Politécnica UFRJ

ESTUDO DE ESCOAMENTOS PARA ARREFECIMENTO DE ELETRÔNICOS ATRAVÉS DE MÉTODO DE ELEMENTOS FINITOS

Ygor da Silva Florentino

Projeto de Graduação apresentado ao Curso de Engenharia Mecânica da Escola Politécnica, Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Engenheiro.

Orientador: Gustavo Rabello dos Anjos

Rio de Janeiro

Junho de 2022

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO

Departamento de Engenharia Mecânica

DEM/POLI/UFRJ

ESTUDO DE ESCOAMENTOS PARA ARREFECIMENTO DE ELETRÔNICOS ATRAVÉS DE MÉTODO DE ELEMENTOS FINITOS

Ygor da Silva Florentino

PROJETO FINAL SUBMETIDO AO CORPO DOCENTE DO DEPARTAMENTO DE ENGENHARIA MECÂNICA DA ESCOLA POLITÉCNICA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO MECÂNICO.

Aprovada por:

Prof. Gustavo Rabello dos Anjos, Ph.D.

Prof. Gabriel Lisbôa Verissimo, D.Sc.

Prof. Gustavo Cesar Rachid Bodstein, Ph.D.

RIO DE JANEIRO, RJ – BRASIL

JUNHO DE 2022

da Silva Florentino, Ygor

Estudo de escoamentos para arrefecimento de eletrônicos através de Método de Elementos Finitos/ Ygor da Silva Florentino. – Rio de Janeiro: UFRJ/Escola Politécnica, 2022.

XVII, 104 p.: il.; 29, 7cm.

Orientador: Gustavo Rabello dos Anjos

Projeto de Graduação – UFRJ/ Escola Politécnica/ Curso de Engenharia Mecânica, 2022.

Referências Bibliográficas: p. 89 – 91.

1. Arrefecimento de Eletrônicos. 2. Mecânica dos Fluidos Computacional. 3. Elementos Finitos. 4. Método Taylor-Galerkin. 5. Formulação Corrente-Vorticidade. I. Rabello dos Anjos, Gustavo. II. Universidade Federal do Rio de Janeiro, UFRJ, Curso de Engenharia Mecânica. III. Estudo de escoamentos para arrefecimento de eletrônicos através de Método de Elementos Finitos.

"Nobody ever figures out what life is all about, and it doesn't matter. Explore the world. Nearly everything is really interesting if you go into it deeply enough." — *Richard P. Feynman*

Agradecimentos

"Se eu vi mais longe, foi por estar nos ombros de gigantes". Retirada de um trecho de uma carta de Isaac Newton para Robert Hooke, em 5 de Fevereiro de 1676, esta frase não poderia descrever melhor o que a conclusão deste projeto de Graduação significa para mim. Olhando para trás, tudo o que sinto é gratidão por todos amigos e colegas que fiz durante o caminho, por tudo aquilo que passei e dos excelentes professores e profissionais da UFRJ que tive o prazer de conhecer.

Primeiramente, gostaria de agradecer a minha família e amigos por todo o apoio ao longo desses anos de faculdade, em especial meu pai, Severino, e minha mãe, Irene. Poucos sabiam que eu era filho de um porteiro e uma empregada doméstica, mas, felizmente, isso nunca interferiu nas nossas relações. Com o apoio deles, e de muito esforço e ajuda de amigos, como Maurício e Adriana, consegui chegar até aqui.

Aos meus inestimáveis colegas, que tive a sorte de conhecer e acompanhar praticamente durante todos os períodos e em quase todas as matérias que cursei. Gostaria agradecer aos meus colegas mais próximos: Tiago, Rafael, Felipe, Eduardo, Luiz e Luís, Marcelle, André, Eric, Gabriel, Vitor, Larissa e Vinícius. Em especial, gostaria de agradecer também ao Yan e ao Jefferson, que me deram muito suporte na minha reta final de formação. Certamente deixei de mencionar alguém, mas garanto que minha gratidão se estende a todos que caminharam junto comigo nessa universidade.

Não poderia também de reconhecer a importância dos professores e membros da faculdade, que me ajudaram nesse trajeto e agregaram novos e interessantes conhecimentos na nossa formação. Em especial, gostaria de deixar meus agradecimentos a todos os professores do departamento de Engenharia Mecânica, particularmente aos professores Nísio Brum, Hélcio Orlande, Flávio de Marco, Carolina Cotta e Antônio Figueiredo, que direta ou indiretamente participaram da minha formação, sejam

sendo meus professores, ou do meu irmão gêmeo.

Por fim, ao meu orientador, professor Gustavo dos Anjos, que há pouco se juntou ao corpo docente da universidade, não somente pelo indispensável suporte neste projeto, mas trouxe consigo também novas aspirações e novos interesses para a faculdade de engenharia, sendo, para mim, um dos melhores professores com o qual tive o prazer de cursar uma disciplina.

Resumo do Projeto de Graduação apresentado à Escola Politécnica/UFRJ como parte dos requisitos necessários para a obtenção do grau de Engenheiro Mecânico

ESTUDO DE ESCOAMENTOS PARA ARREFECIMENTO DE ELETRÔNICOS ATRAVÉS DE MÉTODO DE ELEMENTOS FINITOS

Ygor da Silva Florentino

Junho/2022

Orientador: Gustavo Rabello dos Anjos

Programa: Engenharia Mecânica

Este projeto apresenta um estudo sobre escoamentos em diversas configurações com uso das equações do movimento de fluido, sob a formulação de função corrente - vorticidade. O objetivo do trabalho é entender e visualizar a influência das parâmetros de malhas nos vetores e campos escalares de interesse em um escoamento. Os dados do estudo foram obtidos a partir de simulações numéricas, desenvolvidas em linguagem *Python*, das equações de governo. O algoritmo responsável por essas simulações faz uso do método de elementos finitos junto ao esquema de Taylor-Galerkin para discretização das equações no tempo e no espaço. Além disso, foram feitas simulações numéricas de casos clássicos e recorrentes da literatura, como o escoamento de Poiseuille e o escoamento em canal com cilindro, na faixa de Reynolds variando de 1, 10 e 100, para validação dos dados obtidos.

Palavras-chave: Arrefecimento de Eletrônicos. Simulação Numérica. Elementos Finitos. Método Taylor-Galerkin. Formulação Função Corrente-Vorticidade.

Abstract of Undergraduate Project presented to POLI/UFRJ as a partial fulfillment of the requirements for the degree of Mechanical Engineer

STUDY OF FLOWS FOR COOLING OF ELECTRONICS WITH INFINITE ELEMENT METHOD

Ygor da Silva Florentino

June/2022

Advisor: Gustavo Rabello dos Anjos

Department: Mechanical Engineering

This project presents a study on flows in various configurations using fluid movement equations, under the streamfunction - vorticity formulation. The objective of the present work is to try to understand and visualize the influence of different mesh parameters in the vectors and scalar fields in a flow. The study data were obtained from numerical simulations, written and developed in *Python*, from the government equations. The algorithm responsible for these simulations makes use of the finite element method alongside the Taylor-Galerkin scheme to discretize equations in time and space. In addition, numerical simulations of classical and recurrent cases of the literature, such as the Poiseuille flow and the flow with an inner cylinder, within the Reynolds range between 1, 10 and 100, were performed to validate the data obtained.

Keywords: Electronics Cooling. Numerical Simulation. Finite Elements. Taylor-Galerkin method. Streamfunction-Vorticity formulation.

Sumário

Lista de Figuras **x**

Lista de Tabelas **xiii**

Lista de Símbolos **xv**

1 Introdução **1**

1.1 Objetivo 2

1.2 Organização do Trabalho 3

2 Revisão Bibliográfica **5**

2.1 Dinâmica dos Fluidos Computacional 5

2.2 Método de Elementos Finitos 6

2.3 Transferência de Calor em Fluidos 8

3 Fundamentação Teórica **10**

3.1 Dinâmica dos Fluidos 10

3.2 Adimensionalização das Equações 13

3.3 Vorticidade e Função Corrente 15

3.3.1 Vorticidade 16

3.3.2 Função Corrente 17

3.4 Formulação Corrente Vorticidade 18

4 Modelagem da Simulação **20**

4.1 Método de Elementos Finitos 20

4.2 Discretização do Domínio e seus Elementos 23

4.3 MEF na Formulação Corrente Vorticidade 27

4.4 Método de Taylor-Galerkin 30
4.5 MEF e a Condição CFL 32

5 Validação do Modelo **34**
5.1 Escoamento de Hagen-Poiseuille 35
5.1.1 Escoamento em Canal para Re = 1 38
5.1.2 Escoamento em Canal para Re = 10 42
5.1.3 Escoamento em Canal para Re = 100 46
5.2 Escoamento em Canal com Cilindro 50
5.2.1 Escoamento com Cilindro para Re = 10 52
5.2.2 Escoamento com Cilindro para Re = 100 56

6 Resultados dos Estudos **60**
6.1 Diferentes Números de Canais Internos 65
6.1.1 5 Canais Internos 67
6.1.2 8 Canais Internos 69
6.1.3 11 Canais Internos 71
6.1.4 Comparação entre N5, N8 e N11 72
6.2 Diferentes Dimensões dos Canais Internos 74
6.2.1 8 Canais Internos com Larguras Diferentes 74
6.2.2 Comparação entre N8, 8L1 e 8L2 78
6.2.3 8 Canais Internos com Alturas Diferentes 80
6.2.4 Comparação entre N8, 8A1 e 8A2 84

7 Conclusões **86**
7.1 Considerações Finais 86
7.2 Trabalhos Futuros 88

Referências Bibliográficas **89**

A Códigos em Python utilizados no projeto **92**
A.1 Script para casos de validação 92
A.2 Script para casos de estudo 97
A.2.1 Script para casos sem fluido escoando 97
A.2.2 Script para casos com fluido escoando 100

Lista de Figuras

1.1 Esquema de funcionamento de um resfriador (“cooler”) de líquido. . . 1
1.2 Sistema de refrigeração utilizando água pura. 2

2.1 Modelo em elementos finitos de articulação de joelho humano. 7
2.2 Análise DFC de um dissipador de calor. 9

3.1 Esquema demonstrando domínio e contorno de um volume de controle arbitrário. 11
3.2 Efeitos do número de Reynolds em escoamento ao redor de um cilindro. Adaptado de WHITE (2021). 17

4.1 Esquema de solução utilizando MEF. 21
4.2 Malha de elementos finitos com 24 quadriláteros e 35 nós; malha de elementos finitos com 48 triângulos e 35 nós. 23
4.3 Exemplo de malha de triângulos utilizada no método de elementos finitos. 24
4.4 Elemento triangular de malha no MEF. 24
4.5 Esboço do método de Taylor-Galerkin. 31

5.1 Escoamento Hagen-Poiseuille. 36
5.2 Malhas para simulação de escoamento de Poiseuille. 37
5.3 Condições iniciais das simulações de Hagen-Poiseuille. 37
5.4 Função corrente para escoamento de Poiseuille com Re = 1. 39
5.5 Função vorticidade para escoamento de Poiseuille com Re = 1. 39
5.6 Velocidade horizontal u para escoamento de Poiseuille com Re = 1. . 39
5.7 Velocidade vertical v para escoamento de Poiseuille com Re = 1. . . . 40
5.8 Temperatura para escoamento de Poiseuille com Re = 1. 40

5.9 Gráfico de velocidades u com solução analítica para escoamento de Poiseuille com Re = 1. 41
5.10 Gráfico de distribuição de temperatura para escoamento de Poiseuille com Re = 1. 42
5.11 Velocidade horizontal u para escoamento de Poiseuille com Re = 10. . 43
5.12 Velocidade vertical v para escoamento de Poiseuille com Re = 10. . . 43
5.13 Temperatura para escoamento de Poiseuille com Re = 10. 44
5.14 Gráfico de velocidades u com solução analítica para escoamento de Poiseuille com Re = 10. 45
5.15 Gráfico de distribuição de temperatura para escoamento de Poiseuille com Re = 10. 46
5.16 Velocidade horizontal u para escoamento de Poiseuille com Re = 100. 47
5.17 Velocidade vertical v para escoamento de Poiseuille com Re = 100. . . 47
5.18 Temperatura para escoamento de Poiseuille com Re = 100. 47
5.19 Gráfico de velocidades u com solução analítica para escoamento de Poiseuille com Re = 100. 48
5.20 Gráfico de distribuição de temperatura com solução analítica para escoamento de Poiseuille com Re = 100. 49
5.21 Malhas para simulação de escoamento em canal com cilindro. 51
5.22 Condições iniciais das simulações de canal com cilindro. 51
5.23 Função corrente para escoamento com cilindro para Re = 10. 53
5.24 Função vorticidade para escoamento com cilindro para Re = 10. . . . 53
5.25 Velocidade horizontal u para escoamento com cilindro para Re = 10. . 53
5.26 Velocidade vertical v para escoamento com cilindro para Re = 10. . . 54
5.27 Temperatura para escoamento com cilindro para Re = 10. 54
5.28 Solução analítica da velocidade horizontal u para escoamento com cilindro para Re = 10. 55
5.29 Gráfico da distribuição de temperatura para escoamento com cilindro para Re = 10. 56
5.30 Velocidade horizontal u para escoamento com cilindro para Re = 100. 57
5.31 Velocidade vertical v para escoamento com cilindro para Re = 100. . 57
5.32 Temperatura para escoamento com cilindro para Re = 100. 57

5.33 Solução analítica da temperatura para escoamento com cilindro para Re = 100. 58
5.34 Solução analítica da velocidade horizontal u para escoamento com cilindro para Re = 100. 58

6.1 Esquema de componentes de um processador de 4 núcleos. 60
6.2 Escoamento de fluido para resfriamento de componente eletrônico. . . 61
6.3 Condições iniciais e de contorno do problema com canais internos. . . 64
6.4 Vetores da simulação da malha 8L1. 66
6.5 Vetores da simulação da malha N5. 67
6.6 Vetores da simulação da malha N8. 69
6.7 Vetores da simulação da malha N11. 71
6.8 Gráfico de velocidade **u** para malhas com diferentes quantidades de canais. 73
6.9 Malhas para simulação com paredes internas de larguras diferentes. . 75
6.10 Vetores da simulação da malha 8L1. 76
6.11 Vetores da simulação da malha 8L2. 77
6.12 Gráfico de velocidade **u** para malhas com diferentes larguras. 79
6.13 Malhas para simulação com paredes internas de alturas diferentes. . . 80
6.14 Vetores da simulação da malha 8A1. 81
6.15 Vetores da simulação da malha 8A2. 83
6.16 Gráfico de velocidade **u** para malhas com diferentes alturas. 84

Lista de Tabelas

5.1 Configurações e hardware da máquina que executou as simulações. . . 35
5.2 Número de nós e de elementos de cada malha. 36
5.3 Parâmetros do canal e fluido de estudo. 38
5.4 Valores de Δt e N de cada malha. 38
5.5 Erro relativo para $u(y)$ de cada malha. 41
5.6 Valores de Δt e N de cada malha. 43
5.7 Erro relativo para $u(y)$ de cada malha. 45
5.8 Erro relativo para $u(y)$ de cada malha. 48
5.9 Erro relativo para $T(y)$ de cada malha. 49
5.10 Número de nós e de elementos de cada malha. 50
5.11 Parâmetros do canal e fluido estudados. 52
5.12 Valores de Δt e N de cada malha. 52
5.13 Erro relativo para $u(y)$ de cada malha. 55
5.14 Erro relativo para $u(y)$ de cada malha. 59

6.1 Siglas das malhas estudadas 62
6.2 Parâmetros do canal externo. 63
6.3 Parâmetros do fluido estudado. 63
6.4 Número de iterações e passo de tempo do MEF. 63
6.5 Pontos no escoamento onde há geração de calor. 65
6.6 Parâmetros das paredes internas do escoamento. 65
6.7 Quantidade de nós e elementos triangulares das malhas. 66
6.8 Resultados absolutos obtidos da simulação N5, N8 e N11. 73
6.9 Parâmetros das paredes internas do escoamento. 74
6.10 Quantidade de nós e elementos triangulares das malhas. 75

6.11 Resultados absolutos obtidos da simulação N8, 8L1 e 8L2. 79
6.12 Parâmetros das paredes internas do escoamento. 80
6.13 Quantidade de nós e elementos triangulares das malhas. 81
6.14 Resultados absolutos obtidos da simulação. 85

Lista de Símbolos

Símbolo	Descrição	Unidade
t	Tempo	s
Δt	Intervalo de tempo	s
N	Número de iterações	-
$\mathbf{V}$	Vetor velocidade	m/s
u	Componente do vetor velocidade na direção x	m/s
v	Componente do vetor velocidade na direção y	m/s
w	Componente do vetor velocidade na direção z	m/s
∇	Operador gradiente	m^{-1}
ρ	Domínio do volume de controle	$\mathrm{kg/m}^3$
g	Vetor gravidade	$\mathrm{m/s}^2$
p	Pressão absoluta	Pa
μ	Viscosidade dinâmica	Pa.s
ν	Viscosidade cinemática	m^2/s
λ	Viscosidade volumétrica	Pa.s
$\mathbf{\Omega}$	Domínio	-
$\mathbf{\Gamma}$	Contorno do domínio	-
$\mathbf{I}$	Matriz identidade	-
$\mathbf{M}$	Matriz de massa	-
$\mathbf{G}$	Matriz de gradiente	-
$\mathbf{K}$	Matriz de rigidez	-
$\mathbf{K_{est}}$	Matriz de estabilização	-
h	Entalpia mássica específica	J/kg
k	Condutividade térmica	$\mathrm{W.m}^{-1}\mathrm{K}^{-1}$
α	Difusividade térmica	m^2/s

Símbolo	Descrição	Unidade
c_p	Calor específico à pressão constante	J/kg.K
T	Temperatura	K
T_W	Temperatura da parede	K
$\mathbf{\Phi}$	Função dissipação	J/m^{3}s
Q	Calor fornecido ao escoamento	W.m^{-3}
U	Velocidade de referência	m/s
L	Comprimento de referência	m
Re	Número de Reynolds	-
Pr	Número de Prandtl	-
Fr	Número de Froude	-
Ec	Número de Eckert	-
Pe	Número de Peclet	-
ω	Vetor vorticidade	s^{-1}
ω_x	Componente do vetor vorticidade na direção x	s^{-1}
ω_y	Componente do vetor vorticidade na direção y	s^{-1}
ω_z	Componente do vetor vorticidade na direção z	s^{-1}
ψ	Função corrente	m^{2}s^{-1}
$\mathbf{A}$	Área	m^2
$\dot{m}$	Vazão mássica	kg/s
$\mathbf{N_{i,j,k}}$	Função de forma	-

Capítulo 1

Introdução

O arrefecimento de equipamentos e componentes eletrônicos se tornou um grande desafio nas últimas décadas por conta da crescente necessidade desses aparelhos para aplicações residenciais e industriais. Os diversos avanços dessa tecnologia, seja no seu design ou na sua construção, fez com que essas partes se tornassem cada vez mais eficientes e de menores dimensões. Pesquisas e debates entre engenheiros têm ocorrido para discutir sobre conceitos e métodos para determinar quais destes formas de resfriamento são mais economicamente viáveis. A indústria vem buscando, por exemplo, formas mais eficientes de minimizar o consumo de eletricidade e reduzir o custo de resfriamento. Um exemplo é o sistema de resfriamento líquido de processadores, como apresentado na Figura 1.1 abaixo.

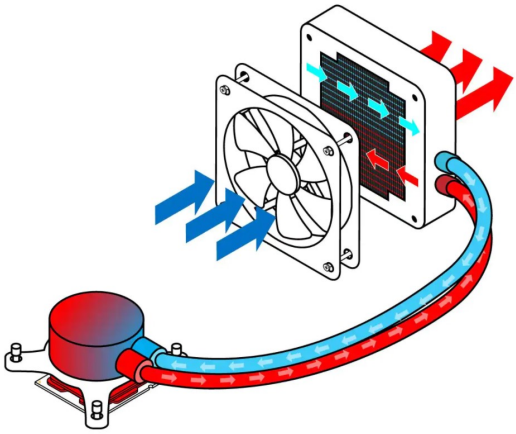


Figura 1.1: Esquema de funcionamento de um resfriador ("cooler") de líquido.

Fonte: Adaptado de Intel Corporation [1]

1.1 Objetivo

Dentre as possíveis formas de otimização no resfriamento de eletrônicos, existem duas vertentes principais. A primeira é encontrar fluidos refrigerantes mais eficientes, isto é, com coeficientes de transferência de calor maiores do que o do ar. O ar, por sua vez, é um dos fluidos de refrigeração mais empregados nas últimas décadas nos sistemas de arrefecimento (como dissipadores de calor e sistemas de ventilação). Não obstante disso, dentre os refrigerantes, os melhores fluidos são aqueles que possuem o melhor custo-benefício, grau toxicidade e propriedades termofísicas para uma dada aplicação [2].

O fluido refrigerante mais comum no mercado, além do ar, é a água. Sistemas à base de água são, geralmente, mais eficientes, porém, devem ter maiores cuidados quanto às consequências que seu uso pode provocar. Uma delas é a necessidade de haver controle periódico nos parâmetros de corrosão, pH, resistividade do fluido, entre outros [3]. A água pura, ou deionizada, é utilizado em aplicações de arrefecimento e possui propriedades termoquímicas que permitem seu uso junto aos equipamentos eletrônicos. A Figura 1.2 ilustra um esquema que faz uso desse tipo de fluido [4].

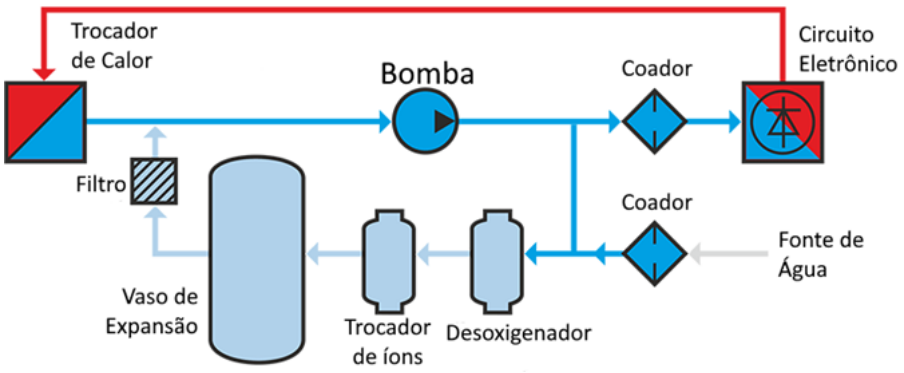


Figura 1.2: Sistema de refrigeração utilizando água pura.

Fonte: Adaptado de SWEP Industrial Handbook [4]

A segunda principal vertente é a de buscar inovações e progressos nas estruturas dos dissipadores de calor, *e.g.* estudar a interferência de estruturas de aletas nesses dissipadores, como feito em YANG et al. (2017) [5], em vários formatos diferentes. Nesse sentido, pesquisas e simulações, muitas vezes numéricas, são feitas para estudar o efeito de diferentes configurações de dissipadores e como estes parâmetros

podem afetar o desempenho do sistema como um todo, com formas e tamanhos distintos.

Seguindo essa última direção, o objetivo deste trabalho é analisar a interferência das formas e contornos em escoamentos em canais com números de Reynolds distintos, além de analisar também a transferência de calor em cada caso. Para tal, foram feitas simulações numéricas em linguagem Python, fazendo uso da formulação função corrente-vorticidade e do método de elementos finitos (MEF) baseado no esquema de Taylor-Galerkin. Dessa forma, é possível desacoplar os termos de pressão e velocidade na equação de Navier-Stokes. Com isso, e com auxílio das equações de governo, tem-se uma das formas da literatura de se obter informações (como velocidade e temperatura) de um escoamento. No trabalho, foram analisadas algumas configurações com diferentes fluidos e números de Reynolds. Além disso, para validar as simulações, também foram testados alguns casos clássicos da literatura, como o escoamento de *Poiseuille* em canais [6].

1.2 Organização do Trabalho

O trabalho está dividido em sete capítulos, e a seguir será descrito qual conteúdo de cada tópico.

- No Capítulo 2, "Revisão Bibliográfica", são apresentados os trabalhos existentes sobre os temas método de elementos finitos e dinâmica de fluidos computacional que motivaram o presente trabalho.
- No Capítulo 3, "Fundamentação Teórica", são revisadas as principais equações de governo, conceitos e formulações utilizadas como base para as simulações.
- No Capítulo 4, "Modelagem da Simulação", são apresentados os parâmetros da simulação, com base nos tópicos introduzidos no capítulo 4.
- No Capítulo 5, "Validação Numérica", são exibidos os resultados obtidos nas simulações dos casos clássicos da literatura, para servir de comparação do método aplicado e modelado de acordo com o Capítulo 4.
- No Capítulo 6, "Resultados dos Estudos", são exibidos os resultados das simulações feitas no trabalho.

- No Capítulo 7, “Conclusão”, é feita uma análise sobre os resultados obtidos no capítulo anterior, e serão oferecidas ideias para temas de trabalhos futuros.

Capítulo 2

Revisão Bibliográfica

Neste capítulo é apresentado ao leitor os principais trabalhos e bibliografia utilizados como base para o presente trabalho. Aqui, são incluídos livros e projetos em temas como dinâmica de fluidos computacional (DFC), transferência de calor e elementos finitos.

2.1 Dinâmica dos Fluidos Computacional

A dinâmica dos fluidos computacional, também conhecida pela sigla "DFC" [7], é a área da computação científica responsável pelo estudo de métodos e algoritmos computacionais para simulação de fenômenos que envolvem o movimento de fluidos. A transferência de calor decorrente da agitação desse fluido também pode ser avaliada.

O estudo da DFC inclui não apenas o movimento de fluidos em torno de objetos e formas, como é o caso de aerofólios no automobilismo e na aviação [8], mas também no interior de tubulações e turbinas, avaliando possíveis melhorias e otimizações para essas situações [9]. Diversos trabalhos na engenharia utilizam a DFC em diferentes casos para diferentes áreas buscando obter dados e informações de seu interesse. No caso do estudo de escoamentos, geralmente se procura analisar as distribuições de velocidade, pressão e temperatura nas regiões de fluxo. Com isto, um engenheiro pode ser capaz de propor melhorias no projeto, seja para reduzir seus custos operacionais, como para otimizar o desempenho das partes do modelo – por exemplo, reduzir o arrasto aerodinâmico de uma aeronave ou automóvel permite

uma redução no seu consumo de combustível.

Outro ponto importante a se ressaltar na DFC é a escolha do software ou da ferramenta computacional a ser utilizada. Este trabalho propõe o uso do *Python*, que é uma linguagem de programação consideravelmente recente comparada a outras aplicadas na indústria. Apesar disso, ela teve sua aplicação e desenvolvimento em constante crescimento nos últimos anos, principalmente por ser uma linguagem gratuita de código aberto (do termo em inglês, "*open source*"). Isso possibilitou seu emprego não apenas na engenharia, mas também em áreas administrativas e financeiras, por exemplo.

Dentro dos programas escritos em *Python*, são usadas bibliotecas como "*NumPy*" [10] e "*Meshio*" [11]. A primeira disponibiliza diversas funções e ferramentas matemáticas que são essenciais para solucionar problemas algébricos. A segunda biblioteca, por sua vez, facilita a leitura de malhas, que é um procedimento vital para o método numérico utilizado neste trabalho. Além dessas, várias outras bibliotecas foram utilizadas, não apenas para modelar os problemas de estudo, mas também para manipular e exibir melhor os seus resultados, seja em tabelas, em gráficos ou figuras. Os códigos deste projeto se encontram no Apêndice A.

2.2 Método de Elementos Finitos

Na engenharia e nas ciências, muitos fenômenos físicos podem ser descritos e modelados em termos de equações diferenciais parciais. Em geral, resolver essas equações por clássicos métodos analíticos para formas arbitrárias é praticamente impossível. Segundo FISH (2007) [12], o método de elementos finitos (conhecido por MEF ou FEM, na sigla em inglês) é um tipo de abordagem numérica pela qual as formas são divididas em elementos e as equações diferenciais parciais podem ser resolvidas aproximadamente para cada elemento. Do ponto de vista da engenharia, o MEF é um método para resolver problemas como análise de estresse de estruturas, transferência de calor, escoamento de fluidos e problemas eletromagnéticos através de simulação computacional.

O método de elementos finitos aplicado neste trabalho é descrito em detalhes no Capítulo 4, onde são apresentados os parâmetros e considerações para os casos de

estudo deste projeto. Além do estudo de arrefecimento de eletrônicos, o MEF tem diversas aplicações nas ciências, tais como [12]:

(a) Análise térmica e de estresse de partes industriais como chips eletrônicos, aparelhos elétricos, válvulas, etc.

(b) Vasos de pressão, motores automotivos e de aeronaves.

(c) Análise de colisão de carros, trens e aeronaves.

(d) Escoamento de fluidos de tanques de líquidos refrigerantes, poluentes e contaminantes, além de ar em sistemas de ventilação.

(e) Análise de procedimentos cirúrgicos, como cirurgias plásticas e reconstrução de partes do corpo humano, como apresentado na Figura 2.1.

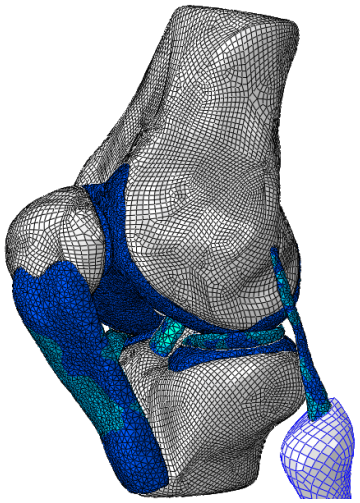


Figura 2.1: Modelo em elementos finitos de articulação de joelho humano.

Fonte: Adaptado de Wikimedia Commons [13]

Como é possível observar, nos últimos anos diversas novas áreas de aplicação do MEF vem surgindo, principalmente relacionadas em áreas de saúde. Tanto o modelo apresentado na Figura 2.1, quanto os demais modelos elaborados em elementos finitos podem tomar muitas horas para serem criados, porém o retorno é uma drástica redução no número de protótipos no processo de design, otimizando os custos do projeto.

Como introduzido no Capítulo 1, o presente trabalho faz uso do método de elementos finitos para estudar o arrefecimento de eletrônicos, em específico, a simular

escoamentos e transferência de calor em canais com diferentes formatos. As equações parciais que governam tal sistema são apresentadas no Capítulo 3.

2.3 Transferência de Calor em Fluidos

A engenharia faz uso de diversos princípios extraídos da termodinâmica e outras ciências, como a mecânica dos fluidos e a transferência de calor e massa, para analisar e elaborar projetos destinados a atender às necessidades da humanidade [14]. Os engenheiros sempre buscaram desenvolver projetos otimizados com alto desempenho, procurando fatores como o aumento da produção de algum produto desejado, a redução de algum insumo de um recurso escasso, a redução dos custos totais de um projeto ou reduzir os impactos ambientais deste.

Os princípios da termodinâmica na engenharia desempenham um papel importante na realização desses objetivos [15], principalmente considerando a termodinâmica como a ciência da energia, com especial destaque ao *calor* como uma das formas de energia térmica [14]. Dentro dessa tecnologia, existe um conceito de grande relevância para a engenharia como um todo, que é a definição de *transferência de calor*. BERGMAN e LAVINE (2017) [16] utilizam uma forma bem simples para defini-la, como sendo a energia térmica (calor) "em trânsito" devido à uma diferença espacial de temperatura em um meio.

Aliado ao desenvolvimento de ferramentas computacionais na dinâmica dos fluidos (DFC), foram desenvolvidas técnicas para a discretização e solução de sistemas mais complexos de equações da transferência de calor, como apresentado na Figura 2.2. Essas técnicas surgiram principalmente a partir do uso de métodos numéricos, como o MEF utilizado no presente trabalho. Esses sistemas são construídos predominantemente por equações diferenciais parciais (EDPs), algumas das quais as soluções podem ser obtidas analiticamente ou a partir de métodos numéricos.

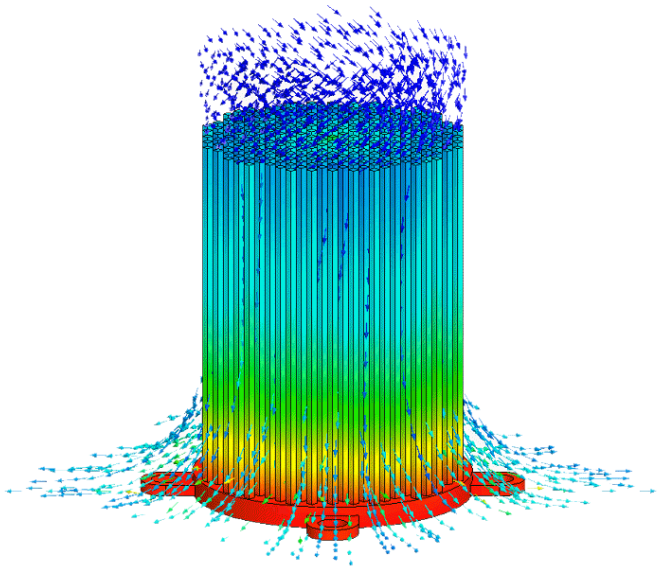


Figura 2.2: Análise DFC de um dissipador de calor.

Fonte: Adaptado de Wikimedia Commons [17]

Dessa forma, o uso conjunto da DFC, dos métodos númericos e das ferramentas computacionais na análise e elaboração de projetos envolvendo transferência de calor e massa. se tornou não apenas possível, mas também acessível e mais barato do que fazer diversos experimentos. Além disso, essas tecnologias possibilitaram um grande avanço em diversos campos de estudos como, por exemplo, no arrefecimento de eletrônicos - que é o foco deste trabalho.

Capítulo 3

Fundamentação Teórica

Neste capítulo é discutido os fundamentos da teoria que dá suporte ao estudo deste projeto. Aqui, são apresentadas hipóteses e parâmetros do problema estudado, além de elaborar as principais equações de governo e condições de contorno, principalmente aquelas que regem a mecânica do fluido e a transferência de calor.

3.1 Dinâmica dos Fluidos

Para modelar escoamentos de fluidos nos sistemas apresentados neste trabalho, é necessário estabelecer algumas definições. Segundo a literatura, um escoamento pode ser definido como o movimento conjunto das moléculas de um fluido, de forma que as propriedades deste possam ser tratadas pontualmente, isto é, podendo variar a cada ponto.

Na mecânica dos fluidos, as propriedades de um fluido em movimento podem ser medidas de duas maneiras diferentes. A primeira forma seria realizando a medição de uma propriedade em um ponto fixo no espaço à medida que as partículas do fluido passam. A segunda forma, por sua vez, seria seguindo uma parcela de fluido ao longo de sua linha de corrente, que é uma linha de campo no escoamento. A derivada de um campo em relação a uma posição fixa no espaço é conhecida como "derivada euleriana", enquanto a derivada seguindo uma parcela móvel é chamada de "derivada material" (ou Lagrangiana) [6].

