Politécnica
UFRJ

ESTUDO DE GEOMETRIAS DE CANAIS EM PLACA FRIA PARA ARREFECIMENTO DE ELETRÔNICOS ATRAVÉS DO MÉTODO DE ELEMENTOS FINITOS

Lucas Mendes Miranda

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 2023

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
Departamento de Engenharia Mecânica
DEM/POLI/UFRJ

ESTUDO DE GEOMETRIAS DE CANAIS EM PLACA FRIA PARA ARREFECIMENTO DE ELETRÔNICOS ATRAVÉS DO MÉTODO DE ELEMENTOS FINITOS

Lucas Mendes Miranda

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.

Aprovado por:

Prof. Gustavo Rabello dos Anjos

Prof. Gustavo César Rachid Bodstein

Prof. Manuel Ernani de Carvalho Cruz

RIO DE JANEIRO, RJ - BRASIL

Junho DE 2023

Mendes Miranda, Lucas

ESTUDO DE GEOMETRIAS DE CANAIS EM PLACA FRIA PARA ARREFECIMENTO DE ELETRÔNICOS ATRAVÉS DO MÉTODO DE ELEMENTOS FINITOS/Lucas Mendes Miranda. – Rio de Janeiro: UFRJ/ Escola Politécnica, 2023.

XI, 73 p.: il.; 29, 7cm.

Orientador: Gustavo Rabello dos Anjos

Projeto de Graduação – UFRJ/ Escola Politécnica/ Curso de Engenharia Mecânica, 2023.

Referências Bibliográficas: p. 62 – 64.

1. Método de elementos finitos. 2. Transferência de calor. 3. Mecânica dos fluidos. I. Rabello dos Anjos, Gustavo. II. Universidade Federal do Rio de Janeiro, Escola Politécnica, Curso de Engenharia Mecânica. III. Título.

Agradecimentos

Gostaria de agradecer primeiramente aos meus pais, Maria Lúcia Mendes dos Santos e Marcelo Miranda, que, mesmo com muitos sacrifícios, se empenharam em me proporcionar a melhor educação possível desde cedo. Sempre acreditaram em mim e me deram orientação, apoio e energia para que eu trilhasse sempre o melhor caminho que eu acreditava. Assim como agradecer às minhas irmãs, Isabella e Luisa por todo o apoio e carinho que tem me dado até aqui.

Agradeço também à Mikaela Ana Sherman de Moraes, quem me ajudou a manter o foco durante esse trabalho, ao proporcionar todo o amor, carinho e paciência que alguém poderia pedir de se quem ama. Que sempre esteve presente apesar da distância e que tem feito os dia serem melhores.

Agradeço a todos os meus amigos, mas em especial ao Gabriel Fischer e João Victor de Azeredo por toda a troca de conhecimentos durante os anos da faculdade. Também ao Caio Serra, Gabriela Serra, Rafael Canário, Marco Antônio de Azeredo, Caio Cordeiro, Guilherme Couto, Luiz Paulo de Azeredo, Thales Cavaliere, Leonardo Garrido, Brenno Bontempo, Felipe Bastos e Brunno Barros com quem dividi tantos momentos de alegria junto.

E finalmente agradecer ao meu orientador, Gustavo Rabello, quem sempre se empenhou em dar a melhor aula possível e com isso fomentou o meu interesse pelo tema do trabalho. Que se dispôs a me atender e oferecer o melhor suporte possível, ajudando na concepção e desenvolvimento do trabalho. Sempre disposto a transmitir suas ideias a fim de ajudar na construção dos alunos, no âmbito educacional ou social, tornando-o um dos melhores professores com quem tive contato durante o curso.

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 GEOMETRIAS DE CANAIS EM PLACA FRIA PARA ARREFECIMENTO DE ELETRÔNICOS ATRAVÉS DO MÉTODO DE ELEMENTOS FINITOS

Lucas Mendes Miranda

Junho/2023

Orientador: Gustavo Rabello dos Anjos

Curso: Engenharia Mecânica

Apresenta-se, neste projeto, o desenvolvimento de um algorítimo de simulação computacional para a realização de estudos de geometrias de aletas em placas frias que propiciam melhor transferência de calor para refrigeração de componentes eletrônicos. Para alcançar esse objetivo foi utilizada a formulação corrente-vorticidade para a solução bidimensional do problema de mecânica dos fluidos e a equação geral do calor para o acoplamento térmico. Utilizando a linguagem de programação *Python*, é feita a implementação numérica a partir do esquema Galerkin para o método de elementos finitos. São realizadas as simulações do escoamento de Poiseuille entre placas paralelas e estudadas geometrias de aletas onduladas, que obtiveram resultados satisfatórios.

Palavras-chave: Método de elementos finitos, Transferência de calor, Mecânica dos fluidos

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

STUDY OF COLD PLATE FIN GEOMETRIES TO IMPROVE HEAT TRANSFER CAPABILITIES OF ELETRONICS COOLING SOLUTIONS UTILIZING THE FINITE ELEMENT METHOD

Lucas Mendes Miranda

June/2023

Advisor: Gustavo Rabello dos Anjos

Course: Mechanical Engineering

In this work, we present the development of a computer simulation algorithm in order to study cold plate fin geometries that improve heat transfer capabilities of eletronics cooling solutions. The stream function-vorticity formulation was used to solve the two-dimensional problem of fluid dynamics and the heat equation for the heat transfer. Using *Python*, the finite element method was implemented utilizing the Galerkin scheme. With the help of the algorithm, the Poiseuille flow between two parallel plates was simulated, alongside the waved fin geometries chosen, that had good results.

Keywords: Finite Element Method, Heat transfer, Fluid Mechanics.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

O campo computacional tem se desenvolvido exponencialmente nos últimos anos, com componentes cada vez mais potentes e miniaturizados. Empresas líderes de mercado, como Intel, AMD e Apple, tem dedicado seus esforços para produzir componentes cada vez mais eficientes e poderosos.

Para alcançar resultados cada vez melhores tem-se buscado por dois principais caminhos, um sendo o da eficiência, que busca diminuir cada vez mais as distâncias entre cada transistor, que hoje chegam a 5nm. Esse processo leva uma menor dissipação de energia, através do efeito joule, entre as conexões e a *clocks* (velocidade de processamento) mais altos.

O outro caminho seria o aumento da potência inserida no *CPU*, esse processo leva a *clocks* mais altos, mas também a mais energia dissipada e consequentemente um aumento de temperatura. Entusiastas de tecnologia costumam adotar esse procedimento sempre que alguma das companhias líderes lançam seus novos modelos. Realizando o chamado "*overclock*", esses entusiastas programam o processador para consumir mais energia e alcançar o maior valor de *clock* possível, e para isso precisam empregar soluções térmicas elaboradas que vão até a resfriamento criogênico dos componentes.

1.1 Motivação

Especialistas apontam que podemos estar próximos de um limite sobre a miniaturização dos transistores. Conforme *Mukesh Khare*, vice-presidente do grupo de pesquisas de semicondutores da IBM [1] "Agora é sobre introduzir novos materiais, novas estruturas, novas inovações", "É sobre inovação, não escala". Assim, um dos caminhos a que podem levar a um maior poder de processamento é o aumento da potência inserida à *CPU*, como feito pelos entusiastas em *overclock*.

O que leva a motivação desse trabalho, já que com mais energia inserida no nosso sistema, devemos observar uma maior geração de calor neles e para evitar

temperaturas prejudicais aos componentes eletrônicos, devemos aplicar soluções de dissipação de calor cada vez melhores. Um estudo conduzido por *Florentino* [2], relata como a quantidade e as dimensões das aletas empregadas em uma placa fria influenciam a dissipação de calor gerada por um processador. Assim, esse projeto surge como um desdobramento desse estudo, visando pesquisar por geometrias em aletas internas de uma placa fria de *water cooling*, fugindo aos padrões uniformes comumente aplicados na indústria.

1.2 Metodologia

Para alcançar esse objetivo será adotada uma saída comum na área de pesquisa e desenvolvimento, a simulação computacional, que nos permitirá observar padrões que proporcionem uma dissipação de calor mais eficiente, resultando em temperaturas menores observadas no sistema.

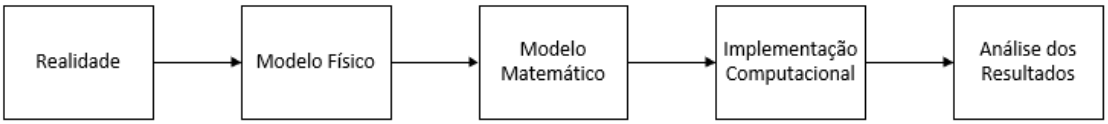


Figura 1.1: Fluxograma básico para uma simulação numérica

Assim, a metodologia proposta irá utilizar da formulação função-corrente - vorticidade para resolver a parte de dinâmica dos fluidos e com os resultados será resolvida a equação de temperatura para cada passe de tempo. Para a implementação computacional, as equações de governos serão desenvolvidas para o método de elementos finitos com a formulação Galerkin utilizando a linguagem *Python* de programação. Para o desenho e discretização das geometrias estudadas será empregado o software livre *GMSH* [3], que será responsável por gerar as malhas utilizadas pelo algorítimo, e para a visualização dos resultados será utilizado o, também software livre, *Paraview* [4].

1.3 Organização do trabalho

- **Capitulo 2** - Apresenta os conceitos básicos e gerais que serão utilizados para o desenvolvimento teórico e da metodologia aplicada.
- **Capitulo 3** - Desenvolve as equações de governo utilizadas, sendo realizada a adimensionalidade delas.
- **Capitulo 4** - Demonstra a formulação das equações governantes no método de elementos finitos.

- **Capitulo 5** - Apresenta a implementação do algorítimo.
- **Capitulo 6** - Apresenta o resultado das validações realizadas do algorítimo desenvolvido.
- **Capitulo 7** - Apresenta os resultados obtidos durante o estudo.
- **Capitulo 8** - Expõe as conclusões do trabalho, discute possibilidades de melhoria e sugere trabalhos futuros.

Capítulo 2

Revisão Bibliográfica

2.1 Arrefecimento de Componentes Eletrônicos

Segundo a *Intel* [5], *TDP* (*Thermal Design Power*) é o valor, em Watts, do consumo de um *CPU* quando está sob a máxima carga teórica. Assim, para um processador com 105 Watts de *TDP*, será necessária uma solução térmica capaz de dissipar esses mesmos 105 Watts, para que a temperatura dos componentes não cheguem a valores prejudiciais, portanto diminuindo a vida útil do processador.

Para isso todo *CPU* tem uma solução básica integrada à própria estrutura para ajudar na dissipação de calor gerado pelos núcleos de processamento.

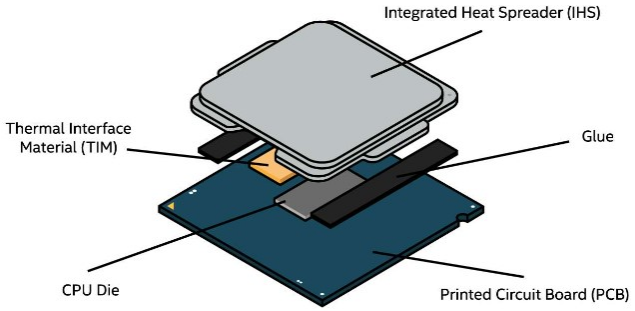


Figura 2.1: Componentes de uma *CPU*
Fonte: *Intel* [6]

Como mostrado na figura 2.1, o calor gerado pelo processador é transmitido a um dissipador de calor integrado, em inglês *Integrated Heat Spreader* ou *IHS*. Assim, a energia dissipada pelo processamento é dissipada por uma área maior, o que permite o acoplamento de uma solução de resfriamento externa para manter os componentes em uma temperatura adequada.

Uma das soluções que tem ganhado espaço no mercado é o resfriamento a água, ou *water cooling* em inglês, objeto de estudo deste trabalho. Ela é composta por

três componentes principais, sendo uma placa fria, componente que está em contato com o processador e responsável pelo resfriamento dele. Uma bomba, utilizada para manter o fluxo constante no sistema. E um radiador, responsável pela refrigeração do fluido utilizado.

Figura 2.2: Exemplo de um *water cooler* básico

Assim, o *water cooler* utiliza de um líquido que chega resfriado ao trocador de calor com o processador, maximizando a troca de calor, e o leva até o radiador, onde ele será resfriado novamente em uma troca térmica com o ar. Logo, é possível perceber que o líquido refrigerante utilizado é de extrema importância. Conforme a fabricante de soluções de resfriamento a água *Boyd* [7], a maioria dos sistemas utiliza água ou uma mistura de água e glicol, que possuem boas propriedades térmicas dadas pela água e de manutenção dadas pelo glicol (quando misturado).

Vale ainda citar que outra alternativa para resfriamento ativo é o resfriamento por ar, que utiliza uma solução de condução térmica, geralmente por canais de cobre, para transmitir o calor gerado pelo processador até um conjunto de aletas e uma ventoinha para resfriar essas aletas por convecção forçada.

