonal method and artificial neural networks. *Applied Thermal Engineering*, Elsevier BV, v. 62, n. 1, p. 48–57, jan. 2014. Disponível em: <https://doi.org/10.1016/j.applthermaleng.2013.09.006>.

TEMAM, R. Sur l'approximation de la solution des équations de Navier-Stokes par la méthode des pas fractionnaires (ii). *Archive for rational mechanics and analysis*, Springer, v. 33, n. 5, p. 377–385, 1969.

VERSTEEG, H. K. *An introduction to computational fluid dynamics : the finite volume method*. Harlow, England New York: Pearson Education Ltd, 2007. ISBN 978-0131274983.

WHITE, F. M. *Mecânica Dos Fluidos*. Mc Graw Hill, 2019. ISBN 8580556066. Disponível em: <https://www.xarg.org/ref/a/8580556066/>.

ZHU, L. et al. Machine learning methods for turbulence modeling in subsonic flows around airfoils. *Physics of Fluids*, American Institute of Physics Inc., v. 31, n. 1, 2019.

ANEXOS

I. NOTEBOOKS UTILIZADOS

Link Para Github:

https://github.com/rowcrf/projeto-de-graduacao-02