A derivada material é definida como o operador não linear:

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + (\mathbf{V} \cdot \nabla) \tag{3.1}$$

onde $\mathbf{V}$ é vetor de velocidades do escoamento e ∇ é o operador gradiente, definidos como:

$$\mathbf{V} = (u, v, w) \tag{3.2}$$

$$\nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z} \right) \tag{3.3}$$

onde u, v, w são as componentes do vetor de velocidades nas direções x, y e z.

O primeiro termo no lado direito da Equação 3.1 é a derivada euleriana comum, representando mudanças em um ponto fixo em relação ao tempo; enquanto que o segundo termo representa mudanças de uma propriedade em relação à posição levando em consideração a velocidade $\mathbf{V}$ do fluido. Esta derivada "especial" pode ser obtida através da aplicação da regra da cadeia em que todas as variáveis independentes são verificadas para alteração ao longo do caminho.

No estudo de escoamento de fluidos, existem duas equações governo que são comumente empregadas: as equações de Navier-Stokes, e a equação de conservação de massa, também conhecida como equação de continuidade [6]. As equações de Navier-Stokes baseiam-se na suposição de que o fluido é uma substância contínua (em vez de partículas discretas), além de supor que todos os campos de interesse, como pressão, velocidade, massa específica e temperatura são diferenciáveis. Essas equações são derivadas dos princípios básicos de continuidade da massa, da quantidade de movimento e da energia considerando um volume de controle (como o mostrado na Figura 3.1), sobre o qual esses princípios podem ser aplicados.

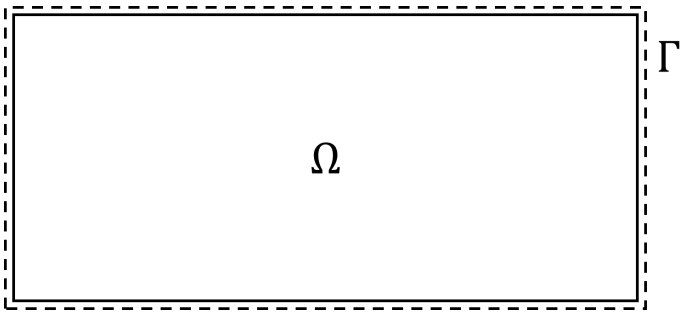


Figura 3.1: Esquema demonstrando domínio e contorno de um volume de controle arbitrário.

Esse volume finito é denotado por ω e seu contorno por Γ. Esse volume de controle pode permanecer fixo no espaço ou pode se mover com o fluido. Nele, as equações de continuidade e de Navier-Stokes podem ser descritas como:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{V}) = 0 \tag{3.4}$$

$$\rho\left(\frac{\partial \mathbf{V}}{\partial t} + \mathbf{V} \cdot \nabla \mathbf{V}\right) = \rho \mathbf{g} - \nabla p + \nabla \cdot \left[\mu\left(\nabla \mathbf{V} + (\nabla \mathbf{V})^T\right)\right] + \nabla \cdot [\lambda(\nabla \cdot \mathbf{V})\mathbf{I}] \tag{3.5}$$

onde ρ é a massa específica, p é a pressão, ν é a viscosidade dinâmica do fluido, λ é a viscosidade volumétrica, $\mathbf{I}$ é a matriz identidade (3x3 em caso tridimensional) e $\mathbf{g}$ é o vetor gravidade.

Além das equações de continuidade e de Navier-Stokes, outra equação que será usada neste trabalho é a de conservação de energia, dada na forma geral como:

$$\rho \frac{Dh}{Dt} = \frac{Dp}{Dt} + \nabla \times (k \nabla T) + \mathbf{\Phi} \tag{3.6}$$

onde h é a entalpia, k é a condutividade térmica, T é a temperatura do fluido e Φ é um termo positivo conhecido como função de dissipação do escoamento.

Outra forma de expressar a transferência de calor num escoamento a partir da Equação 3.6 segundo ANJOS (2020) [18] é da forma:

$$\frac{\partial (\rho c_p T)}{\partial t} + \mathbf{V} \cdot \nabla (\rho c_p T) = \nabla \cdot k \nabla T + Q \tag{3.7}$$

onde Q é o calor fornecido ao volume de controle, em Wm^{-3}. As condições de contorno, *"c.c."*, para a transferência de calor nesse volume devem ser consideradas e impostas na equação 3.7 acima.

As demonstrações para obter as Equações 3.4, 3.5 e 3.6 podem ser encontradas em WHITE (2021) [6]. Nesses e em diversos outros problemas de mecânica de fluídos podem ser feitas diversas hipóteses que "simplificam"as equações e facilitam o seu cálculo. Por isso, neste projeto, em todas as simulações, podemos destacar as seguintes hipóteses:

- Escoamento incompressível
- Propriedades constantes ao longo do fluido
- Regime laminar
- Escoamento estável/contínuo

- Sem escorregamento nas paredes

Por conta das hipóteses assumidas acima, uma das consequências na análise do escoamento é que a massa específica, ρ, não varia no tempo:

$$\frac{\partial \rho}{\partial t} = 0 \tag{3.8}$$

Com isso, junto as demais hipóteses listadas anteriormente, as Equações 3.4, 3.5 e 3.7 podem ser reescritas como:

$$\nabla \cdot \mathbf{V} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0 \tag{3.9}$$

$$\frac{\partial \mathbf{V}}{\partial t} + \mathbf{V} \cdot \nabla \mathbf{V} = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \mathbf{V} + \mathbf{g} \tag{3.10}$$

$$\frac{\partial (T)}{\partial t} + \mathbf{V} \cdot \nabla T = \alpha \nabla^2 T + \frac{Q}{\rho c_p} \tag{3.11}$$

onde ν é a viscosidade cinemática do fluido e α a difusividade térmica do fluido.

Por fim, se considerarmos um escoamento bidimensional, como é o caso dos estudos deste trabalho, as equações de continuidade e de Navier-Stokes se tornam:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{3.12}$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \tag{3.13}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial y} + \nu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) + g \tag{3.14}$$

Apesar de não apresentado, também é possível aplicar a bidimensionalidade acima para a equação de temperatura. Além disso, é importante notar que as equações acima possuem um complexo acoplamento entre velocidades e pressão. Para mudar isso e facilitar a modelagem da solução dessas equações, uma formulação é usada e esta é discutida na Seção 3.4.

3.2 Adimensionalização das Equações

Uma vez que as equações básicas de movimento de um fluido são relativamente complexas de se analisar em geral, é de interesse no estudo desses problemas usando as equações nas suas formas mais eficientes, aumentando assim a utilidade de quaisquer soluções que for encontrada. Isso é feito a partir da adimensionalização das

equações e das condições de fronteira. O teorema π de Buckingham é um dos teoremas centrais na análise dimensional e fornece um método para computar conjuntos de parâmetros adimensionais a partir das variáveis dadas do problema.

Em estudos de dinâmica de fluidos, as quatro "variáveis" p, ρ, $\mathbf{V}$ e T (pressão, massa específica, velocidade e temperatura, respectivamente), dependerão não apenas do espaço e do tempo, mas também de diversos outros parâmetros que ocorrem nas equações básicas e condições de contorno, como viscosidade, condutividade térmica, enter outros. Presume-se que esses parâmetros sejam conhecidos a priori, a partir de dados ou relações de estado termodinâmicos.

Outro passo a ser tomado é determinar as propriedades de referência constantes apropriadas ao escoamento. Em geral, escoamentos viscosos estáveis não possuem um tempo característico próprio, então o tempo de partículas L/U é usado como referência. As variáveis sem dimensão são denotadas por um asterisco.

1. Velocidade de referência U (a velocidade do fluxo livre)

2. Comprimento de referência L (a literatura sugere o comprimento do corpo para fluxos externos ou o diâmetro hidráulico para fluxos internos)

3. Outras propriedades do fluxo livre, como p_0, ρ_0, T_0, μ_0 e k_0

$$x_i^* = \frac{x_i}{L} \qquad t^* = t\frac{U}{L} \qquad \mathbf{V}^* = \frac{\mathbf{V}}{U} \qquad p^* = \frac{p - p_0}{\rho U^2} \qquad \Phi^* = \frac{L^2}{\mu_0 U^2}\Phi$$
$$\rho^* = \frac{\rho}{\rho_0} \qquad T^* = \frac{T - T_0}{T_w - T_0} \qquad \mu^* = \frac{\mu}{\mu_0} \qquad k^* = \frac{k}{k_0} \qquad \nabla^* = L\nabla \tag{3.15}$$

Substituindo as variáveis adimensionais da Equação 3.15 nas equações de continuidade, Navier-Stokes e de transferência de calor, tem-se:

$$\nabla^* \mathbf{V}^* = \frac{\partial u^*}{\partial x^*} + \frac{\partial v^*}{\partial y^*} = 0 \tag{3.16}$$

$$\frac{\partial \mathbf{V}^*}{\partial t^*} + \mathbf{V}^* \cdot \nabla \mathbf{V}^* = -\frac{1}{\rho^*}\nabla^* p^* + \frac{1}{Re}\nabla^{*2}\mathbf{V}^* + \frac{1}{Fr^2}\hat{\mathbf{g}} \tag{3.17}$$

$$\frac{\partial T^*}{\partial t^*} + \mathbf{V}^* \cdot \nabla^* T^* = \frac{1}{Re.Pr}\nabla^{*2}T^* + \frac{Ec}{Re}Q \tag{3.18}$$

Nota-se que a equação de continuidade não possui parâmetros, enquanto que na Equação 3.17, de Navier-Stokes, possui dois novos parâmetros:

$$\text{Número de Reynolds:} \qquad Re = \frac{\rho_0 U L}{\mu_0} \tag{3.19}$$

$$\text{Número de Froude:} \qquad Fr = \frac{U^2}{gL} \tag{3.20}$$

O número de Reynolds é o parâmetro adimensional mais importante na mecânica dos fluidos. Quase todos os fenômenos de escoamentos viscosos dependem do número de Reynolds. O número de Froude, por sua vez, só é importante se houver uma superfície livre no fluxo. A equação de transferência de calor adimensional também possui dois parâmetros adicionais, além de conter o número de Reynolds:

$$\text{Número de Prandtl:} \qquad Pr = \frac{\mu_0 c_p}{k_0} \tag{3.21}$$

$$\text{Número de Eckert:} \qquad Ec = \frac{U^2}{c_p T_0} \tag{3.22}$$

Segundo WHITE (2021), o número de Prandtl é essencial em problemas de transferência de calor por convecção. Para escoamentos de alta velocidade, tanto o número de Reynolds quanto o de Prandtl são importantes, porém, para baixas velocidades, os termos com o número de Eckert são geralmente negligenciados, e a Equação 3.18 se torna:

$$\frac{\partial T^*}{\partial t^*} + \mathbf{V}^* \cdot \nabla^* T^* = \frac{1}{Re.Pr} \nabla^{*2} T^* \tag{3.23}$$

Por fim, na literatura é comum chamar o produto $Re.Pr$ como o número de Peclet, Pe. Com a análise dimensional desse número, é possível verificar que com o aumento da velocidade do escoamento (U), ou o aumento do comprimento de referência (L), ou a redução da difusividade térmica (α), o número de Peclet aumenta.

$$Pe = Re.Pr = \frac{U}{\alpha / L} \tag{3.24}$$

As Equações 3.17 e 3.23 apresentadas acima serão utilizadas no Capítulo 4, onde é apresentado e explicado o Método de Elementos Finitos (MEF). Elas são extremamente importantes pois, entre outros fatores, facilitam a modelagem dos escoamentos a partir dos principais números adimensionais, como Reynolds (**Re**) e Prandtl (**Pr**).

3.3 Vorticidade e Função Corrente

Nesta seção, são definidos dois conceitos primordiais para a análise que é feita neste trabalho: a vorticidade e a função corrente de um escoamento. Antes de introduzir essas definições, é importante lembrar das definições vetoriais de divergente

e rotacional. Com base nas Equações 3.2 e 3.3, para o vetor de velocidades $\mathbf{V}$, o divergente e o rotacional podem ser definidos como:

$$div\mathbf{V} = \nabla \cdot \mathbf{V} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \tag{3.25}$$

$$rot\mathbf{V} = \nabla \times \mathbf{V} = \left(\frac{\partial w}{\partial y} - \frac{\partial v}{\partial z}, \frac{\partial u}{\partial z} - \frac{\partial w}{\partial x}, \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right) \tag{3.26}$$

Para os casos de estudo apresentados neste trabalho, considerando escoamentos bidimensionais de fluidos newtonianos incompressíveis, tem-se que $w = 0$ e que todas as derivadas $\partial/\partial z$ também valem zero. Assim, as Equações 3.25 e 3.26 são reescritas como:

$$div\mathbf{V} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \tag{3.27}$$

$$rot\mathbf{V} = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\hat{\mathbf{k}} \tag{3.28}$$

3.3.1 Vorticidade

De acordo com WHITE (2021) [6], define-se a quantidade chamada vetor vorticidade ω para representar o dobro da velocidade angular do escoamento. Apesar de não ser geralmente uma variável primária para analisar a dinâmica de um fluido, como a velocidade, a vorticidade é uma importante medida dos efeitos de rotação de um fluxo. Dessa forma, ela é definida como:

$$\omega = (\omega_x, \omega_y, \omega_z) = \nabla \times \mathbf{V} \tag{3.29}$$

De forma geral, em todos os fluxos viscosos, a vorticidade está geralmente presente, sendo gerada pelo movimento relativo perto de paredes sólidas, onde a velocidade é zero. Por outro lado, se o número de Reynolds for grande o suficiente, a vorticidade é deslocada ao longo do escoamento e permanece perto da parede. Um escoamento com $\omega = 0$ é chamado de escoamento irrotacional. Para ilustrar a influência do número de Reynolds, na Figura 3.2 são apresentados escoamentos ao redor de um cilindro para vários valores de Reynolds.

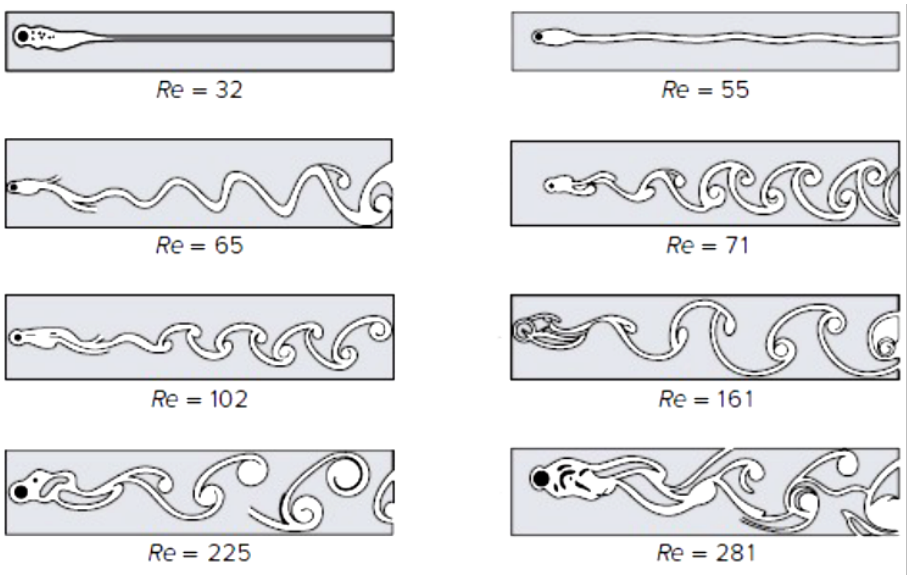


Figura 3.2: Efeitos do número de Reynolds em escoamento ao redor de um cilindro. Adaptado de WHITE (2021).

Considerando um escoamento bidimensional no plano xy, como são os casos deste projeto, o vetor vorticidade ω se resume a sua componente na direção z:

$$\omega_z = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \tag{3.30}$$

3.3.2 Função Corrente

Outra quantidade importante para o presente estudo é a função corrente. Ela é definida tanto para escoamentos compressíveis como para incompressíveis. Para cada caso respectivamente, ela é definida como [6]:

$$\rho u = \partial\psi/\partial y \quad \text{e} \quad \rho v = \partial\psi/\partial x \tag{3.31}$$

$$u = \partial\psi/\partial y \quad \text{e} \quad v = \partial\psi/\partial x \tag{3.32}$$

Matematicamente, a função corrente ψ é tangente ao vetor de velocidades $\mathbf{V}$ e possui um significado físico para um dos casos acima. Para um escoamento compressível, tem-se:

$$\begin{aligned} d\psi &= \frac{\partial\psi}{\partial x}dx + \frac{\partial\psi}{\partial y}dy = -\rho v\, dx + \rho u\, dy \\ &= \rho\mathbf{V} \cdot d\mathbf{A} = d\dot{m} \end{aligned} \tag{3.33}$$

Assim, linhas de corrente constantes ($d\psi = 0$) são linhas onde não há fluxo de massa ($\dot{m} = 0$). Além disso, é possível identificar que a diferença no valor entre duas linhas de função corrente constantes é numericamente igual a diferença entre o fluxo de massa entre essas linhas. O mesmo vale para a definição da função corrente para escoamentos incompressíveis, porém, sem a massa específica ρ, o significado físico da função corrente passa de fluxo mássico para fluxo volumétrico ($d\psi = d\dot{V}$).

3.4 Formulação Corrente Vorticidade

A estrutura do modelo elaborado neste trabalho se sustenta em algo chamado de Formulação Corrente Vorticidade, onde são combinados os conceitos de função corrente ψ e vorticidade ω. Primeiro, é necessário avaliar a vorticidade na equação de Navier-Stokes, mas, para isso, é preciso fazer uso de duas identidades vetoriais. Para simplificação, as identidades a seguir são escritas em termos de vorticidade, levando em conta a Equação 3.29.

$$(\mathbf{V} \cdot \nabla)\mathbf{V} = \nabla\frac{\mathbf{V}^2}{2} - \mathbf{V} \times \omega \tag{3.34}$$

$$\nabla^2\mathbf{V} = \nabla(\nabla \cdot \mathbf{V}) - \nabla \times \omega \tag{3.35}$$

Nota-se que, considerando a equação de Continuidade, $\nabla \cdot \mathbf{V} = 0$, o primeiro termo do lado direito da Equação 3.35 é zero. Assim, se substituirmos as Equações 3.34 e 3.35 na equação de Navier-Stokes (3.10), reorganizando os termos, e tomando a gravidade como $\mathbf{g} = -g\hat{\mathbf{k}}$, tem-se que:

$$\rho\frac{\partial\mathbf{V}}{\partial t} + \nabla\Big(p + \frac{1}{2}\rho\mathbf{V}^2 + \rho g z\Big) = \rho\mathbf{V} \times \omega - \mu\nabla \times \omega \tag{3.36}$$

É interessante observar na equação acima que se o escoamento viscoso for irrotacional, o que se encontra do lado esquerdo é a famosa Equação de Bernoulli. De toda forma, se for tomado o rotacional da Equação 3.36 acima, tem-se a Equação de Transporte de Vorticidade:

$$\frac{\partial\omega}{\partial t} + \mathbf{V} \cdot \nabla\omega = (\omega \cdot \nabla)\mathbf{V} + \nu\nabla^2\omega \tag{3.37}$$

Nessa equação, se considerado um escoamento viscoso bidimensional, o primeiro termo do lado direito é zero porque o vetor velocidade $\mathbf{V}$ só possui componentes nas

direções x e y, enquanto o vetor de vorticidade ω só possui componente no eixo z (ou seja, ω e $\mathbf{V}$ são vetores perpendiculares um ao outro). Assim, o Transporte de Vorticidade se torna:

$$\frac{\partial \omega}{\partial t} + \mathbf{V} \cdot \nabla \omega = \nu \nabla^2 \omega \tag{3.38}$$

Além disso, se combinadas os conceitos de vorticidade, na Equação 3.30, função corrente, na Equação 3.32, e a Equação de Continuidade (3.12), tem-se que:

$$\begin{aligned} \omega_z &= \left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right) \\ &= \left(-\frac{\partial}{\partial x}\frac{\partial \psi}{\partial x} - \frac{\partial}{\partial y}\frac{\partial \psi}{\partial y} \right) \\ &= -\nabla^2 \psi \end{aligned} \tag{3.39}$$

Dessa forma, reduzimos o problema de mecânica dos fluidos para escoamentos bidimensionais incompressíveis às Equações 3.38 e 3.39, reduzindo o número de variáveis de três (p, u e v) para duas (ω e ψ). Além disso, é importante fazer algumas considerações:

- Com a formulação corrente vorticidade, a pressão foi "desacoplada" da equação de Navier-Stokes, tornado-a menos complexa. Porém, neste projeto não será apresentado a solução para a pressão por conta da falta de informações para as condições de contorno da pressão nos domínios estudados.

- Sobre as condições de contorno dos problemas, estas serão melhor apresentadas no próximo Capítulo 4, onde os domínios serão introduzidos mais detalhadamente.

- Para estudo de escoamentos viscosos bidimensionais de fluidos incompressíveis - onde as propriedades μ e ρ são constantes ao longo do fluxo -, como são os casos deste projeto, a análise das equações de energia e de conservação de quantidade de movimento (Navier-Stokes) são desacopladas. Ou seja, a distribuição da temperatura $\mathbf{T}$ como apresentado na Equação 3.11 pode ser feito depois da análise das velocidades do escoamento.

Capítulo 4

Modelagem da Simulação

Neste capítulo é apresentado a discretização dos problemas de estudo e seus domínios, assim como a teoria e aplicação do Método de Elementos Finitos nessas situações, a discretização espacial-temporal de Taylor-Galerkin [19] nos elementos da malha de cada problema.

4.1 Método de Elementos Finitos

Como introduzido anteriormente, o método de elementos finitos é um método numérico separado em algumas etapas. Resumidamente, o método começa pela divisão do domínio em pontos e em partes, chamados de nós e elementos, respectivamente. Depois disso, alguns cálculos usando as equações de governo de cada problema são feitos em cada elemento e nó para, por fim, obter a solução para todo o domínio. A Figura 4.1 a seguir ilustra os principais passos do MEF.

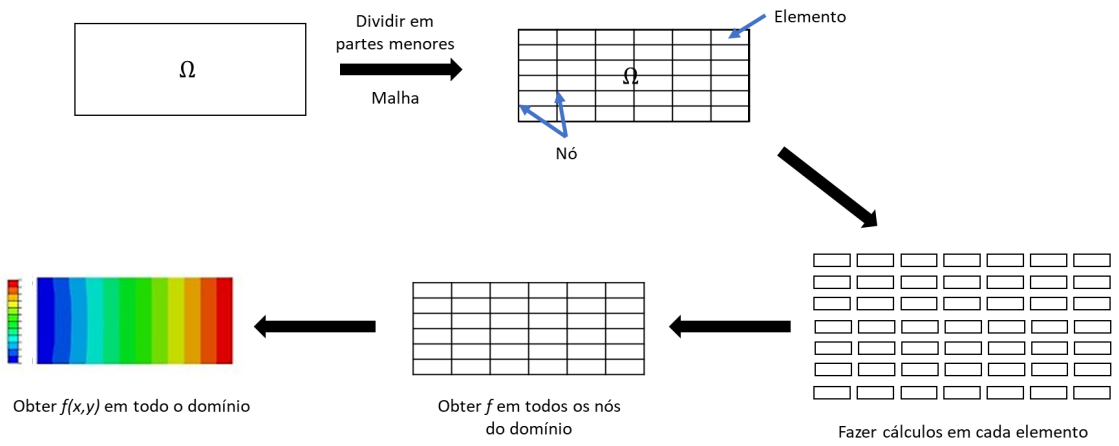


Figura 4.1: Esquema de solução utilizando MEF.

Após a discretização do domínio em elementos, que será detalhado na Seção 4.2, o primeiro passo do método é transformar as equações de governo da sua forma forte para a sua forma fraca. De acordo com FISH (2007) [12], a forma forte de uma equação é o seu formato diferencial original (EDP), e a forma fraca é quando se multiplica a forma forte por uma função peso *w(x,y)* e integra-se ao longo do domínio.

Exemplo de equação na forma forte:
$$\nabla^2 \mathbf{u} + \nabla \mathbf{u} + \mathbf{u} = 0 \tag{4.1}$$

Exemplo de equação na forma fraca:
$$\int_\Omega w[\nabla^2 \mathbf{u} + \nabla \mathbf{u} + \mathbf{u}] = 0 \tag{4.2}$$

Com a equação na forma fraca, o segundo passo é reduzir a ordem do termo de derivada segunda através do uso do Teorema de Green. Para campos vetoriais, o teorema de Green pode ser reescrito como:

$$\int_\Omega w \,.\, \nabla^2 \mathbf{u} \, \mathrm{d}\Omega = \int_\Gamma (w \,.\, \nabla \mathbf{u}) \,.\, \mathbf{n} \, \mathrm{d}\Gamma - \int_\Omega \nabla \mathbf{u} \,.\, \nabla w \, \mathrm{d}\Omega \tag{4.3}$$

Nesta etapa, é importante analisar as condições de contorno do problema. Segundo ANJOS (2020) [18], na mecânica dos fluidos e transferência de calor, existem três grupos de condição de contorno: *Dirichlet*, *Neumann* e *Robin*. Um problema pode possuir desde um tipo de condição para todo o seu contorno, ou uma combinação desses tipos.

- Dirichlet: o valor da variável de interesse da EDP é conhecido.
- Neumann: o valor da derivada da variável de interesse da EDP é conhecido.

- Robin: quando há combinação das condições do tipo Dirichlet e Neumann.

O tipo de condição de contorno definirá no MEF o valor da função peso *w(x,y)* no contorno. Por exemplo, para o caso de condição do tipo *Dirichlet*, $w = 0$ no contorno Γ_D.

O próximo passo do método é reescrever as funções/vetores (**u** e **w** no exemplo) como uma série de soma infinita, porém truncada até a quantidade de elementos da malha do domínio. Ou seja,

$$\mathbf{w} = \sum_{i=0}^{\infty} N_i(x,y)w_i \approx \sum_{i=0}^{n} N_i(x,y)w_i \tag{4.4}$$

$$\mathbf{u} = \sum_{j=0}^{\infty} N_j(x,y)u_j \approx \sum_{j=0}^{n} N_j(x,y)u_j \tag{4.5}$$

onde N_i e N_j são as funções de forma, segundo FISH (2007) [12]. Essas funções podem ser lineares, quadráticas, cúbicas, entre outras. Para simplificação, neste trabalho faz uso de funções de forma lineares e do método de Galerkin, em que as funções de forma são iguais.

$$N_i(x,y) = N_j(x,y) \tag{4.6}$$

Em seguida, o penúltimo passo é substituir a aproximação das funções/vetores em séries de soma (Equações 4.4 e 4.5) na forma fraca (Equação 4.2), e obter as formas matriciais dos termos de função de forma nas integrais sobre o domínio. Essas matrizes são chamadas de matrizes globais e são definidas como:

Integral sem derivadas:

$$\int_{\Omega} N_i N_j \, \mathrm{d}\Omega \quad \longrightarrow \quad \text{Matriz de massa} \quad \mathbf{M} \tag{4.7}$$

Integral com um termo sem derivada e um termo com gradiente:

$$\int_{\Omega} N_j \nabla N_j \, \mathrm{d}\Omega \quad \longrightarrow \quad \text{Matriz do gradiente} \quad \mathbf{G} \tag{4.8}$$

Integral com multiplicação escalar entre gradiente:

$$\int_{\Omega} \nabla N_i \cdot \nabla N_j \, \mathrm{d}\Omega \quad \longrightarrow \quad \text{Matriz de rigidez (ou viscosa)} \quad \mathbf{K} \tag{4.9}$$

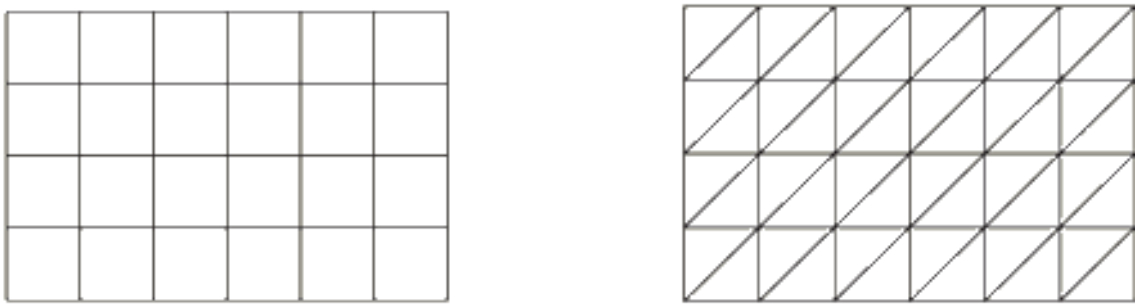
A construção das matrizes **M**, **G** e **K** são feitas através do processo de "assembling" [18], isto é, cada matriz global é construída a partir de matrizes criadas para de cada elemento da malha ($\mathbf{m}^e$, $\mathbf{g}^e$ e $\mathbf{k}^e$).

Por fim, o próximo e último procedimento é escrever a equação global utilizando as matrizes globais e estabelecer como os termos da equação serão avaliados no tempo (seja em n ou n+1). Em geral, utiliza-se um esquema de diferença finita para discretizar uma função/vetor no tempo no MEF. Por exemplo, para a vorticidade ω_z, tem-se:

$$\frac{\partial\, \omega_z}{\partial t} \approx \frac{\omega_z^{n+1} - \omega_z^n}{\Delta t} \tag{4.10}$$

4.2 Discretização do Domínio e seus Elementos

A discretização do domínio de um problema se resume em dividi-lo em partes menores, constituídas por pontos (nós) e elementos no que a literatura chama de "malha" (tradução do termo em inglês, "*mesh*"). No método de elementos finitos, os cálculos são baseados em cada elemento, que podem ser triangulares, quadriláteros ou outras formas geométricas. É possível também a combinação de diferentes formas de elementos em uma mesma malha. Além disso, uma malha é dita "estruturada" caso esta seja feita a partir de elementos iguais, seja qual for o formato deste. Caso contrário, ela é chamada de malha "não-estruturada". Na Figura 4.2 são ilustradas duas malhas, uma estruturada quadrada (com elementos quadrados/retangulares) e outra triangular (com elementos triangulares).



(a) Elemento quadrangular (b) Elemento triangular

Figura 4.2: Malha de elementos finitos com 24 quadriláteros e 35 nós; malha de elementos finitos com 48 triângulos e 35 nós.

Como introduzido no Capítulo 2, a discretização do domínios em malhas pode se tornar consideravelmente complexa. Para facilitar este processo, existem ferramentas/softwares que geram malhas a partir de desenhos ou "scripts" em alguma

linguagem de programação. Neste trabalho, foi utilizado o software "*GMSH*" [20] para gerar todas as malhas dos casos de validação e de estudo, como a Figura 4.3 abaixo.

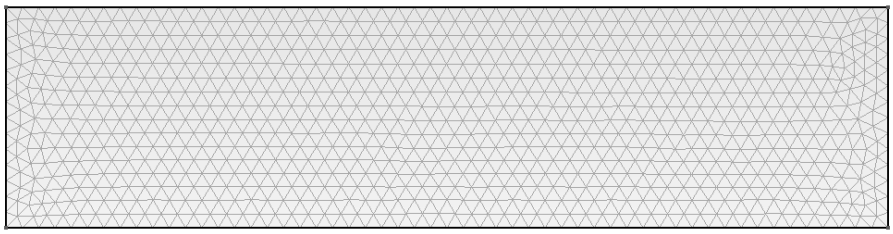


Figura 4.3: Exemplo de malha de triângulos utilizada no método de elementos finitos.

Na geração da malha de cada problema, a escolha do elemento de malha é de escolha do usuário. Essa escolha definirá parâmetros e variáveis do algoritmo do método de elementos finitos. Neste trabalho, os domínios são divididos em elementos triangulares para o MEF. Segundo FISH (2007) [12], a escolha de triângulos é recomendada por conta das curvas em partes do interior e contorno de alguns domínios estudados, partes essas que não são bem descritas por elementos retangulares. Na Figura 4.4 abaixo, é esquematizado como é definido cada elemento de malha.

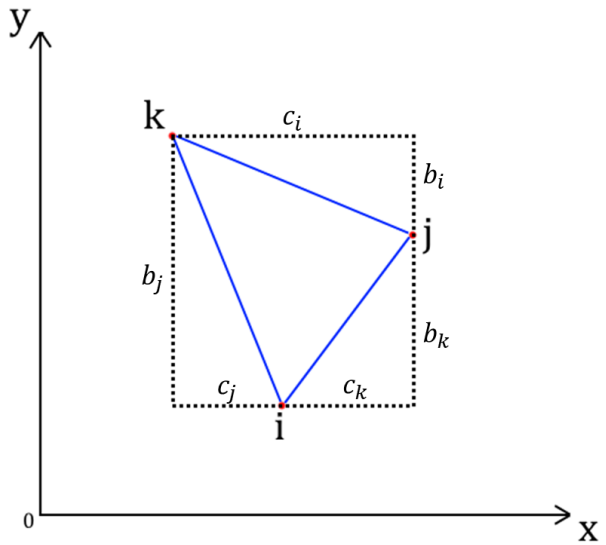


Figura 4.4: Elemento triangular de malha no MEF.