2.2 Transferência de Calor

Conforme *Çengel* [8] a termodinâmica pode ser definida como a ciência da energia. Uma das leis mais fundamentais da natureza é a conservação de energia, a qual dita que a energia pode mudar de uma forma para outra, mas sua quantidade total não muda. Sendo a termodinâmica o campo que estuda a transferência de energia de um sistema com sua vizinhança, via interações chamadas de calor e trabalho.

O estudo da transferência de calor busca entender os mecanismos físicos que descrevem a troca de energia através do calor. Calor, que segundo *Çengel*, é definido como a forma de energia transferida através da diferença de temperatura entre dois sistemas (ou um sistema e sua vizinhança). Assim, calor é energia em transição e só

é reconhecida ao cruzar a fronteira de um sistema. E sua transferência é dada por três modos, sendo eles: condução, convecção e radiação.

2.2.1 Condução

O processo de condução está ligado às atividades atômicas e moleculares e descreve a transferência de energia de partículas mais energéticas para menos energéticas, no qual as partículas com mais energias se chocam com as que possuem menos, transferindo a energia do meio com mais para o com menos. Em uma escala macroscópica, esse fenômeno se traduz como a transferência de calor de corpos com maior temperatura para corpos com menor temperatura. A equação que descreve a taxa na qual a transferência de calor por condução ocorre é conhecida como lei de Fourier:

$$q_{cond} = -k\nabla T \tag{2.1}$$

Que podemos ler como: o fluxo de calor por área, $q_{cond}[W/m^2]$, é igual ao gradiente de temperatura, $\nabla T[K/m]$, multiplicado pela condutividade térmica do material, $k[W/mK]$, e o sinal de negativo indica que essa taxa ocorre do corpo de maior temperatura para o de menor temperatura.

2.2.2 Convecção

A condução descreve a troca energética entre partículas estacionárias, que transferem energia através de suas vibrações inerentes ao nível de energia interna que possuem. Já a convecção descreve a troca energética entre partículas que estão estacionárias e partículas em movimento. A convecção abrange dois mecanismos, a transferência de energia devido ao movimento molecular aleatório, difusão, e através do movimento global ou macroscópico do fluido, advecção. Nesse campo, a interação que mais interessa é a troca de energia entre um sólido e um fluido, e essa interação da origem a uma região do fluido em que sua velocidade varia entre zero, no contato com a superfície, até a velocidade associada ao escoamento do fluido, U_∞, essa região é conhecida como camada limite hidrodinâmica. Associada a esse desenvolvimento hidrodinâmico, se houver uma diferença de temperatura entre a superfície e o fluido de escoamento, haverá também uma região na qual a temperatura variará da temperatura associada a superfície, T_s até a associada ao fluido, T_∞, conhecida como camada limite térmica.

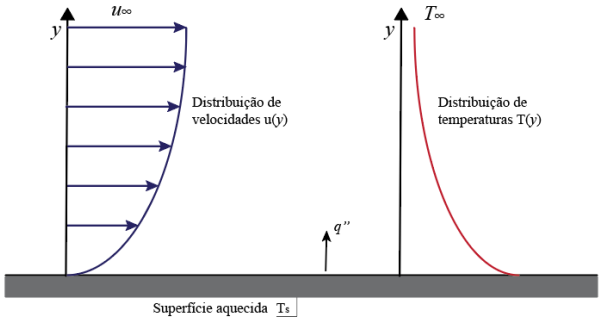


Figura 2.3: Ilustração da camada limite térmica e hidrodinâmica

Assim, a transferência de calor na parte da parede, onde a velocidade é zero, ocorre somente por difusão e conforme se afasta da parede a transferência de calor migra para a advecção, evidenciando a existência desses dois mecanismos.A convecção é dividida em duas classes, sendo elas:

- **Convecção natural:** A convecção natural ocorre pelas forças de empuxo que se originam da diferença de densidade causadas pela variação de temperatura do fluido. Um fluido em contato com uma superfície mais quente, tem sua temperatura elevada e sua densidade diminuída, logo o empuxo leva essas moléculas a subirem e darem lugar a moléculas mais frias e densas.

- **Convecção forçada:** A convecção forçada ocorre quando o escoamento é causado por meios externos, como um ventilador ou uma bomba, que forçam o fluido a escoar sobre uma superfície.

Para as duas classes de convecção, a taxa de transferência de calor é dada por:

$$q_{conv} = h(T_s - T_\infty) \tag{2.2}$$

Que pode-se ler como: o fluxo de calor por área, $_{conv}[W/m^2]$, é proporcional a diferença de temperatura entre o fluido e a superfície multiplicada pelo coeficiente de transferência de calor por convecção, $h[W/m^2K]$. Vale notar que os coeficientes h são maiores para convecção forçada que para convecção natural, justificando o uso de ventiladores e bombas para melhorar a taxa de transferência de calor de fluidos.

2.2.3 Radiação

Conforme descrito em *Incropera* [9], a radiação térmica é energia emitida pela matéria que se encontra em uma temperatura não nula. Independente da forma da matéria, a emissão de pode ser atribuída a mudanças nas configurações eletrônicas de átomos ou moléculas que constituem a matéria. E diferentemente da condução ou convecção, a radiação não necessita de um meio material para ocorrer a transferência

de calor, ocorrendo de forma mais eficiente no vácuo. A emissividade é dada pela lei de Stefan-Boltzmann:

$$E = \epsilon\sigma T^4 \tag{2.3}$$

Que lê-se como a emissividade, $E[W/m^2]$, sendo equivalente à emissividade do material, $\epsilon(0 \leq \epsilon \leq 1)$, multiplicada pela constante de Stefan-Boltzmann, $\sigma = 5.67 \times 10^{-8} W/m^2K$, e pela temperatura do corpo a quarta potência.

E a taxa líquida de transferência de calor por radiação é dada por:

$$q''_{rad} = \frac{q_{rad}}{A} = \epsilon\sigma(T_s^4 - T_{viz}^4) \tag{2.4}$$

Onde T_{viz} é a temperatura da vizinhança na qual a superfície está inserida. Em algumas aplicações é conveniente expressar a troca de calor por radiação como:

$$q_{rad} = h_r A(T_s - T_{viz}) \tag{2.5}$$

em que $h_r = \epsilon\sigma(T_s + T_{viz})(T_s^2 + T_{viz}^2)$ é o coeficiente de transferência de calor por radiação.

2.3 Mecânica dos Fluidos

Como seu próprio nome sugere, a mecânica dos fluidos é o campo da ciência que estuda fluidos em repouso ou em movimento. Com ampla empregabilidade no campo da engenharia, ela tem sido aplicada no desenvolvimento de bombas, compressores, aerodinâmica de automóveis e condicionamento de ar e edifícios. Os fluidos podem se apresentar como líquidos ou gases. Segundo *Fox* [10] um fluido é uma substância que se deforma continuamente sob a aplicação de cisalhamento (tangencial), não importando quão pequeno seja seu valor. Ao analisar qualquer problema de mecânica dos fluidos, devemos considerar as leis básicas de governo de um fluido e o conceito de volume de controle:

2.3.1 Volume de controle

Um volume de controle é um volume arbitrário no espaço e sua fronteira geométrica é chamada de superfície de controle. E é utilizado como referência geométrica para o desenvolvimento de problemas térmicos e de dinâmica dos fluidos.

Figura 2.4: Exemplo de volume de controle arbitrário

2.3.2 Conservação de massa:

Dado um fluido incompressível ($\rho_{constante}$) em um volume de controle, temos que a quantidade de massa que entra no volume de controle deve ser a mesma quantidade que sai do sistema pela lei da conservação de massa.

$$\dot{m} = \int_{M(sistema)} dm = \int_{\forall(sistema)} d\forall = constante \tag{2.6}$$

Assim,

$$\rho_e \forall_e A_e = \rho_s \forall_s A_s \tag{2.7}$$

2.3.3 A segunda lei de Newton:

A segunda lei de newton é uma das leis da física mais conhecidas e é um dos princípios fundamentais da dinâmica. Segundo ela, a soma das forças atuantes em um sistema é igual à variação no tempo de momento linear em um sistema:

$$\vec{F} = \left.\frac{d\vec{P}}{dt}\right|_{sistema} \tag{2.8}$$

No qual a quantidade de movimento linear é dada por:

$$\vec{P}_{sistema} = \int_{M(sistema)} \vec{V} dm = \int_{\forall(sistema)} \vec{V} \rho d\forall \tag{2.9}$$

2.3.4 Princípio da quantidade de movimento angular:

O princípio da quantidade de movimento angular para um sistema diz que a soma de todos os torques atuantes em um sistema é igual à variação de momento angular no tempo:

$$\vec{T} = \frac{d\vec{H}}{dt}|_{sistema} \tag{2.10}$$

Sendo o momento angular dado por:

$$\vec{H}|_{sistema} = \int_{M(sistema)} \vec{r} \times \vec{V} dm = \int_{\forall(sistema)} \vec{r} \times \vec{V} \rho d\forall \tag{2.11}$$

2.3.5 Primeira lei da termodinâmica:

A primeira lei da termodinâmica foi apresentada anteriormente em 2.2, e diz que o total de energia em um sistema não se altera, mas se transforma. E se traduz como:

$$\delta Q - \delta W = dE \tag{2.12}$$

Onde a energia total de um sistema pode ser dada por:

$$\vec{E}|_{sistema} = \int_{M(sistema)} e dm = \int_{\forall(sistema)} e \rho d\forall \tag{2.13}$$

e

$$e = u + \frac{V^2}{2} + gz \tag{2.14}$$

Em que ΔU é a variação de energia interna, Q é a quantidade de calor e τ é o trabalho realizado pelo (ou no) sistema.

2.3.6 Segunda lei da termodinâmica:

A segunda lei da termodinâmica nos diz que o calor sempre flui do corpo mais quente, e com maior temperatura, para o mais frio, e menor temperatura. Essa transferência de calor ocorre de forma espontânea até os corpos atingirem equilíbrio térmico ao chegarem na mesma temperatura. Matematicamente a segunda lei se traduz em:

$$\frac{dS}{dt} \geq \frac{\dot{Q}}{T} \tag{2.15}$$

Ao aplicarmos ela a um volume de controle, obtém-se:

$$\frac{\partial}{\partial t} \int_{vc} s \rho d\forall + \int_{sc} s \rho \vec{V} \cdot d\vec{A} \geq \int_{sc} \frac{1}{T} \left(\frac{\dot{Q}}{A} \right) dA \tag{2.16}$$

Sendo vc e sc volume de controle e superfície de controle, T a temperatura local e $\dot{Q}/A$ é o fluxo de calor local.

Capítulo 3

Fundamentação teórica

Neste capitulo serão discutidos os tópicos pertinentes para o desenvolvimento das equações governantes de calor e mecânica dos fluidos a serem utilizadas no MEF. Elas serão responsáveis por traduzir de forma numérica o que pode-se esperar que aconteça na realidade.

3.1 Equação de Navier Stokes

Desenvolvidas por *Claude-Louis Navier* e *George Gabriel Stokes*, as equações de Navier-Stokes utilizam de equações diferenciais parciais derivadas da segunda lei de newton para determinar os campos de pressão e velocidade de fluido em uma região de interesse. A partir das equações 2.8 e 2.9, podemos escrever que para um sistema infinitesimal a segunda lei de newton se equivale à:

$$d\vec{F} = dm\frac{D\vec{V}}{Dt} = \rho dV\left[u\frac{\partial\vec{V}}{\partial x} + v\frac{\partial\vec{V}}{\partial y} + w\frac{\partial\vec{V}}{\partial z} + \frac{\partial\vec{V}}{\partial t}\right] \tag{3.1}$$

Visto que o operador D/Dt é chamado de derivada material, que relaciona a derivação relacionada com o movimento do fluido, ou derivada Lagrangeana, com a derivada de um ponto fixo no espaço, derivada de Euler. Ela é dada por:

$$\frac{D(-)}{Dt} = \frac{\partial(-)}{\partial t} + (\vec{V}\cdot\nabla)(-) \tag{3.2}$$

Pelo outro lado, deve-se considerar o somatório de forças dada pela tensão em cada direção:

$$dF_x = \left(\rho\mathbf{g}_x + \frac{\partial\sigma_{xx}}{\partial x} + \frac{\partial\tau_{yx}}{\partial y} + \frac{\partial\tau_{zx}}{\partial z}\right)dxdydz \tag{3.3}$$

$$dF_y = \left(\rho\mathbf{g}_y + \frac{\partial\tau_{xy}}{\partial x} + \frac{\partial\sigma_{yy}}{\partial y} + \frac{\partial\tau_{zy}}{\partial z}\right)dxdydz \tag{3.4}$$

$$dF_z = \left(\rho \mathbf{g}_z + \frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z}\right) dxdydz \tag{3.5}$$

Substituindo as equações acima em 3.1, obtemos as equações diferenciais de movimento:

$$\rho \mathbf{g}_x + \frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} + \frac{\partial \tau_{zx}}{\partial z} = \rho\left(u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + w\frac{\partial u}{\partial z} + \frac{\partial u}{\partial t}\right) \tag{3.6}$$

$$\rho \mathbf{g}_y + \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \tau_{zy}}{\partial z} = \rho\left(u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + w\frac{\partial v}{\partial z} + \frac{\partial v}{\partial t}\right) \tag{3.7}$$

$$\rho \mathbf{g}_z + \frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z} = \rho\left(u\frac{\partial w}{\partial x} + v\frac{\partial w}{\partial y} + w\frac{\partial w}{\partial z} + \frac{\partial w}{\partial t}\right) \tag{3.8}$$

Supondo um fluido newtoniano, com viscosidade constante e escoamento incompressível, é possível obter as equações de Navier-Stokes após as devidas substituições dos termos de tensão:

$$\rho g_x - \frac{\partial p}{\partial x} + \mu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right) = \rho\left(u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + w\frac{\partial u}{\partial z} + \frac{\partial u}{\partial t}\right) \tag{3.9}$$

$$\rho g_y - \frac{\partial p}{\partial y} + \mu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2}\right) = \rho\left(u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + w\frac{\partial v}{\partial z} + \frac{\partial v}{\partial t}\right) \tag{3.10}$$

$$\rho g_z - \frac{\partial p}{\partial z} + \mu\left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2}\right) = \rho\left(u\frac{\partial w}{\partial x} + v\frac{\partial w}{\partial y} + w\frac{\partial w}{\partial z} + \frac{\partial w}{\partial t}\right) \tag{3.11}$$

Dividindo as equações por ρ podemos simplificar a equação por:

$$\frac{D\vec{V}}{Dt} = \vec{g} - \frac{1}{\rho}\nabla p + \nu\nabla^2\vec{V} \tag{3.12}$$

3.1.1 Função corrente

A função corrente visa descrever as linhas de corrente observadas em um fluido. As linhas de corrente são aquelas que, em dado instante de tempo, se posicionam tangente ao escoamento do fluido em cada ponto. E em duas dimensões, para um fluido incompressível, é descrita por:

$$u \equiv \frac{\partial \Psi}{\partial y} \quad \text{e} \quad v \equiv -\frac{\partial \Psi}{\partial x} \tag{3.13}$$

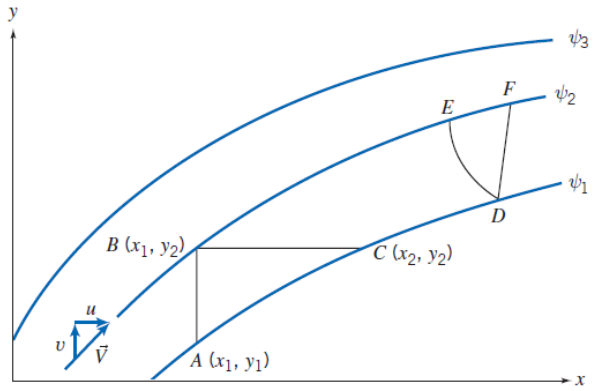


Figura 3.1: Linhas de corrente
Fonte: *Fox and McDonald's Introduction to Fluid Mechanics. 9th*

Assim, ao analisar o escoamento, devemos sempre considerar que entre duas linhas de corrente, sempre passará a mesma quantidade de fluido. Calculando a vazão, não importa o caminho que pegamos, a vazão entre duas linhas de corrente será a mesma:

$$Q = \int_{y_1}^{y_2} u dy = \int_{y_1}^{y_2} \frac{\partial \Psi}{\partial y} dy = \Psi_2 - \Psi_1 \tag{3.14}$$

$$Q = \int_{x_1}^{x_2} v dx = -\int_{x_1}^{x_2} \frac{\partial \Psi}{\partial x} dx = \Psi_2 - \Psi_1 \tag{3.15}$$

3.1.2 Vorticidade

A vorticidade é a grandeza física que quantifica a rotação das partículas de um fluido em movimento, sendo definida por:

$$\vec{\omega} = \nabla \times \vec{V} \tag{3.16}$$

Sendo que para duas dimensões temos:

$$\omega_z = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \tag{3.17}$$

E se relaciona com a função corrente através da substituição da equação 3.13 na equação de vorticidade 3.17, obtendo:

$$\omega_z = \frac{\partial(-\partial \Psi)}{\partial x^2} - \frac{\partial(\partial \Psi)}{\partial y^2} = -\nabla^2 \Psi \tag{3.18}$$

3.1.3 Formulação função-corrente - vorticidade

A formulação corrente vorticidade é um modo de representar as equações de Navier-Stokes em função da função de corrente e do transporte de vorticidade em

um escoamento. Ela tem como grande vantagem o desacoplamento do campo de pressões da equação. Para obter a equação de transporte vorticidade, calculamos:

$$\frac{\partial N.S._x}{\partial y} - \frac{\partial N.S._y}{\partial x} \tag{3.19}$$

Abrindo a equação, temos:

$$\frac{\partial}{\partial y}\left(\frac{Du}{Dt}\right) - \frac{\partial}{\partial x}\left(\frac{Dv}{Dt}\right) = \frac{\partial}{\partial y}\left(\vec{g} - \frac{1}{\rho}\frac{\partial p}{\partial x} + \nu\nabla^2\vec{V}\right) - \frac{\partial}{\partial x}\left(\vec{g} - \frac{1}{\rho}\frac{\partial p}{\partial y} + \nu\nabla^2\vec{V}\right) \tag{3.20}$$

Abrindo as derivadas materiais e considerando a gravidade constante, temos:

$$\begin{aligned}\frac{\partial}{\partial y}\left(\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}\right) - \frac{\partial}{\partial x}\left(\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}\right) \\ = \frac{1}{\rho}\left(\frac{\partial^2 p}{\partial x \partial y} - \frac{\partial^2 p}{\partial y \partial x}\right) + \frac{\partial}{\partial y}\left(\nu\nabla^2\vec{V}\right) - \frac{\partial}{\partial x}\left(\nu\nabla^2\vec{V}\right)\end{aligned} \tag{3.21}$$

Utilizando da propriedade:

$$\frac{\partial}{\partial x_i}\left(\frac{\partial}{\partial x_j}\right) = \frac{\partial}{\partial x_j}\left(\frac{\partial}{\partial x_i}\right) \tag{3.22}$$

Podemos riscar o termo da pressão, e reorganizar a equação na forma:

$$\begin{aligned}\frac{\partial}{\partial t}\left(\frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}\right) + u\frac{\partial}{\partial x}\left(\frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}\right) + v\frac{\partial}{\partial y}\left(\frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}\right) \\ = \nu\left[\frac{\partial^2}{\partial x^2}\left(\frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}\right) + \frac{\partial^2}{\partial y^2}\left(\frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}\right)\right]\end{aligned} \tag{3.23}$$

Substituindo 3.17 na equação acima:

$$\frac{\partial \omega}{\partial t} + u\frac{\partial \omega}{\partial x} + v\frac{\partial \omega}{\partial y} = \nu\left(\frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2}\right) \tag{3.24}$$

Que já é a equação de transporte de vorticidade, mas que pode ser simplificada como:

$$\frac{\partial \omega}{\partial t} + \vec{V}\cdot\nabla\omega = \nu\nabla^2\omega \tag{3.25}$$

3.2 Equação de Temperatura

Para a obtenção da equação da temperatura começamos com a aplicação da conservação de energia da primeira lei da termodinâmica:

$$\left(h + \frac{\overline{V^2}}{2} + gz\right)_e \delta\dot{m}_e - \left(h + \frac{\overline{V^2}}{2} + gz\right)_s \delta\dot{m}_s + \delta\dot{Q} - \delta\dot{W} + \delta E_{gerada} = \frac{dE_{VC}}{dt} \quad (3.26)$$

Em que $\dot{m}_e$ e $\dot{m}_s$ são as taxas de entrada e saída de massa do volume de controle, respectivamente. Tomando $e = h + \overline{V}^2/2 + gz$, sua forma integral é:

$$\dot{Q} - \dot{W} + E_{gerada} = \frac{\partial}{\partial t}\left(\int_{VC} e\rho d\forall\right) + \int_{VC}\left(e + \frac{p}{\rho}\right)\rho(\vec{V}\cdot\vec{n})dA \quad (3.27)$$

Para um volume de controle muito pequeno, a integral de volume se reduz à:

$$\frac{\partial}{\partial t}(e\rho d\forall) = \frac{\partial}{\partial t}(e\rho)dx\ dy\ dz \quad (3.28)$$

O que nos da:

$$\dot{Q} - \dot{W} + E_{gerada} = \left(\rho\frac{\partial e}{\partial t} + \vec{V}\cdot\vec{\nabla}p + p\vec{\nabla}\cdot\vec{V}\right)dx\ dy\ dz \quad (3.29)$$

Para o termo $\dot{Q}$ desconsideraremos a radiação, considerando somente a condução de calor pelos elementos, e para um volume de controle infinitesimal, temos:

$$\dot{Q} = -\left(\frac{\partial}{\partial x}q_x + \frac{\partial}{\partial y}q_y + \frac{\partial}{\partial z}q_z\right)dx\ dy\ dz = -\vec{\nabla}\cdot\vec{q}\ dx\ dy\ dz \quad (3.30)$$

e por fim:

$$\dot{Q} = \vec{\nabla}\cdot(k\vec{\nabla}T)\ dx\ dy\ dz \quad (3.31)$$

Utilizando o termo $\dot{W}$ como:

$$\dot{W} = -\vec{\nabla}\cdot(\vec{V}\cdot\vec{\tau}_{ij})\ dx\ dy\ dz \quad (3.32)$$

Substituindo $\dot{Q}$ e $\dot{W}$ na equação 3.29, obtém-se:

$$\rho\frac{\partial e}{\partial t} + \vec{V}\cdot\vec{\nabla}p + p\vec{\nabla}\cdot\vec{V} = \vec{\nabla}\cdot(k\vec{\nabla}T) + \vec{\nabla}\cdot(\vec{V}\cdot\vec{\tau}_{ij}) + E_{gerada} \quad (3.33)$$

Que, segundo *Incropera* [9], após mais manipulações, a equação de calor para duas dimensões e sem dissipação viscosa fica:

$$\frac{\partial T}{\partial t} + \vec{V} \cdot \nabla T = \alpha \nabla^2 T + \frac{Q}{\rho c_p} \tag{3.34}$$

Em que Q é a geração de energia volumétrica no sistema dado em $[W/m^3]$.

3.3 Adimensionalização

A adimensionalidade é uma ferramenta importante na análise de fluidos, ao levar a análise do particular para o global. Os números adimensionais dizem como diferentes características de um sistema se relacionam, como é o exemplo do número de Reynolds que avalia a estabilidade do fluxo, ou o número de Prandtl que relaciona a viscosidade dinâmica com a difusividade térmica.

Assim, para simplificar a análise do escoamento estudado no trabalho, olhando suas características globais ao invés das particulares, é interessante realizar a adimensionalidade das equações governantes vistas anteriormente 3.13, 3.25, 3.18 e 3.34, a ideia é utilizar valores de referência que permitam que a variável adimensional cumpra $0 \leq [-] \leq 1$.