Cada elemento de malha é dividido em 3 nós i, j e k, cada qual com coordenadas (x_i, y_i), (x_j, y_j) e (x_k, y_k), respectivamente. É importante manter a orientação destes

nós de cada elemento de forma uniforme na malha. Aqui, a orientação dos nós é no sentido anti-horário. Além disso, na Figura 4.4 é possível observar também duas variáveis de forma essenciais para o MEF: b_i e c_i. Além delas, para cada elemento há também uma variável a_i. Essas variáveis são definidas por FISH (2007) [12] como:

$$a_i = x_j y_k - x_k y_j \qquad b_i = y_j - y_k \qquad c_i = x_k - x_j \tag{4.11}$$

$$a_j = x_k y_i - x_i y_k \qquad b_j = y_k - y_i \qquad c_j = x_i - x_k \tag{4.12}$$

$$a_k = x_i y_j - x_j y_i \qquad b_k = y_i - y_j \qquad c_k = x_j - x_i \tag{4.13}$$

Não obstante disso, outro fator importante para o MEF é determinar a área de cada elemento. Para tal, é possível obter essa área algebricamente através da Equação 4.14:

$$\mathbf{A} = \frac{1}{2}\det \begin{bmatrix} 1 & x_i & y_i \\ 1 & x_j & y_j \\ 1 & x_k & y_k \end{bmatrix} \tag{4.14}$$

Ainda segundo FISH (2007) [12], depois de escolhido o formato para o elemento de malha, neste caso triangular, é possível definir um vetor de funções de forma $\mathbf{N}$, como apresetado nas Equações 4.4 e 4.5, como:

$$\mathbf{N}(x, y) = [N_i \; N_j \; N_k] \tag{4.15}$$

onde cada componente é definido como:

$$N_i = \frac{1}{2A}(a_i + b_i x + c_i y) \tag{4.16}$$

$$N_j = \frac{1}{2A}(a_j + b_j x + c_j y) \tag{4.17}$$

$$N_k = \frac{1}{2A}(a_k + b_k x + c_k y) \tag{4.18}$$

Assim, determinadas as funções de forma N, para construir as matrizes $\mathbf{M}$, $\mathbf{G}$ e $\mathbf{K}$ pelo processo de "assembling". Para tal, é necessário primeiro determinar o gradiente da função de forma, $\nabla\mathbf{N}$, como é possível observar nas integrais das Equações 4.8 e 4.9. Segundo FISH (2007) [12], para cada elemento da malha, tem-se que:

$$\nabla\mathbf{N} = B\ \mathbf{N} = \begin{bmatrix} g_x^e \\ g_y^e \end{bmatrix} \tag{4.19}$$

onde B é uma matriz, g_x e g_y são matrizes gradiente de cada elemento. Cada uma dessas matrizes, denominadas "matrizes elementares", é calculada a cada nó do

elemento. Exitem outras duas matrizes definidas na literatura: a matriz elementar de massa, $\mathbf{m}^e$, e a matriz elementar de rigidez, $\mathbf{k}^e$. Elas são calculadas a partir das variáveis a_i, b_i, c_i, e da área $\mathbf{A}$ da seguinte forma [12]:

$$\mathbf{g_x^e} = \frac{1}{6}\begin{bmatrix} b_i & b_j & b_k \\ b_i & b_j & b_k \\ b_i & b_j & b_k \end{bmatrix} \qquad \mathbf{g_y^e} = \frac{1}{6}\begin{bmatrix} c_i & c_j & c_k \\ c_i & c_j & c_k \\ c_i & c_j & c_k \end{bmatrix} \tag{4.20}$$

$$B = \frac{1}{2A}\begin{bmatrix} b_i & b_j & b_k \\ c_i & c_j & c_k \end{bmatrix} \qquad \mathbf{m^e} = \frac{A}{12}\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} \tag{4.21}$$

Diferentemente das anteriores, o cálculo da matriz elementar $\mathbf{k}^e$ depende de algumas hipóteses. De forma geral, ela é dada por:

$$k^e = \int_\Omega \mathbf{B^T D B} d\Omega \tag{4.22}$$

$$k^e = A\mathbf{B^T D B} = k_x\mathbf{k_x^e} + k_y\mathbf{k_y^e} \tag{4.23}$$

onde D é a matriz de coeficientes do laplaciano, A é a área do elemento, k_x e k_y são os coeficientes do laplaciano, e $\mathbf{k}_x^e$ e $\mathbf{k}_x^e$ são matrizes de rigidez elementares nas direções x e y, respectivamente. A matriz D é definida por:

$$\mathbf{D} = \begin{bmatrix} k_x & 0 \\ 0 & k_y \end{bmatrix} \tag{4.24}$$

Os coeficientes k_x e k_y dependem do material (neste caso, do fluido de escoamento). Um material é dito ter simetria isotrópica [12] se as propriedades do material são as mesmas em qualquer sistema de coordenadas. Ou seja, quando $k_x = k_y = k$. Para simplificação, neste trabalho se considera $k = 1$, e, por tanto, a matriz D se torna uma matriz identidade. Logo, a Equação 4.23 pode ser reescrita como:

$$k^e = A\mathbf{B^T I B} = \mathbf{k_x^e} + \mathbf{k_y^e} \tag{4.25}$$

onde

$$\mathbf{k_x^e} = \frac{1}{4A}\begin{bmatrix} b_i^2 & b_ib_j & b_ib_k \\ b_jb_i & b_j^2 & b_jb_k \\ b_kb_i & b_kb_j & b_k^2 \end{bmatrix} \qquad \mathbf{k_y^e} = \frac{1}{4A}\begin{bmatrix} c_i^2 & c_ic_j & c_ic_k \\ c_jc_i & c_j^2 & c_jc_k \\ c_kc_i & c_kc_j & c_k^2 \end{bmatrix} \tag{4.26}$$

4.3 MEF na Formulação Corrente Vorticidade

Como apresentado na Seção 4.1, o primeiro passo do método de elementos finitos é passar as equações de governo da forma forte para a forma fraca. Primeiro, aplicando o MEF na Equação 3.38 de Transporte de Vorticidade, e utilizando o Teorema de Green pra reduzir a ordem do laplaciano, tem-se:

$$\int_\Omega w(x,y)\left(\frac{\partial \omega_z}{\partial t} + \mathbf{V} \cdot \nabla\omega_z - \nu\nabla^2\omega_z\right)\mathrm{d}\Omega = 0 \tag{4.27}$$

$$\int_\Omega w\frac{\partial \omega_z}{\partial t}\mathrm{d}\Omega + \int_\Omega w\mathbf{V} \cdot \nabla\omega_z\mathrm{d}\Omega - \int_\Omega w\nu\nabla^2\omega_z\mathrm{d}\Omega = 0 \tag{4.28}$$

$$\int_\Omega w\frac{\partial \omega_z}{\partial t}\mathrm{d}\Omega + \int_\Omega w\mathbf{V} \cdot \nabla\omega_z\mathrm{d}\Omega - \int_\Gamma w\nu\nabla\omega_z\mathrm{d}\Gamma + \int_\Omega \nabla w\nu\nabla\omega_z\mathrm{d}\Omega = 0 \tag{4.29}$$

A seguir, a função de peso w e a vorticidade ω_z são aproximadas por séries de somas para todos os n nós na malha.

$$\omega_z \approx \sum_{i=0}^{n} N_i(x,y)\omega_{z_i} \tag{4.30}$$

$$w \approx \sum_{j=0}^{n} N_j(x,y)w_j \tag{4.31}$$

Nota-se ainda que tanto w quanto ω_z são constantes no espaço (em cada nó), mas variam no tempo. Ou seja, $w = w(t)$ e $\omega_z = \omega_z(t)$. Por outro lado, as funções de forma N_i e N_j não dependem do tempo. Dessa forma, se substituirmos os somatórios em i e j da Equação 4.29 por somatórios a cada elemento e, retirarmos a vorticidade ω_z da integrais espaciais e dividirmos toda a equação por w_j, teremos:

$$\begin{aligned}\sum_e \frac{\partial \omega_{z_i}}{\partial t}\int_\Omega N_iN_j\mathrm{d}\Omega + \sum_e \omega_{z_i}\int_\Omega \mathbf{V} \cdot (N_j\nabla N_i)\mathrm{d}\Omega \\ - \sum_e \omega_{z_i}\int_\Gamma N_j(\nu\nabla N_i)\mathrm{d}\Gamma + \sum_e \omega_{z_i}\int_\Omega \nabla N_j \cdot (\nu\nabla N_i)\mathrm{d}\Omega = 0\end{aligned} \tag{4.32}$$

O próximo passo consiste em substituir as Equações 4.7 a 4.10 na Equação 4.32 acima, para obter:

$$m^e_{i,j}\left(\frac{\omega^{n+1}_{z_i} - \omega^n_{z_i}}{\Delta t}\right) + \mathbf{V} \cdot g^e_{i,j}\,\omega_{z_i} + \nu k^e_{i,j}\,\omega_{z_i} = 0 \tag{4.33}$$

em que por conta da condição do tipo Dirichlet para o Tranporte de Vorticidade, a função w_j no contorno é nula.

Após o "assembling" das matrizes $\mathbf{M}$, $\mathbf{G}$ e $\mathbf{K}$, chega-se a equação global do problema:

$$\mathbf{M}\left(\frac{\omega^{n+1}_{z_i} - \omega^n_{z_i}}{\Delta t}\right) + \mathbf{V} \cdot \mathbf{G}\,\omega_{z_i} + \nu\mathbf{K}\,\omega_{z_i} = 0 \tag{4.34}$$

onde o termo de convecção ($\mathbf{V}\,.\,\mathbf{G}$) pode ser reescrito como:

$$\mathbf{V}\,.\,\mathbf{G}\;\omega_{z_i} = u\mathbf{I}(\mathbf{G}_x\;\omega_{z_i}) + v\mathbf{I}(\mathbf{G}_y\;\omega_{z_i}) \tag{4.35}$$

Por fim, como comentado ao final da Seção 4.1, é possível ter esquemas de avaliação temporal para os termos convectivo e viscoso ($\nu\mathbf{K}$): implícito e explícito. No caso da equação de Transporte de Vorticidade, se o termo está multiplicando ω_z^{n+1}, ele é dito "implícito" e, por outro lado, se o termo estiver multiplicando ω_z^n, este é dito "explícito". Mantendo-se o termo viscoso *sempre* implícito, os esquemas temporais para o termo de convecção são da forma:

$$\text{Termo de convecção implícito:} \quad \left(\frac{\mathbf{M}}{\Delta t} + \nu\mathbf{K} - \mathbf{V}\,.\,\mathbf{G}\right)\omega_z^{n+1} = \frac{\mathbf{M}}{\Delta t}\omega_z^n \tag{4.36}$$

$$\text{Termo de convecção explícito:} \quad \left(\frac{\mathbf{M}}{\Delta t} + \nu\mathbf{K}\right)\omega_z^{n+1} = \left(\frac{\mathbf{M}}{\Delta t} - \mathbf{V}\,.\,\mathbf{G}\right)\omega_z^n \tag{4.37}$$

De acordo com ANJOS (2020) [18], a literatura indica o uso do esquema implícito ao invés do explícito por conta de limitações que este último provoca no intervalo de tempo Δt. Neste trabalho, em todos os casos validados e estudados, é utilizado o esquema implícito.

Não obstante disso, outra equação de governo dos problemas de mecânica dos fluidos é a Equação 3.39 da formulação corrente vorticidade. O processo do MEF não é diferente para essa equação, que pode ser escrita na forma matricial global da forma:

$$\mathbf{K}\psi = \mathbf{M}\omega_z \tag{4.38}$$

Os domínios apresentados neste projeto em que a formulação função corrente-vorticidade é aplicada, em sua maioria, possuem condição do tipo Dirichlet em todo o contorno. Por conta disso, é necessário incluir as condições de contorno (c.c.) nas equações de governo, quando houver. Por exemplo, a vorticidade possui condições de contorno do tipo Dirchlet que, diferente dos demais vetores e escalares, é calculada a cada passo de tempo no MEF. Depois de calculado a vorticidade no contorno - utilizando a Equação 4.36 no esquema implícito -, estes valores são impostos na solução da equação de vorticidade.

Como apresentado no Capítulo 3, a formulação função corrente-vorticidade é acompanhada de funções auxiliares. As equações auxiliares na forma matricial são

definidas como:

$$\mathbf{M}u = \mathbf{G_y}\psi \tag{4.39}$$

$$\mathbf{M}v = -\mathbf{G_x}\psi \tag{4.40}$$

$$\mathbf{M}\omega_z = \mathbf{G_x}u - \mathbf{G_y}v \tag{4.41}$$

Analogamente a formulação vorticidade-função corrente, a Equação 3.11 de transferência de calor também pode e deve ser inserida no método de elementos finitos. Assim como na equação de vorticidade, a forma matricial da equação de transferência de calor possui esquemas temporais implícito e explícito. Mantendo-se o termo difusivo ($\alpha\mathbf{K}$) implícito, os esquemas para o termo convectivo são dados por:

$$\text{Esquema implícito:} \qquad \left(\frac{\mathbf{M}}{\Delta t} + \alpha\mathbf{K} - \mathbf{V}\,\mathbf{.}\,\mathbf{G}\right)\mathbf{T}^{n+1} = \frac{\mathbf{M}}{\Delta t}\,\mathbf{T}^{n} \tag{4.42}$$

$$\text{Esquema explícito:} \qquad \left(\frac{\mathbf{M}}{\Delta t} + \alpha\mathbf{K}\right)\mathbf{T}^{n+1} = \left(\frac{\mathbf{M}}{\Delta t} - \mathbf{V}\,\mathbf{.}\,\mathbf{G}\right)\mathbf{T}^{n} \tag{4.43}$$

Além disso, a partir das Equações 4.42 e 4.43, é possível determinar a transferência de calor na ausência de escoamento. Nesse caso, os termos que envolvem o vetor velocidade $\mathbf{V}$ desaparecem, restando apenas termos dependentes apenas da temperatura. Segundo ANJOS (2020) [18], temos então:

$$\text{Esquema implícito:} \qquad \left(\frac{\mathbf{M}}{\Delta t} + \mathbf{K}\right)\mathbf{T}^{n+1} = \frac{\mathbf{M}}{\Delta t}\,\mathbf{T}^{n} + \mathbf{M}\mathbf{Q}_{i,j} \tag{4.44}$$

$$\text{Esquema explícito:} \qquad \left(\frac{\mathbf{M}}{\Delta t}\right)\mathbf{T}^{n+1} = \left(\frac{\mathbf{M}}{\Delta t} - \mathbf{K}\right)\mathbf{T}^{n} + \mathbf{M}\mathbf{Q}_{i,j} \tag{4.45}$$

O algoritmo para determinar a solução de um problema através do MEF está resumida na lista a seguir [18].

1. Criar a malha, os vetores de coordenadas e o vetor de conectividade (**IEN**)
2. Criar os vetores u, v, ω_z e ψ e ajustar suas condições de contorno
3. Início do processo iterativo para cada passo de tempo Δt:
   (a) Montar as matrizes globais $\mathbf{K}_x$, $\mathbf{K}_y$, $\mathbf{K}_{xy}$, $\mathbf{K}_{est}$, $\mathbf{M}$, $\mathbf{G}_x$ e $\mathbf{G}_y$
   (b) Calcular a vorticidade (Equação 4.41)
   (c) Resolver o Transporte de Vorticidade (Equação 4.36)
   (d) Resolver a Função Corrente-Vorticidade (Equação 4.38)

(e) Calcular velocidades (Equações 4.39 e 4.40)

(f) Resolver a Transferência de Calor (Equação 4.42)

O vetor de conectividade **IEN** serve para definir quais nós definem cada elemento. Já as matrizes globais $\mathbf{K}_{xy}$ e $\mathbf{K}_{est}$ serão melhor detalhadas na Seção 4.4.

4.4 Método de Taylor-Galerkin

A formulação corrente-vorticidade possui suas vantagens, como desacoplar a pressão das equações de Navier-Stokes. Todavia, ela traz consigo também algumas limitações. Uma delas é que, segundo a literatura, como apresentado em ANJOS (2020) [18], o termo convectivo $\mathbf{V} \cdot \mathbf{G}$ presente na equação de Transporte de Vorticidade pode gerar oscilações no MEF para números de Reynolds elevados. Por conta disso, quando aplicado nessas situações, o método de Galerkin deve ser corrigido. Algumas das técnicas para correção no MEF indicados pela literatura são:

- Método de Taylor-Galerkin (do inglês, "characteristic Galerkin")
- Método SUPG (do inglês, "streamline upwind Petrov-Galerkin")
- Método semi-langrangeano
- Método de Captura de Choque (do inglês, "shock-capturing")
- Método CBS (do inglês, "characteristic-based split")

Neste trabalho, foi escolhido o método de Taylor-Galerkin, descrito em DONEA (1984) [19]. O método alivia a geração de oscilações expandindo termos da equação de Transporte de Vorticidade. Para uma discretização de primeira ordem. como na equação, é feita uma aproximação através de uma reta, representada na Figura 4.5 abaixo.

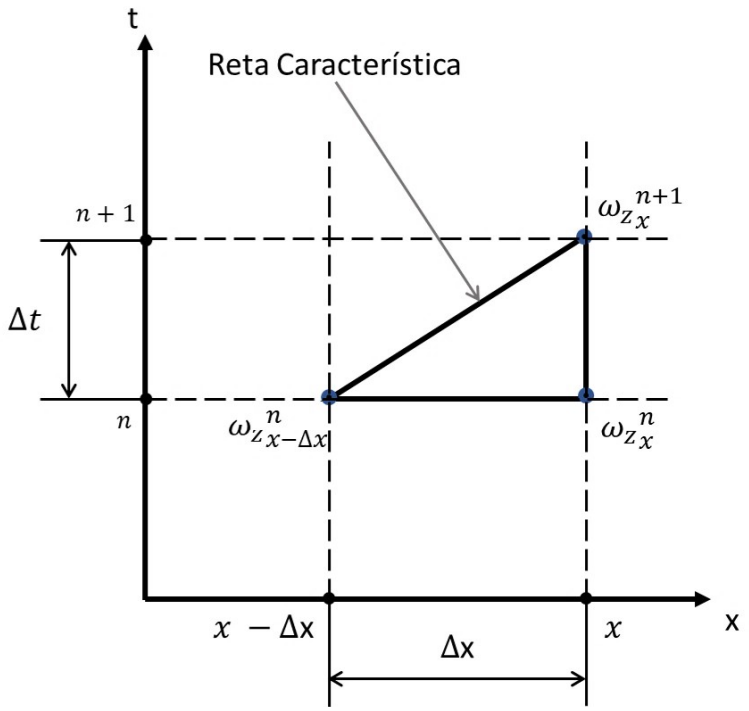


Figura 4.5: Esboço do método de Taylor-Galerkin.

De acordo com DONEA (1984) [19], o método faz expansões em séries de Taylor para os termos da equação de transporte, da forma:

$$\omega^n_{z_{x-\Delta x}} = \omega^n_{z_x} - \frac{\partial \omega^n_{z_x}}{\partial x}\frac{\Delta t}{1!} + \frac{\partial^2 \omega^n_{z_x}}{\partial x^2}\frac{\Delta t^2}{2!} - \dots \tag{4.46}$$

$$\begin{aligned}\frac{\partial}{\partial x}\left(\nu\frac{\partial \omega_z}{\partial x}\right)^n_{x-\Delta x} = \frac{\partial}{\partial x}\left(\nu\frac{\partial \omega_z}{\partial x}\right)^n_x - \frac{\partial}{\partial x}\left[\frac{\partial}{\partial x}\left(\nu\frac{\partial \omega_z}{\partial x}\right)^n_x\right]\frac{\Delta x}{1!} \\ + \frac{\partial^2}{\partial x^2}\left[\frac{\partial}{\partial x}\left(\nu\frac{\partial \omega_z}{\partial x}\right)^n_x\right]\frac{\Delta x^2}{2!} - \dots\end{aligned} \tag{4.47}$$

Com o método de Taylor-Galerkin e sua discretização, dois termos de estabilização são adicionados à equação de transporte resultante, sendo um para a direção x e outro para a direção y.

$$\begin{aligned}\frac{\omega_z^{n+1} - \omega_z^n}{\Delta t} = &- \mathbf{V}\,.\,\mathbf{G}\omega_z^n + \nu\nabla^2\omega_z^n + \\ &u\frac{\Delta t}{2}\frac{\partial}{\partial x}\left[u\frac{\partial \omega_z}{\partial x} + v\frac{\partial \omega_z}{\partial y}\right]^n + v\frac{\Delta t}{2}\frac{\partial}{\partial y}\left[u\frac{\partial \omega_z}{\partial x} + v\frac{\partial \omega_z}{\partial y}\right]^n\end{aligned} \tag{4.48}$$

Analogamente, o método de estabilização também deve ser aplicado na equação de Transferência de Calor, uma vez que esta equação também possui um termo

convectivo $\mathbf{V} \cdot \mathbf{G}\mathrm{T}$. A equação de calor resultante é dada por:

$$\begin{aligned}\frac{\mathbf{T}^{n+1}-\mathbf{T}^{n}}{\Delta t} = &-\mathbf{V} \cdot \mathbf{G}\mathbf{T}^{n}+\alpha \nabla^{2}\mathbf{T}^{n}+ \\ &u\frac{\Delta t}{2}\frac{\partial}{\partial x}\left[u\frac{\partial \mathbf{T}}{\partial x}+v\frac{\partial \mathbf{T}}{\partial y}\right]^{n}+v\frac{\Delta t}{2}\frac{\partial}{\partial y}\left[u\frac{\partial \mathbf{T}}{\partial x}+v\frac{\partial \mathbf{T}}{\partial y}\right]^{n}\end{aligned} \tag{4.49}$$

Finalmente, é possível escrever a equação de transporte num esquema temporal implícito ou explícito. O sistema linear explícito com termos de convecção, difusão e estabilização é dado por:

$$\frac{\mathbf{M}}{\Delta t}\omega_{z}^{n+1}=\left(\frac{\mathbf{M}}{\Delta t}-\mathbf{V} \cdot \mathbf{G}-\nu \mathbf{K}\right)\omega_{z}^{n}-\mathbf{K}_{\mathbf{est}}\omega_{z}^{n} \tag{4.50}$$

onde a matriz de estabilização $\mathbf{K}_{est}$ é construída usando o procedimento padrão por meio da matriz do elemento $\mathbf{k}_{est}^{e}$, que depende do formato do elemento. Para um elemento triangular, a matriz elementar é dada por:

$$\mathbf{k}_{\mathbf{est}}^{\mathbf{e}}=\bar{u}\frac{\Delta t}{2}[\bar{u}\mathbf{k}_{\mathbf{x}}^{\mathbf{e}}+\bar{v}\mathbf{k}_{\mathbf{xy}}^{\mathbf{e}}]+\bar{v}\frac{\Delta t}{2}[\bar{u}\mathbf{k}_{\mathbf{xy}}^{\mathbf{e}}+\bar{v}\mathbf{k}_{\mathbf{y}}^{\mathbf{e}}] \tag{4.51}$$

onde $\mathbf{k}_{x}^{e}$ e $\mathbf{k}_{y}^{e}$ estão definidos nas Equação 4.26, enquanto $\bar{u}$, $\bar{v}$ e $\mathbf{k}_{xy}^{e}$ são definidos como:

$$\bar{u}=\frac{u_i+u_j+u_k}{3}; \quad \bar{v}=\frac{v_i+v_j+v_k}{3}; \quad \mathbf{k}_{\mathbf{xy}}^{\mathbf{e}}=\frac{1}{4A}\begin{bmatrix} b_ic_i & b_ic_j & b_ic_k \\ b_jc_i & b_jc_j & b_jc_k \\ b_kc_i & b_kc_j & b_kc_k \end{bmatrix} \tag{4.52}$$

4.5 MEF e a Condição CFL

Um fator importante a ser considerado na aplicação de métodos numéricos para problemas de CFD, como é o caso do Método de Elementos Finitos (MEF), é a condição de Courant–Friedrichs–Lewy, mais conhecida como "condição CFL". Segundo CAMINHA (2019) [21], essa é uma condição para estabilidade de métodos numéricos estáveis que modelam convecção, como é o caso deste projeto.

De forma resumida, essa condição - que sozinha não garante a estabilidade do método numérico - define e calcula o passo de tempo (Δt) que método deve usar para assegurar a estabilidade do problema e sua solução. Ou seja, para garantir que o domínio físico de um dado problema está contido no domínio numérico, a condição de CFL expressa que a distância que qualquer informação percorre durante o tamanho do passo de tempo Δt dentro da malha deve ser menor do que a distância entre

os elementos de malha. Em outras palavras, informações de um nó ou elemento de malha devem se propagar apenas para seus vizinhos imediatos.

Neste projeto como um todo, nos problemas e simulações apresentados nos Capítulos 5 e 6, não foi feito expressamente um cálculo para determinar o passo de tempo que atenda a condição de CFL. Por outro lado, para garantir a estabilidade dos resultados obtidos, foram feitas diversos testes e simulações com diferentes valores para o passo de tempo para diferentes malhas, o que afeta diretamente o custo computacional de cada estudo. No momento em que os resultados foram satisfatórios para o dado custo computacional (principalmente, o tempo de processamento), os valores de Δt foram organizados e são apresentados nos respectivos capítulos.

Capítulo 5

Validação do Modelo

Como introduzido anteriormente, neste capítulo são apresentados resultados de simulações utilizando o MEF para casos clássicos da literatura de escoamentos viscosos. Ao todo, são duas propostas: o escoamento de Hagen–Poiseuille em canal, na Seção 5.1; e o escoamento em canal com cilindro, na Seção 5.2. Cada caso possui soluções analíticas apresentadas pela literatura e estas, assim como os parâmetros de cada problema, são apresentadas nas respectivas seções. Em seguida, são comparados os resultados entre as simulações numéricas e as soluções analíticas para diferentes números de Reynolds.

Outro ponto importante que é abordado neste capítulo é a avaliação das malhas utilizadas em cada situação. Como em ANJOS (2020) [18], uma das formas de julgar a qualidade de uma malha é realizar sucessivas simulações com geometrias gradualmente mais refinadas. Para tal, algumas adaptações em parâmetros do MEF são feitas, como ajustes na quantidade total de iterações e intervalo de tempo (Δt) de cada passo iterativo. Quanto mais refinada a malha, maior a quantidade de iterações e menor deve ser o passo de tempo. Assim que o resultado é satisfatório em relação a um valor de tolerância predeterminado, pode-se dizer que a malha convergiu.

Para as simulações deste capítulo, foi utilizado um computador com as seguintes configurações:

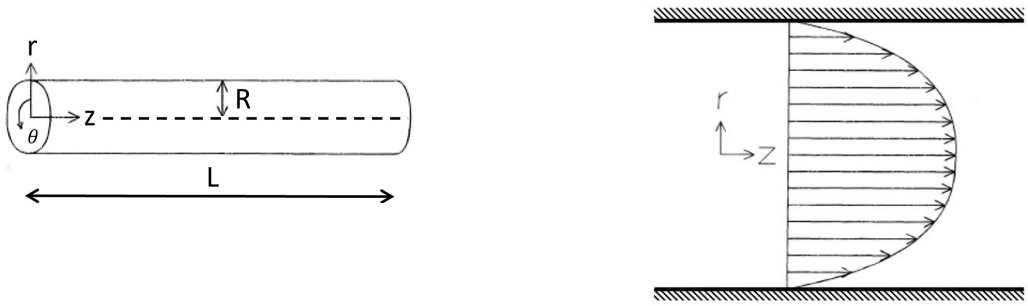
Tabela 5.1: Configurações e hardware da máquina que executou as simulações.

Hardware	**Descrição**
Modelo	Dell Inspiron 7580
Processador	Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz
Memória RAM	16 GB
Sistema Operacional	Windows 11 Pro
Armazenamento	1TB HD & 128GB SSD

5.1 Escoamento de Hagen-Poiseuille

Os escoamentos gerados somente por gradientes de pressão são chamados na literatura de escoamentos de Poiseuille, e possuem aplicações principalmente em dutos. De acordo com WHITE (2021) [6], esses escoamentos surgem em situações com dutos com formato arbitrário, porém constantes, em que devido à condição de não deslizamento na superfície, a velocidade nas paredes é zero ($u_w = 0$). Dessa forma, passar a existir um efeito de entrada, isto é, uma fina camada inicial e aceleração no centro. As camadas crescem e se encontram, e o centro desaparece dentro de um espaço bastante curto, denominado comprimento de entrada L_e. Ainda segundo a literatura, esse comprimento pode ser correlacionado com o número de Reynolds, definido na Equação 3.19.

Em particular, o escoamento de Hagen-Poiseuille ocorre em um duto de formato circular. Após o comprimento de entrada L_e, a velocidade se torna puramente axial e não varia mais com a coordenada z (axial) do fluxo. Neste caso, o escoamento é dito "completamente desenvolvido". Nas Figuras 5.1a e 5.1b abaixo, é possível ver os parâmetros utilizados no estudo desse tipo de escoamento, assim como o perfil de velocidade esperado, respectivamente.



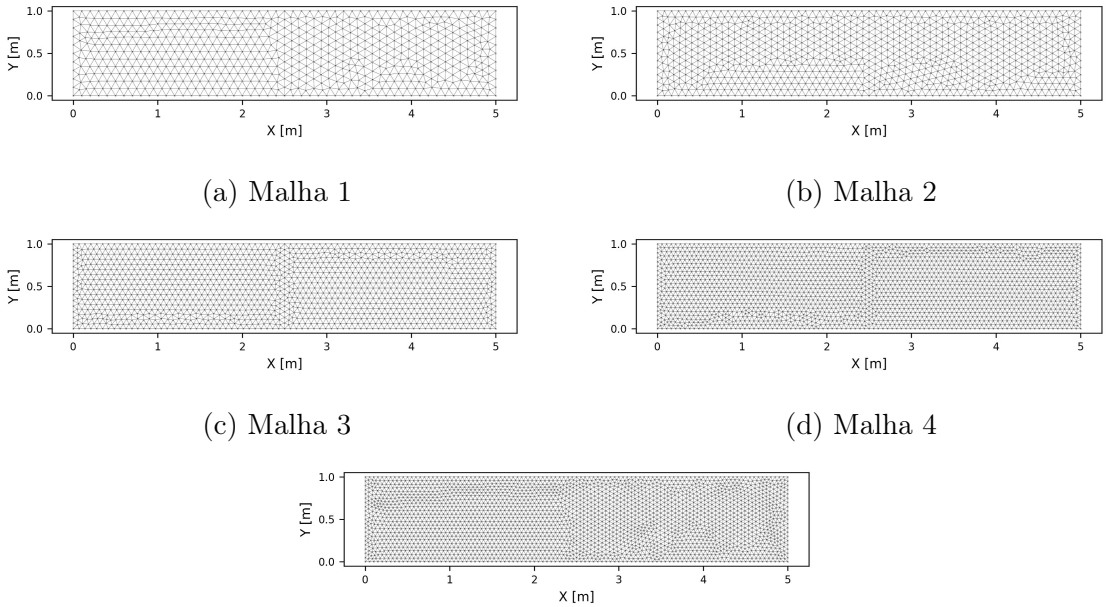
(a) Parâmetros do escoamento (b) Velocidade no escoamento

Figura 5.1: Escoamento Hagen-Poiseuille.

Uma vez que esse tipo de escoamento é amplamente conhecido e estudado na literatura, torna-se viável seu uso para validar o modelo implementado neste projeto. Para tal, foram estabelecidas algumas condições, assim como um domínio dividido em uma malha triangular. Para determinar a convergência de malha, foram feitas simulações para 5 malhas com elementos triangulares de tamanhos cada vez menores para afirmar em qual delas houve convergência. Na Tabela 5.2 a seguir estão as informações de cada malha.

Tabela 5.2: Número de nós e de elementos de cada malha.

Malha	**N^0 de nós**	**N^0 de elementos**
1 (mais grossa)	668	1214
2	937	1728
3	1348	2520
4	2148	4072
5 (mais fina)	2549	4856



(a) Malha 1

(b) Malha 2

(c) Malha 3

(d) Malha 4

(e) Malha 5

Figura 5.2: Malhas para simulação de escoamento de Poiseuille.

Por fim, foram feitas simulações para 3 diferentes números de Reynolds (entre 1, 10 e 100) para cada uma das malhas acima. Com isso, não apenas é possível verificar a convergência das malhas, mas também comparar os resultados do modelo computacional com os resultados analíticos obtidos da literatura. Todas as simulações nas Seções 5.1.1, 5.1.2 e 5.1.3 a seguir, possuem as mesmas condições iniciais, descritas na Figura 5.3 abaixo.

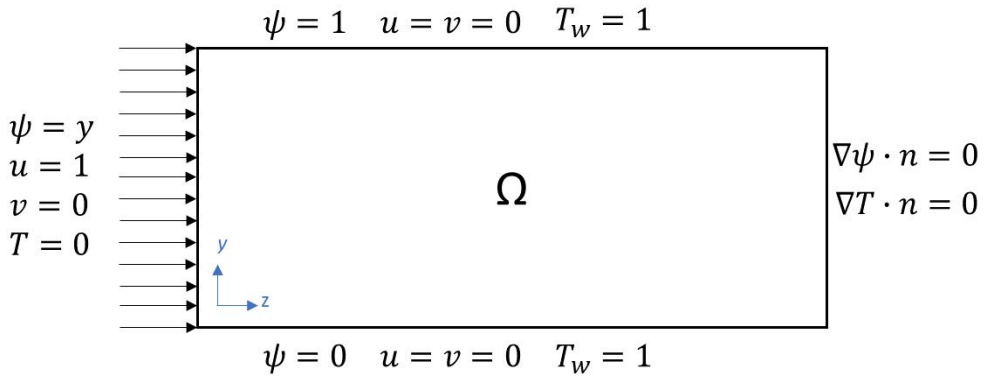


Figura 5.3: Condições iniciais das simulações de Hagen-Poiseuille.

No domínio estudado, existem condições de Dirichlet e de Neumann tanto para o a equação de função corrente ψ, quanto para o perfil de temperatura $\mathbf{T}$.

5.1.1 Escoamento em Canal para Re = 1

Para a simulação com $Re = 1$, optou-se por um canal de altura unitária e comprimento relativamente maior. Além disso, é necessário escolher o fluido de estudo, uma vez que as equações adimensionais exigem tanto o número de Reynolds, quanto o número de Prandtl, que por sua vez dependem de características do fluido. Os parâmetros escolhidos para a simulação estão resumidos na Tabela 5.3 abaixo.

Tabela 5.3: Parâmetros do canal e fluido de estudo.

Altura do canal	1 m
Comprimento do canal	5 m
Fluido	R1234ZE
Número de Prandtl	0,7968

Por outro lado, há parâmetros que precisam ser definidos para o MEF, sendo eles o passo de tempo Δt de cada iteração do método e o número total de iterações realizadas pelo modelo N. Essas duas quantidades são vitais para o modelo, uma vez que com malhas cada vez mais refinadas, e portanto, para processar matrizes globais maiores, o MEF exige passos de tempo menores e, para se preservar o total de tempo de simulação (produto entre Δt e N), o valor de N deve ser maior. Outro impacto dessas medidas é, logo, o tempo de processamento de cada simulação. Para as Malhas 1 a 5, foram determinados os seguintes valores de Δt e N:

Tabela 5.4: Valores de Δt e N de cada malha.

Malha	**Δt**	**N**
1 (mais grossa)	0,01	120
2	0,01	120
3	0,005	240
4	0,005	240
5 (mais fina)	0,004	300

Como apresentado na Seção 4.5, os valores da Tabela 5.4 acima, assim como os valores das Tabelas 5.6 e 5.12, foram determinados a partir de vários testes e

simulações consecutivos para atender a condição CFL, que assegura a estabilidade dos resultados obtidos.

Definidas as condições do domínio, as características do fluido de estudo e os parâmetros do MEF, a seguir são apresentados os resultados obtidos para as simulações com $Re = 1$. Para evitar o excesso de figuras, somente serão apresentados os gráficos de distribuição dos vetores de interesse da malha mais refinada, o que permite uma melhor visualização do comportamento dos vetores, sem perca de generalidade entre as malhas.

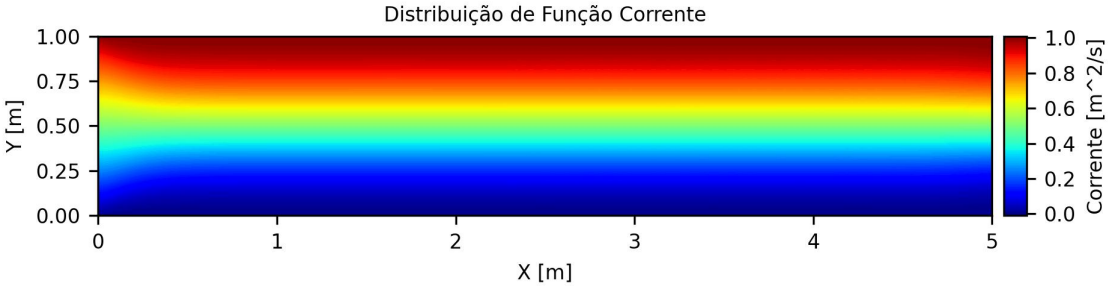


Figura 5.4: Função corrente para escoamento de Poiseuille com Re = 1.