Começando pela função corrente-vorticidade, defini-se como os equivalentes adimensionais das variáveis da função:

$$\begin{aligned} x^* &= \frac{x}{L_0} & y^* &= \frac{y}{L_0} & t^* &= t\frac{U_0}{L} \\ u^* &= \frac{u}{U_0} & v^* &= \frac{v}{U_0} & \nu^* &= \frac{\nu}{\nu_0} \end{aligned} \tag{3.35}$$

Substituindo as variáveis adimensionais na função vorticidade:

$$\omega_z = \frac{\partial v^* U_0}{\partial x^* L_0} - \frac{\partial u^* U_0}{\partial y^* L_0} = \frac{U_0}{L_0}\left(\frac{\partial v^*}{\partial x^*} - \frac{\partial u^*}{\partial y^*}\right) = \frac{U_0}{L_0}\omega^* \tag{3.36}$$

Com a vorticidade adimensional, podemos substituí-la na função corrente-vorticidade:

$$\begin{aligned} \frac{d(U_0/L_0\omega_z^*)}{d(L_0/U_0 t^*)} + U_0 u^* \frac{d(U_0/L_0\omega_z^*)}{d(L_0 x^*)} + U_0 v^* \frac{d(U_0/L_0\omega_z^*)}{d(L_0 y^*)} \\ = (\nu_0 \nu^*)\left[\frac{d^2(U_0/L_0\omega_z^*)}{d(L_0^2 x^{*^2})} + \frac{d^2(U_0/L_0\omega_z^*)}{d(L_0^2 y^{*^2})}\right] \end{aligned} \tag{3.37}$$

Arrumando as variáveis da equação, obtém-se:

$$\frac{U_0^2}{L_0^2}\frac{d\omega_z^*}{dt^*} + \frac{U_0^2}{L_0^2} v_x^* \frac{d\omega_z^*}{dx^*} + \frac{U_0^2}{L_0^2} v_y^* \frac{d\omega_z^*}{dy^*} = \nu^* \frac{\nu_0 U_0}{L_0^3}\left[\frac{d^2\omega_z}{dx^{*^2}} + \frac{d^2\omega_z}{dy^{*^2}}\right] \tag{3.38}$$

Dividindo a equação toda por U_0^2/L_0^2 e escolhendo $\nu_0 = \nu$ de modo que $\nu^* = 1$:

$$\frac{d\omega_z^*}{dt^*} + v_x^* \frac{d\omega_z^*}{dx^*} + v_y^* \frac{d\omega_z^*}{dy^*} = \frac{\nu_0}{L_0 U_0} \left[\frac{d^2\omega_z}{dx^{*2}} + \frac{d^2\omega_z}{dy^{*2}} \right] \tag{3.39}$$

Nota-se o número adimensional de Reynolds é definido no problema como:

$$R_e = \frac{U_0 L_0}{\nu} \tag{3.40}$$

Realizando a substituição na equação 3.41:

$$\frac{d\omega_z^*}{dt^*} + v_x^* \frac{d\omega_z^*}{dx^*} + v_y^* \frac{d\omega_z^*}{dy^*} = \frac{1}{R_e} \left[\frac{d^2\omega_z}{dx^{*2}} + \frac{d^2\omega_z}{dy^{*2}} \right] \tag{3.41}$$

Reescrevendo a equação de forma simplificar sua leitura:

$$\frac{d\omega_z}{dt} + \vec{V} \cdot \nabla\omega = \frac{1}{R_e} \nabla^2\omega \tag{3.42}$$

E para a função corrente, seu equivalente adimensional é:

$$\Psi^* = \frac{\Psi}{L_0 U_0} \tag{3.43}$$

Substituindo em 3.18:

$$-\frac{U_0}{L_0}\omega^* = \frac{\partial(\partial(U_0 L_0 \Psi^*))}{\partial(L_0 x^*)^2} + \frac{\partial(\partial(U_0 L_0 \Psi^*))}{\partial(L_0 y^*)^2} \tag{3.44}$$

Após simplificar as variáveis, a forma adimensional do acoplamento corrente-vorticidade fica igual a sua forma dimensional:

$$-\omega = \nabla^2\Psi \tag{3.45}$$

E a velocidade é adimensionalizada para:

$$U_0 \vec{V}^* = \left(\frac{\partial(L_0 U_0 \Psi^*)}{\partial L_0 y^*}, \frac{\partial(L_0 U_0 \Psi^*)}{\partial L_0 x^*} \right) \tag{3.46}$$

Que se resume a mesma forma de sua variante dimensional:

$$\vec{V} = \left(\frac{\partial\Psi}{\partial y}, \frac{\partial\Psi}{\partial x} \right) \tag{3.47}$$

Para a equação da temperatura, devemos adotar as variáveis adimensionais abaixo:

$$\rho^* = \frac{\rho}{\rho_0} \qquad c_p^* = \frac{c_p}{c_{p_0}} \qquad \alpha^* = \frac{\alpha}{\alpha_0} \tag{3.48}$$
$$T^* = \frac{T - T_0}{T_s - T_0} \qquad Q^* = \frac{L_0}{h_0(T_s - T_0)}Q \qquad \nabla^* = L_0\nabla$$

Aplicando elas em 3.34, obtém-se:

$$\frac{\partial((T_s - T_0)T^*)}{\partial(L_0/U_0 t^*)} + U_0\vec{V}^*\frac{\vec{\nabla}^*}{L_0}((T_s - T_0)T^*) = \alpha^*\alpha_0\frac{\nabla^{*^2}}{L_0^2}((T_s - T_0)T^*) + \frac{h_0(T_s - T_0)Q^*}{L_0\rho_0\rho^* c_{p_0}c_p^*} \tag{3.49}$$

Podemos arrumar a equação na forma:

$$\frac{(T_s - T_0)U_0}{L_0}\frac{\partial T^*}{\partial t^*} + \frac{(T_s - T_0)U_0}{L_0}\vec{V}^*\vec{\nabla}^*T^* = \frac{(T_s - T_0)\alpha^*\alpha_0}{L_0^2}\nabla^{*^2}T^* + \frac{h_0(T_s - T_0)}{L_0\rho_0 c_{p_0}}\frac{Q^*}{\rho^* c_p^*} \tag{3.50}$$

Adotando:

$$\alpha_0 = \alpha,\ \rho_0 = \rho,\ c_{p_0} = c_p \tag{3.51}$$

Considerando $\alpha^* = \rho^* = c_p^* = 1$, e dividindo a equação toda por $(T_0U_0)/L_0$:

$$\frac{\partial T^*}{\partial t^*} + \vec{V}^*\vec{\nabla}^*T^* = \frac{\alpha_0}{U_0L_0}\nabla^{*^2}T^* + \frac{h_0}{U_0\rho_0 c_{p_0}}Q^* \tag{3.52}$$

Nesse momento a nossa equação está adimensional, onde:

$$\frac{\alpha_0}{U_0L_0} = \frac{\mu}{U_0L_0}\frac{\alpha_0}{\mu} = \frac{1}{R_e}\frac{1}{Pr} = \frac{1}{P_e} \tag{3.53}$$

e

$$\frac{h_0}{U_0\rho_0 c_{p_0}} = \frac{N_u}{R_eP_r} = \frac{N_u}{P_e} = St \tag{3.54}$$

Em que P_e é o número adimensional de Péclet, que relaciona o número adimensional de Reynolds com o número adimensional de Prandlt que relaciona a difusividade de momento com a difusividade térmica. E S_t é o número de Stanton, que indica a quantidade de calor absorvida pelo fluido quando existe transferência de calor entre sólido e fluido, e quanto maior seu valor, mais eficiente o calor é transferido ao fluido.

$$\frac{\partial T}{\partial t} + \vec{V}\vec{\nabla}T = \frac{1}{P_e}\nabla^2 T + S_t Q \tag{3.55}$$

As equações de governo 3.42, 3.45, 3.47 e 3.55 estão na forma adimensional em que serão utilizadas no trabalho.

Capítulo 4

Método de Elementos Finitos

Muitos fenômenos físicos da ciência ou engenharia podem ser descritos em equações diferenciais parciais. Tais equações, muitas vezes, são quase impossíveis de serem resolvidas analiticamente. Assim, uma saída é resolver, de forma aproximada, por métodos computacionais. O método de elementos finitos, ou MEF, aparece com uma solução computacional para descrever aproximadamente um fenômeno de interesse.

Conforme *Fish* [11], a ideia básica do MEF é dividir um corpo em um número finito de elementos, conectados por nós, para obter a solução aproximada. A figura abaixo mostra como ocorre essa divisão de elementos e a rede que se forma com ela.

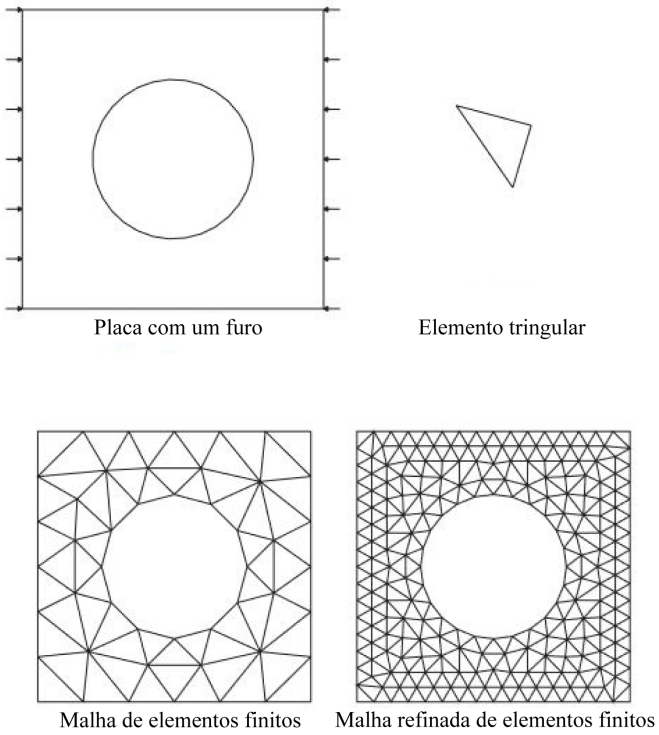


Figura 4.1: Geometrias e malhas de elemento finito
Adaptado de: *A First Course in Finite Elements*

O método de elementos finitos provê uma metodologia sistemática que toma equações analíticas, muitas vezes equações diferenciais parciais, a discretiza para um sistema não contínuo, chamada de formulação fraca, e a aplica os elementos formados por geometrias simples que compõe a geometria mais complexa do problema geral. Com isso o MEF obtém um sistema linear de equações, na forma $Ax = b$, utilizado para a solução computacional completa do problema.

4.1 Formulação forte

A forma forte consiste nas equações governantes e nas condições de contorno do sistema. No caso estudado nesse trabalho, serão utilizadas as equações de governo 3.42, 3.45, 3.47 e 3.55, já desenvolvidas no capítulo anterior.

Para resolver o problema de mecânica dos fluidos precisaremos das três equações abaixo:

$$\frac{\partial \omega_z}{\partial t} + \mathbf{v} \cdot \nabla \omega_z = \frac{1}{R_e} \nabla^2 \omega_z \tag{4.1}$$

$$\nabla^2 \Psi = -\omega \tag{4.2}$$

$$\mathbf{v} = (v_x, v_y) = \left(\frac{\partial \Psi}{\partial y}, -\frac{\partial \Psi}{\partial x} \right) \tag{4.3}$$

Válidas no domínio $\Omega \subset \mathbb{R}$, com as condições de contorno de Dirichlet:

$$\omega = \overline{\omega} \tag{4.4}$$

$$\Psi = \overline{\Psi} \tag{4.5}$$

$$\mathbf{v} = \overline{\mathbf{v}} \tag{4.6}$$

e para o problema térmico,

$$\frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T = \frac{1}{P_e} \nabla^2 T \tag{4.7}$$

Válidas no domínio $\Omega \subset \mathbb{R}$, com condições de contorno de Dirichlet e Neumann, dependendo do contorno analisado. Assim é conveniente dividir o contorno Γ em Γ_E e Γ_N, onde $\Gamma_N \cup \Gamma_E = \Gamma$. Assim, as condições de contorno de temperatura ficam:

$$T = \overline{T} \quad em \quad \Gamma_E \tag{4.8}$$

$$\frac{\partial T}{\partial n} = 0 \quad em \quad \Gamma_N \tag{4.9}$$

4.2 Formulação fraca

Para a discretização das funções diferenciais acima, devemos desenvolver a formulação fraca do problema. De acordo com Anjos [12], para encontrar a forma fraca devemos encontrar uma função tentativa, u, que satisfaça:

$$\int_{\Omega}\left(\frac{du}{dx}\right)^{2}dx < \infty \tag{4.10}$$

A função que satisfizer essa expressão é chamada de função H^1 e representa o espaço Sobolev.

$$H^{1}(\Omega) = \left\{u \in L^{2}(\Omega), \frac{\partial u}{\partial x_i} \in L^{2}(\Omega), i = 1, 2, ..., n\right\} \tag{4.11}$$

Em que $L^2(\Omega)$ é o espaço:

$$L^{2}(\Omega) = \left\{u : \Omega \longrightarrow \mathbb{R}, \int_{\Omega} u^{2}d\Omega \leq \infty\right\} \tag{4.12}$$

Junto a função tentativa, é utilizada uma função peso, w, também do tipo H^1, cuja característica é assumir o valor zero, $w = 0$, quando a condição de contorno é de Dirichlet.