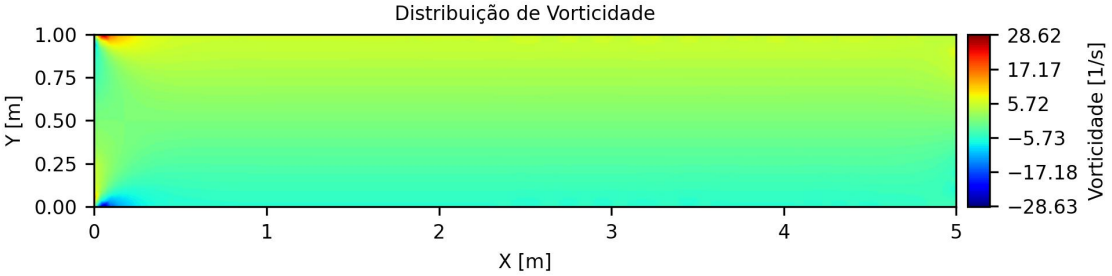


Figura 5.5: Função vorticidade para escoamento de Poiseuille com Re = 1.

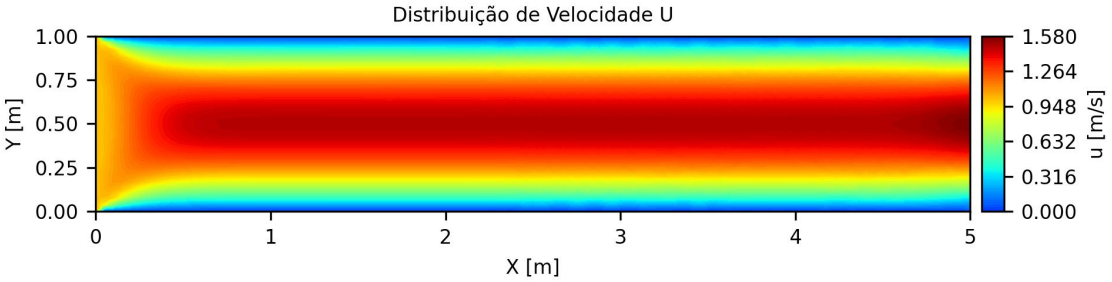


Figura 5.6: Velocidade horizontal u para escoamento de Poiseuille com Re = 1.

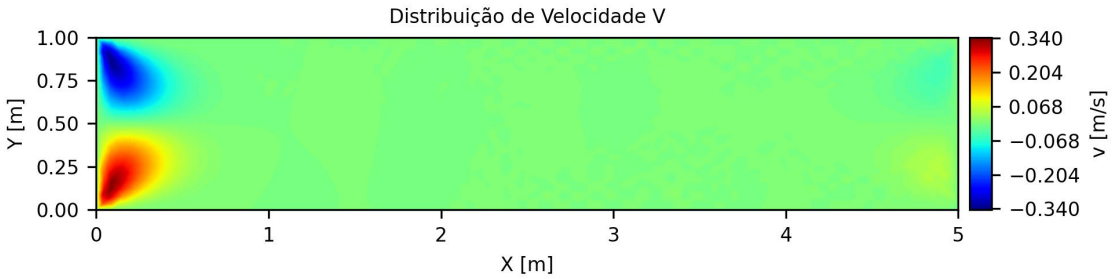


Figura 5.7: Velocidade vertical v para escoamento de Poiseuille com Re = 1.

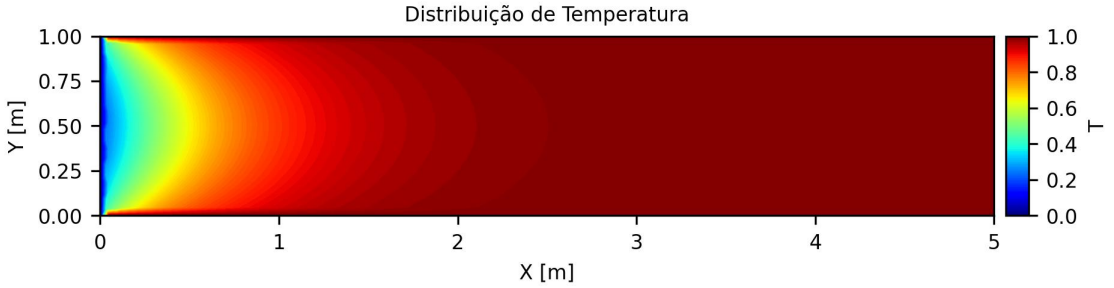


Figura 5.8: Temperatura para escoamento de Poiseuille com Re = 1.

Como já é possível observar na distribuição da função corrente na Figura 5.4, o MEF manteve a condição de $\psi = y$ por todo o domínio, sendo possível observar a formação da comprimento de entrada também no gráfico de distribuição de velocidades na Figura 5.6. Além disso, nesse último é também possível observar a formação de um perfil parabólico de velocidade, assim como esperado pela Figura 5.1b.

Por outro lado, na Figura 5.8, representando o vetor de temperatura do domínio, observa-se que pelo baixo valor de Reynolds, não foi possível que o escoamento se desenvolva completamente termicamente.

Algo importante a se destacar é que os gráficos acima, assim como os demais gráficos deste projeto que foram e serão apresentados, são resultados do modelo do MEF aplicado no Python. A seguir são apresentados gráficos comparando as velocidades obtidas no centro do canal ($z = 2,5$) em relação à solução analítica apresentada por WHITE (2021) [6].

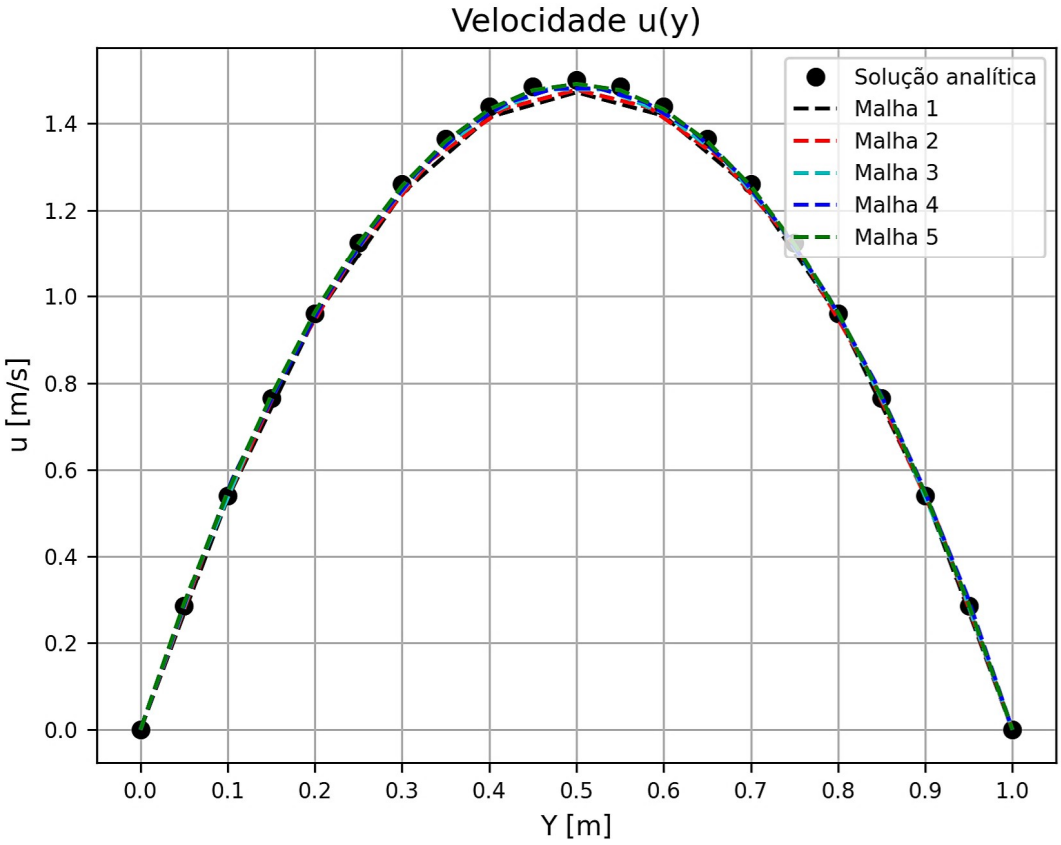


Figura 5.9: Gráfico de velocidades u com solução analítica para escoamento de Poiseuille com Re = 1.

Como é possível observar pelo Gráfico 5.9, o vetor de velocidade **u** ficou bem próximo em todas as malhas. Além disso, outro fator importante a se considerar é o tempo de processamento que cada malha leva para executar o modelo. Para melhor quantificar a proximidade dos resultados com a solução analítica, na Tabela 5.5 estão os erros relativos de cada malha e o seu tempo de processamento.

Tabela 5.5: Erro relativo para $u(y)$ de cada malha.

	Malha 1	**Malha 2**	**Malha 3**	**Malha 4**	**Malha 5**
Erro relativo (%)	1,470	1,253	0,851	0,905	0,515
Tempo (s)	10	14	57	174	352

A seguir, é apresentado uma comparação entre os valores de temperatura de cada malha.

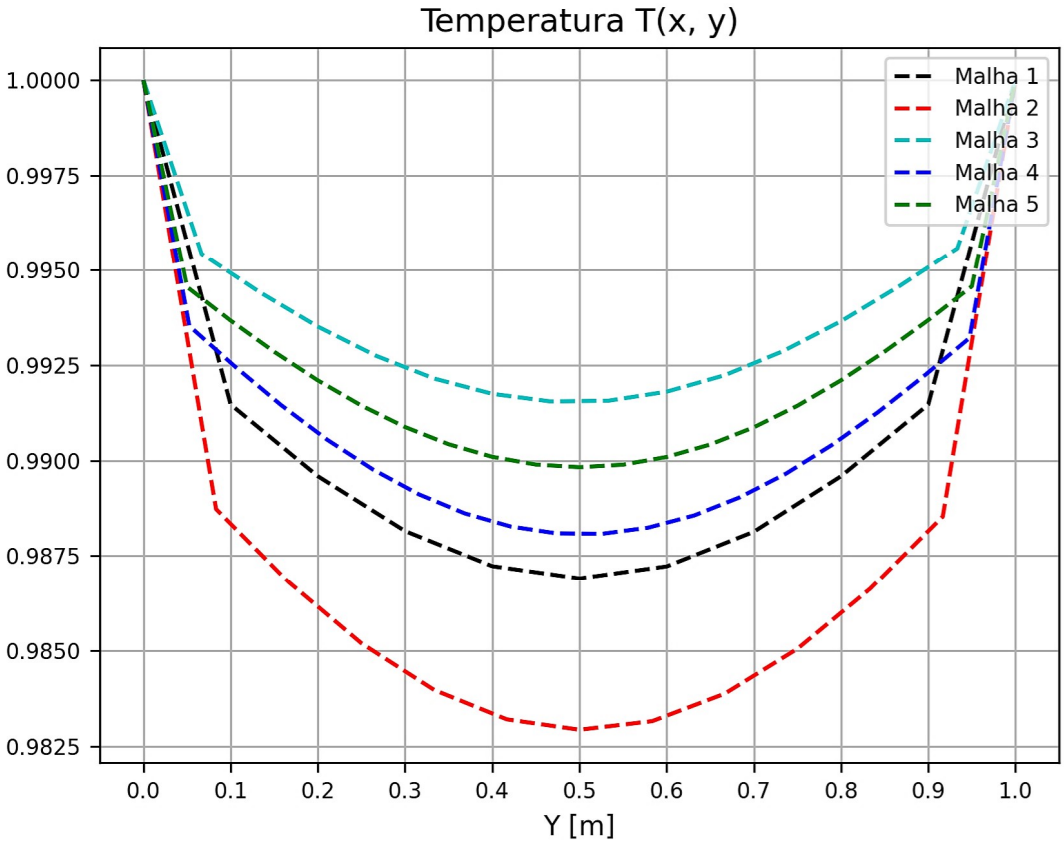


Figura 5.10: Gráfico de distribuição de temperatura para escoamento de Poiseuille com Re = 1.

Apesar de parecerem distintas, se observado a escala do gráfico, é possível afirmar que os valores temperatura ficaram bem próximos. O campo escalar de temperatura em $z = 2,5$ (no centro do canal), como visível na Figura 5.8, pouco é afetado pela temperatura baixa ($T = 0$) que o fluido carrega na entrada. Isso se deve porque o escoamento não se desenvolveu termicamente de forma completa, provavelmente pelo baixo valor de Reynolds.

5.1.2 Escoamento em Canal para Re = 10

Para a simulação com $Re = 10$, optou-se pelo mesmo canal e fluido utilizados na subseção anterior, cujos parâmetros estão resumidos na Tabela 5.3.

Por outro lado, diferentemente da seção anterior, por conta do número de Reynolds maior, foi possível alterar os valores de Δt e N para a Malha 3, para menos iterações e um passo de tempo maior, uma vez que esses valores afetam diretamente o tempo de processamento da simulação.

Tabela 5.6: Valores de Δt e N de cada malha.

Malha	**Δt**	**N**
1 (mais grossa)	0,01	120
2	0,01	120
3	0,01	120
4	0,005	240
5 (mais fina)	0,004	300

Assim, definidas as condições do domínio, as características do fluido e os parâmetros do MEF, a seguir são apresentados os resultados obtidos para as simulações com $Re = 10$, com exceção dos resultados para ψ e ω. Uma vez que esses se aproximam dos obtidos nos Gráficos 5.4 e 5.5, para evitar o excesso de figuras, somente serão apresentados os gráficos de vetores de interesse **u**, **v** e o perfil de temperatura **T**.

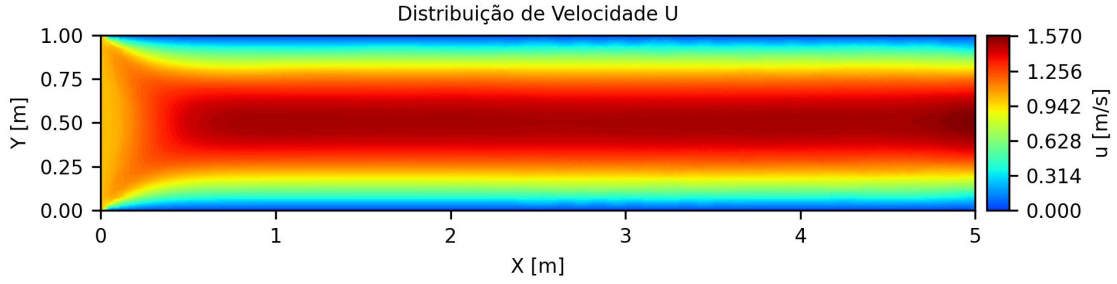


Figura 5.11: Velocidade horizontal u para escoamento de Poiseuille com Re = 10.

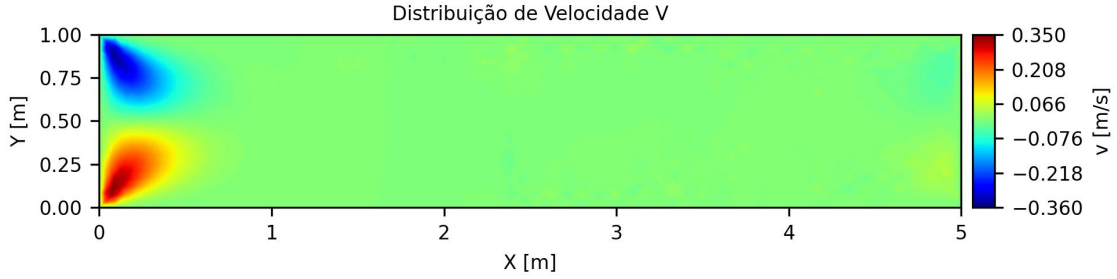


Figura 5.12: Velocidade vertical v para escoamento de Poiseuille com Re = 10.

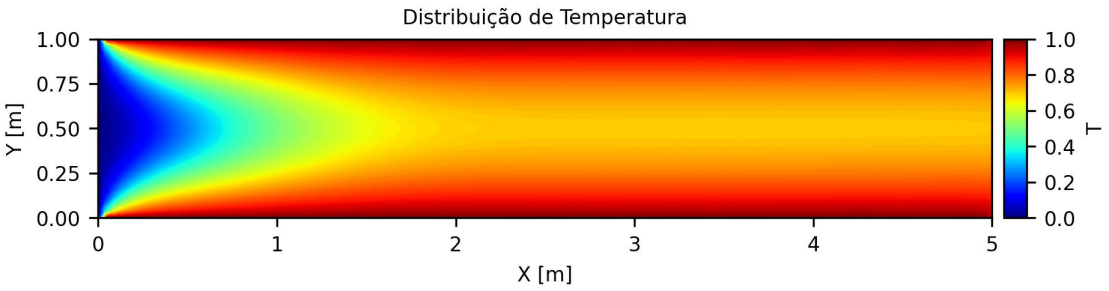


Figura 5.13: Temperatura para escoamento de Poiseuille com Re = 10.

Como nos resultados apresentados para $Re = 1$, os Gráficos de velocidade **u** e **v** (5.11 e 5.12, respectivamente) para $Re = 10$ se comportaram como esperado, formando, no caso do primeiro por exemplo, um perfil parabólico de velocidade ao longo do canal e com uma camada na entrada.

Além disso, no perfil de temperatura **T** foi possível ver que com o o número de Reynolds maior, houve um maior desenvolvimento térmico, se comparado ao primeiro caso. Todavia, ainda não se pode dizer que houve no escoamento o completo desenvolvimento térmico, uma vez que, segundo BERGMAN (2017) [16], para as condições estabelecidas para o canal com paredes de temperatrua constante, o perfil de temperatura também deveria ser parabólico ao longo do canal. Apesar desse comportamento surgir no Gráfico 5.15, a temperatura mínima está ainda consideravelmente acima da temperatura mínima, neste caso a temperatura de entrada ($\mathbf{T} = 0$ em $z = 0$).

Utilizando-se dos resultados acima, foram extraídos os seguintes gráficos comparando as soluções computacionais e analíticas para a velocidade **u** e para a temperatura **T**:

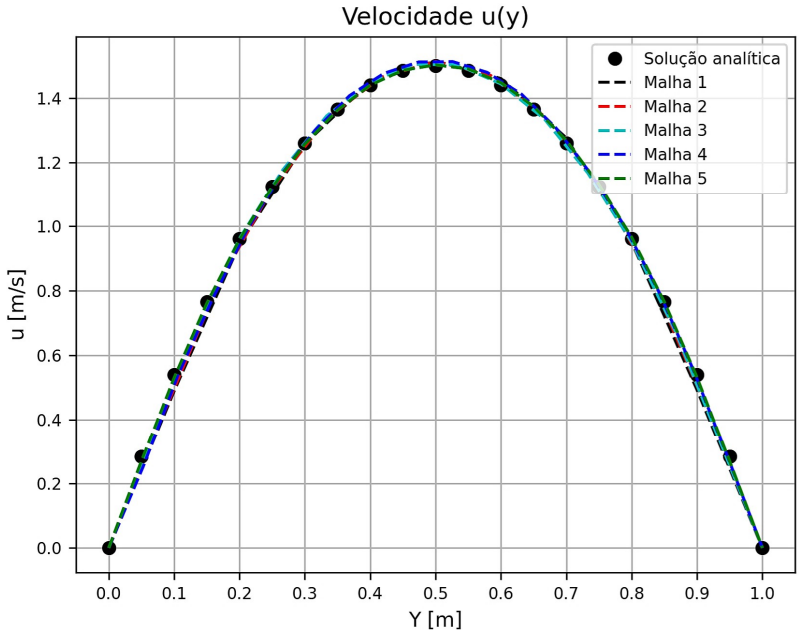


Figura 5.14: Gráfico de velocidades u com solução analítica para escoamento de Poiseuille com Re = 10.

O comportamento do Gráfico 5.14 de velocidade segue de acordo com o esperado pela literatura. Porém, diferentemente dos resultados obtidos da distribuição de temperatura, a velocidade ficou satisfatoriamente próximos à solução analítica. De fato, na Tabela 5.7 abaixo é listado o erro relativo entre solução analítica e a solução obtido para cada malha, assim como o tempo que cada modelo levou para processar.

Tabela 5.7: Erro relativo para $u(y)$ de cada malha.

	Malha 1	**Malha 2**	**Malha 3**	**Malha 4**	**Malha 5**
Erro relativo (%)	2,245	1,600	1,254	1,610	0,639
Tempo (s)	10	14	29	175	353

É possível notar que, se comparado com a Tabela 5.5, para um número de Reynolds maior, os erros relativos também aumentaram. Por outro lado, os tempos de processamento se mantiverem praticamente inalterados.

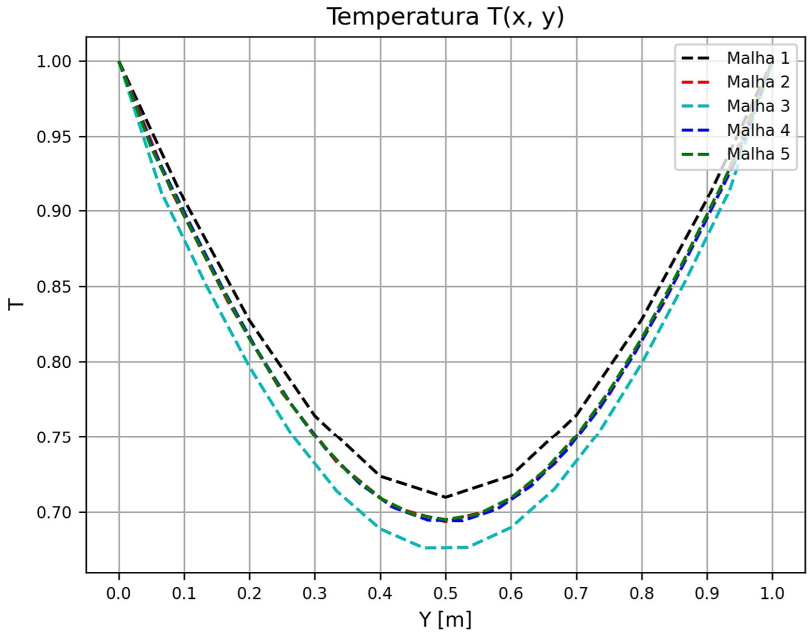


Figura 5.15: Gráfico de distribuição de temperatura para escoamento de Poiseuille com Re = 10.

Por fim, como antecipado nesta subseção, o comportamento do perfil de temperatura no Gráfico 5.15 já é parabólico, porém ainda não é possível comparar os resultados acima com a solução analítica, visto que o escoamento não se desenvolveu termicamente de forma completa. Novamente, isso provavelmente se deve às condições do próprio escoamento e seu fluido, como o baixo número de Reynolds.

5.1.3 Escoamento em Canal para Re = 100

Finalmente, para a última simulação de escoamento de Poiseuille com $Re = 100$, optou-se pelo mesmo canal, fluido e parâmetros do MEF utilizados na subseção anterior, cujos parâmetros estão resumidos nas Tabelas 5.3 e 5.6.

A seguir, são apresentados os resultados obtidos para a simulação utilizando o MEF. Assim como na Subseção 5.1.2, os resultados de função corrente e vorticidade, ψ e ω, respectivamente, foram omitidos por apresentarem pouca ou nenhuma diferença (pelo menos visual).

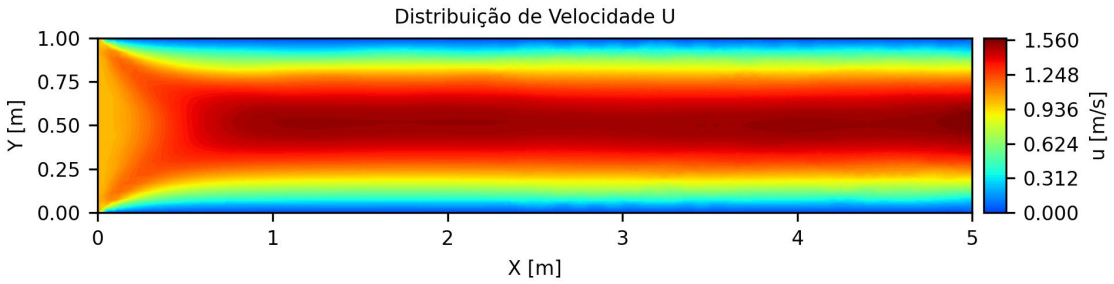


Figura 5.16: Velocidade horizontal u para escoamento de Poiseuille com Re = 100.

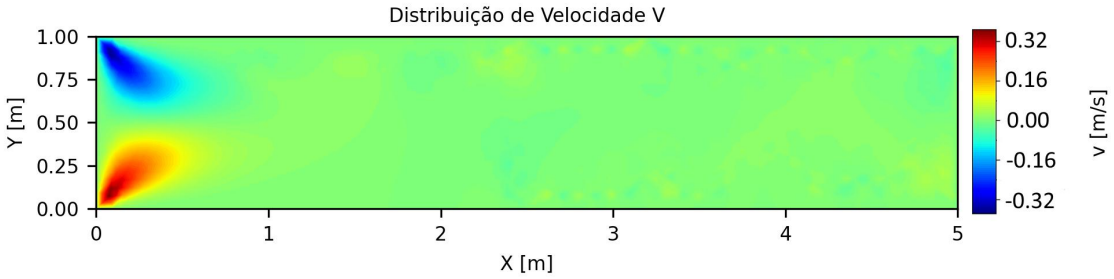


Figura 5.17: Velocidade vertical v para escoamento de Poiseuille com Re = 100.

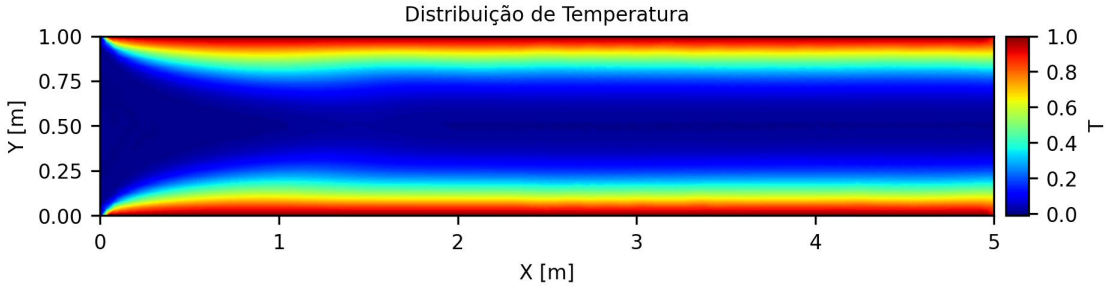


Figura 5.18: Temperatura para escoamento de Poiseuille com Re = 100.

Enquanto as soluções para velocidades **u** e **v**, nas Figuras 5.16 e 5.17, ficaram bem próximos aos obtidos para os outros números de Reynolds, aqui é possível destacar o quão diferente a distribuição de temperatura **T** ficou em relação aos demais obtidos nas subseções anteriores. Neste caso, o escoamento pôde se desenvolver muito mais termicamente, e o perfil de temperatura possui formato parabólico e parece permanecer constante durante todo o canal. De fato, segundo BISWAS (2015) [22], para escoamentos laminares e hidrodinamicamente desenvolvidos, o comprimento térmico de entrada pode ser dado pela expressão $L_{e,T} \approx 0.05 Re_D Pr$. Ou seja, para um $L_{e,T} < 1$, então $Re > 25$, considerando o número de Prandtl do fluido de estudo.

Logo, podemos dizer que para $Re = 100$, o escoamento de Poiseuille neste caso é termicamente desenvolvido.

Da mesma forma, podemos extrair dos resultados acima uma comparação com as soluções analíticas, só que desta vez também incluindo a solução para temperatura. Assim, obtemos o seguinte:

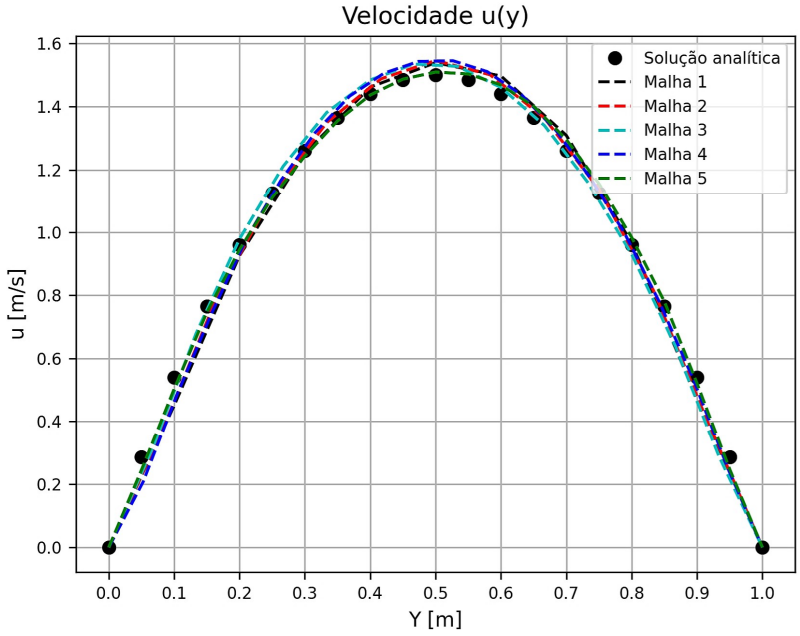


Figura 5.19: Gráfico de velocidades u com solução analítica para escoamento de Poiseuille com Re = 100.

O vetor de velocidade **u** obtido das malhas ficaram bem próximos do resultado analítico. De fato, na Tabela 5.8 é possível observar os erros relativos e o tempo de cada simulação.

Tabela 5.8: Erro relativo para $u(y)$ de cada malha.

	Malha 1	**Malha 2**	**Malha 3**	**Malha 4**	**Malha 5**
Erro relativo (%)	4,197	3,463	3,645	3,828	2,189
Tempo (s)	14	17	32	184	373

Conforme constatado na Subseção 5.1.2, com o maior número de Reynolds, enquanto o tempo de processamento praticamente se manteve, o erro relativo entre soluções computacional e analítica aumentou.

Abaixo, a comparação entre os resultados do perfil de temperatura obtidos na simulação e os esperados pela solução analítica, retirada de BISWAS (2015) [22].

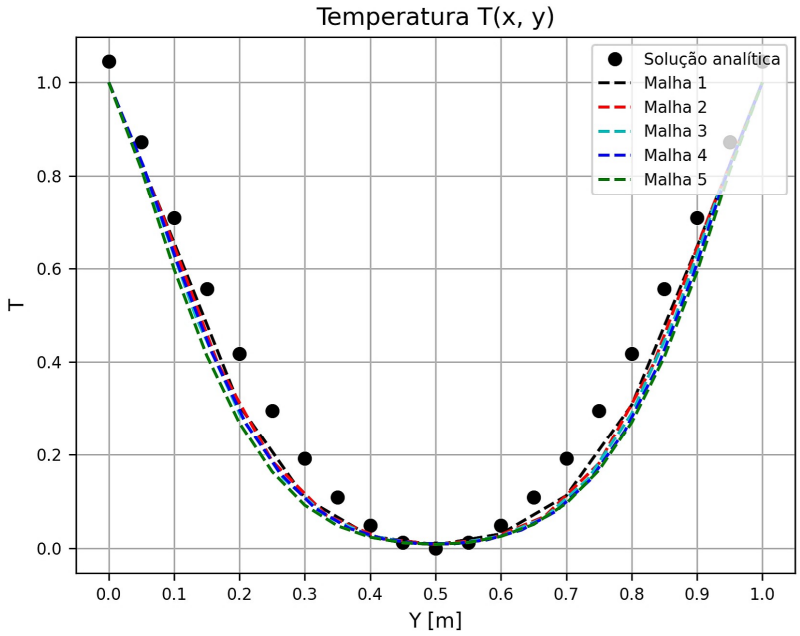


Figura 5.20: Gráfico de distribuição de temperatura com solução analítica para escoamento de Poiseuille com Re = 100.

Visualmente, no Gráfico 5.20 acima os pontos de cada malha e os analíticos parecem bem próximos. Porém, na Tabela 5.9 abaixo é apresentado os valores dos erros relativos.

Tabela 5.9: Erro relativo para $T(y)$ de cada malha.

	Malha 1	**Malha 2**	**Malha 3**	**Malha 4**	**Malha 5**
Erro relativo (%)	12,730	14,241	16,156	17,157	20,008

Como é possível ver, os erros superam a ordem dos 10%, crescendo para as malhas mais refinadas. Apesar de não ser o desejado, os resultados são satisfatórios se considerar que a própria solução analítica possui algumas limitações e aproximações feitas. Em BISWAS (2015) [22], por exemplo, é utilizado o Método de Newton-Raphson para aproximar a solução de equações diferenciais de temperatura para dutos circulares. Logo, podemos afirmar que a solução térmica do MEF, dentro de um limite, é razoável.

Além disso, considerando os erros relativos pequenos para todos os 3 números de

Reynolds testados, e os tempos de processamento baixos, é válido dizer que houve convergência na Malha 3. A partir dela, os erros tendem a aumentar, junto com os tempos de processamento.

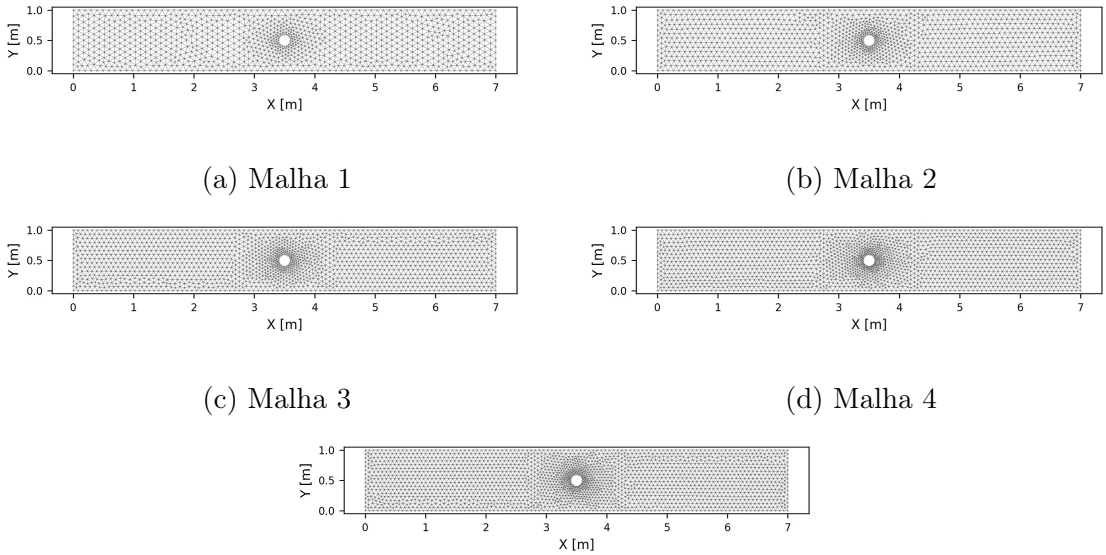
5.2 Escoamento em Canal com Cilindro

Assim como o escoamento de Hagen-Poiseuille em dutos circulares, os escoamentos em canais com objetos em seus interiores é outro tópico amplamente estudado na literatura. No presente projeto, para validar o MEF, foi escolhido o escoamento de canal com cilindro no seu interior. Em particular, foi escolhido o caso em que as paredes estão próximas do cilindro. É importante ressaltar esse detalhe pois existem casos estudando escoamentos com paredes distantes, o que influencia no cálculo do número de Reynolds. Em ambos, o escoamento também ocorre por diferença de pressão entre as pontas do canal, e existe um objeto no interiro (cilindro) que "atrapalha"o fluxo direto do fluido.

Nesta seção, portanto, são apresentados os resultados de simulações para 5 malhas, assim como no escoamento de Poiseuille na Seção 5.1, também para averiguar a convergência de malha; e para números de Reynolds iguais a 10 e 100. Por conta da maior complexidade das malhas, aqui não é feita a simulação para $Re = 1$. Na Tabela 5.10 a seguir estão as informações de cada malha e nas Figuras 5.21 estão esquematizadas as malhas geradas pelo *GMESH*.

Tabela 5.10: Número de nós e de elementos de cada malha.

Malha	**N^0 de nós**	**N^0 de elementos**
1 (mais grossa)	1072	1966
2	1366	2530
3	1679	3134
4	1925	3610
5 (mais fina)	2180	4102



(a) Malha 1

(b) Malha 2

(c) Malha 3

(d) Malha 4

(e) Malha 5

Figura 5.21: Malhas para simulação de escoamento em canal com cilindro.

É importante destacar, assim como na seção anterior, que em cada malha vai exigir valores de N (quantidade de iterações do MEF) e Δt (passo de tempo de cada iteração) diferentes por conta das diferentes quantidades de nós e elementos.

Além das características das malhas estudadas, é preciso também desenhar o problema de estudo e estabelecer as suas condições de contorno, seja para a velocidade, como para a temperatura. Abaixo, na Figura 5.22 está resumido as condições que serão comumente discutidas nas Subseções 5.2.1 e 5.2.2.

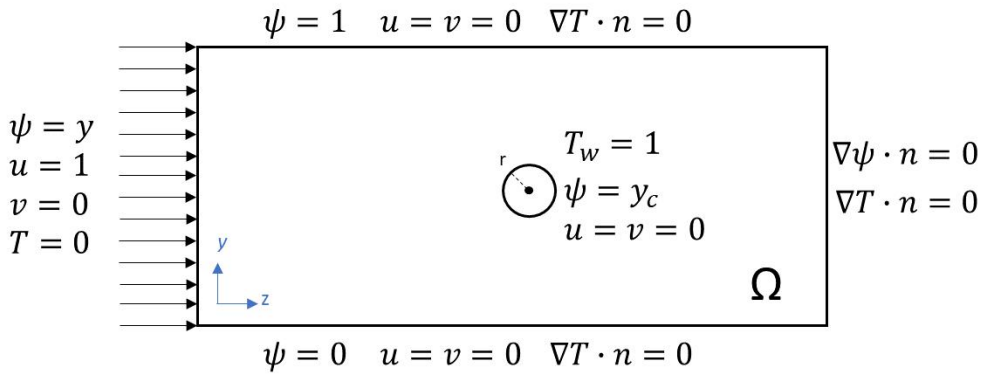


Figura 5.22: Condições iniciais das simulações de canal com cilindro.

Tabela 5.11: Parâmetros do canal e fluido estudados.

Altura do canal	1 m
Comprimento do canal	7 m
Raio do cilindro	$0,1$ m
Fluido	R1234ZE
Número de Prandtl	0,7968

Nota-se que no domínio de estudo, existem condições de contorno do tipo Dirchlet e Neumann tanto para o vetor função corrente ψ quanto para a temperatura $\mathbf{T}$. Há condições de não deslizamento tanto nas paredes do canal quanto nas paredes do cilindro, que possui temperatura fixada em $T_w = 1$. Além disso, para ajudar na solução do MEF para a função corrente, fixou-se também que para toda a parede do cilindro o valor de ψ unitário.

5.2.1 Escoamento com Cilindro para Re = 10

Estabelecidas as condições da malha, do domínio do problema e do fluido estudado, resta definir para a simulação de $Re = 10$ os parâmetros do MEF utilizados. Após testes, estabeleceram-se os seguintes valores para cada malha:

Tabela 5.12: Valores de Δt e N de cada malha.

Malha	**Δt**	**N**
1 (mais grossa)	0,01	120
2	0,01	120
3	0,005	240
4	0,005	240
5 (mais fina)	0,004	300

Com isso, através do MEF, obteve-se os vetores velocidade $\mathbf{u}$ e velocidade $\mathbf{v}$, da função corrente ψ, da vorticidade ω e o perfil de temperatura $\mathbf{T}$ para todas as malhas. Para evitar expor um grande número de resultados gráficos, serão apresentados a seguir as distribuições de vetores para a malha mais refinada (Malha 5).

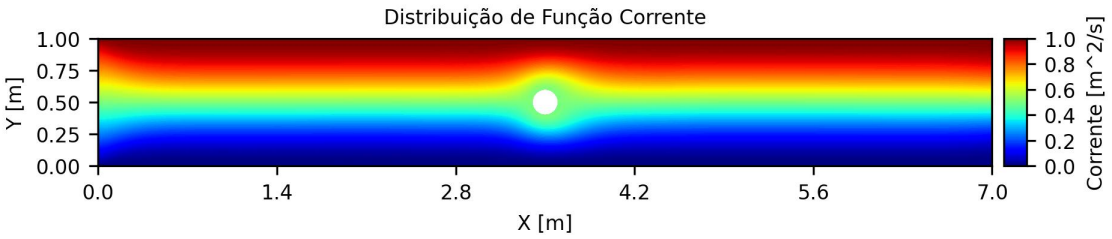


Figura 5.23: Função corrente para escoamento com cilindro para Re = 10.

A função corrente apresentada na Figura 5.23, como planejado, apresentou comportamento esperado, onde a função corrente varia linearmente com a altura do canal, contornando o cilindro ($\psi = y$).

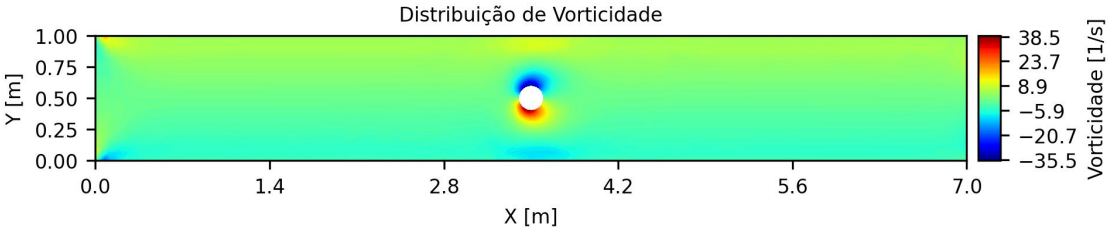


Figura 5.24: Função vorticidade para escoamento com cilindro para Re = 10.

A vorticidade, por outro lado, só possui valores expressivos próximos ao cilindro. Na parte superior, a vorticidade assume valores negativos, enquanto que na parte inferior esta assume valores positivos. Isso se deve por conta do movimento do fluido contornando o cilindro.

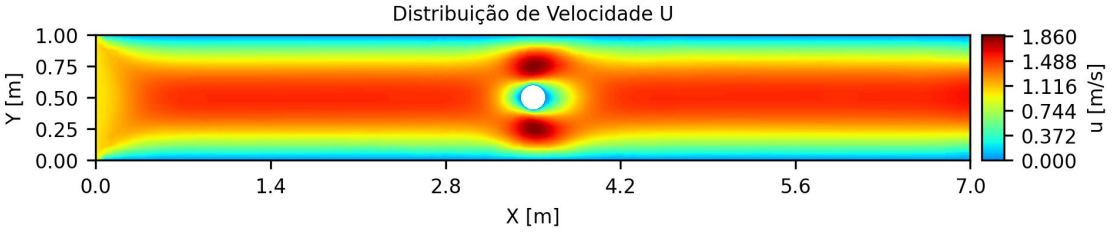


Figura 5.25: Velocidade horizontal u para escoamento com cilindro para Re = 10.

A velocidade **u** por sua vez também apresentou comportamento esperado, assumindo um perfil parabólico ao longo do canal, exceto na região mais próxima ao cilindro, principalmente por conta da condição de não deslizamento. Outro ponto importante a se destacar aqui, e que era o esperado na literatura, é a simetria axial

apresentada pelo gráfico. Uma vez que o dominio em si possui tal simetria, esperava-se que a velocidade também apresentasse uma simetria através do meio do cilindro (em $y = 0,5$ ou $r = 0$ em coordenada polar).

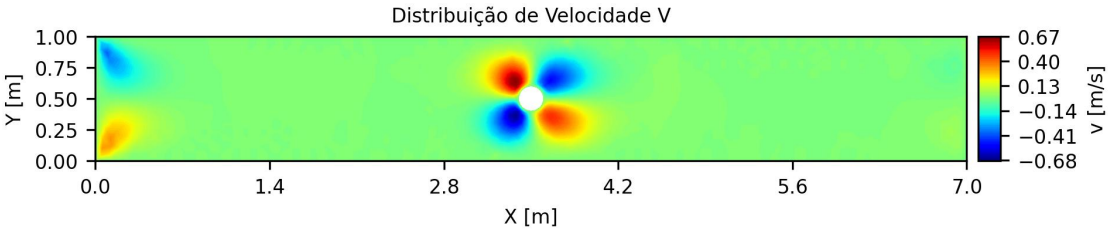


Figura 5.26: Velocidade vertical v para escoamento com cilindro para Re = 10.

Assim como a velocidade **u**, a velocidade **v** também apresentou uma certa simetria axial, onde os valores positivos e negativos indicam como se deu a passagem do fluido no contorno do cilindro.

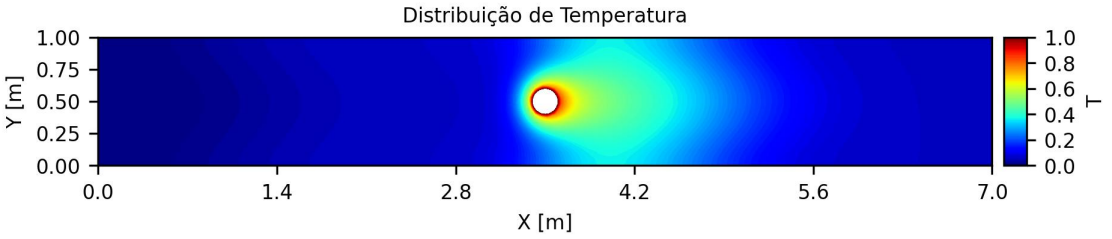


Figura 5.27: Temperatura para escoamento com cilindro para Re = 10.

Por fim, e não menos importante, a temperatura **T** obtida pelo MEF também apresentou o comportamento esperado. Dadas as condições, a temperatura do cilindro se difundiria pelo canal e seria "distorcida"pelo movimento do fluido frio através do cilindro.

Além dos resultados acima, também foi possível extrair pontos do vetor **u** e da temperatura **T** em dois pontos da malha: para $z = 2,75$ e $z = 4,25$, respectivamente. A razão da escolha desses pontos é: para a velocidade, buscou-se um ponto anterior ao cilindro onde não havia interferência do mesmo na velocidade; para a temperatura, um ponto posterior ao cilindro, para se visualizar a dispersão da temperatura no canal. No primeiro caso, a escolha permite comparar a solução com a solução analítica da velocidade, assim como no escoamento de Poiseuille. Por outro lado, dada a complexidade do problema, não se encontrou uma solução analí-

tica dadas as condições de estudo. Nos Gráficos 5.28 e 5.29 são apresentados essas comparações.

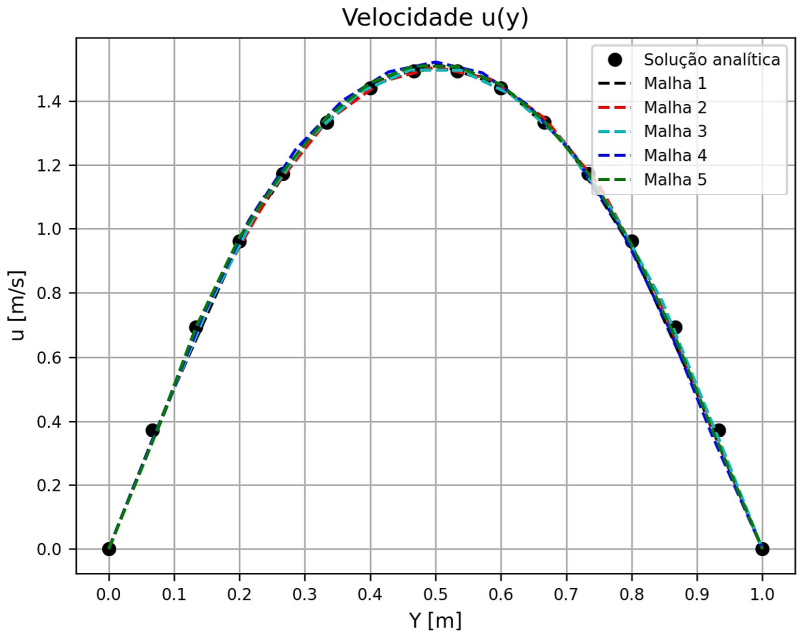


Figura 5.28: Solução analítica da velocidade horizontal u para escoamento com cilindro para Re = 10.

Aqui é possível ver que a solução computacional e analítica para a velocidade **u** ficaram bem próximas. De fato, na Tabela 5.13 é possível observar os erros relativos e os tempos de processamento de cada malha. Para $Re = 10$, todos os erros ficaram em valores menores ou iguais a $2,5\%$.

Tabela 5.13: Erro relativo para $u(y)$ de cada malha.

	Malha 1	**Malha 2**	**Malha 3**	**Malha 4**	**Malha 5**
Erro relativo (%)	2,088	1,541	1,292	2,539	1,908
Tempo (s)	49	69	98	264	417

Já em relação a temperatura, notou-se no Gráfico 5.29 que os comportamentos foram bem próximos entre as malhas. A diferença entre valores obtidos para cada simulação pode ter raiz nos valores de N e Δt dados para cada caso, onde as malhas 1 e 2 tiveram resultados próximos, assim como as malhas 3, 4 e 5 entre si.

Quanto ao tempo, os valores ficaram parecidos com os obtidos nos estudos de escoamento de Poiseuille, porém, por conta da complexidade adicional, os valores para

as malhas menos refinadas ficaram um pouco maiores se comparadas a simulações em canal livre na Seção 5.1.

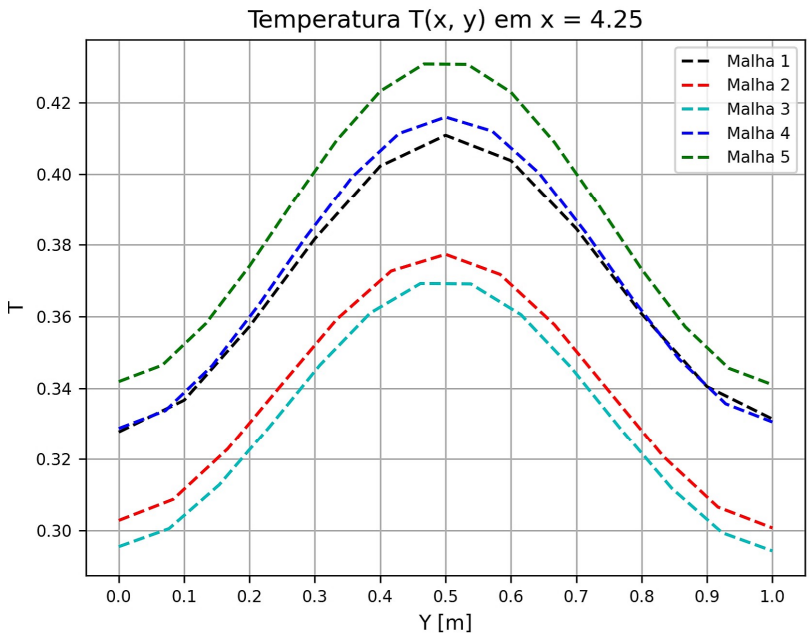


Figura 5.29: Gráfico da distribuição de temperatura para escoamento com cilindro para Re = 10.

Nota-se, porém, que no Gráfico 5.29 acima, que não há linearidade nos resultados obtidos para cada malha, isto é, os resultados não variariam diretamente de acordo com o refinamento da malha. A possível razão para esse comportamento é a forma como essas malhas de nós e elementos foram gerados. Cada malha foi gerada e refinada individualmente no software *"GMSH"*. Se, por outro lado, fosse criado uma malha "grossa"e que tivesse seus nós e elementos consecutivamente divididos em partes menores, isso faria com que esses resultados apresentassem um comportamento que acompanham o refinamento das malhas. Essa não-linearidade no comportamento nas malhas gradualmente refinadas também é observado no Gráfico 5.33.

5.2.2 Escoamento com Cilindro para Re = 100

Finalmente, para as simulações com $Re = 100$ se manteve as mesmas condições de domínio, características de fluido e parâmetros do MEF apresentados anteriormente (ver Tabelas 5.11 e 5.12). Novamente, sem perder generalidade e reduzir a

quantidade excessiva de imagens, omitiu-se aqui os resultados para ψ e ω, uma vez que estes pouco se diferem entre os diferentes números de Reynolds.

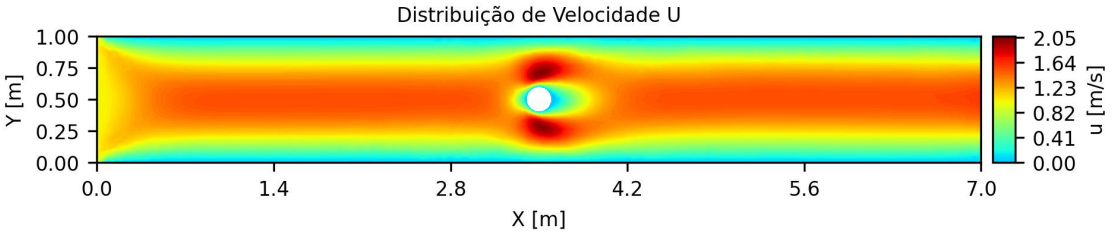


Figura 5.30: Velocidade horizontal u para escoamento com cilindro para Re = 100.

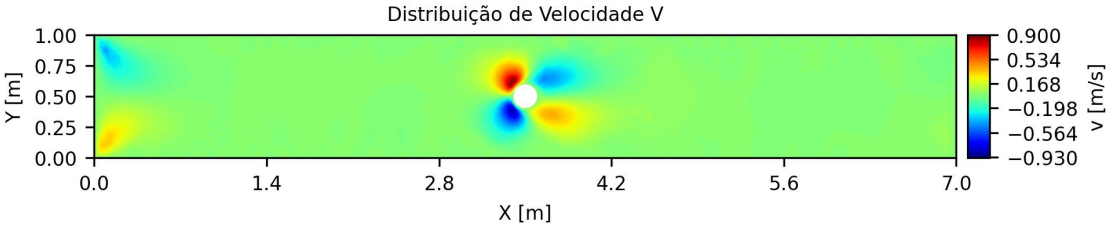


Figura 5.31: Velocidade vertical v para escoamento com cilindro para Re = 100.

Em ambos os gráficos de velocidade, percebe-se o mesmo comportamento destacado na subseção anterior, porém com uma maior distorção por conta da maior velocidade gerada pelo maior valor de Re.

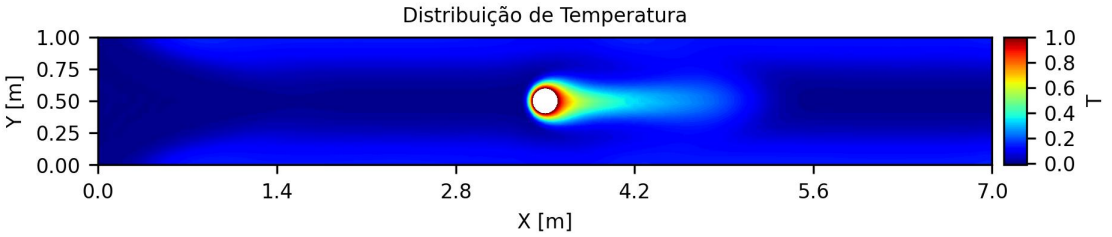


Figura 5.32: Temperatura para escoamento com cilindro para Re = 100.

O perfil de temperatura, no Gráfico 5.32, por sua vez, sofreu maior distorção por conta da maior velocidade (número de Reynolds). Por conta disso, a região entre o cilindro e a parede do canal ficou mais fria, se comparada com o resultado obtido para $Re = 10$. Na Figura 5.33 é possível observar melhor essa distorção.

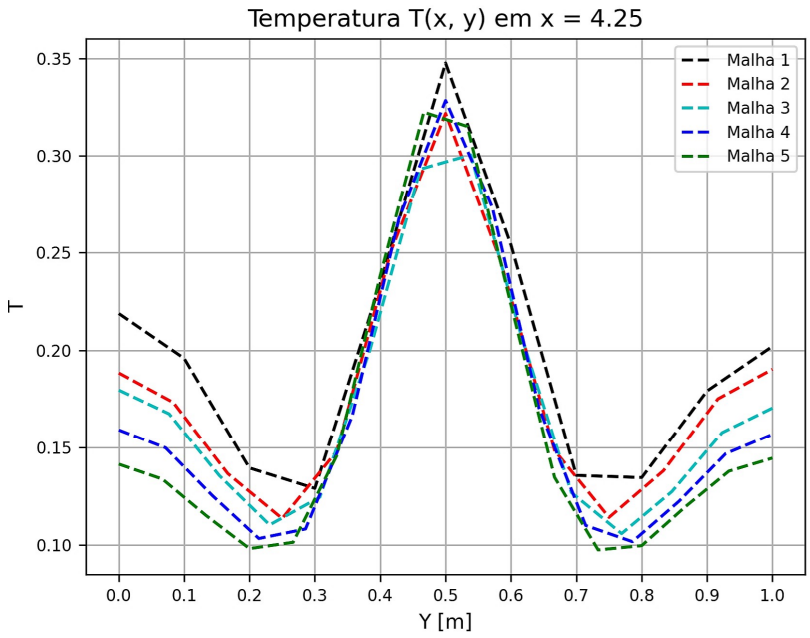


Figura 5.33: Solução analítica da temperatura para escoamento com cilindro para Re = 100.

Por fim, assim como na Seção 5.2.1 anterior, também é feita uma comparação entre a solução de velocidade antes do cilindro com a solução analítica esperada pela literatura.

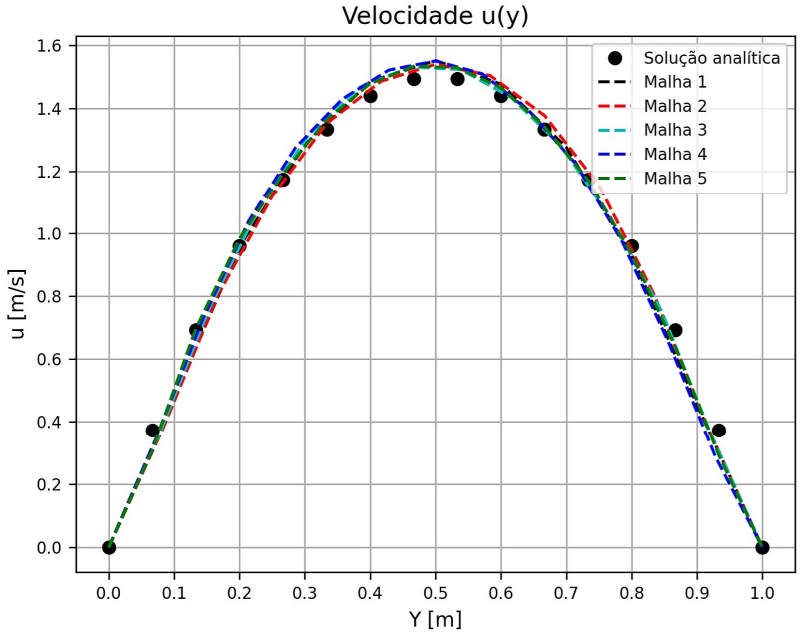


Figura 5.34: Solução analítica da velocidade horizontal u para escoamento com cilindro para Re = 100.

Tabela 5.14: Erro relativo para $u(y)$ de cada malha.

	Malha 1	**Malha 2**	**Malha 3**	**Malha 4**	**Malha 5**
Erro relativo (%)	3,971	3,769	3,464	4,650	3,379
Tempo (s)	49	69	97	262	415

Como esperado, os resultados computacionais e analíticos foram bem próximos, mas, assim como no escoamento de Poiseuille, o aumento do número de Reynolds refletiu num erro relativo entre as soluções maior, variando entre 3% e $4,5\%$. De toda forma, é razoável dizer que o resultado obtido foi satisfatório. Os tempos permaneceram praticamente inalterados.

Com isso, considerando os erros relativos pequenos para ambos os números de Reynolds testados, e os tempos de processamento intermediários, é válido dizer que houve convergência na Malha 3. A partir dela, os erros tendem a aumentar, junto com os tempos de processamento.

Capítulo 6

Resultados dos Estudos

Como apresentado na Introdução (Capítulo 1) deste projeto, um dos objetivos aqui é também estudar o arrefecimento de equipamentos eletrônicos. Nesta seção, com uso do algoritmo do MEF e do esquema de Taylor-Galerkin, trabalhados e validados nas seções anteriores, é modelado um estudo de situação prática de resfriamento de eletrônicos. Um dos equipamentos eletrônicos que exigem um sistema de resfriamento para funcionar, e estão relacionados ao nosso dia-a-dia, são os processadores de computadores. Na Figura 6.1 é desenhado uma simplificação dos principais componentes de um processador comum com 4 núcleos.

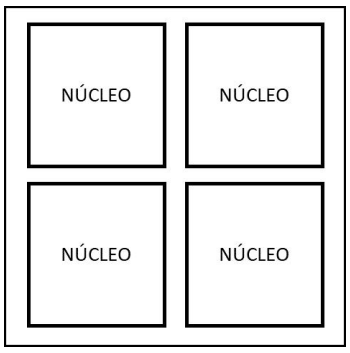


Figura 6.1: Esquema de componentes de um processador de 4 núcleos.

Esses e outros equipamentos eletrônicos possuem, no geral, pequenas dimensões, exceto algumas com finalidades específicas. Por conta disso, além do fluido utilizado, o estudo dessas aplicações geralmente envolvem pequenas dimensões e valores de número de Reynolds pequenos, como será o caso apresentado mais adiante.

No caso dos processadores, para seu correto funcionamento, um sistema de resfriamento é necessário, uma vez que durante seu funcionamento muito calor é gerado.

As soluções mais comuns são resfriamento a ar, nos denominados *"air coolers"*; ou com resfriamento com líquidos, nos chamados *"water coolers"*. Neste projeto, o ponto de interesse está nesse último caso, por vários motivos, entre eles podemos citar:

- os líquidos, em sua maioria, são praticamente incompressíveis, o que evitaria uma complexidade a mais a se lidar no MEF;
- a possibilidade de simplificar um modelo de *"water cooler"* como esse em canais, assim como foi feito no Capítulo 5 anterior.

Por isso, na Figura 6.2 a seguir é apresentado um esquema básico da proposta deste projeto: simular e estudar o escoamento através de um canal externo, com canais e paredes internas, sobre um equipamento eletrônico que gera calor.

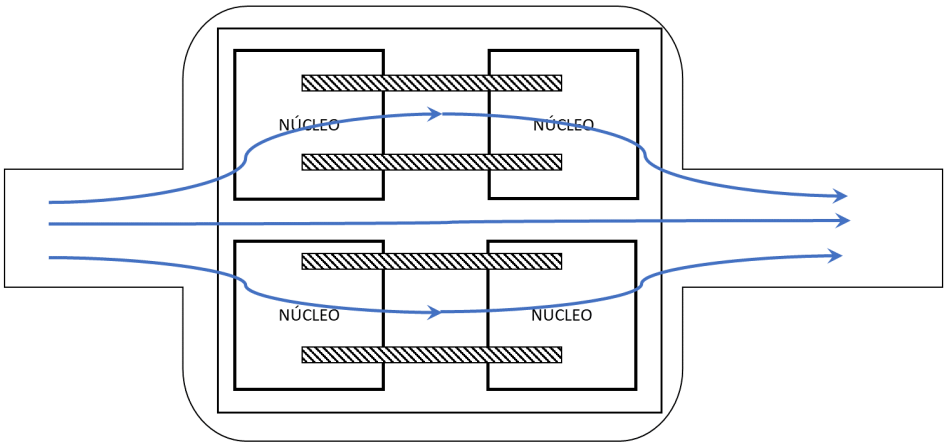


Figura 6.2: Escoamento de fluido para resfriamento de componente eletrônico.

Na Figura 6.2 acima, os núcleos serão responsáveis pela geração de calor do problema. Por isso, eles serão tratados no problema como massas pontuais nos cantos do escoamento gerando calor. Isso pode ser melhor visualizado na Figura 6.3 mais adiante.

Para tentar entender a influência desses canais internos no escoamento e na troca de calor no fluido refrigerante no interior do canal externo, foram feitas simulações fazendo alterações nas paredes internas. Ao total, foram feitas 7 simulações, e nelas foram modificados:

1. A quantidade de paredes internas (e, consequentemente, a quantidade de canais internos), variando entre 5, 8 e 11 canais internos;

2. A largura das paredes internas, entre um comprimento padrão, outro mais curto e um terceiro caso com largura mais longa;

3. A altura das paredes internas, entre um valor padrão menor, uma altura intermediária e outra mais alta.

Pela quantidade de simulações feitas, cada situação é testada apenas para uma malha com elementos triangulares pequenos e próximos aos apresentados nos casos de validação do Capítulo 5. Por isso, não será analisada a convergência de malha, que, para todos os efeitos, será considerada como convergente. Além disso, para facilitar a identificação de cada situação testada, na Tabela 6.1 são estabelecidos as abreviações de cada malha. Essas abreviações serão utilizadas ao longo deste capítulo para identificar cada caso de estudo.

Tabela 6.1: Siglas das malhas estudadas

Sigla	**Descrição**
N5	Malha com 5 canais internos e paredes internas com formato padrão
N8	Malha com 8 canais internos e paredes internas com formato padrão
N11	Malha com 11 canais internos e paredes internas com formato padrão
8L1	Malha com 8 canais internos e paredes internas com largura menor
8L2	Malha com 8 canais internos e paredes internas com largura maior
8A1	Malha com 8 canais internos e paredes internas com altura intermediária
8A2	Malha com 8 canais internos e paredes internas com altura maior

Como é possível observar, para os casos em que as altura e as larguras das paredes internas variam, fixou-se o número de canais internos em 8, apenas por ser o valor intermediário entre os canais estudados. Para cada caso acima, os parâmetros de cada malha para o canal externo e as características do fluido refrigerante escolhidos para as simulações estão resumidos nas Tabelas 6.2 e 6.3 a seguir.

Tabela 6.2: Parâmetros do canal externo.

Altura do canal (m)	12,0
Comprimento do canal (m)	16,0
Raio das quinas do canal (m)	0,2

Tabela 6.3: Parâmetros do fluido estudado.

Fluido do escoamento	R1234ZE
Massa específica (kg/m^3)	4.774
Viscosidade dinâmica (mPa.s)	0.0124
Velocidade de entrada (cm/s)	1,0
Diâmetro característico (μm)	60,0
Número de Reynolds* (Re)	0,23
Número de Prandtl (Pr)	0,7968

O número de Reynolds, Re, é calculado a partir dos valores de massa específica, velocidade, diâmetro e viscosidade, a partir da Equação 3.19. Além disso, para permitir um comparação consistente entre as simulações, o fluido escolhido foi o *R1234ZE*.

Para os valores das dimensões das paredes de cada caso, estas serão apresentadas em suas respectivas seções.

Em seguida, por serem malhas muito parecidas, com números de nós e elementos bem próximos (no geral), os parâmetros do MEF para as simulações foram definidos como apresentado na Tabela 6.4 abaixo.

Tabela 6.4: Número de iterações e passo de tempo do MEF.

N (iterações)	240
Δt (s)	0,005

É importante, porém, ressaltar que por conta da particularidade e da relativa complexidade dos estudos que serão apresentados adiante, não se encontrou na literatura uma solução analítica para os vetores de velocidade (**u** e **v**) ou de temperatura

(**T**). Por isso, diferentemente do que fora apresentado no Capítulo 5, onde foi feita uma comparação entre esses vetores obtidos pelo algoritmo e os valores obtidos pela literatura, neste capítulo serão somente apresentados, comparados e analisados soluções obtidas direta e exclusivamente do algoritmo do MEF implementado no projeto. Com o uso da Equação 4.44, num esquema implícito, é possível gerar soluções para o problema sem a presença do fluido. Com posse desse resultado, poderemos ver a influência do escoamento sobre o canal com geração de calor.

Além diso, também é importante descrever quais as condições de contorno dos problemas. Como são simulações parecidas, as condições inicias e as condições de contorno (do tipo Dirichlet e Neumann) do domínio também são comuns e estão descritas na Figura 6.3. Em geral, nas paredes há condição de não deslizamento, e a função corrente ψ varia entre 0 e 1 linearmente com o y de cada ponto do canal. Para cada parede interna o valor da função corrente é fixado como o a altura média y_p de cada parede.

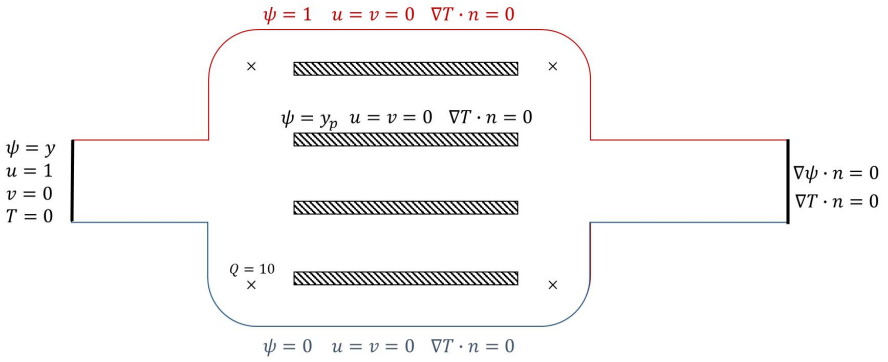


Figura 6.3: Condições iniciais e de contorno do problema com canais internos.