Assim, como expresso por *Luis Carnevale* [13], podemos definir os espaços das funções 4.1, 4.2, 4.3, 4.7 como:

$$\mathbb{W} = \{\omega | \omega \in H^{1}(\Omega), \omega = \overline{\omega} \text{ em } \Gamma\} \tag{4.13}$$

$$\mathbb{T} = \{T | T \in H^{1}(\Omega), T = \overline{T} \text{ em } \Gamma_E\} \tag{4.14}$$

$$\mathbb{V} = \{\mathbf{v} | \mathbf{v} \in H^{1}(\Omega), \mathbf{v} = \overline{\mathbf{v}} \text{ em } \Gamma\} \tag{4.15}$$

$$\mathbb{P} = \{\Psi | \Psi \in H^{1}(\Omega), \Psi = \overline{\Psi} \text{ em } \Gamma\} \tag{4.16}$$

E os espaços das funções pesos, respectivamente atribuídas como:

$$\mathbb{S} = \{\varsigma | \varsigma \in H^{1}(\Omega), \varsigma = 0 \text{ em } \Gamma\} \tag{4.17}$$

$$\mathbb{D} = \{\delta | \delta \in H^{1}(\Omega), \delta = 0 \text{ em } \Gamma_E\} \tag{4.18}$$

$$\mathbb{H} = \{\eta | \eta \in H^{1}(\Omega), \eta = 0 \text{ em } \Gamma\} \tag{4.19}$$

$$\mathbb{F} = \{\phi | \phi \in H^{1}(\Omega), \phi = 0 \text{ em } \Gamma\} \tag{4.20}$$

Multiplicando as equações governantes pelas respectivas funções pesos atribuídas a cada uma, tem-se:

$$\int_\Omega \varsigma \left[\frac{\partial \omega_z}{\partial t} + \mathbf{v} \cdot \nabla \omega_z - \frac{1}{R_e} \nabla^2 \omega_z \right] d\Omega = 0 \tag{4.21}$$

$$\int_\Omega \delta \left[\frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T - \frac{1}{P_e} \nabla^2 T \right] d\Omega = 0 \tag{4.22}$$

$$\int_\Omega \eta \left[\mathbf{v} - \left(\frac{\partial \Psi}{\partial y}, -\frac{\partial \Psi}{\partial x} \right) \right] d\Omega = 0 \tag{4.23}$$

$$\int_\Omega \phi \left[\nabla^2 \Psi + \omega \right] d\Omega = 0 \tag{4.24}$$

As quais podem ser expressas como:

$$\int_\Omega \varsigma \frac{\partial \omega_z}{\partial t} d\Omega + \int_\Omega \varsigma \mathbf{v} \cdot \nabla \omega_z d\Omega - \int_\Omega \varsigma \frac{1}{R_e} \nabla^2 \omega_z d\Omega = 0 \tag{4.25}$$

$$\int_\Omega \delta \frac{\partial T}{\partial t} d\Omega + \int_\Omega \delta \mathbf{v} \cdot \nabla T d\Omega - \int_\Omega \delta \frac{1}{P_e} \nabla^2 T d\Omega = 0 \tag{4.26}$$

$$\int_\Omega \eta \mathbf{v} d\Omega - \int_\Omega \eta \left(\frac{\partial \Psi}{\partial y}, -\frac{\partial \Psi}{\partial x} \right) d\Omega = 0 \tag{4.27}$$

$$\int_\Omega \phi \nabla^2 \Psi d\Omega + \int_\Omega \phi \omega d\Omega = 0 \tag{4.28}$$

Para os termos de maior ordem, ∇^2, se faz conveniente reduzir suas ordens através do teorema de Green:

$$\int_\Omega \varsigma \frac{\partial \omega_z}{\partial t} d\Omega + \int_\Omega \varsigma \mathbf{v} \cdot \nabla \omega_z d\Omega + \int_\Omega \nabla \varsigma \frac{1}{R_e} \nabla \omega_z d\Omega - \int_\Gamma \varsigma \frac{1}{R_e} (\nabla \omega_z \cdot \mathbf{n}) d\Gamma = 0 \tag{4.29}$$

$$\int_\Omega \delta \frac{\partial T}{\partial t} d\Omega + \int_\Omega \delta \mathbf{v} \cdot \nabla T d\Omega + \int_\Omega \nabla \delta \frac{1}{P_e} \nabla T d\Omega - \int_\Gamma \delta \frac{1}{P_e} (\nabla T \cdot \mathbf{n}) d\Gamma = 0 \tag{4.30}$$

$$- \int_\Omega \nabla \phi \nabla \Psi d\Omega + \int_\Gamma \phi (\nabla \Psi \cdot \mathbf{n}) d\Gamma + \int_\Omega \phi \omega d\Omega = 0 \tag{4.31}$$

As funções peso possuem a característica de serem iguais a zero quando a função tentativa tem seu valor constante e definido no ponto, ou seja, nas condições de contorno de Dirichlet. Com isso, podemos eliminar a parte das funções que estão avaliadas em um contorno com condição Dirichlet e as formas fracas das equações

governantes ficam:

$$\int_\Omega \varsigma \frac{\partial \omega_z}{\partial t} d\Omega + \int_\Omega \varsigma \mathbf{v} \cdot \nabla \omega_z d\Omega + \int_\Omega \nabla \varsigma \frac{1}{R_e} \nabla \omega_z d\Omega = 0 \tag{4.32}$$

$$\int_\Omega \delta \frac{\partial T}{\partial t} d\Omega + \int_\Omega \delta \mathbf{v} \cdot \nabla T d\Omega + \int_\Omega \nabla \delta \frac{1}{P_e} \nabla T d\Omega = 0 \tag{4.33}$$

$$\int_\Omega \eta \mathbf{v} d\Omega - \int_\Omega \eta \left(\frac{\partial \Psi}{\partial y}, -\frac{\partial \Psi}{\partial x} \right) d\Omega = 0 \tag{4.34}$$

$$- \int_\Omega \nabla \phi \nabla \Psi d\Omega + \int_\Omega \phi \omega d\Omega = 0 \tag{4.35}$$

4.3 Método Galerkin

O próximo passo para a utilização do MEF é a discretização das equações via aproximações. Para isso será utilizado o método Galerkin, que desenvolve aproximações finitas para as funções tentativa e peso, u^e e w^e respectivamente, de cada uma das funções governantes na forma:

$$u(x) \approx \sum_{n=1}^{n_p} u_i N_i(x) \tag{4.36}$$

$$w(x) \approx \sum_{n=1}^{n_p} w_j N_j(x) \tag{4.37}$$

Em que u_i e w_i são constantes a serem determinadas e N_i e N_j são funções de forma, que no caso do método Galerkin são iguais, $N_i^e = N_j^e$, e definidas por:

Aplicando o gradiente e aplicando as aproximações nas funções 4.32, 4.33, 4.34 e 4.35 vistas anteriormente, obtém-se:

$$\int_\Omega \sum_j^{n_p} \varsigma_j N_j^e \frac{\partial}{\partial t} \left(\sum_i^{n_p} \omega_i N_i^e \right) d\Omega + \int_\Omega \sum_j^{n_p} \varsigma_j N_j^e \mathbf{v} \cdot \nabla \left(\sum_i^{n_p} \omega_i N_i^e \right) d\Omega + \int_\Omega \nabla \sum_j^{n_p} \varsigma_j N_j^e \frac{1}{R_e} \nabla \left(\sum_i^{n_p} \omega_i N_i^e \right) d\Omega = 0 \tag{4.38}$$

$$\int_\Omega \sum_j^{n_p} \delta_j N_j^e \frac{\partial}{\partial t}\left(\sum_i^{n_p} T_i N_i^e\right) d\Omega + \int_\Omega \sum_j^{n_p} \delta_j N_j^e \mathbf{v}\cdot\nabla\left(\sum_i^{n_p} T_i N_i^e\right) d\Omega + \int_\Omega \nabla \sum_j^{n_p} \delta_j N_j^e \frac{1}{P_e} \nabla\left(\sum_i^{n_p} T_i N_i^e\right) d\Omega = 0 \quad (4.39)$$

$$-\int_\Omega \nabla\left(\sum_j^{n_p} \phi_j N_j^e\right)\nabla\left(\sum_i^{n_p} \Psi_i N_i^e\right) d\Omega + \int_\Omega \sum_j^{n_p} \phi_j N_j^e \sum_i^{n_p} \omega_i N_i^e d\Omega = 0 \quad (4.40)$$

Para a equação da velocidade, a mesma será desmembrada nas componentes $\mathbf{v} = (u, v)$:

$$\int_\Omega \sum_j^{n_p} \eta_j N_j^e \sum_i^{n_p} u_i N_i^e d\Omega - \int_\Omega \sum_j^{n_p} \eta_j N_j^e \frac{\partial}{\partial y}\left(\sum_i^{n_p} \Psi_i N_i^e\right) d\Omega = 0 \quad (4.41)$$

$$\int_\Omega \sum_j^{n_p} \eta_j N_j^e \sum_i^{n_p} v_i N_i^e d\Omega + \int_\Omega \sum_j^{n_p} \eta_j N_j^e \frac{\partial}{\partial x}\left(\sum_i^{n_p} \Psi_i N_i^e\right) d\Omega = 0 \quad (4.42)$$

Aplicando a propriedade das integrais na qual a soma das integrais é igual à integral das somas, pode-se passar os somatórios para fora das integrais. Como em 4.41 e 4.42, u e v são tratadas como constantes, essas também são passadas para fora das integrais:

$$\sum_j^{n_p}\sum_i^{n_p} \varsigma_j \frac{\partial \omega_i}{\partial t} \int_\Omega N_j^e N_i^e d\Omega + \sum_j^{n_p}\sum_i^{n_p} \varsigma_j \omega_i \mathbf{v} \int_\Omega N_j^e \nabla N_i^e d\Omega + \sum_j^{n_p}\sum_i^{n_p} \varsigma_j \omega_i \frac{1}{R_e} \int_\Omega \nabla N_j^e \nabla N_i^e d\Omega = 0 \quad (4.43)$$

$$\sum_j^{n_p}\sum_i^{n_p} \delta_j \frac{\partial T_i}{\partial t} \int_\Omega N_j^e N_i^e d\Omega + \sum_j^{n_p}\sum_i^{n_p} \delta_j T_i \mathbf{v} \int_\Omega N_j^e \nabla N_i^e d\Omega + \sum_j^{n_p}\sum_i^{n_p} \delta_j T_i \frac{1}{P_e} \int_\Omega \nabla N_j^e \nabla N_i^e d\Omega = 0 \quad (4.44)$$

$$-\sum_j^{n_p}\sum_i^{n_p} \phi_j \Psi_i \int_\Omega \nabla N_j^e \nabla N_i^e d\Omega + \sum_j^{n_p}\sum_i^{n_p} \phi_j \omega_i \int_\Omega N_j^e N_i^e d\Omega = 0 \quad (4.45)$$

$$\sum_{j}^{n_p}\sum_{i}^{n_p}\eta_j u_i \int_\Omega N_j^e N_i^e d\Omega - \sum_{j}^{n_p}\sum_{i}^{n_p}\eta_j \int_\Omega N_j^e \Psi_i \frac{\partial N_i^e}{\partial y} d\Omega = 0 \tag{4.46}$$

$$\sum_{j}^{n_p}\sum_{i}^{n_p}\eta_j v_i \int_\Omega N_j^e N_i^e d\Omega + \sum_{j}^{n_p}\sum_{i}^{n_p}\eta_j \int_\Omega N_j^e \Psi_i \frac{\partial N_i^e}{\partial x} d\Omega = 0 \tag{4.47}$$

Os termos em integral possuem correspondentes matriciais, como indicado por *Anjos* [12]:

- $\int_\Omega N_j^e N_i^e d\Omega \longrightarrow \mathbf{m}^e$
- $\int_\Omega N_j^e \nabla N_i^e d\Omega \longrightarrow \mathbf{g}^e$
- $\int_\Omega \nabla N_j^e \nabla N_i^e d\Omega \longrightarrow \mathbf{k}^e$

Essas são chamadas de matrizes locais, cada uma dessas matrizes devem passar por um processo chamado de *assembling*, em que cada matriz local é utilizada no processo de montagem da matriz global.

$$\mathbf{M} = \mathcal{A}_{e=1}^{n_e}\mathbf{M}^e; \quad \mathbf{G} = \mathcal{A}_{e=1}^{n_e}\mathbf{G}^e; \quad \mathbf{K} = \mathcal{A}_{e=1}^{n_e}\mathbf{K}^e \tag{4.48}$$

Obtém-se a forma fraca para implementação numérica:

$$\mathbf{M}\frac{\partial \omega}{\partial t} + (\mathbf{v}\cdot\mathbf{G} + \frac{1}{R_e}\mathbf{K})\omega = 0 \tag{4.49}$$

$$\mathbf{M}\frac{\partial T}{\partial t} + (\mathbf{v}\cdot\mathbf{G} + \frac{1}{P_e}\mathbf{K})T = S_t\mathbf{M}Q \tag{4.50}$$

$$\mathbf{K}\Psi = \mathbf{M}\omega \tag{4.51}$$

$$\mathbf{M}u_i = \mathbf{G}\Psi_i \tag{4.52}$$

$$\mathbf{M}v_i = -\mathbf{G}\Psi_i \tag{4.53}$$

A discretização temporal de primeira ordem é dada por:

$$\frac{\square^{n+1} - \square^n}{\Delta t} \tag{4.54}$$

Assim, para as equações de temperatura e vorticidade é possível utilizar duas formas de discretização:

- Implícita: No qual a velocidade é calculada no tempo $n+1$:

$$\left(\mathbf{v}\cdot\mathbf{G}+\frac{1}{R_e}\mathbf{K}+\frac{\mathbf{M}}{\Delta t}\right)\omega_z^{n+1}=\frac{\mathbf{M}}{\Delta t}\omega_z^{n} \tag{4.55}$$

$$\left(\mathbf{v}\cdot\mathbf{G}+\frac{1}{P_e}\mathbf{K}+\frac{\mathbf{M}}{\Delta t}\right)T^{n+1}=\frac{\mathbf{M}}{\Delta t}T^{n}+S_t\mathbf{M}Q \tag{4.56}$$

- Explícita: No qual a velocidade é calculada no tempo n e utilizada para o cálculo de $n+1$:

$$\left(\frac{1}{R_e}\mathbf{K}+\frac{\mathbf{M}}{\Delta t}\right)\omega_z^{n+1}=\left(\frac{\mathbf{M}}{\Delta t}-\mathbf{v}\cdot\mathbf{G}\right)\omega_z^{n} \tag{4.57}$$

$$\left(\frac{1}{P_e}\mathbf{K}+\frac{\mathbf{M}}{\Delta t}\right)T^{n+1}=\left(\frac{\mathbf{M}}{\Delta t}-\mathbf{v}\cdot\mathbf{G}\right)T^{n}+S_t\mathbf{M}Q \tag{4.58}$$

A formulação explícita pode trazer instabilidade se não cumprida as "condições de CFL"para estabilidade do método, mas foi o método escolhido por permitir que as matrizes de rigidez e massa de ω^{n+1} e T^{n+1} possam ser calculadas e fixadas antes dos passos de tempo, poupando assim poder computacional e acelerando a simulação.