E finalmente, nos estudos de caso apresentados a seguir neste trabalho, existem 4 pontos em cada malha (Tabela 6.5) onde há geração de calor Q, como previsto no lado direito da Equação 3.11. No caso da Equação 4.44, para as simulações em de transferência de calor sem escoamento, o termo de geração de calor também aparece no lado direito, multiplicado pela matriz de massa **M**. Nos estudos, após testes com valores distintos, decidiu-se fixar a geração de calor em $\mathbf{Q} = 10$ nesses 4 pontos.

Tabela 6.5: Pontos no escoamento onde há geração de calor.

X (m)	**Y** (m)
4,0	1,0
4,0	11,0
12,0	1,0
12,0	11,0

Nas seções a seguir, serão expostas as condições específicas de cada simulação e os seus resultados, que foram agrupados de acordo com as modificações feitas, isto é, de acordo com a quantidade, altura e largura dos canais e paredes internos.

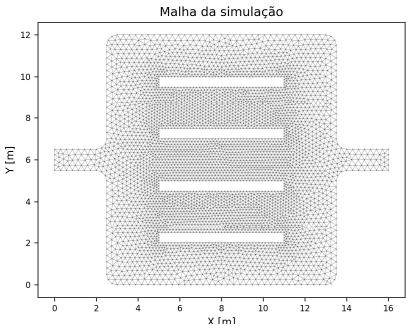
6.1 Diferentes Números de Canais Internos

Como introduzido anteriormente, os primeiros cenários de estudo são aqueles nos quais se variou a quantidade de paredes internas e, por consequência, a quantidade de canais internos. A quantidade de canais foi definida arbitrariamente, variando entre 5, 8 e 11 canais (ou 4, 7 e 10 paredes internas). Na Tabela 6.2 foram definidos os parâmetros fixos do canal externo onde o estudo é feito, porém, é necessário também apresentar outras propriedades específicas do domínio que variam nos estudos adiante. Na Tabela 6.6 abaixo estão listados as dimensões das paredes internas dos casos desta seção.

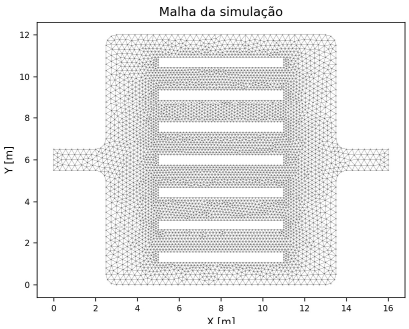
Tabela 6.6: Parâmetros das paredes internas do escoamento.

Malha	N5	N8	N11
Altura (m)	0,5	0,5	0,5
Largura (m)	6,0	6,0	6,0

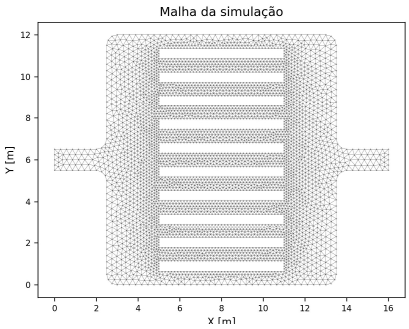
Definidas todas as dimensões do domínio, foram criadas as malhas para as simulações com números distintos de canais. Como mencionado antes, as malhas foram criadas utilizando o software *"GMSH"*, e estas estão expostas na Figura 6.4 a seguir.



(a) Malha N5



(b) Malha N8



(c) Malha N11

Figura 6.4: Vetores da simulação da malha 8L1.

Cada malha, por sua vez, apresenta atributos diferentes: a quantidade de nós e de elementos. Na Tabela 6.7 são listados tais valores para cada caso, além do tempo que cada simulação levou para processar.

Tabela 6.7: Quantidade de nós e elementos triangulares das malhas.

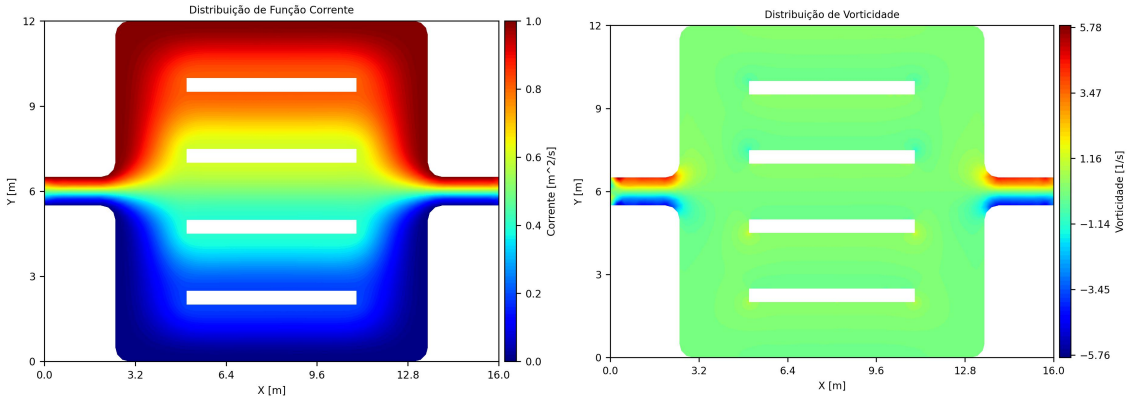
Malha	N5	N8	N11
Número de nós	4660	4868	4640
Número de elementos	8786	8944	8230
Tempo (s)	1680	1839	1616

Algo importante a se notar na tabela acima é o tempo de processamento de cada malha. Nota-se que quanto maior a quantidade de nós e elementos de malha, maior o tende a ser o tempo de processamento. Nesta seção, os tempos ficaram entre 25 e 30 minutos para cada simulação.

Definidas todas as condições do domínio, as características do fluido refrigerante, os parâmetros das malhas e do MEF, a seguir serão exibidos os resultados obtidos.

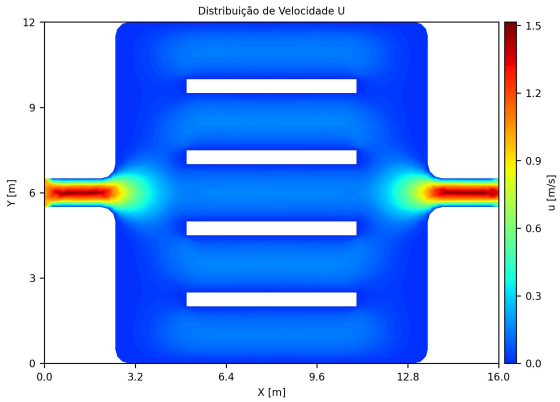
6.1.1 5 Canais Internos

No escoamento com 5 canais internos, ou 4 paredes internas, na Figura 6.5 estão os resultados para todos os vetores de interesse: função corrente ψ, vorticidade ω, velocidade **u**, velocidade **v** e temperaturas **T** com e sem escoamento passando pelo canal.

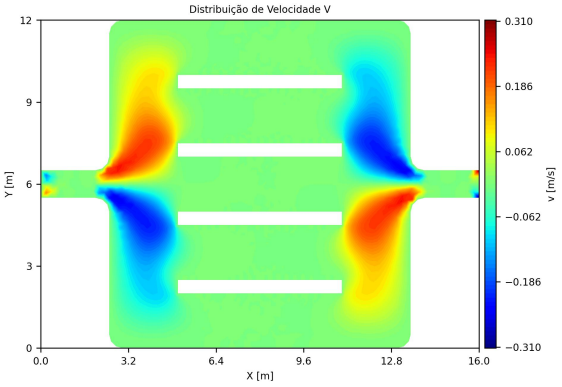


(a) Função corrente ψ

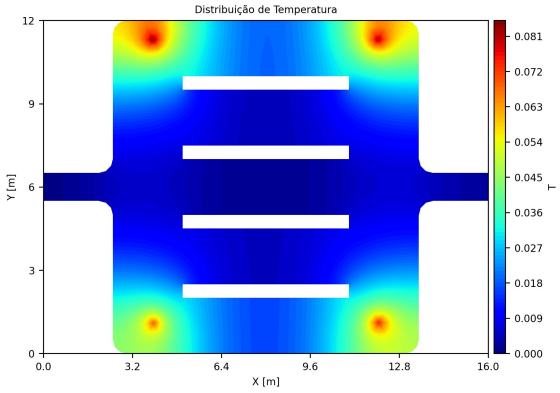
(b) Vorticidade ω



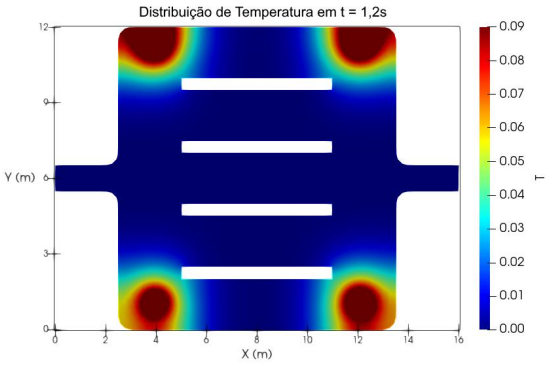
(c) Velocidade **u**



(d) Velocidade **v**



(e) Temperatura **T**



(f) Temperatura **T** (sem escoamento)

Figura 6.5: Vetores da simulação da malha N5.

No Gráfico 6.5a é possível observar o resultado da função corrente ψ, assim como observado nos estudos de validação, com um comportamento linear de acordo com a altura y, variando entre 0 e 1 nas paredes inferior e superior, respectivamente. Existe, porém, uma certa distorção nesse vetor no interior do escoamento, entre $x = 3$ e $x = 13$, por conta da mudança súbita de altura interior, por conta do fluido ter mais espaço para escoar. Tal distorção também é observada nos demais vetores. Por exemplo, nas regiões de entrada e saída do Gráfico 6.5c - entre $x \leq 3$ e $x \geq 13$, respectivamente -, o vetor velocidade $\mathbf{u}$ apresenta o comportamento parabólico - como visto na simulação de canal livre -, porém, com a abertura (e fechamento) do canal externo, a velocidade tende a diminuir no interior do escoamento. Nesse vetor também é possível ver que nas paredes a velocidade vai a zero, por conta da condição de não deslizamento preestabelecida.

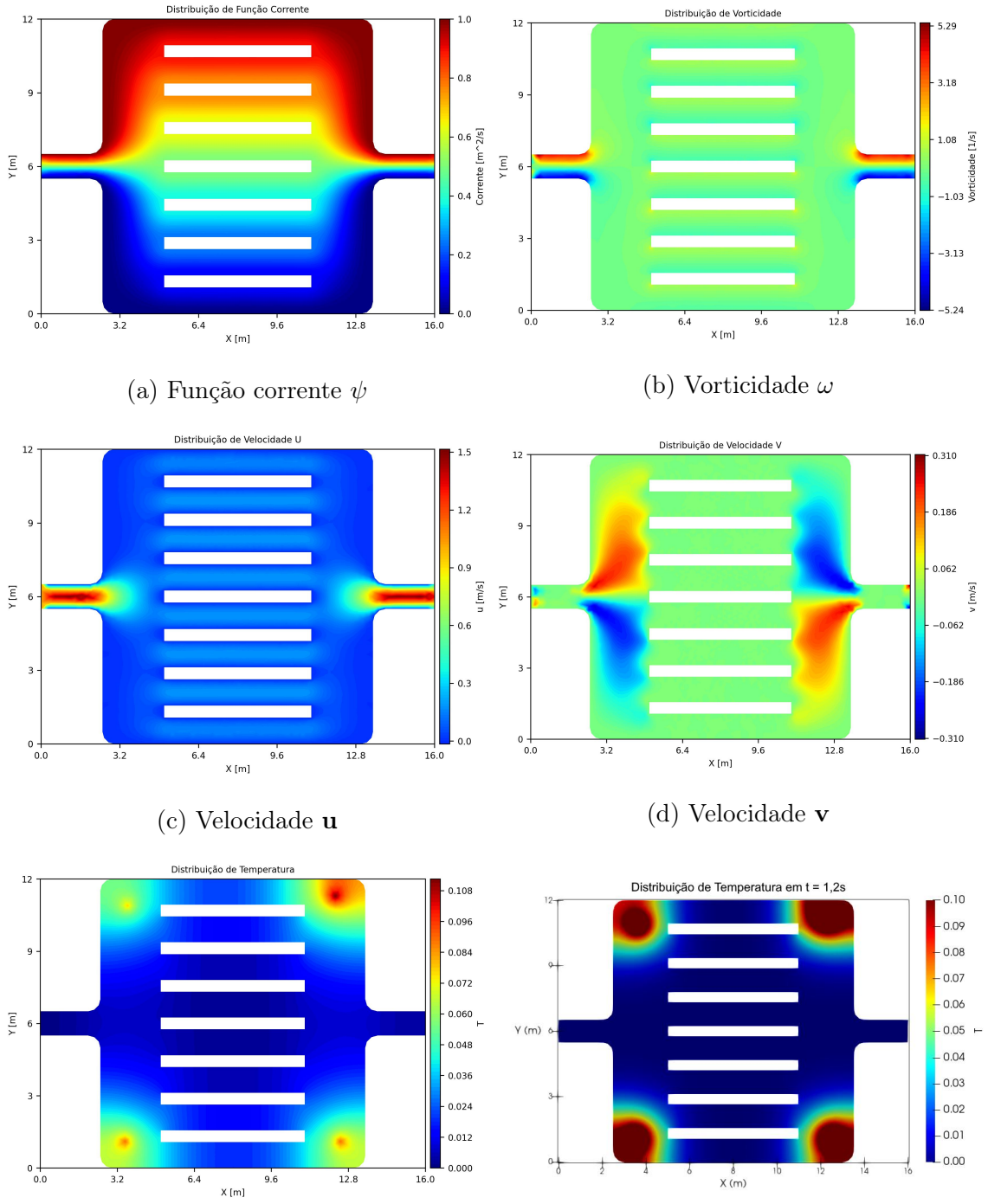
Em contrapartida, próximo ao que foi visto no canal livre, a vorticidade ω na Gráfico 6.5b apresentou valores pequenos, e estes estão majoritariamente localizados nas regiões de entrada e saída. No interior do escoamento a vorticidade possui valores bem próximos a zero.

O vetor velocidade $\mathbf{v}$, no Gráfico 6.5d, por sua vez, também apresentou comportamento similar ao presenciado no canal livre nas regiões de entrada e saída, mas, com a necessidade do fluido em escoar para o interior dos canais internos, podemos ver claramente um aumento no módulo desse vetor nas regiões de abertura e fechamento. Quando os valores de $\mathbf{v}$ estão positivos, significa que as partículas do fluido estão indo para cima, enquanto que para valores negativos, o fluido está se deslocando para baixo.

Por fim, quanto ao vetor temperatura $\mathbf{T}$ no Gráfico 6.5e, é possível observar a geração de calor em 4 pontos do interior do escoamento, conforme estabelecido na proposta do escoamento. Além disso, verifica-se que com a passagem do fluido refrigerante frio (com $T = 0$), o calor gerado nesses pontos tende a se espalhar na direção da parede externa, isto é, para longe das paredes internas. Por outro lado, se compararmos a distribuição de temperatura obtida pela simulação sem escoamento, no Gráfico 6.5f, obtido no *ParaView*, observamos que as temperaturas no canal sem fluido são significativamente maiores, chegando a ser de 3 a 4 vezes maior nos pontos de geração de calor, onde a temperatura é máxima.

6.1.2 8 Canais Internos

Nesta subseção, analisando a simulação de escoamento com 8 canais internos, ou 7 paredes internas, na Figura 6.6, são exibidos os resultados para os vetores ψ, ω, $\mathbf{u}$, $\mathbf{v}$ e $\mathbf{T}$.



(a) Função corrente ψ

(b) Vorticidade ω

(c) Velocidade $\mathbf{u}$

(d) Velocidade $\mathbf{v}$

(e) Temperatura $\mathbf{T}$

(f) Temperatura $\mathbf{T}$ (sem escoamento)

Figura 6.6: Vetores da simulação da malha N8.

Assim como nos resultados da malha N5 exibidos anteriormente, os vetores da malha N8 apresentaram comportamentos similares, com pequenas diferenças em seus valores. A função corrente, Gráfico 6.6a, por exemplo, apresentou variação linear nas regiões de entrada e saída, e em seu interior, os valores se adaptaram as condições impostas, como $\psi = y_p$ para cada parede. Já a vorticidade, Gráfico 6.6b, manifestou tanto comportamento quanto valores próximos ao da malha N5.

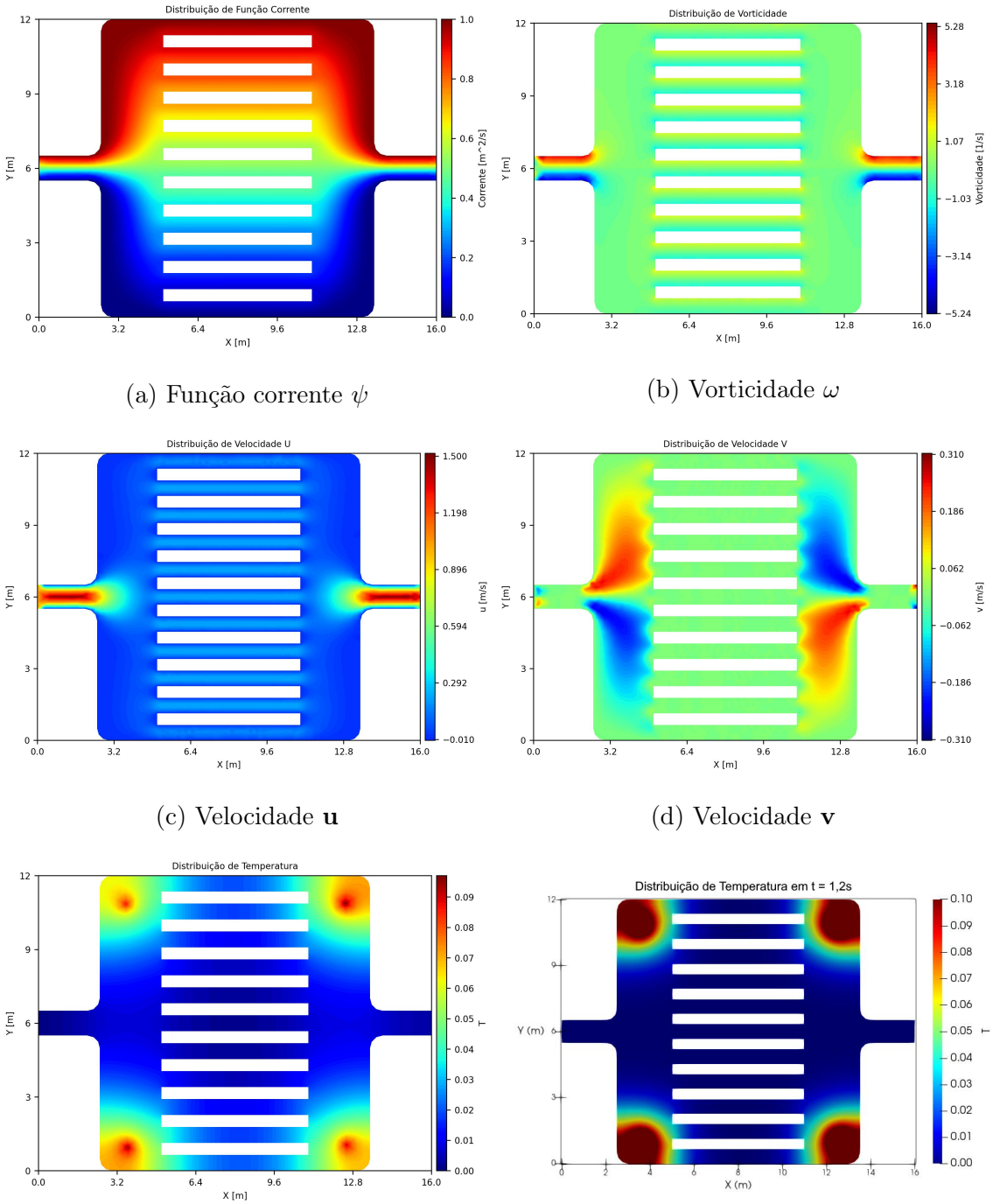
Em compensação, as velocidades **u** (Gráfico 6.6c) e **v** (Gráfico 6.6d) ostentaram valores ligeiramente distintos do caso anterior. Por conta da maior quantidade de paredes no interior do escoamento e, consequentemente, menor o espaço nos canais internos. Logo, para a mesma massa de fluido passar, as velocidades **u** e **v** são maiores. Por conta da escala desses gráficos, isso pode ser um pouco difícil de se observar diretamente. Portanto, na Subseção 6.1.4 é exposto um comparativo para essas e outras quantidades variando entre as simulações.

Finalmente, no Gráfico 6.6e, a solução de temperatura **T** apresentou resultados semelhantes em ambas as malhas N5 e N8, observando-se que os gráficos possuem escalas ligeiramente diferentes. Além disso, verificamos que assim como em N5, na simulação N8 de geração de calor sem escoamentos - Gráfico 6.6f obtido pelo *ParaView* - observamos temperaturas relativamente maiores do que no caso com escoamento fluindo, e mais contidas nos pontos onde o calor é produzido.

A razão da diferença dessas escala nesses e outros gráficos é por conta de limitações dos programas escritos em Python. Para se evitar escrever dezenas de funções para cada traçado de gráfico, criou-se uma função única que traça os vetores no domínio, que identifica por si só os limites de escala.

6.1.3 11 Canais Internos

Finalmente, na Figura 6.7 abaixo, sobre o escoamento com 11 canais internos, ou 10 paredes internas, são apresentados os gráficos resultantes para os vetores ψ, ω, **u**, **v** e **T**, assim como exposto nos casos anteriores.



(a) Função corrente ψ

(b) Vorticidade ω

(c) Velocidade **u**

(d) Velocidade **v**

(e) Temperatura **T**

(f) Temperatura **T** (sem escoamento)

Figura 6.7: Vetores da simulação da malha N11.

Como nos estudos das malhas N5 e N8, os vetores finais para a malha N11 são bem parecidos com os vetores observados nos gráficos anteriores. Em particular, observamos pouca ou nenhuma variação nos vetores ψ, Gráfico 6.7a, e ω, Gráfico 6.7b, com exceção do fato de existirem mais canais no escoamento. Além disso, novamente é possível ver um pequeno aumento no vetor $\mathbf{u}$, Gráfico 6.7c, entre as paredes internas; e um aumento na velocidade $\mathbf{v}$, Gráfico 6.7d, na região de abertura e fechamento do canal externo.

No Gráfico 6.7e para a temperatura $\mathbf{T}$ da malha N11, por sua vez, são observados tanto comportamento como valores de temperatura mais próximos aos obtidos no gráfico de $\mathbf{T}$ para a malha N5. Aqui também expõe-se que o calor gerado internamente no escoamento é difundido na direção da parede externa com a passagem de fluido refrigerante mais frio. Além disso, no Gráfico 6.7f, gerado pelo *ParaView*, também observamos claramente valores expressivamente maiores de temperaturas próximas aos pontos de geração de calor.

Outra conclusão que podemos extrair das distribuições de temperatura $\mathbf{T}$ em todos os casos, seja N5, N8 ou N11, é que a presença e movimento do fluido refrigerante no canal contribui efetivamente para reduzir as temperaturas no interior do escoamento.

6.1.4 Comparação entre N5, N8 e N11

Além da observação e análise superficial das distribuições dos vetores para os casos N5, N8 e N11, considerou-se também importante nesta subseção trazer uma comparação mais profunda sobre os resultados anteriores. No Gráfico 6.8, extraído dos resultados gerados pelo Python e lidos no software *ParaView*, está exposta a evolução da velocidade $\mathbf{u}$ no meio do escoamento, isto é, em $x = 8$, em cada uma das três malhas simuladas.

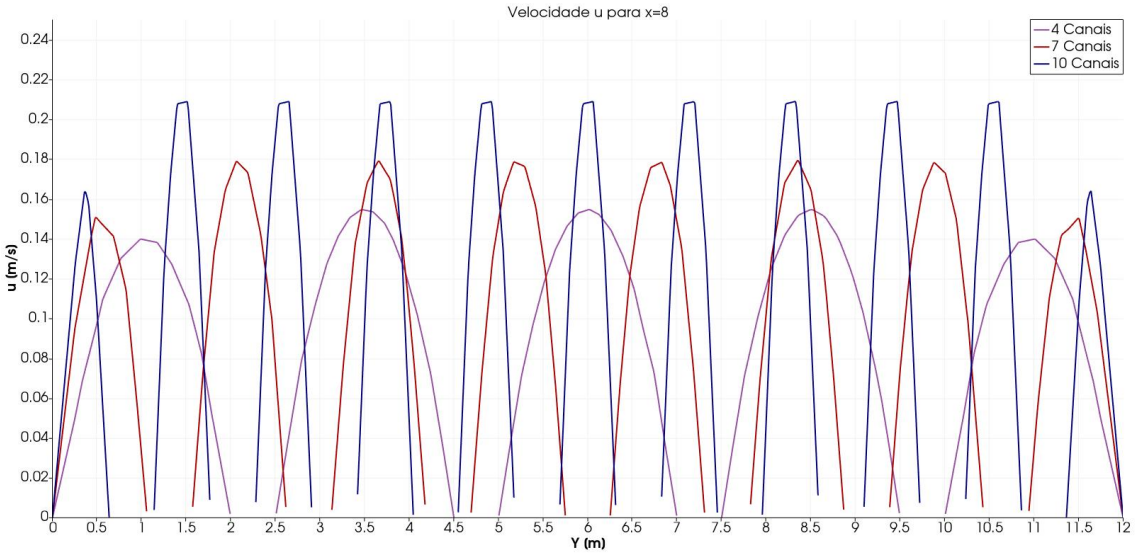


Figura 6.8: Gráfico de velocidade **u** para malhas com diferentes quantidades de canais.

Conforme esperado e descrito anteriormente, observamos que com um maior número de canais no interior do escoamento, maior é a velocidade no interior das paredes internas. Ou seja, em N5, o valor de **u** é o menor se comparado com os obtidos em N8 e N11, em que neste último caso são obtidos os maiores valores. Novamente, dependendo do objetivo do projeto que poderia sair com base nesses estudos, poderia-se optar por uma configuração de canais ou outra.

Por fim, outras informações podem ser extraídas das simulações N5, N8 e N11. Na Tabela 6.8 são resumidos alguns dados absolutos de cada simulação, como a grandeza máxima dos vetores **u**, **v** e **T** no interior do escoamento. Nessa tabela, é possível notar a diferença entre as temperaturas em canais com e sem escoamento.

Tabela 6.8: Resultados absolutos obtidos da simulação N5, N8 e N11.

Malha	Malha N5	Malha N8	Malha N11
Velocidade u máx. (m/s)	1,503	1,501	1,503
Velocidade v máx. (m/s)	0,309	0,309	0,309
Temperatura T máx. com escoamento	0,084	0,112	0,096
Temperatura T máx. sem escoamento	0,326	0,425	0,359

6.2 Diferentes Dimensões dos Canais Internos

Após o estudo de escoamentos com diferentes números de canais internos, outra proposta deste trabalho é estudar escoamentos com um número fixo de canais, porém com dimensões internas diferentes. Como introduzido anteriormente, foram feitos dois grupos de simulações: primeiro, variando-se a largura das paredes internas; e depois, modificando as alturas dessas paredes.

Com essas variações, é possível fazer comparações entre as soluções para os vetores de interesse encontradas anteriormente. Por isso, para as simulações apresentadas adiante, fixou-se o número de canais em 8, arbitrariamente, por ser o valor intermediário de canais.

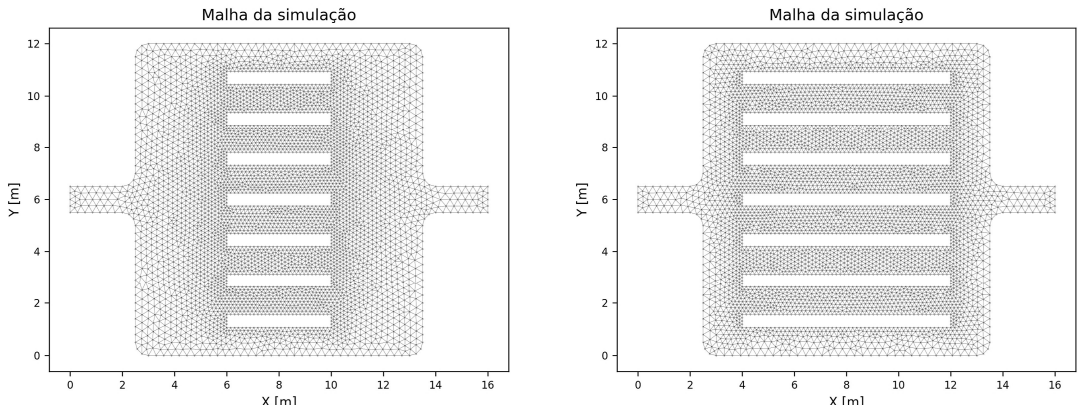
6.2.1 8 Canais Internos com Larguras Diferentes

Nesta seção, foram feitas simulações com base em domínios - e também malhas - cujas paredes internas possuem dimensões de largura distintos. Considerando a malha N8 como padrão, optou-se então por simular malhas com paredes curtas, a malha 8L1, e com paredes longas, malha 8L2.

O primeiro passo a ser feito é definir quais as dimensões da malha foram alteradas. Na Tabela 6.9 abaixo estão listados os dados de largura e altura da malha N8, assim como as novas dimensões das malhas 8L1 e 8L2. Essas malhas, criadas e obtidas pelo "*GMSH*", estão representadas na Figura 6.9.

Tabela 6.9: Parâmetros das paredes internas do escoamento.

Malha	N8	8L1	8L2
Altura (m)	0,5	0,5	0,5
Largura (m)	6,0	4,0	8,0



(a) Malha 8L1 (b) Malha 8L2

Figura 6.9: Malhas para simulação com paredes internas de larguras diferentes.

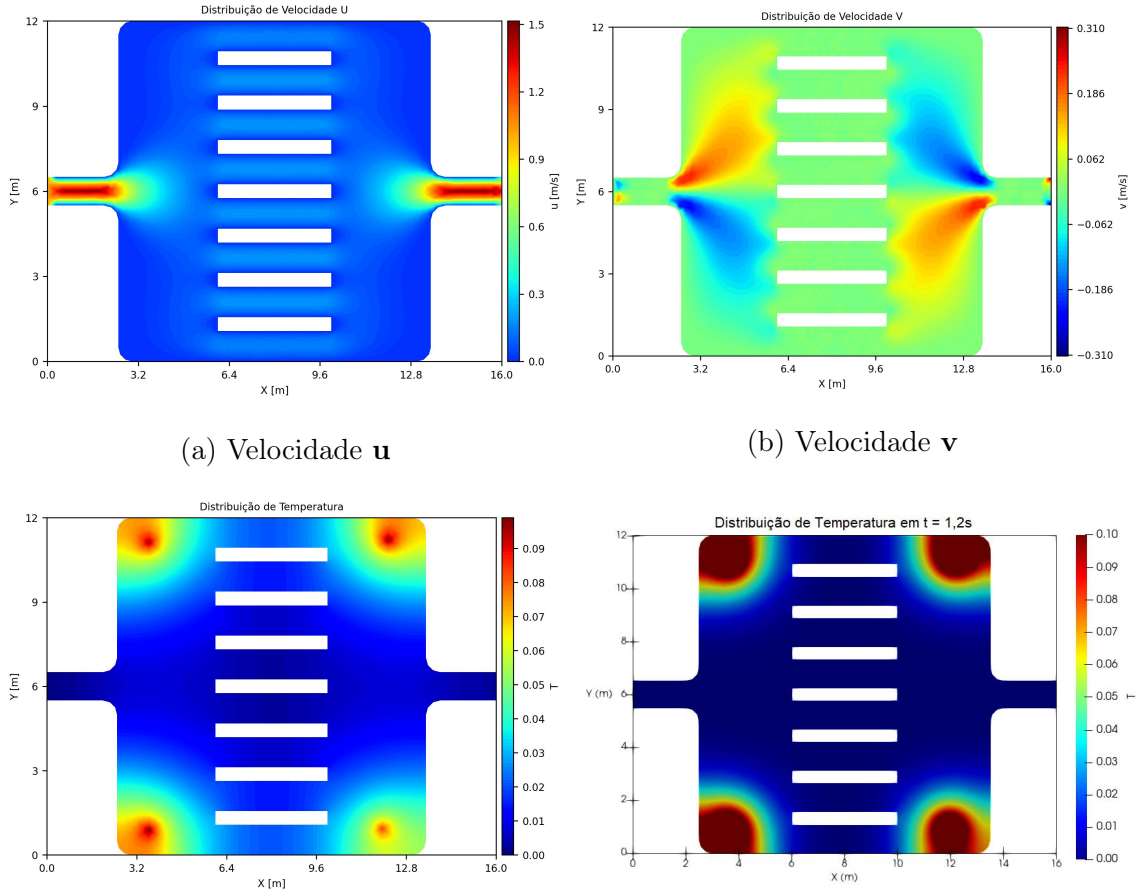
Como visto anteriormente, para a simulação de cada malha acima desenhada é essencial destacar quais os números de nós e de elementos. Na Tabela 6.10 esses dados estão listados para cada situação, além do tempo de simulação que cada caso tomou.

Tabela 6.10: Quantidade de nós e elementos triangulares das malhas.

Malha	Malha N8	Malha 8L1	Malha 8L2
Número de nós	4868	4675	5159
Número de elementos	8944	8740	9330
Tempo (s)	1839	1663	1760

A partir dessa tabela, nota-se então que por conta da quantidade próxima de nós e elementos, os tempos de simulação também ficaram bem próximos, novamente na ordem de 25 a 30 minutos por malha.

Uma vez que os demais parâmetros e condições do domínio, do fluido e do MEF permaneceram fixos, todos os requisitos para o estudo estão definidos. Na Figura 6.10 abaixo são apresentados as distribuições finais dos vetores de velocidade **u**, velocidade **v** e temperatura **T** da malha 8L1.



(a) Velocidade **u**

(b) Velocidade **v**

(c) Temperatura **T**

(d) Temperatura **T** (sem escoamento)

Figura 6.10: Vetores da simulação da malha 8L1.

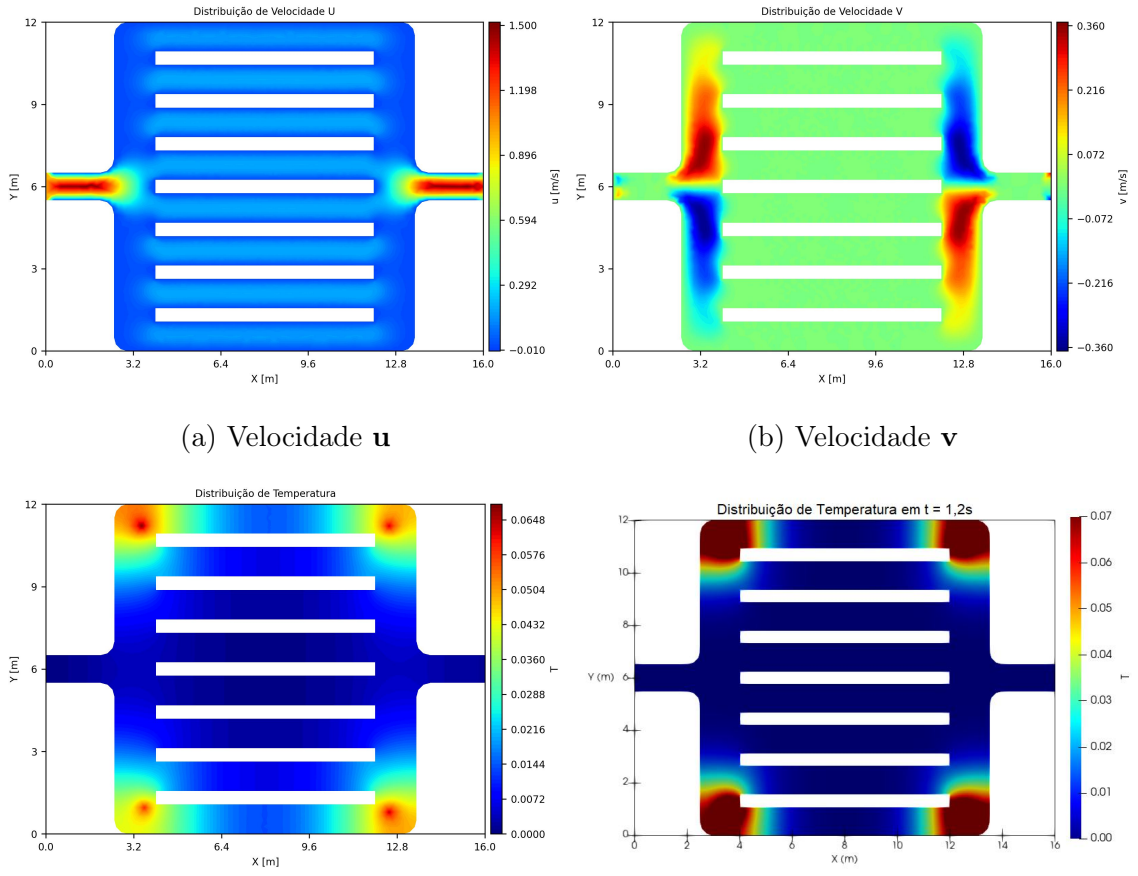
Primeiramente, nota-se a ausência dos gráficos da função corrente ψ e vorticidade ω. Uma vez que o comportamento e valores desses dois vetores pouco variou se comparado com as malhas N5, N8 e N11, optou-se então por omitir essas soluções.

De qualquer forma, com paredes internas mais curtas, notou-se algumas mudanças nos gráficos de **u** e **v**. No primeiro caso, no Gráfico 6.10a, novamente observa-se que na região de entrada e saída a velocidade **u** apresentou comportamento parabólico, assim como no caso de escoamento em canal livre. Além disso, por conta das paredes internas mais distantes da região de abertura (a partir de $x = 3$) e de fechamento (próximo de $x = 13$), o vetor **u** varia mais lentamente, seja diminuindo na região de entrada, seja aumentando na região de fechamento. Visualmente, não é possível afirmar se a velocidade **u** é maior ou menor do que visto na malha N8. Analogamente, por conta desse maior espaço para o fluido se movimentar, o vetor **v**, Gráfico 6.10b, também varia mais lentamente, indicando um movimento vertical

mais suave no fluido.

Por outro lado, a solução da temperatura **T** para a malha 8L1 também demonstrou comportamento parecido com o obtido na malha N8, isto é, o calor gerado internamente no escoamento tende a se espalhar na direção das paredes externas. Porém, por serem mais afastadas, pouco calor chega nas paredes internas e a temperatura nelas é menor daquelas observadas em N8. Além disso, observando o Gráfico 6.10d para temperatura sem escoamento, nota-se o mesmo comportamento observado em N8, em que a temperatura sem fluxo é muito maior se comparado ao caso com fluido refrigerante no interior.

A seguir, definidos os requisitos da simulação, na Figura 6.11 abaixo são apresentados os gráficos dos vetores de velocidade **u**, velocidade **v** e temperatura **T** da malha 8L2.



(a) Velocidade **u**

(b) Velocidade **v**

(c) Temperatura **T**

(d) Temperatura **T** (sem escoamento)

Figura 6.11: Vetores da simulação da malha 8L2.

Nos gráficos acima, com os canais mais longos em 8L2, é nítido a diferença com

as demais simulações. É claro, a distribuição geral dos vetores permaneceu bem próxima com o comportamento observado tanto em N8 como em 8L1, mas algumas distorções surgiram. Por conta do espaço estreito, a diferença mais nítida está no vetor **v**, no Gráfico 6.11b, onde há um aumento brusco e significativo na velocidade vertical, principalmente nas regiões de abertura e de fechamento. A velocidade **u**, no Gráfico 6.11a, por sua vez, apenas apresentou uma pequena distorção nessas regiões porque, com as paredes bem próximas, a condição de não deslizamento faz com que a velocidade caia a zero mais rápido do que visto em 8L1.

Por fim, observa-se no Gráfico 6.11c com a distribuição de temperatura que as paredes internas estão bem mais próximas das fontes de calor. Logo, apesar da tendência do escoamento espalhar o calor na direção da parede externa se repetir também na malha 8L2, as paredes internas apresentam temperaturas maiores do que observado em N8 e em 8L1. Novamente, no Gráfico 6.11d se nota que a temperatura no caso sem escoamento é consideravelmente maior.

6.2.2 Comparação entre N8, 8L1 e 8L2

Assim como na Seção 6.1.4, aqui também é feito um comparativo entre os resultados obtidos para as malhas com larguras distintas N8, 8L1 e 8L2.

Como analisado anteriormente nos gráficos de velocidade **u**, não é possível identificar, visivelmente, alguma variação desse vetor entre as malhas com mesmo número de canais. De fato, se observado o Gráfico 6.12 abaixo, veremos que, independentemente das dimensões das larguras das paredes internas, a velocidade no meio do escoamento (em $x = 8$) é a mesma.

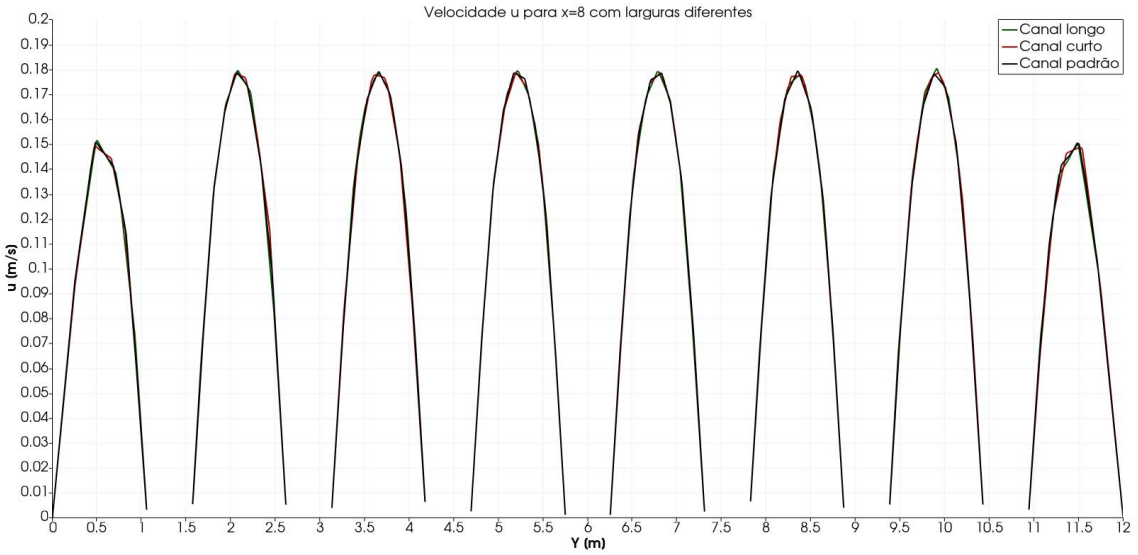


Figura 6.12: Gráfico de velocidade **u** para malhas com diferentes larguras.

Apesar das velocidades coincidirem no meio do canal, é importante ressaltar que estas devem ser distintas caso fosse analisado algum ponto próximo da abertura (ou fechamento) do canal externo, onde há distorção por conta das paredes próximas ou afastadas.

Outros dados absolutos das simulações são apresentadas na Tabela 6.11. Nela, observamos aquilo que havíamos constatado para a velocidade **v**: na simulação 8L2, por conta do espaço reduzido, o valor máximo da velocidade vertical é maior que as demais simulações. Além disso, nota-se a consistente diferença de temperatura entre simulações com e sem escoamento.

Tabela 6.11: Resultados absolutos obtidos da simulação N8, 8L1 e 8L2.

Malha	Malha N8	Malha 8L1	Malha 8L2
Velocidade u máx. (m/s)	1,501	1,501	1,504
Velocidade v máx. (m/s)	0,309	0,309	0,361
Temperatura T máx. com escoamento	0,112	0,099	0,060
Temperatura T máx. sem escoamento	0,425	0,301	0,193

6.2.3 8 Canais Internos com Alturas Diferentes

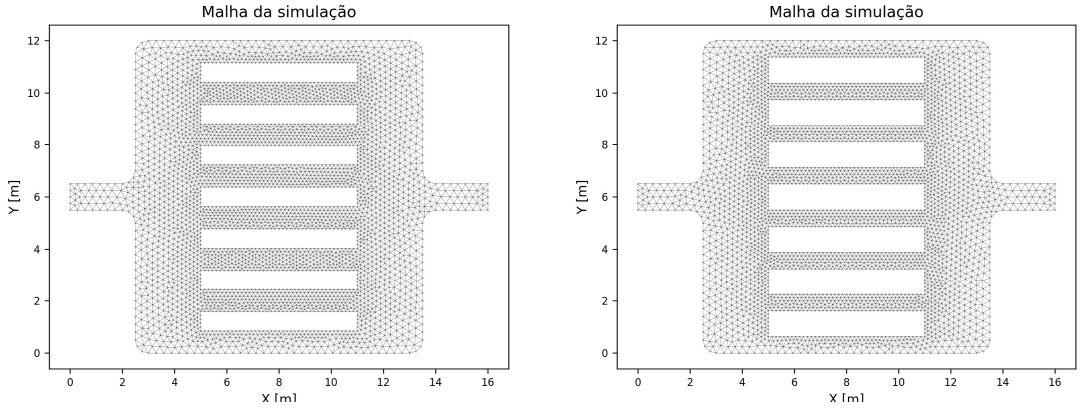
O último grupo de simulações a serem apresentados neste trabalho consiste em simular malhas com um número fixo de canais, variando-se apenas a altura das paredes internas. Diferentemente do estudo com diferentes larguras, nesta seção essas larguras internas serão fixas e em ambas as malhas 8A1 e 8A2 as alturas são maiores do que o padrão apresentado anteriormente, que possuía altura de $0,5$m.

Novamente, o primeiro passo consiste em definir quais mudanças foram aplicadas em cada malha simulada. Na Tabela 6.12 são listados as características das paredes internas das malhas N8, 8A1 e 8A2.

Tabela 6.12: Parâmetros das paredes internas do escoamento.

Malha	N8	8A1	8A2
Altura (m)	0,5	0,75	1,0
Largura (m)	6,0	6,0	6,0

Definidas as dimensões específicas de cada caso, foi possível criar as malhas com uso do "*GMSH*". Na Figura 6.13 é exibido as malhas 8A1 e 8A2 obtidas pelo software.



(a) Malha 8A1 (b) Malha 8A2

Figura 6.13: Malhas para simulação com paredes internas de alturas diferentes.

Uma vez criadas as malhas de estudo, outros parâmetros que precisam ser definidos para as simulações são os números de nós e de elementos de cada malha implementada. Na Tabela 6.13 estão listados esses valores, assim como o tempo

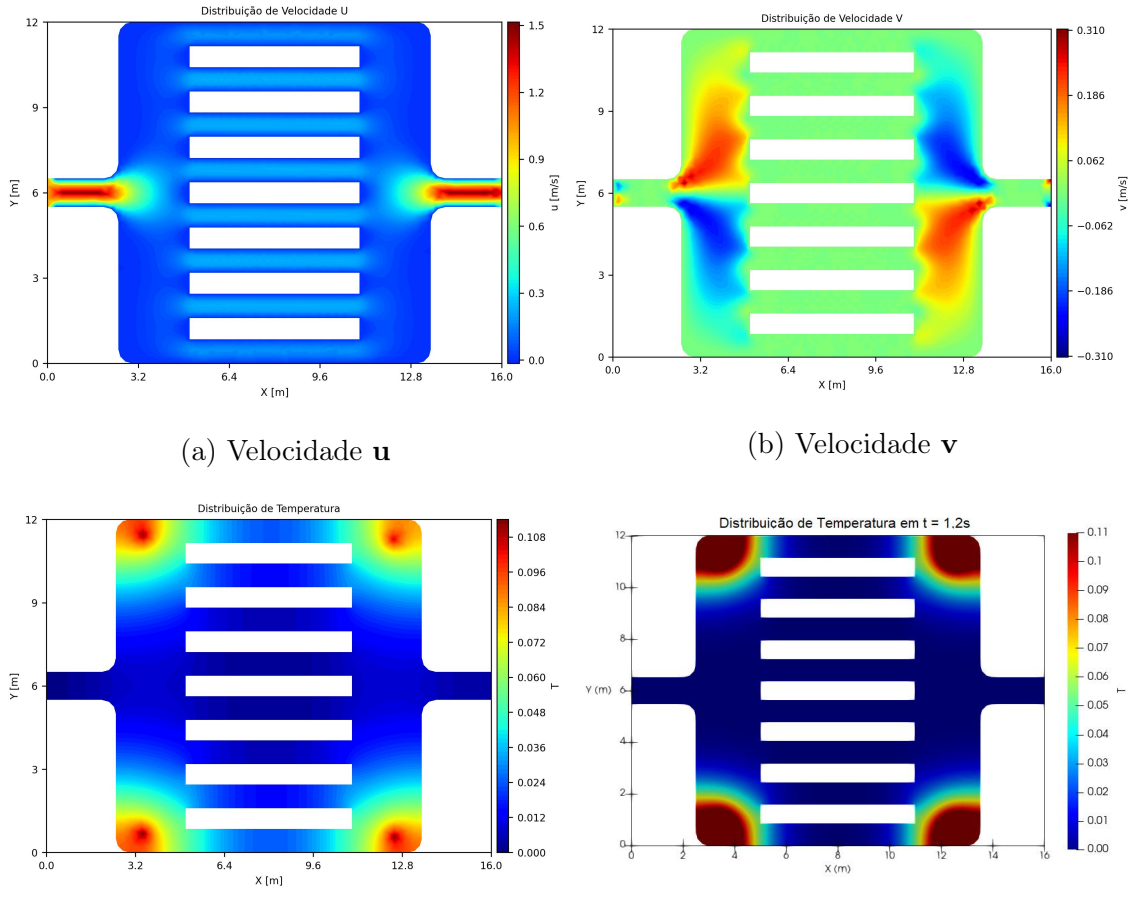
consumido no processamento de cada malha.

Tabela 6.13: Quantidade de nós e elementos triangulares das malhas.

Malha	Malha N8	Malha 8A1	Malha 8A2
Número de nós da malha	4868	4051	3659
Número de elementos da malha	8944	7296	6484
Tempo (s)	1839	1253	977

Na tabela acima é possível ver que, com o número bem disperso de nós e elementos, o tempo de processamento de cada simulação também variou significativamente, indo desde cerca de 15 minutos para a malha com menos nós (8A2) até 30 minutos para malha mais refinada (N8).

Definidos todos os requisitos da simulação, na Figura 6.14 são exibidos os gráficos dos vetores de velocidade **u**, velocidade **v** e temperatura **T** da malha 8A1.



(a) Velocidade **u**

(b) Velocidade **v**

(c) Temperatura **T**

(d) Temperatura **T** (sem escoamento)

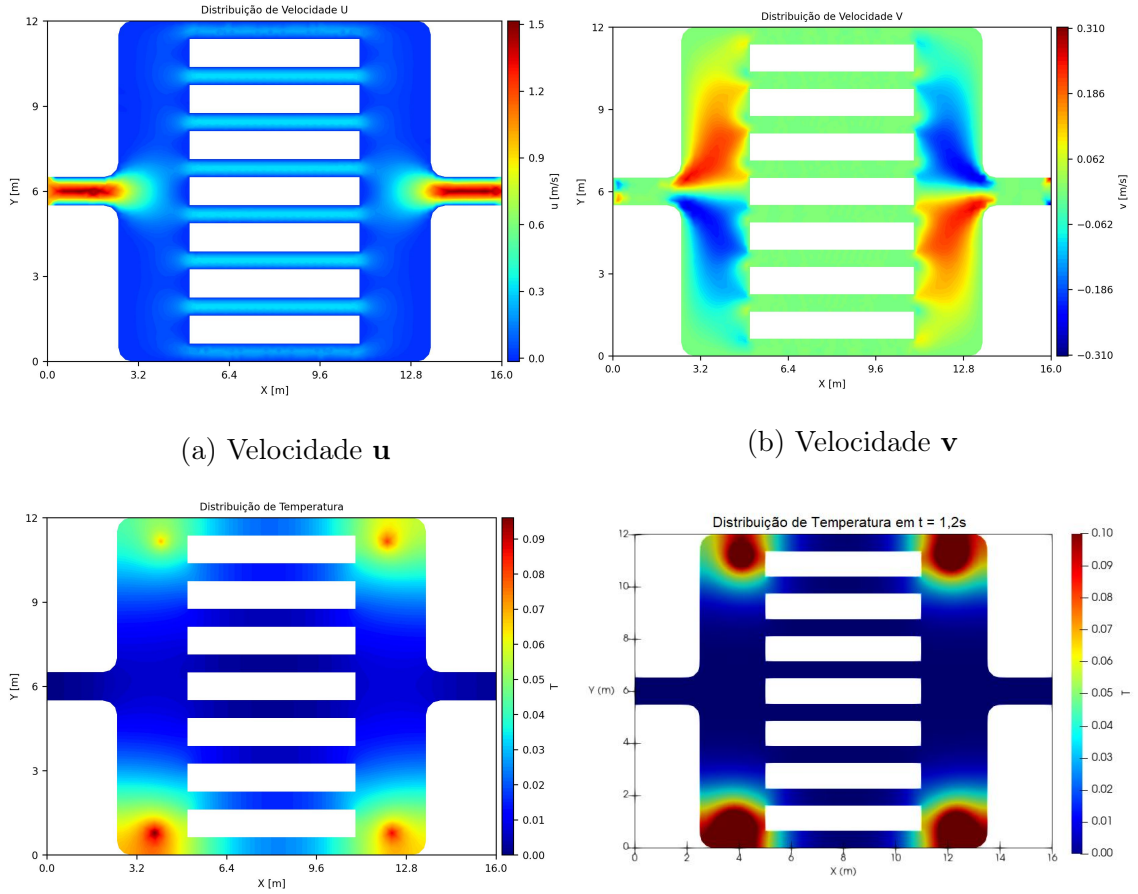
Figura 6.14: Vetores da simulação da malha 8A1.

O Gráfico 6.14b de velocidade **v** obtido na malha 8A1 é muito parecido, seja em valores quanto em comportamento, com o gráfico obtido para a malha N8, onde as paredes internas não causam significativa distorção na velocidade vertical do fluido. Nesse gráfico, como nos demais de velocidade **v**, é possível identificar o movimento do fluido subindo e descendo no escoamento.

Por outro lado, no Gráfico 6.14a de distribuição do vetor **u**, podemos identificar uma similaridade em relação ao obtido na malha N8 nas regiões de entrada, abertura, fechamento e saída ($x \leq 3$ e $x \geq 13$). Todavia, no meio do escoamento, com o espaço vertical mais estreito, podemos ver um leve aumento no módulo desse vetor. Assim como aconteceu no caso de quantidade de canais diferentes, como a mesma quantidade de fluido deve passar de um lado a outro do canal, com um espaço menor esperava-se e observou-se uma maior velocidade horizontal nos canais mais estreitos, como é o caso da malha 8A1.

Além disso, também identificamos certa proximidade na distribuição de temperatura **T**, no Gráfico 6.14c, com a solução obtida para a malha N8. Mas, podemos ver também que as paredes internas mais próximas da parede externa apresentam temperaturas um pouco maiores do que aquelas observadas em N8. Um dos possíveis motivos se deve a, como comentado acima, as velocidades horizontais entre as duas malhas são levemente distintas, sendo maior no caso da simulação de 8A1. Já no Gráfico 6.14d, novamente observamos que a temperatura é significativamente maior na simulação sem escoamento.

Com os mesmo requisitos estabelecidos para a malha 8A1, na Figura 6.15 são exibidos os gráficos de distribuição dos vetores de **u**, **v** e **T** obtidos para solução da malha 8A2.



(a) Velocidade **u**

(b) Velocidade **v**

(c) Temperatura **T**

(d) Temperatura **T** (sem escoamento)

Figura 6.15: Vetores da simulação da malha 8A2.

A grande mudança visível entre os resultados de 8A1 e 8A2 está no vetor **u**, no Gráfico 6.15a. Como explicado anteriormente, o espaço consideravelmente mais estreito nos canais internos faz com que a velocidade no meio do escoamento seja igualmente maior.

Já o resultado obtido para a velocidade vertical **v**, exposta no Gráfico 6.15b, é praticamente o mesmo apresentado pela simulação 8A1. É possível também observar certa similaridade entre o vetor de temperatura **T** da malha 8A2, no Gráfico 6.15c, e o obtido na malha 8A1 e N8; porém, considerando a diferença no vetor **u**, a temperatura nas paredes mais próximas do exterior são maiores. O comportamento do fluido de difundir o calor gerado no escoamento na direção da parede externa também é observado aqui. Além disso, nota-se significativo aumento na temperatura no estudo sem escoamento, Gráfico 6.15d, assim como observado nas demais malhas simuladas.

6.2.4 Comparação entre N8, 8A1 e 8A2

Por fim, assim como nas demais seções deste capítulo, também foi feito uma análise comparativa entre os resultados obtidos em N8, 8A1 e 8A2, para buscar entender a diferença que as alturas das paredes internas exercem sobre o escoamento.

Como comentado para os resultados obtidos nos gráficos precedentes, com o espaço vertical mais estreito entre os canais por conta das paredes com alturas maiores, foi possível identificar uma variação na velocidade horizontal **u**. De fato, no Gráfico 6.16 é exposto a distribuição desse vetor no meio do escoamento, isto é, em $x = 8$. Nesse gráfico se nota efetivamente que as velocidades nas malhas com alturas maiores são também maiores.

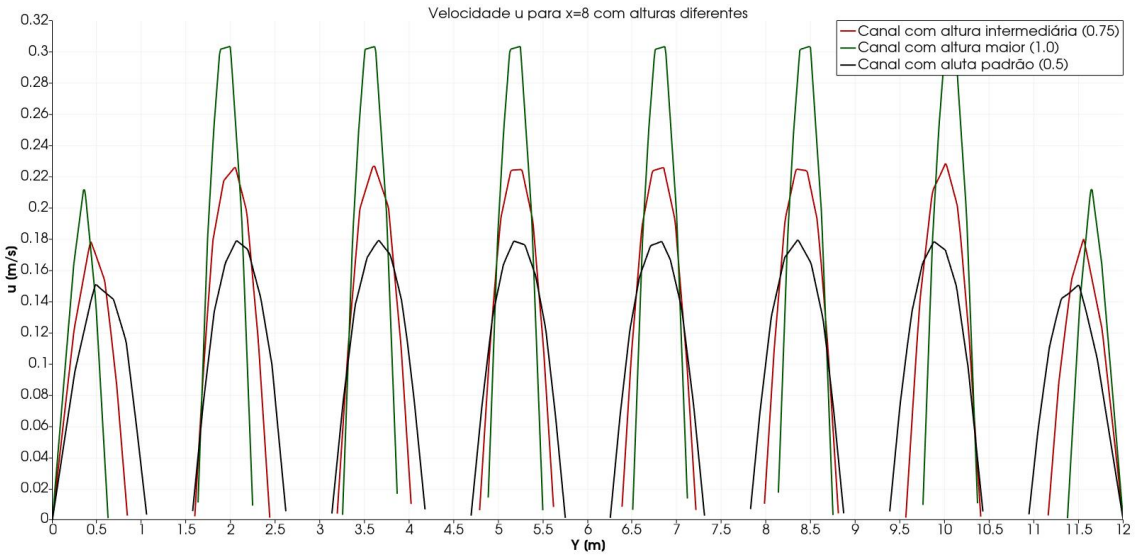


Figura 6.16: Gráfico de velocidade **u** para malhas com diferentes alturas.

Além disso, é importante notar na Tabela 6.14 abaixo que, entre as malhas estudadas nesta seção, houve pouca ou nenhuma diferença - globalmente - nos valores máximos absolutos das velocidades **u** e **v** e na temperatura **T**. É claro que, se observarmos localmente em diferentes pontos do escoamento, certamente se encontram variações nas soluções calculadas pelo algoritmo. Na tabela também é possível ver, numericamente, o significativo aumento na temperatura nos estudos com e sem escoamento.

Tabela 6.14: Resultados absolutos obtidos da simulação.

Malha	Malha N8	Malha 8A1	Malha 8A2
Velocidade u máx. (m/s)	1,501	1,501	1,501
Velocidade v máx. (m/s)	0,309	0,309	0,309
Temperatura T máx. com escoamento	0,112	0,113	0,096
Temperatura T máx. sem escoamento	0,425	0,328	0,274

Capítulo 7

Conclusões

7.1 Considerações Finais

O presente projeto tinha como objetivo mesclar os principais conceitos e equações elementares das áreas de mecânica dos fluidos e de transferência de calor e massa, aos princípios e ferramentas da computação científica, da dinâmica dos fluidos computacional (CFD) e de métodos numéricos (como o MEF) para estudar o escoamento de fluidos em diferentes situações. Além de casos clássicos e recorrentes da literatura de mecânica dos fluidos, o trabalho também buscou analisar situações reais e mais complexas, em particular, o escoamento no interior de canais com diferentes configurações de paredes (aletas) internas, também analisando como cada configuração afeta esse escoamento.

No primeiro momento, com o uso integrado entre a linguagem Python e o Método de Elementos Finitos (MEF), sob o esquema de Taylor-Galerkin, criou-se um algoritmo que modela e obtém dados de casos de escoamentos mais conhecidos da literatura, sendo eles o escoamento em canal de Hagen-Poiseuille e o escoamento em canal com cilindro. Além disso, para cada um desses dois casos, foram feitas simulações para diferentes malhas e diferentes números de Reynolds, a fim de se analisar a influência destes no modelo criado. O fato de simular casos conhecidos da literatura permitiu que os dados obtidos pela solução computacional fossem comparados com as soluções analíticas obtidos por autores dessas áreas. Com isso, foi possível validar o algoritmo e julgar que os resultados gerados se aproximaram significativamente aos obtidos pela bibliografia, com erros relativamente baixos, principalmente

com relação aos vetores de velocidade.

O projeto também não apenas se limitou a estudar somente os escoamentos mais simples e conhecidos. Na segunda parte do trabalho são propostas configurações comuns na área de arrefecimento de partes eletrônicos. Em particular, buscou-se analisar o comportamento, transporte e transferência de calor de um fluido refrigerante através de um canal que se abre e que possui paredes em seu interior, aletas, dividindo o canal externo em pequenos canais internos. Para verificar as influências dos parâmetros estabelecidos para esse problema, foram feitos três grupos de simulações: primeiro, variando-se a quantidade de paredes internas; e em seguida, dois grupos variando as dimensões das paredes internas (largura e altura). Nestes dois últimos casos, para que fosse possível uma comparação com o primeiro estudo, foi fixado a quantidade de canais internos em um número intermediário.

Por outro lado, por se tratarem de estudo de problemas específicos e mais complexos do que os clássicos da literatura, na parte final do trabalho não foi feita uma comparação entre soluções computacionais e analíticas. Em compensação, foram feitas comparações entre os resultados do algoritmo para cada grupo de simulações, sejam elas com ou sem fluido escoando. Como resultado, observamos não apenas que cada configuração e formas internas interfere nos parâmetros dos escoamentos, aumentando ou diminuindo velocidades e temperaturas; mas também que a presença do fluido no canal contribui significativamente para a redução da temperatura no interior do escoamento - já que com o fluido há convecção. Por exemplo, assim como era esperado, com o aumento no número de aletas internas, menor era o espaço para o fluido passar, o que acarretou em um aumento de velocidades, assim como interferiu na dissipação do calor. Dessa forma, os resultados foram consistentes e apresentaram comportamentos esperados para as condições inciais fornecidas.

Uma vez que as simulações e os resultados obtidos são satisfatórios e seguem o que se era esperado do modelo e das condições fornecidas a cada problema, o algoritmo pode ser adaptado a outras situações. Isso permite o engenheiro que criou o algoritmo, por exemplo, a estudar como melhorar um produto já criado, ou, por outro lado, pensar e criar um protótipo com bases nas suas observações computacionais.

Por fim, é importante ressaltar o quanto a computação cientifica vem crescendo

vertiginosamente nos últimos anos, com a melhoria desde novas bibliotecas e ferramentas nas antigas e novas linguagens de programação, assim como a criação de softwares que, assim como o algoritmo aqui criado, resolvam problemas de engenharia cada vez mais complexos. Uma das vantagens de criar manualmente o código responsável pelas simulações, como a deste projeto, é dar ao engenheiro não apenas a possibilidade de pensar e elaborar problemas que exigiriam elevados custos para testes práticos - que ainda são importantes em muitas aplicações -, mas também trazer as noções de vantagens, desvantagens e limitações que a computação cientifica traz e poderá trazer para a engenharia no futuro.

7.2 Trabalhos Futuros

O presente trabalho foi feito com o objetivo de estudar o arrefecimento de eletrônicos a partir de escoamentos em canais com uso do MEF. Com base nos resultados obtidos e o que fora apresentado neste projeto, como sugestão de trabalhos futuros para fins de comparação, recomenda-se:

- Fazer uso dos códigos elaborados aqui para estudar simulações em canais com outras configurações e formas de aletas;
- Implementar e comparar os resultados obtidos por outros métodos numéricos além do MEF, como Volumes Finitos ou Diferenças Finitas;
- Com uso dos códigos, elaborar e implementar simulações que possam ser feitas fisicamente para comparação e validação;
- Utilizar outras linguagens de programação e softwares, como *ANSYS* e *OpenFOAM*, para simular casos parecidos e comparar resultados;
- Implementar e expandir as simulações apresentadas aqui, aproveitando-se do MEF já implementado nos códigos, para rodar os mesmos estudos em 3D;
- Adicionar a implementação do código fluidos que possuam partículas em seus interiores, adicionando o movimento dessas partículas no estudo de resfriamento;

Referências Bibliográficas

[1] INTEL, "Cooling with Liquid", *CPU cooler: Liquid Cooling vs. Air Cooling*, 2021, <https://www.intel.com/content/www/us/en/gaming/resources/cpu-cooler-liquid-cooling-vs-air-cooling.html>. Último acesso em 25/10/2021.

[2] MOHAPATRA, S., "An overview of liquid coolants for electronics cooling", *Electronics Cooling*, v. 12, pp. 1–6, 01 2006.

[3] ONG, C. L., LAMAISON, N., MARCINICHEN, J. B., et al., "Two-phase minithermosyphon electronics cooling, Part 1: Experimental investigation". IEEE, May 2016.

[4] INTERNACIONAL, S., *SWEP Industrial Handbook*. 2021, Último acesso em 25/10/2021.

[5] YANG, D., WANG, Y., DING, G., et al., "Numerical and experimental analysis of cooling performance of single-phase array microchannel heat sinks with different pin-fin configurations", *Applied Thermal Engineering*, v. 112, pp. 1547–1556, 2017.

[6] WHITE, F., *Viscous Fluid Flow*. McGraw-Hill: New York, NY, 2021.

[7] FORTUNA, A. O., *Técnicas Computacionais para Dinâmica dos Fluídos Vol. 30*. Edusp, 2000.

[8] ZHAO, M., ZOU, Y., FU, Q., et al., "Effects of airfoil on aerodynamic performance of flapping wing", v. 1, pp. 100004, June 2021.

[9] ALI, A., RISI, R. D., SEXTOS, A., "Finite element modeling optimization of wind turbine blades from an earthquake engineering perspective", v. 222, pp. 111105, Nov. 2020.

[10] HARRIS, C. R., MILLMAN, K. J., VAN DER WALT, S. J., et al., "Array programming with NumPy", *Nature*, Sept. 2020.

[11] SCHLÖMER, N., "meshio", Aug. 2021.

[12] FISH, J., *A first course in finite elements*. John Wiley & Sons: Chichester, England Hoboken, NJ, 2007.

[13] COMMONS, W., "Human knee joint FE model", 2021, Último acesso em 15 de novembro de 2021.

[14] ÇENGEL, Y. A., BOLES, M. A., KANOĞLU, M., *THERMODYNAMICS: AN ENGINEERING APPROACH*. 9th ed. McGraw-Hill Education: New York, NY., 2019.

[15] MORAN, M. J., SHAPIRO, H. N., *Fundamentals of Engineering Thermodynamics*. 5th ed. John Wiley & Sons: Chicester, UK, 2006.

[16] BERGMAN, T., LAVINE, A., INCROPERA, F., et al., *Fundamentals of Heat and Mass Transfer*. Wiley, 2017.

[17] COMMONS, W., "CFD Forced Convection Heat Sink v2", 2021, Último acesso em 15 de novembro de 2021.

[18] ANJOS, G. R., *Computação Científica para Engenheiros*. UFRJ, 2020.

[19] DONEA, J., "A Taylor-Galerkin method for convective transport problems", v. 20, n. 1, pp. 101–119, Jan. 1984.

[20] GEUZAINE, CHRISTOPHE AND REMACLE, JEAN-FRANCOIS, "Gmsh", .

[21] CAMINHA, G., "The CFL Condition and How to Choose Your Timestep Size", Oct. 2019, Library Catalog: www.simscale.com Section: CAE Hub.

[22] BISWAS, P. G., "Notas de Aula do curso de Convecção e Transferência de Massa", 2015, <https://www.iitg.ac.in/gtm/files/Chapter3.pdf> Último acesso em 24 de maio de 2022.

[23] SANTOS, F. O., "Simulação numérica de escoamentos de fluidos utilizando diferenças finitas generalizadas", *ICMC-USP*, Oct. 2005.