4.4 Elementos de Malha

Como visto no início do capítulo, O MEF tem como premissa dividir uma região de interesse em um número finito de elementos menores, criando a chamada malha. Assim, a forma em que a divisão da região é feita afeta diretamente como aplicamos o método.

A divisão pode ser feita por elementos de n formas, sendo as mais comuns de elementos quadráticos e triangulares. Neste trabalho serão utilizados elementos triangulares, por apresentarem maior flexibilidade em contornos de formas geométricas.

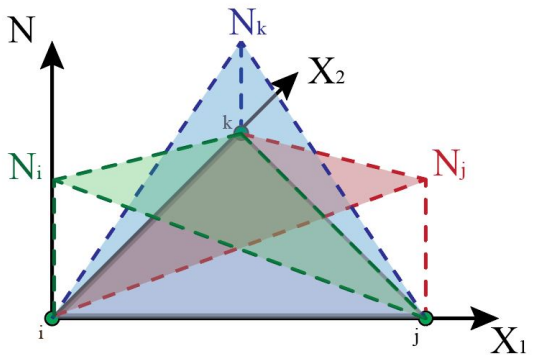


Figura 4.2: Geração de vórtices conforme aumento do número de Reynolds

As funções de forma utilizadas nesse trabalho são dadas pelo vetor:

$$\mathbf{N} = \begin{bmatrix} N_i & N_j & N_k \end{bmatrix} \tag{4.59}$$

Em que:

$$N_i = \frac{1}{2A}(a_i + b_i x + c_i y) \tag{4.60}$$

Sendo A a área do elemento, dada por:

$$A = \frac{1}{2}\begin{bmatrix} 1 & x_i & y_i \\ 1 & x_j & y_j \\ 1 & x_k & y_k \end{bmatrix} \tag{4.61}$$

Para o elemento triangular, as matrizes elementares ficam:

$$\mathbf{m}^e = \frac{A}{12}\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} \tag{4.62}$$

$$\mathbf{g}^e = (\mathbf{g}_x^e, \mathbf{g}_y^e) = \left(\frac{1}{6}\begin{bmatrix} b_i & b_j & b_k \\ b_i & b_j & b_k \\ b_i & b_j & b_k \end{bmatrix}, \frac{1}{6}\begin{bmatrix} c_i & c_j & c_k \\ c_i & c_j & c_k \\ c_i & c_j & c_k \end{bmatrix}\right) \tag{4.63}$$

$$\mathbf{k}^e = \mathbf{k}_x^e + \mathbf{k}_y^e = \frac{1}{4A}\begin{bmatrix} b_i^2 & b_i b_j & b_i b_k \\ b_j b_i & b_j^2 & b_j b_k \\ b_k b_i & b_k b_j & b_k^2 \end{bmatrix} + \frac{1}{4A}\begin{bmatrix} c_i^2 & c_i c_j & c_i c_k \\ c_j c_i & c_j^2 & c_j c_k \\ c_k c_i & c_k c_j & c_k^2 \end{bmatrix} \tag{4.64}$$

Capítulo 5

Metodologia

O trabalho foca no desenvolvimento de um algorítimo em *Python* totalmente livre como comentado em 1.2. Com a parte teórica formulada no capítulo anterior, esse irá abordar a escolha das geometrias de canais testadas, condições de contorno escolhidas e implementação computacional para realização da simulação.

5.1 Geometrias estudadas

As aletas onduladas possuem uma geometria que varia conforme uma função seno, resultando em um formato parecido com uma onda. Conforme *Ramgadia* [14], os canais ondulados causam a mistura da camada de fluido perto da parede com a do centro do canal, que, junto às oscilações próprias da geometria, proporcionam uma melhor transferência de calor. Um dos primeiros estudos sobre o tema foi realizado por *Goldstein* e *Sparrow* [15] e mostrou que para escoamentos com Reynolds na faixa entre 6000 e 8000 a transferência de calor aumentava em até 3 vezes.

Nishimura et al. [16] realizou estudos experimentais de mecânica dos fluidos para canais ondulados na faixa entre 1 a 670 Reynolds e demonstrou essa recirculação e descolamento da camada limite.

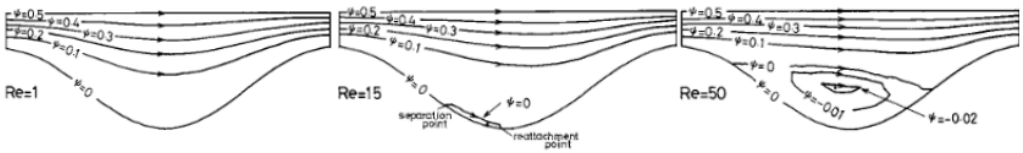


Figura 5.1: Geração de vórtices conforme aumento do número de Reynolds
Fonte: Flow Characteristics in a channel with symmetric wavy wall for steady flow, de Nishimura et al. [16]

Nesse estudo *Nishimura* mostra que o descolamento da camada limite começa a ocorrer em Reynolds igual a 15, formando um vórtex que tem seu tamanho e circulação aumentados conforme o número de Reynolds do fluxo aumenta. O vórtex

mantém um padrão de movimento até $Re = 350$, quando começa a mudar de um padrão estável para um instável.

Wang et al. [17] utiliza dos estudos de *Nishimura* para estudar a convecção e transferência de calor nesses canais, ele mostra que a eficiência desse tipo de geometria é maior para escoamentos não laminares, mas que para escoamentos laminares já existe um pequeno ganho na melhora de transferência de calor. A figura 5.2 abaixo mostra como o perfil da temperatura normalizada se altera na região de recirculação e como se obtém maiores temperaturas ao centro do canal. Esse acréscimo de temperatura é atribuída a recirculação mencionada anteriormente, que permite o fluido absorver mais calor das aletas.

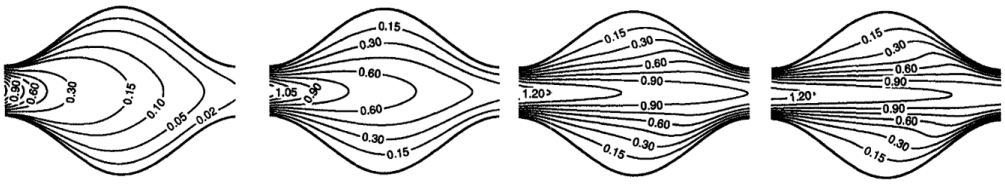


Figura 5.2: Mudança do perfil de temperatura conforme aumento do Reynolds
Fonte: *Convective heat transfer in periodic wavy passages*, de *Wang et al.* [17]

5.2 Geração de Malha

Após o estabelecimento das geometrias adotadas no trabalho, deve-se gerar as malhas utilizadas nas simulações, tanto nas validações, quanto nos estudos de geometrias. Para isso foi utilizado o software livre *Gmsh* [3], ele permite o desenho da geometria, assim como a geração da malha, com classificação dos nós e elementos conforme necessidade, e a obtenção da matriz de conectividade, conhecida como IEN.

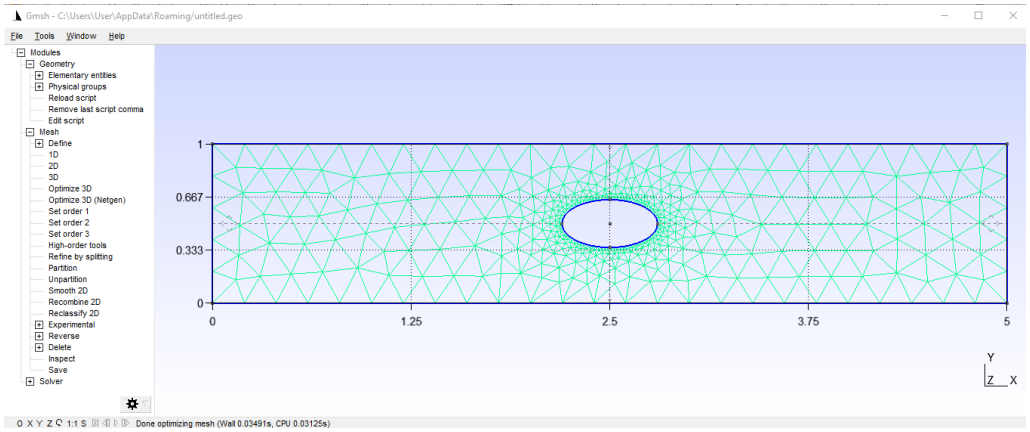


Figura 5.3: Interface gráfica do Gmsh com uma malha 2D triangular

Assim, o primeiro passo tomado é o desenho da geometria de interesse e após

isso, então é feita a classificação de todos os contornos de interesse e por último deve-se gerar a malha para a simulação. Para haver a convergência dos resultados e uma boa resolução, é importante considerar o tamanho dos elementos, ou refino da malha. Além de ser adotada uma malha mais refinada ao redor de uma superfície de interesse, para garantir a estabilidade do método matemático nas regiões mais complicadas sem a necessidade de refinar, pesando no processamento, áreas de menor interesse.

A etapa de geração de malhas ainda pre-vê a geração de outras malhas com níveis de refinamento diferentes, para análise dos resultados gerados por elas e verificar quais malhas apresentam resultados convergentes. Essa etapa é chamada de análise de convergência e nos permite verificar uma malha que traga resultados precisos sem necessariamente utilizar a malha mais refinada e que necessita de maior poder computacional para solução.

5.3 Algorítimo

O algorítimo é será o responsável por transformar o modelo matemático desenvolvido na simulação propriamente dita. Para seu desenvolvimento foi utilizada a linguagem Python, por oferecer uma ampla opção de bibliotecas com todas as ferramentas necessárias para a construção da simulação de maneira acessível e livre. De forma geral, o algorítimo pode ser descrito pelo fluxograma abaixo:

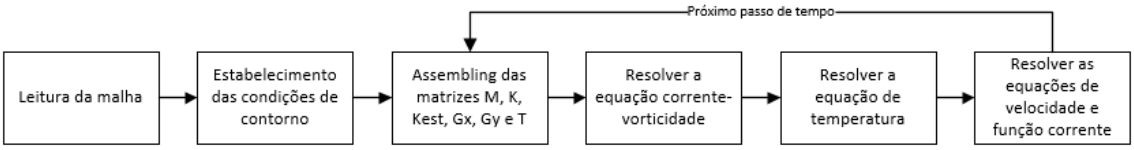


Figura 5.4: Fluxograma do algorítimo

O primeiro passo é fazer a leitura da malha gerada pelo *Gmsh*, para isso é utilizada a biblioteca *meshio*. Com seu uso, são obtidos os vetores com os pontos de cada contorno de interesse e os vetores com os pontos pertencentes a cada uma das superfícies de interesse.

5.3.1 Condição de contorno e iniciais

Com os vetores de cada contorno, é possível assimilar cada ponto com suas respectivas condições de contorno. Nesse trabalho só serão adotadas condições de contorno de Dirichlet, ou de Neumann homogêneo para simplificação do problema proposto. Para as condições de Dirichlet, é feito um loop simples para cada contorno, como é possível observar nas linhas 277 à 290 no código explícito no apêndice A.3.

Já para aplicar as condições de Neumann homogêneo, basta não definir valor para o devido ponto de contorno, deixando-o livre como um ponto fora do contorno.