[24] AHRENS, J. P., GEVECI, B., LAW, C. C., "ParaView: An End-User Tool for Large-Data Visualization". In: *The Visualization Handbook*, 2005.

Apêndice A

Códigos em Python utilizados no projeto

A.1 Script para casos de validação

```python
import os
import datetime
import meshio
import json
import time as tm
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.tri as mtri
from sys import platform as sys_pf
from colorama import Fore, Style
from helpers import print_message, create_gif, plot2D, plotParaview, plotMesh, build_kest

# abrindo arquivo de log
DATE = datetime.datetime.utcnow() - datetime.timedelta(hours=3)
DATE_STR = DATE.strftime("%Y_%m_%d_%H_%M")
LOG_NAME = f"logs/simulation_cil_{DATE_STR}.txt"
LOG_FILE = open(LOG_NAME, "w")

# parametros do output
DIR_NAME  = os.getcwd()
MSH_DIR = "msh"
FIG_DIR = "figures"
FILE_NAME = 'retangleFinN4.msh'
FIG_PATH = os.path.join(DIR_NAME, FIG_DIR)
# FLUID_NAME = "Water"
FLUID_NAME = "R1234ZE"

def simulation(reynolds_num=100, dtime=0.01, niter=120, mesh=FILE_NAME, q_list=[], sufix=""):
    print_message(f"Iniciando simula  o de Escoamento.\n----------{DATE}----------\n\n", LOG_FILE)
    start_process = datetime.datetime.now()
    msh_path = os.path.join(DIR_NAME, MSH_DIR, mesh)
    msh = meshio.read(msh_path)

    # leitura dos elementos da malha
    X = msh.points[:,0]
    Y = msh.points[:,1]
```

```
37      IEN = msh.cells['triangle']
38      IENbound = msh.cells['line']
39      IENboundTypeElem = list(msh.cell_data['line']['gmsh:physical'] - 1)
40      boundNames = list(msh.field_data.keys())
41      IENboundElem = [boundNames[elem] for elem in IENboundTypeElem]
42
43      print_message(f"Obtido malha {mesh[:-4]}\nQuantidade de n s: {len(X)}\nQuantidade de
        elementos na malha: {len(IEN)}\n\n", LOG_FILE)
44
45      # cc is the boundary condition indices
46      cc = np.unique(IENbound.reshape(IENbound.size))
47
48      # parametros do simulador
49      npoints = len(X)
50      ne = len(IEN)
51
52      # parametro geom trico
53      triang = mtri.Triangulation(X, Y, IEN)
54
55      # propriedades do fluido
56      props = {}
57      with open("properties.json") as fp:
58          props = json.load(fp)
59      fluid_props = props[FLUID_NAME]
60
61      # definicao das constantes do problema
62      dt = dtime
63      veloc = 1.0
64
65      reynolds = reynolds_num
66      prandtl = fluid_props["Pr"]
67      peclet = reynolds*prandtl
68
69      #wall_temp = 0.0
70      iter_limit = niter
71
72      # inicializando as matrizes K e M e os vetores F e Q
73      K =  np.zeros((npoints,npoints), dtype='float')
74      M =  np.zeros((npoints,npoints), dtype='float')
75      Gx =  np.zeros((npoints,npoints), dtype='float')
76      Gy =  np.zeros((npoints,npoints), dtype='float')
77
78      # loop dos elementos da malha
79      print_message(f"Criando matrizes de forma K, M, Gx e Gy...\n", LOG_FILE, color = Fore.YELLOW)
80      for e in range(0,ne):
81          ien_elem = IEN[e]
82          # area do elemento = 1/2 * det([X,Y,1]) = 1/2 * det([X_i, Y_i, 1], [X_j, Y_j, 1], [X_k,
        Y_k, 1])
83          det = X[ien_elem[2]]*(Y[ien_elem[0]]-Y[ien_elem[1]]) \
84              + X[ien_elem[0]]*(Y[ien_elem[1]]-Y[ien_elem[2]]) \
85              + X[ien_elem[1]]*(-Y[ien_elem[0]]+Y[ien_elem[2]])
86
87          area_elem = det/2.0
88
89          # matrizes do elemento linear
90          m = (area_elem/12.0) * np.array([ [2.0, 1.0, 1.0],
91                                            [1.0, 2.0, 1.0],
92                                            [1.0, 1.0, 2.0] ])
93
94          # formula do k do elementro triangular linear
95          b1 = Y[ien_elem[1]]-Y[ien_elem[2]]
96          b2 = Y[ien_elem[2]]-Y[ien_elem[0]]
97          b3 = Y[ien_elem[0]]-Y[ien_elem[1]]
98
99          c1 = X[ien_elem[2]]-X[ien_elem[1]]
100         c2 = X[ien_elem[0]]-X[ien_elem[2]]
```

```
101         c3 = X[ien_elem[1]]-X[ien_elem[0]]
102
103         # matriz do gradiente
104         B = (1.0/(2.0*area_elem)) * np.array([ [b1, b2, b3],
105                                                [c1, c2, c3] ])
106         # matriz do divergente
107         BT = B.transpose()
108
109         kele = area_elem*np.dot(BT,B)
110
111         gxele = (1.0/6.0)*np.array([ [b1, b2, b3],
112                                      [b1, b2, b3],
113                                      [b1, b2, b3] ])
114         gyele = (1.0/6.0)*np.array([ [c1, c2, c3],
115                                      [c1, c2, c3],
116                                      [c1, c2, c3] ])
117
118         for i in range(0,3):
119             ii = IEN[e,i]
120             for j in range(0,3):
121                 jj = IEN[e,j]
122
123                 # montagem (assembling) das matrizes K e M
124                 K[ii,jj] = K[ii,jj] + kele[i,j]
125                 M[ii,jj] = M[ii,jj] + m[i,j]
126                 Gx[ii,jj] = Gx[ii,jj] + gxele[i,j]
127                 Gy[ii,jj] = Gy[ii,jj] + gyele[i,j]
128
129     print_message("Iniciando processo iterativo...\n\n", LOG_FILE, color = Fore.WHITE)
130
131     # inicializar os campos vx, vy e T
132     vx = np.zeros( (npoints,1),dtype='float')
133     vy = np.zeros( (npoints,1),dtype='float')
134     temp = np.zeros( (npoints,1),dtype='float')
135     q = np.zeros( (npoints,1),dtype='float')
136     # for i in [1391, 2383, 2280, 1663]:
137     for i in q_list:
138         q[i] = 10.0
139
140     # impondo cc para vx, vy, temp e pressure
141     eps = 1e-6
142     for i in cc:
143         if X[i] - eps < X.min():
144             vx[i] = veloc
145             vy[i] = 0.0
146
147     # criando array para pontos do interior da malha (sem contorno)
148     inner = [x for x in range(0,npoints) if x not in cc]
149     cc2 = [x for x in cc if not X[x] + eps > X.max()]
150
151     count = 0
152     try:
153         while count < iter_limit:
154             start_time = round(tm.time(), 3)
155             print_message(f"Itera   o {count+1} no intervalo {round(count*dt,2)}    {round(count
        *dt+dt,2)}s\n", LOG_FILE, color = Fore.WHITE, print_cmd=False)
156
157             # calculo do omega para inclusao no contorno
158             b = np.dot(Gx,vy) - np.dot(Gy,vx)
159             omega = np.linalg.solve(M,b)
160             omegacc = omega.copy()
161
162             # zerando os valores de omegacc no interior da malha
163             for i in inner:
164                 omegacc[i] = 0.0
165
```

```
166             # resolvendo o transporte da vorticidade
167             A = (1.0/dt) * M + (1.0/reynolds)*K
168
169             # v \dot \nabla \omega
170             vgo = vx*np.dot(Gx,omega) + vy*np.dot(Gy,omega)
171
172             # matriz de difus o artificial para estabiliza  o
173             Kest = build_kest(X, Y, IEN, vx, vy)
174
175             # vetor do lado direito para eq. de transporte da vorticidade (omega)
176             b_1 = (1.0/dt) * np.dot(M,omega) - vgo - (dt/2.0)*np.dot(Kest, omega)
177
178             # imposicao das ccs para omega
179             for i in cc:
180                 A[i,:] = 0.0 # zerando toda a linha
181                 A[i,i] = 1.0 # impondo 1 na diagonal
182                 b_1[i] = omegacc[i]
183
184
185             # resolve eq. transporte solve
186             omega = np.linalg.solve(A,b_1)
187
188             # resolver funcao corrente K\psi = M*\omega
189             b_2 = np.dot(M,omega)
190
191             # lado esquerdo da eq K\psi = M*\omega
192             A_2 = K.copy()
193
194             # identificacao das ccs de psi
195             psicc =  np.zeros( (npoints,1), dtype='float')
196             eps = 1e-6
197             for i in cc2:
198                 if Y[i] - eps < Y.min():
199                     psicc[i] = 0.0
200                 elif Y[i] + eps > Y.max():
201                     psicc[i] = 1.0
202                 elif X[i] - eps < X.min():
203                     psicc[i] = Y[i] - 5.5
204                 elif X[i] - eps < 3.0 or X[i] + eps > 13.0:
205                     if Y[i] - eps < 5.5:
206                         psicc[i] = 0.0
207                     elif Y[i] + eps > 6.5:
208                         psicc[i] = 1.0
209                 elif Y[i] - eps < 2.5:
210                     psicc[i] = 2.25/12.0
211                 elif Y[i] - eps < 5.0:
212                     psicc[i] = 4.75/12.0
213                 elif Y[i] - eps < 7.5:
214                     psicc[i] = 7.25/12.0
215                 elif Y[i] - eps < 10.0:
216                     psicc[i] = 9.75/12.0
217
218             # imposicao das ccs de \psi
219             for i in cc2:
220                 A_2[i,:] = 0.0 # zerando toda a linha
221                 A_2[i,i] = 1.0 # impondo 1 na diagonal
222                 b_2[i] = psicc[i]
223
224             psi = np.linalg.solve(A_2,b_2)
225
226             # encontrar velocidades M vx = Gy \psi, M vy = -Gx \psi
227
228             b_3 = np.dot(Gy,psi)
229             vx = np.linalg.solve(M,b_3)
230
231             b_4 = -1.0*np.dot(Gx,psi)
```

```
232             vy = np.linalg.solve(M,b_4)
233
234             # impor cc de velocidade nos vetores vx e vy
235             # impondo cc para vx e vy
236             eps = 1e-6
237             for i in cc2:
238                 if X[i] - eps < X.min():
239                     vx[i] = veloc
240                     vy[i] = 0.0
241                 else:
242                     vx[i] = 0.0
243                     vy[i] = 0.0
244
245             # resolvendo a transferencia de calor
246             A_3 = (1.0/dt) * M + (1.0/peclet)*K
247
248             # v \dot \nabla \temperature
249             vgt = vx*np.dot(Gx,temp) + vy*np.dot(Gy,temp)
250
251             # vetor do lado direito para eq. de transferencia de calor (temp)
252             b_5 = (1.0/dt) * np.dot(M,temp) - vgt + np.dot(M, q)
253
254             # resolve eq. transporte solve
255             temp = np.linalg.solve(A_3,b_5)
256             for i in cc2:
257                 if X[i] - eps < X.min():
258                     temp[i] = 0.0
259
260             end_time = round(tm.time(), 3)
261             print_message(f"Tempo de processamento da itera  o: {round(end_time - start_time,
        3)}s\n", LOG_FILE, color = Fore.GREEN, print_cmd=False)
262
263             count += 1
264
265     except KeyboardInterrupt:
266         print_message("------------------------------\nSimula  o interrompida pelo usu rio!",
        LOG_FILE, color = Fore.YELLOW)
267     else:
268         plot2D(triang, npoints, psi, "Distribui  o de Fun  o Corrente", FIG_PATH, f"{sufix}
        Corrente_Re{reynolds}", f"Corrente [m^2/s]")
269         plot2D(triang, npoints, temp, "Distribui  o de Temperatura", FIG_PATH, f"{sufix}
        Temperatura_Re{reynolds}", f"T", manual_clb_ticks=False)
270         plot2D(triang, npoints, omega, "Distribui  o de Vorticidade", FIG_PATH, f"{sufix}
        Vorticidade_Re{reynolds}", f"Vorticidade [1/s]")
271         plot2D(triang, npoints, vx, "Distribui  o de Velocidade U", FIG_PATH, f"{sufix}u_Re{
        reynolds}", "u [m/s]", norm=[vx, vy])
272         plot2D(triang, npoints, vy, "Distribui  o de Velocidade V", FIG_PATH, f"{sufix}v_Re{
        reynolds}", "v [m/s]")
273         plotParaview(msh, f"fin/{sufix}colormap", vx, vy, psi, omega, temp=temp)
274
275         end_process = datetime.datetime.now()
276         process_time = int((end_process - start_process).total_seconds())
277         print_message(f"------------------------------\nFim da simula  o! Total de itera  es: {
        count}\nTempo de processamento: {process_time}s\n\n", LOG_FILE, color = Fore.GREEN)
278     return X, Y, vx, vy, temp
279
280 x,y,u,v,t = simulation(reynolds_num=0.23, dtime=0.005, niter=240, q_list=[1391, 2383, 2280,
        1663], sufix="N4_")
281 with open("N4.txt", "w", encoding="utf-8") as file:
282     file.write(f"Velocidade u m x = {max(u)}\n")
283     file.write(f"Velocidade v m x = {max(v)}\n")
284     file.write(f"Temperatura T m x = {max(t)}\n")
```

A.2 Script para casos de estudo

A.2.1 Script para casos sem fluido escoando

```
import sys
import os
import datetime
import meshio
import sys
import time as tm
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.tri as mtri
from sys import platform as sys_pf
from colorama import Fore, Style
from helpers import print_message, create_gif, plot2D, plotParaview, plotMesh, build_kest

# abrindo arquivo de log
DATE = datetime.datetime.utcnow() - datetime.timedelta(hours=3)
DATE_STR = DATE.strftime("%Y_%m_%d_%H_%M")
LOG_NAME = f"logs/sim_N4_temp_{DATE_STR}.txt"
LOG_FILE = open(LOG_NAME, "w")

# parametros do output
DIR_NAME  = os.getcwd()
MSH_DIR = "msh"
FIG_DIR = "figures"
FILE_NAME = 'retangleFinN4.msh'
FIG_PATH = os.path.join(DIR_NAME, FIG_DIR)
# FLUID_NAME = "Water"
FLUID_NAME = "R1234ZE"

def simulation(dtime=0.01, niter=120, mesh=FILE_NAME, q_list=[], sufix=""):
    print_message(f"Iniciando simula  o de Escoamento.\n----------{DATE}----------\n\n",
    LOG_FILE)
    start_process = datetime.datetime.now()
    msh_path = os.path.join(DIR_NAME, MSH_DIR, mesh)
    msh = meshio.read(msh_path)

    # leitura dos elementos da malha
    X = msh.points[:,0]
    Y = msh.points[:,1]
    IEN = msh.cells['triangle']
    IENbound = msh.cells['line']
    IENboundTypeElem = list(msh.cell_data['line']['gmsh:physical'] - 1)
    boundNames = list(msh.field_data.keys())
    IENboundElem = [boundNames[elem] for elem in IENboundTypeElem]

    print_message(f"Obtido malha {mesh[:-4]}\nQuantidade de n s: {len(X)}\nQuantidade de
    elementos na malha: {len(IEN)}\n\n", LOG_FILE)

    # parametros do simulador
    npoints = len(X)
    ne = len(IEN)

    # parametro geom trico
    triang = mtri.Triangulation(X, Y, IEN)

    # definicao das constantes do problema
    dt = dtime

    #wall_temp = 0.0
    iter_limit = niter

```

```
    # inicializando as matrizes K e M e os vetores F e Q
    K =  np.zeros((npoints,npoints), dtype='float')
    M =  np.zeros((npoints,npoints), dtype='float')

    # loop dos elementos da malha
    print_message(f"Criando matrizes K e M...\n", LOG_FILE, color = Fore.YELLOW)
    for e in range(0,ne):
        ien_elem = IEN[e]
        # area do elemento = 1/2 * det([X,Y,1]) = 1/2 * det([X_i, Y_i, 1], [X_j, Y_j, 1], [X_k,
     Y_k, 1])
        det = X[ien_elem[2]]*(Y[ien_elem[0]]-Y[ien_elem[1]]) \
            + X[ien_elem[0]]*(Y[ien_elem[1]]-Y[ien_elem[2]]) \
            + X[ien_elem[1]]*(-Y[ien_elem[0]]+Y[ien_elem[2]])

        area_elem = det/2.0

        # matrizes do elemento linear
        m = (area_elem/12.0) * np.array([ [2.0, 1.0, 1.0],
                                    [1.0, 2.0, 1.0],
                                    [1.0, 1.0, 2.0] ])

        # formula do k do elementro triangular linear
        b1 = Y[ien_elem[1]]-Y[ien_elem[2]]
        b2 = Y[ien_elem[2]]-Y[ien_elem[0]]
        b3 = Y[ien_elem[0]]-Y[ien_elem[1]]

        c1 = X[ien_elem[2]]-X[ien_elem[1]]
        c2 = X[ien_elem[0]]-X[ien_elem[2]]
        c3 = X[ien_elem[1]]-X[ien_elem[0]]

        # matriz do gradiente
        B = (1.0/(2.0*area_elem)) * np.array([ [b1, b2, b3],
                                        [c1, c2, c3] ])
        # matriz do divergente
        BT = B.transpose()

        kele = area_elem*np.dot(BT,B)

        for i in range(0,3):
            ii = IEN[e,i]
            for j in range(0,3):
                jj = IEN[e,j]

                # montagem (assembling) das matrizes K e M
                K[ii,jj] = K[ii,jj] + kele[i,j]
                M[ii,jj] = M[ii,jj] + m[i,j]

    print_message("Iniciando processo iterativo...\n\n", LOG_FILE, color = Fore.WHITE)

    # inicializar os campos vx, vy e T
    vx = np.zeros( (npoints,1),dtype='float')
    vy = np.zeros( (npoints,1),dtype='float')
    psi = np.zeros( (npoints,1),dtype='float')
    omega = np.zeros( (npoints,1),dtype='float')
    temp = np.zeros( (npoints,1),dtype='float')
    q = np.zeros( (npoints,1),dtype='float')

    for i in q_list:
        print(f"Ponto {i}: X = {X[i]} ; Y = {Y[i]}")
        q[i] = 10.0

    print("Vetores T e Q criados!\n")
    sys.exit(0)
    count = 0
    times = [0]
    results = [temp]
```

```
124     try:
125         writer = meshio.XdmfTimeSeriesWriter(f"./figures/{sufix}temp_paraview.vtk")
126         writer.write_points_cells(msh.points, msh.cells)
127         writer.write_data(0, point_data={"temp": temp})
128         while count < iter_limit:
129             start_time = round(tm.time(), 3)
130             print_message(f"Itera   o {count+1} no intervalo {round(count*dt,2)}    {round(count
        *dt+dt,2)}s\n", LOG_FILE, color = Fore.WHITE, print_cmd=False)
131 
132             # resolvendo a transferencia de calor
133             A = (1.0/dt) * M + K
134 
135             # vetor do lado direito para eq. de transferencia de calor (temp)
136             b = (1.0/dt) * np.dot(M,temp) + np.dot(M, q)
137 
138             # resolve eq. transporte solve
139             temp = np.linalg.solve(A,b)
140 
141             writer.write_data((count+1)*dt, point_data={"temp": temp})
142 
143             end_time = round(tm.time(), 3)
144             print_message(f"Tempo de processamento da itera   o: {round(end_time - start_time,
        3)}s\n", LOG_FILE, color = Fore.GREEN, print_cmd=True)
145             if count == 0:
146                 print("Plotando imagem...")
147                 plot2D(triang, npoints, temp, f"Distribui   o de Temperatura em t=0s", FIG_PATH,
         f"temp/{sufix}Temperatura_Flowless_t0", f"T", norm=[], manual_clb_ticks=False)
148             elif count in [119, 239]:
149                 print("Plotando imagem...")
150                 plot2D(triang, npoints, temp, f"Distribui   o de Temperatura em t={(count+1)*dt}
        s", FIG_PATH, f"temp/{sufix}Temperatura_Flowless_t{count+1}", f"T", norm=[],
        manual_clb_ticks=False)
151             count += 1
152 
153     except KeyboardInterrupt:
154         print_message("-------------------------------\nSimula   o interrompida pelo usu rio!",
        LOG_FILE, color = Fore.YELLOW)
155     else:
156         # plot2D(triang, npoints, temp, "Distribui   o de Temperatura", FIG_PATH, f"temp/{sufix}
        Temperatura_Flowless", f"T", manual_clb_ticks=False)
157         # plotParaview(msh, f"temp/{sufix}colormap_flowless", vx, vy, psi, omega, temp=temp)
158 
159         end_process = datetime.datetime.now()
160         process_time = int((end_process - start_process).total_seconds())
161         print_message(f"-------------------------------\nFim da simula   o! Total de itera   es: {
        count}\nTempo de processamento: {process_time}s\n\n", LOG_FILE, color = Fore.GREEN)
162     return temp
163 
164 # t4 = simulation(dtime=0.01, niter=240, q_list=[1391, 2383, 2280, 1663], sufix="N4_")
165 # with open("N4_Temp.txt", "w", encoding="utf-8") as file:
166 #     file.write(f"Temperatura T m x = {max(t4)}\n")
167 
168 t7 = simulation(mesh='retangleFinN7.msh', dtime=0.01, niter=240, q_list=[1520, 1574, 2184, 1514],
        sufix="N7_")
169 with open("N7_Temp.txt", "w", encoding="utf-8") as file:
170     file.write(f"Temperatura T m x = {max(t7)}\n")
171 
172 t10 = simulation(mesh='retangleFinN10.msh', dtime=0.01, niter=240, q_list=[2013, 2389, 2218,
        1782], sufix="N10_")
173 with open("N10_Temp.txt", "w", encoding="utf-8") as file:
174     file.write(f"Temperatura T m x = {max(t10)}\n")
```

A.2.2 Script para casos com fluido escoando

```
import os
import datetime
import meshio
import json
import time as tm
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.tri as mtri
from sys import platform as sys_pf
from colorama import Fore, Style
from helpers import print_message, plot2D, plot2Danalytical, relative_error

# abrindo arquivo de log

# parametros do output
DIR_NAME  = os.getcwd()
MSH_DIR = "msh"
FIG_DIR = "figures"
FILE_NAME = 'retangleSimpleRef1.msh'
FIG_PATH = os.path.join(DIR_NAME, FIG_DIR, "simple")
# FLUID_NAME = "Water"
FLUID_NAME = "R1234ZE"

def simulation(reynolds_num=100, dtime=0.01, niter=120, mesh=FILE_NAME, mesh_index=""):
    DATE = datetime.datetime.utcnow() - datetime.timedelta(hours=3)
    DATE_STR = DATE.strftime("%Y_%m_%d_%H_%M")
    LOG_NAME = f"logs/simulation_simp{mesh_index}_Re{reynolds_num}_{DATE_STR}.txt"
    LOG_FILE = open(LOG_NAME, "w")

    print_message(f"Iniciando simula  o de Escoamento.\n----------{DATE}----------\n\n",
    LOG_FILE)
    start_process = datetime.datetime.now()
    msh_path = os.path.join(DIR_NAME, MSH_DIR, mesh)
    msh = meshio.read(msh_path)

    # leitura dos elementos da malha
    X = msh.points[:,0]
    Y = msh.points[:,1]
    IEN = msh.cells['triangle']
    IENbound = msh.cells['line']
    IENboundTypeElem = list(msh.cell_data['line']['gmsh:physical'] - 1)
    boundNames = list(msh.field_data.keys())
    IENboundElem = [boundNames[elem] for elem in IENboundTypeElem]

    print_message(f"Obtido malha {mesh[:-4]}\nQuantidade de n s: {len(X)}\nQuantidade de
    elementos na malha: {len(IEN)}\n\n", LOG_FILE)

    # cc is the boundary condition indices
    cc = np.unique(IENbound.reshape(IENbound.size))

    # parametros do simulador
    npoints = len(X)
    ne = len(IEN)

    # parametro geom trico
    triang = mtri.Triangulation(X, Y, IEN)

    # propriedades do fluido
    props = {}
    with open("properties.json") as fp:
        props = json.load(fp)
    fluid_props = props[FLUID_NAME]

    # definicao das constantes do problema
```

```
63      dt = dtime
64      veloc = 1.0
65
66      reynolds = reynolds_num
67      prandtl = fluid_props["Pr"]
68      peclet = reynolds*prandtl
69
70      #wall_temp = 0.0
71      iter_limit = niter
72
73      # inicializando as matrizes K e M e os vetores F e Q
74      K =  np.zeros((npoints,npoints), dtype='float')
75      M =  np.zeros((npoints,npoints), dtype='float')
76      Gx =  np.zeros((npoints,npoints), dtype='float')
77      Gy =  np.zeros((npoints,npoints), dtype='float')
78
79      # loop dos elementos da malha
80      print_message(f"Criando matrizes de forma K, M, Gx e Gy...\n", LOG_FILE, color = Fore.YELLOW)
81      for e in range(0,ne):
82          ien_elem = IEN[e]
83          # area do elemento = 1/2 * det([X,Y,1]) = 1/2 * det([X_i, Y_i, 1], [X_j, Y_j, 1], [X_k,
        Y_k, 1])
84          det = X[ien_elem[2]]*(Y[ien_elem[0]]-Y[ien_elem[1]]) \
85              + X[ien_elem[0]]*(Y[ien_elem[1]]-Y[ien_elem[2]]) \
86              + X[ien_elem[1]]*(-Y[ien_elem[0]]+Y[ien_elem[2]])
87
88          area_elem = det/2.0
89
90          # matrizes do elemento linear
91          m = (area_elem/12.0) * np.array([ [2.0, 1.0, 1.0],
92                                            [1.0, 2.0, 1.0],
93                                            [1.0, 1.0, 2.0] ])
94
95          # formula do k do elementro triangular linear
96          b1 = Y[ien_elem[1]]-Y[ien_elem[2]]
97          b2 = Y[ien_elem[2]]-Y[ien_elem[0]]
98          b3 = Y[ien_elem[0]]-Y[ien_elem[1]]
99
100         c1 = X[ien_elem[2]]-X[ien_elem[1]]
101         c2 = X[ien_elem[0]]-X[ien_elem[2]]
102         c3 = X[ien_elem[1]]-X[ien_elem[0]]
103
104         # matriz do gradiente
105         B = (1.0/(2.0*area_elem)) * np.array([ [b1, b2, b3],
106                                                [c1, c2, c3] ])
107         # matriz do divergente
108         BT = B.transpose()
109
110         kele = area_elem*np.dot(BT,B)
111
112         gxele = (1.0/6.0)*np.array([ [b1, b2, b3],
113                                      [b1, b2, b3],
114                                      [b1, b2, b3] ])
115         gyele = (1.0/6.0)*np.array([ [c1, c2, c3],
116                                      [c1, c2, c3],
117                                      [c1, c2, c3] ])
118
119         for i in range(0,3):
120             ii = IEN[e,i]
121             for j in range(0,3):
122                 jj = IEN[e,j]
123
124                 # montagem (assembling) das matrizes K e M
125                 K[ii,jj] = K[ii,jj] + kele[i,j]
126                 M[ii,jj] = M[ii,jj] + m[i,j]
127                 Gx[ii,jj] = Gx[ii,jj] + gxele[i,j]
```

```
128                 Gy[ii,jj] = Gy[ii,jj] + gyele[i,j]
129
130     print_message("Iniciando processo iterativo...\n\n", LOG_FILE, color = Fore.WHITE)
131
132     # inicializar os campos vx, vy e T
133     vx = np.zeros( (npoints,1),dtype='float')
134     vy = np.zeros( (npoints,1),dtype='float')
135     temp = np.zeros( (npoints,1),dtype='float')
136
137     # impondo cc para vx, vy, temp e pressure
138     eps = 1e-6
139     for i in cc:
140         if X[i] - eps < X.min():
141             vx[i] = veloc
142             vy[i] = 0.0
143         elif Y[i] - eps < Y.min():
144             temp[i] = 1.0
145         elif Y[i] + eps > Y.max():
146             temp[i] = 1.0
147
148     # criando array para pontos do interior da malha (sem contorno)
149     inner = [x for x in range(0,npoints) if x not in cc]
150     cc2 = [x for x in cc if not X[x] + eps > X.max()]
151
152     count = 0
153     try:
154         while count < iter_limit:
155             start_time = round(tm.time(), 3)
156             print_message(f"Itera   o {count+1} no intervalo {round(count*dt,2)}    {round(count
        *dt+dt,2)}s\n", LOG_FILE, color=Fore.BLUE, print_cmd=False)
157
158             # calculo do omega para inclusao no contorno
159             b = np.dot(Gx,vy) - np.dot(Gy,vx)
160             omega = np.linalg.solve(M,b)
161             omegacc = omega.copy()
162
163             # zerando os valores de omegacc no interior da malha
164             for i in inner:
165                 omegacc[i] = 0.0
166
167             # resolvendo o transporte da vorticidade
168             A = (1.0/dt) * M + (1.0/reynolds)*K
169
170             # v \dot \nabla \omega
171             vgo = vx*np.dot(Gx,omega) + vy*np.dot(Gy,omega)
172
173             # vetor do lado direito para eq. de transporte da vorticidade (omega)
174             b_1 = (1.0/dt) * np.dot(M,omega)  - vgo
175
176             # imposicao das ccs para omega
177             for i in cc:
178                 A[i,:] = 0.0 # zerando toda a linha
179                 A[i,i] = 1.0 # impondo 1 na diagonal
180                 b_1[i] = omegacc[i]
181
182
183             # resolve eq. transporte solve
184             omega = np.linalg.solve(A,b_1)
185
186             # resolver funcao corrente K\psi = M*\omega
187             b_2 = np.dot(M,omega)
188
189             # lado esquerdo da eq K\psi = M*\omega
190             A_2 = K.copy()
191
192
```

```
193             # identificacao das ccs de psi
194             psicc =  np.zeros( (npoints,1), dtype='float')
195             eps = 1e-6
196             for i in cc2:
197                 if Y[i] - eps < Y.min():
198                     psicc[i] = 0.0
199                 elif Y[i] + eps > Y.max():
200                     psicc[i] = Y.max()
201                 elif X[i] - eps < X.min():
202                     psicc[i] = Y[i]
203 
204             # imposicao das ccs de \psi
205             for i in cc2:
206                 A_2[i,:] = 0.0 # zerando toda a linha
207                 A_2[i,i] = 1.0 # impondo 1 na diagonal
208                 b_2[i] = psicc[i]
209 
210             psi = np.linalg.solve(A_2,b_2)
211 
212             # encontrar velocidades M vx = Gy \psi, M vy = -Gx \psi
213 
214             b_3 = np.dot(Gy,psi)
215             vx = np.linalg.solve(M,b_3)
216 
217             b_4 = -1.0*np.dot(Gx,psi)
218             vy = np.linalg.solve(M,b_4)
219 
220             # impor cc de velocidade nos vetores vx e vy
221             # impondo cc para vx e vy
222             eps = 1e-6
223             for i in cc2:
224                 if X[i] - eps < X.min():
225                     vx[i] = veloc
226                     vy[i] = 0.0
227                 else:
228                     vx[i] = 0.0
229                     vy[i] = 0.0
230 
231             # resolvendo a transferencia de calor
232             A_3 = (1.0/dt) * M + (1.0/peclet)*K
233 
234             # v \dot \nabla \temperature
235             vgt = vx*np.dot(Gx,temp) + vy*np.dot(Gy,temp)
236 
237             # vetor do lado direito para eq. de transferencia de calor (temp)
238             b_5 = (1.0/dt) * np.dot(M,temp) - vgt
239 
240             # resolve eq. transporte solve
241             temp = np.linalg.solve(A_3,b_5)
242             for i in cc2:
243                 if X[i] - eps < X.min():
244                     temp[i] = 0.0
245                 elif Y[i] - eps < Y.min():
246                     temp[i] = 1.0
247                 elif Y[i] + eps > Y.max():
248                     temp[i] = 1.0
249 
250             end_time = round(tm.time(), 3)
251             print_message(f"Tempo de processamento da itera   o: {round(end_time - start_time,
        3)}s\n", LOG_FILE, color=Fore.GREEN, print_cmd=False)
252 
253             count += 1
254 
255     except KeyboardInterrupt:
256         print_message("-----------------------------\nSimula   o interrompida pelo usu rio!",
        LOG_FILE, color = Fore.YELLOW)
```

```
257     else:
258         plot2D(triang, npoints, psi, "Distribui  o de Fun  o Corrente", FIG_PATH, f"
        Corrente_Re{reynolds}", f"Corrente [m^2/s]")
259         plot2D(triang, npoints, temp, "Distribui  o de Temperatura", FIG_PATH, f"Temperatura_Re
        {reynolds}", f"T")
260         plot2D(triang, npoints, omega, "Distribui  o de Vorticidade", FIG_PATH, f"
        Vorticidade_Re{reynolds}", f"Vorticidade [1/s]")
261         plot2D(triang, npoints, vx, "Distribui  o de Velocidade U", FIG_PATH, f"u_Re{reynolds}"
        , "u [m/s]", norm=[vx, vy])
262         plot2D(triang, npoints, vy, "Distribui  o de Velocidade V", FIG_PATH, f"v_Re{reynolds}"
        , "v [m/s]")
263
264         end_process = datetime.datetime.now()
265         process_time = int((end_process - start_process).total_seconds())
266         print_message(f"-------------------------\nFim da simula  o! Total de itera  es: {
        count}\nTempo de processamento: {process_time}s\n\n", LOG_FILE, color = Fore.GREEN)
267         LOG_FILE.close()
268     return X, Y, vx, vy, temp
269
270 Re = 100
271 msh_list = [1,2,3,4,5]
272 dt = [0.01, 0.01, 0.01, 0.005, 0.004]
273 n = [120, 120, 120, 240, 300]
274 msh_name = FILE_NAME.replace("1", "{}")
275 array = []
276 for i in range(len(msh_list)):
277     res = simulation(reynolds_num=Re, dtime=dt[i], niter=n[i], mesh=msh_name.format(msh_list[i]),
        mesh_index=i+1)
278     array.append(res)
279 array1 = [(row[0], row[1], row[-1]) for row in array]
280
281 analytics = open(f"logs/analytics_simp_Re{Re}.txt", "w")
282 unum, usol = plot2Danalytical(array, Re, suffix="_simp")
283 for i in range(len(msh_list)):
284     err = relative_error(unum[i], usol[i])
285     print_message(f"Malha {i+1}: erro relativo u = {round(err, 3)}%", analytics, color=Fore.WHITE
        )
286
287 tnum, tsol = plot2Danalytical(array1, Re, print_sol=True, vector_name="T", ylabel="T ", title="
        Temperatura T(x, y)", suffix="_simp")
288 for i in range(len(msh_list)):
289     err = relative_error(tnum[i], tsol[i])
290     print_message(f"Malha {i+1}: erro relativo T = {round(err, 3)}%", analytics, color=Fore.WHITE
        )
291 analytics.close()
```