Para a vorticidade nos contornos é necessário ser feito o cálculo através do método de elementos finitos, a partir da equação 3.17 obtém-se:

$$\mathbf{M}\omega = \mathbf{G}_x v - \mathbf{G}_y u \tag{5.1}$$

Que é a solução, já adimensional para a vorticidade, e seu cálculo é realizado através da multiplicação de $\mathbf{M}^{-1}$ em ambos os lados da equação.

$$\omega = \mathbf{M}^{-1}[\mathbf{G}_x v - \mathbf{G}_y u] \tag{5.2}$$

Com ela será obtida a vorticidade do contorno para a condição inicial e cada um dos passos de tempo. A vorticidade, em específico é a única condição de contorno implementada pelo próprio código e inserida após a operação de assembling das matrizes globais vistas na próxima seção.

5.3.2 Assembling

O próximo passo é realizar a montagem das matrizes globais utilizadas em cada uma das equações governantes do problema através do processo de assembling. Para realizar esse processo foi criado um módulo Assembly A.2, nele foi escrito o processo padrão de assembling para os elementos triangulares escolhidos e utilizados no código. A escolha de modularizar essa parte do código se deve ao processo tomar um grande espaço, deixando assim a leitura do código poluída para quando necessitasse ser debugado.

5.3.3 Resolução das equações de governo

Após a montagem das matrizes globais foi implementada a solução das equações de governo. Foi escolhido montar a matriz A antes do loop temporal, visto que essa não irá mudar conforme os passos de tempo, apenas o vetor x. Assim, a implementação das condições de contorno de Dirichlet para a matriz A também foi implementada, sendo feita a substituição de todos os valores da linha referente ao ponto da condição de contorno com zeros(exceto o valor da diagonal, o qual foi substituído com 1) para assim garantir que o valor do ponto será fixo.

Em cada passo de tempo é feita a substituição dos valores das condições de contorno no vetor b e calculado os valores do vetor x para cada uma das equações de governo. Ao final de cada iteração foi utilizada a biblioteca *meshio* para gravar os resultados em arquivos ".VTK", a fim de poder realizar a visualização desses pelo *Paraview* [4]. Para o acompanhamento da progressão do código pelas iterações, foi

feita a utilização da biblioteca *tqdm*, que permite fazer o acompanhamento mediante uma barra de progressão no terminal.

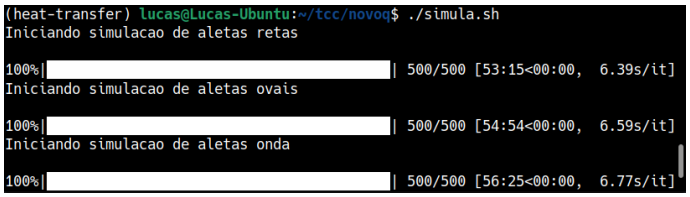


Figura 5.5: visualização da progressão das simulações com o auxílio de tqdm

Ao final das iterações é feito os cálculos de U_{med}, U_{max}, T_{med} e T_{max}, sendo gravados em *Excel* com os dados da simulação, número de pontos, número de elementos, tempo de simulação, Reynolds e Péclet.

5.4 Visualização

Após o cálculo das soluções para cada passo de tempo, é utilizado do *Paraview* para visualização dos dados. Ele oferece ótimos recursos, como animação permitindo a visualização de cada passo de tempo e de como a simulação se desenvolve. Também permitiu observar a correta colocação das equações de contorno, visto que o código tem uma gravação de dados para as condições de contorno.

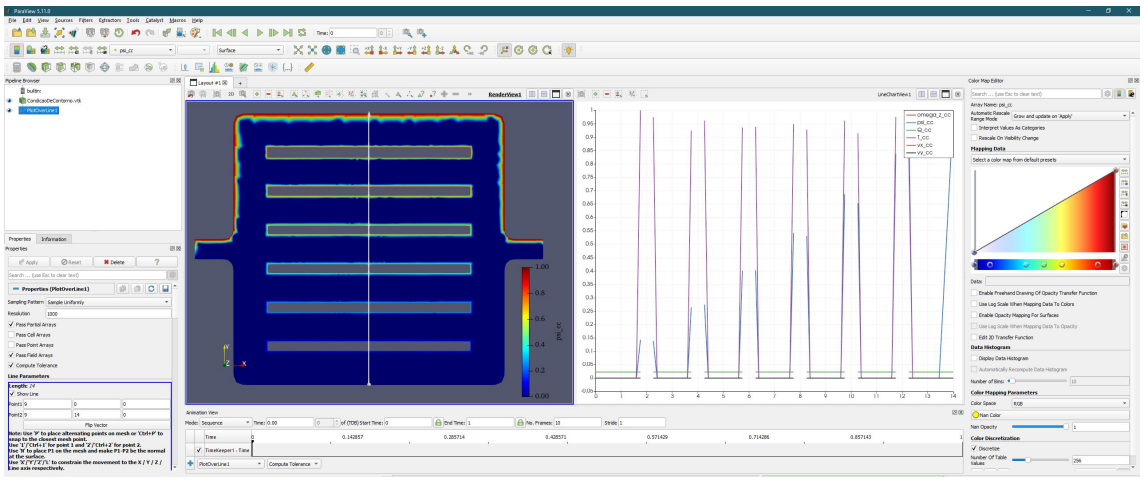


Figura 5.6: Condições de contorno observadas com a ajuda do software *Paraview*

O *Paraview* também permite a análise de seções, podendo serem escolhidos as variáveis que serão plotadas em um gráfico 2D e sendo possível a animação desse também, para verificação em cada passo de tempo.

Capítulo 6

Validação

Pelo trabalho se tratar de uma simulação feita do zero, se faz necessária a prova de que tanto o modelo matemático como o algorítimo funcionam. E para isso é realizada a validação mediante problemas já conhecidos e estudados pela literatura, assim foram escolhidos 3 casos conhecidos para realizar a validação do código e dar base aos resultados encontrados no estudo.

6.1 Escoamento Hagen-Poiseuille

Um dos casos mais conhecidos é o escoamento Hagen-Poiseuille, que descreve um escoamento entre duas placas planas infinitas ou em dutos gerados por um diferencial de pressão. Por ser um problema clássico, esse é um tema muito estudado pela literatura e com várias soluções analíticas dispostas na internet, tornando-o um bom problema para a validação do algorítimo para a parte de mecânica dos fluidos, como para a parte térmica. Para essa validação será simulado um fluxo de Poiseuille com as condições de contorno descritas na figura abaixo:

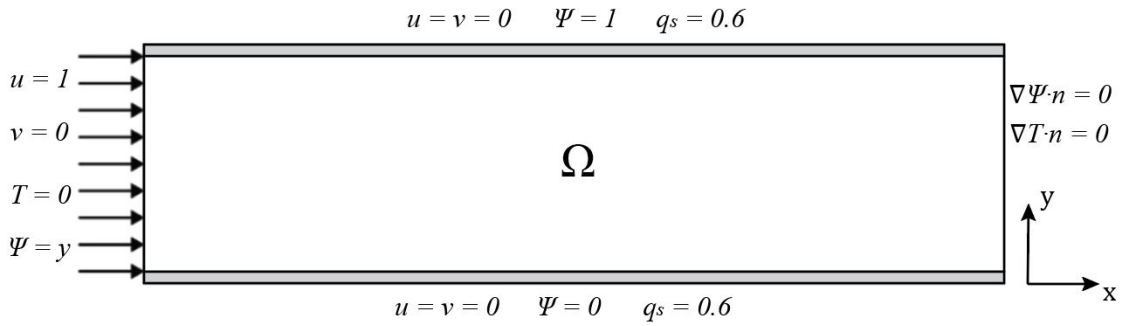


Figura 6.1: Condições de contorno para a simulação do escoamento Hagen-Poiseuille

Para a análise, é importante que o fluido esteja dinâmica e termicamente desenvolvido, para isso é visto que o comprimento de entrada para desenvolvimento do fluido é de:

$$L_e = 0.05 R_e D \tag{6.1}$$

Definindo $D = 1$[-] e $Re = 10$, temos que um comprimento $L > 0.6$[-] nos permitirá observar um escoamento desenvolvido. Além disso, os parâmetros da simulação foram definidos conforme a tabela abaixo:

Tabela 6.1: Parâmetros da simulação do escoamento de Poiseuille

Parâmetro	Valor
Número de iterações	600
Δt	0.01
Número de nós	3797
Número de elementos	7292
Reynolds (Re)	10
Prandlt (Pr)	1

Por fim, antes de realizar a simulação é importante ressaltar as hipóteses levantadas pelo algorítimo proposto:

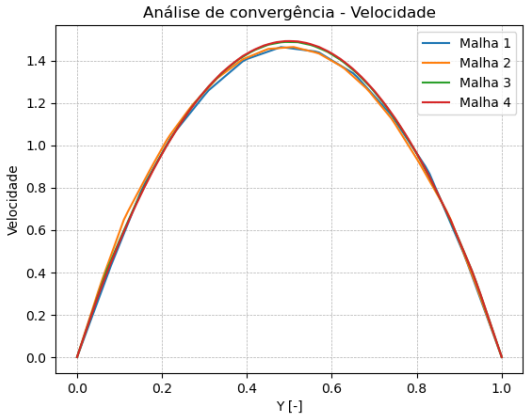
- O fluido utilizado é newtoniano;
- Sua viscosidade é constante;
- O escoamento é incompressível.

Para a análise de convergência de malha, foram feitos testes com 4 refinamentos de malhas até ser obtida a convergência de malha.

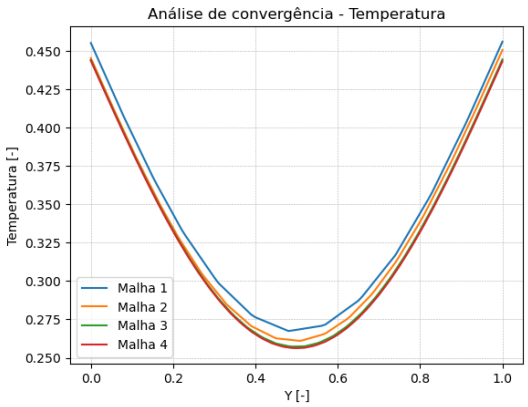
Tabela 6.2: Análise de convergência de malha

Parâmetro	Malha 1	Malha 2	Malha 3	Malha 4
Número de nós	663	1329	3797	6747
Número de elementos	1204	2482	7292	13090
Tempo de processamento [s]	13,2	114,7	1316,4	5760,2

Com essas malhas, foram comparadas as soluções de $\mathbf{u}$ e $\mathbf{T}$ e observado que para as malhas 3 e 4 há a convergência de resultados.



(a) Análise de perfis de velocidade em x =2.5



(b) Análise de perfis de temperatura em x =2.5

A seguir é possível verificar o perfil em mapa de calor de toda a superfície da seção do escoamento analisada:

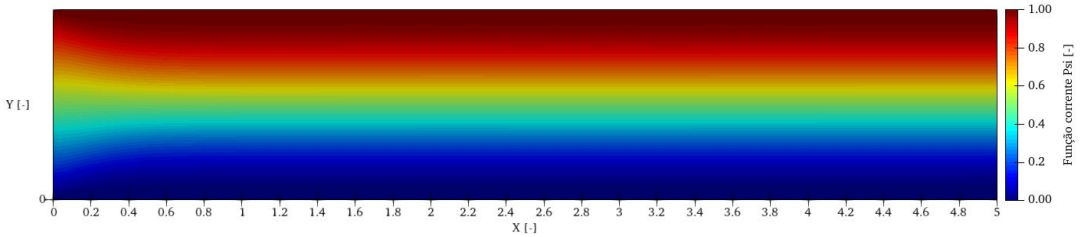


Figura 6.3: Distribuição da função corrente

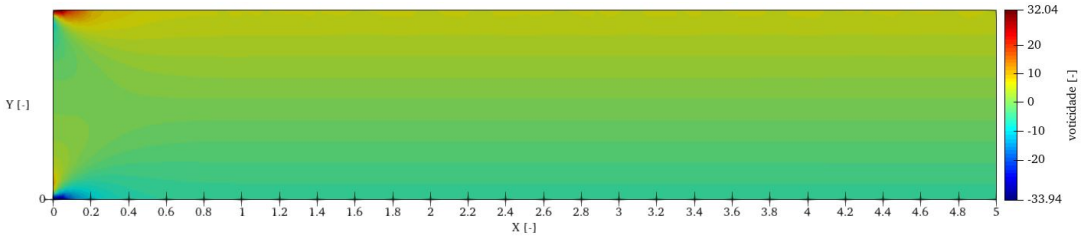


Figura 6.4: Distribuição da vorticidade

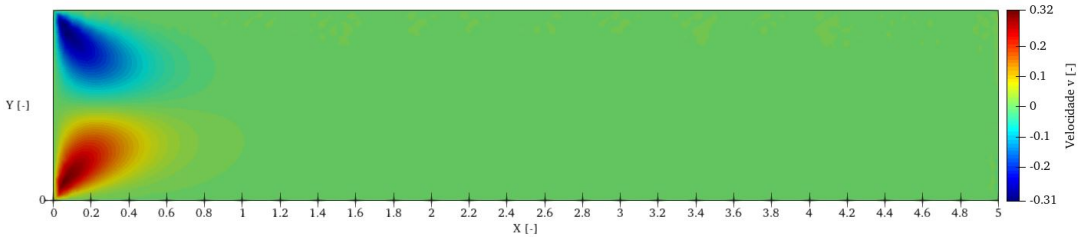


Figura 6.5: Distribuição da velocidade **v**

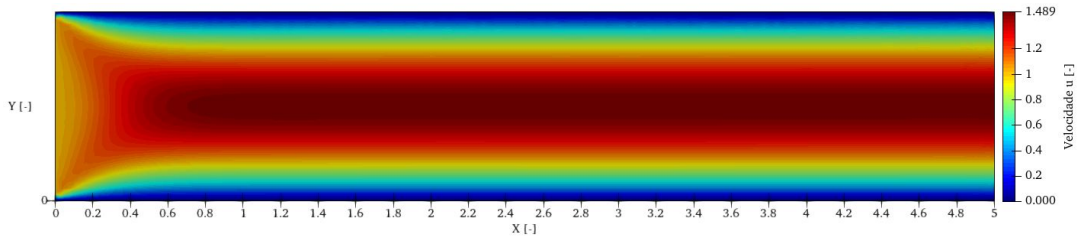


Figura 6.6: Distribuição da velocidade **u**

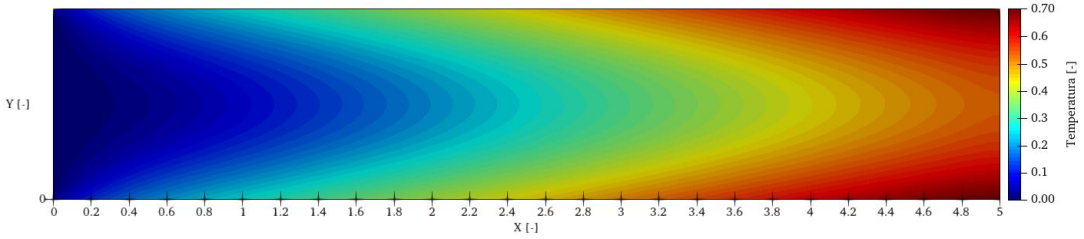


Figura 6.7: Distribuição da temperatura **T**

Conforme a solução analítica contida no livro de *Fenômenos de Transferência* de *Pontes e Mangiavacchi* [18], a equação que descreve a temperatura do fluido na região completamente desenvolvida do fluido é dada por:

$$T(y) = T_0 - \frac{15}{48}q + \frac{2}{Re(-\partial P/\partial x)}qx + 3q\left(\frac{(y - D/2)^2}{2} - \frac{(y - D/2)^4}{3}\right) \tag{6.2}$$

E a solução analítica da velocidade, para $D = 1$ e $-\partial P/\partial x = 12$, é dada por:

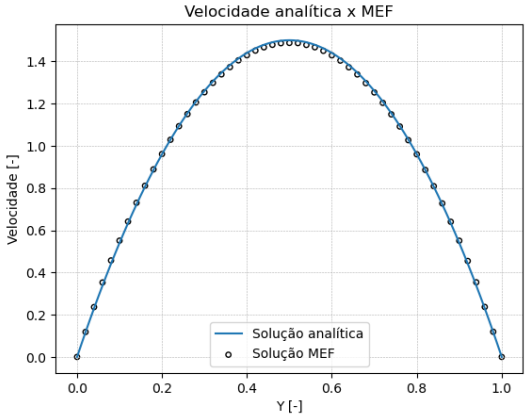
$$u(y) = 6y(1 - y) \tag{6.3}$$

Com as equações analíticas é possível realizar a validação e observar o quanto o escoamento simulado pelo algorítimo proposto se aproxima do perfil real. Abaixo é possível verificar que as comparações feitas tomando o perfil de temperatura e velocidade em $x = 2.5$:

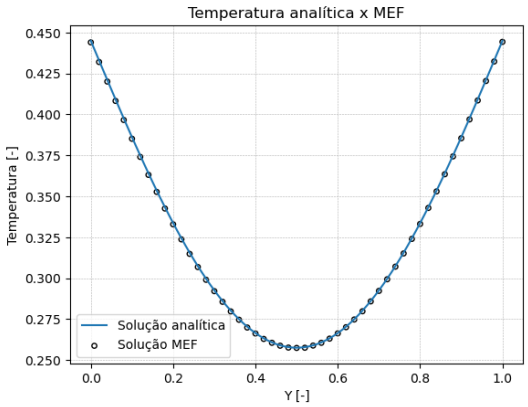
6.2 Simulação em 5 canais - Florentino

Por esse trabalho ser um desdobramento da tese escrita por *Florentino* [2], uma validação importante é tentar reproduzir, a partir do algorítimo e modelo matemático desenvolvidos, os resultados obtidos por ele.

Assim, para comparação, foi desenhada a estrutura inicial com quatro aletas e 4 pontos de aquecimento. As condições de contorno são as mesmas utilizadas por *Florentino*, conforme imagem abaixo:



(a) Perfil de velocidade em x =2.5



(b) Perfil de temperatura em x =2.5

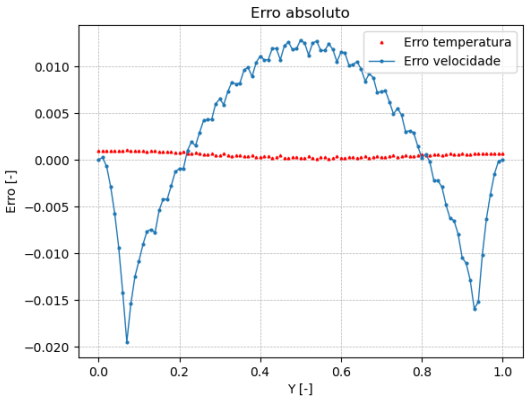


Figura 6.9: Distribuição dos erros absolutos dos perfis de temperatura e velocidade em x =2.5

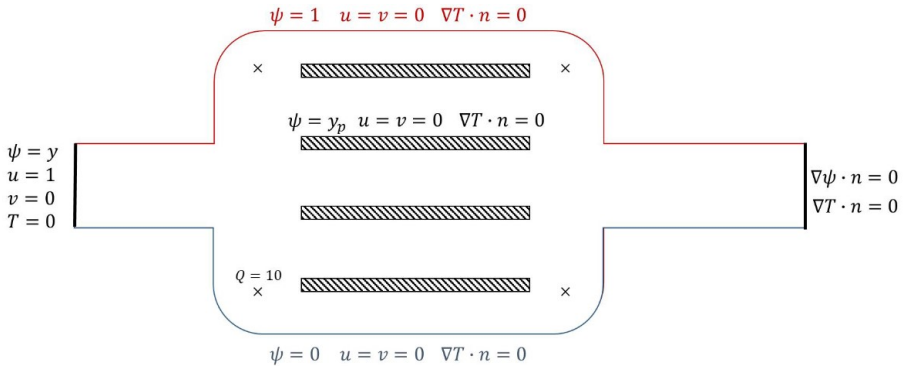


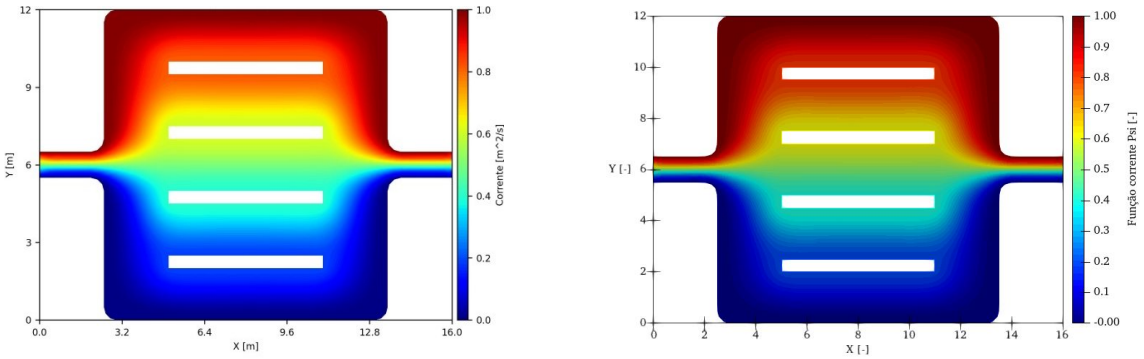
Figura 6.10: Condições de contorno para a simulação de placa fria feita por *Florentino*

Além da geometria e condições de contorno, é importante utilizar os mesmos parâmetros, tanto em relação ao fluido, como à malha e à simulação, assim os parâmetros utilizados estão relacionados abaixo:

Tabela 6.3: Parâmetros da simulação de 5 canais

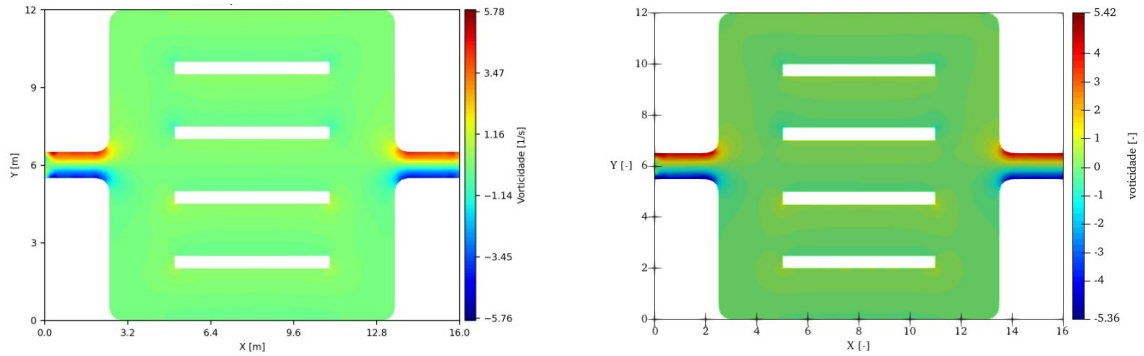
Parâmetro	Valor
Número de iterações	240
Δt	0.005
Número de nós	4618
Número de elementos	8702
Reynolds (Re)	0,23
Prandlt (Pr)	0,7968

Após definir os parâmetros, foi então realizada a simulação com o algorítimo proposto e utilizado o software *Paraview* para visualização dos resultados. Nas figuras 6.11, 6.12, 6.13, 6.14 e 6.15 abaixo estão explicitadas as grandezas: função corrente Ψ, vorticidade ω, velocidade $\mathbf{u}$, velocidade $\mathbf{v}$ e temperatura $\mathbf{T}$. Todas as imagens foram tomadas após as 240 iterações da simulação.



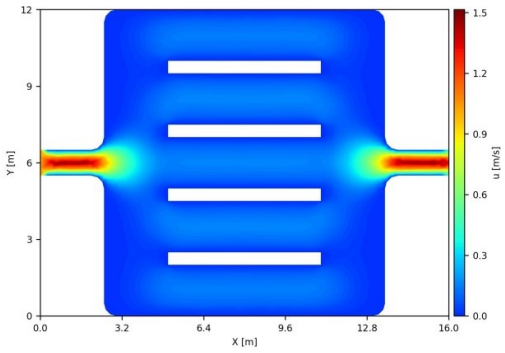
(a) Perfil de **Psi** obtido por *Florentino* (b) Perfil de **Psi** obtido no estudo

Figura 6.11: Comparação, perfil da função corrente **Psi**

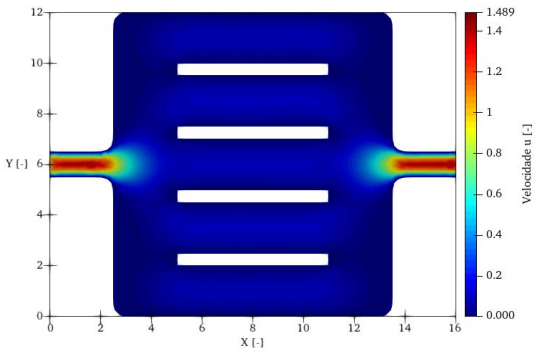


(a) Perfil de $\omega_\mathbf{z}$ obtido por *Florentino* (b) Perfil de $\omega_\mathbf{z}$ obtido no estudo

Figura 6.12: Comparação, perfil da vorticidade $\omega_\mathbf{z}$

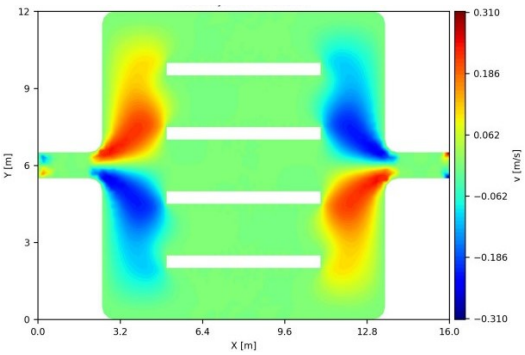


(a) Perfil de **U** obtido por *Florentino*

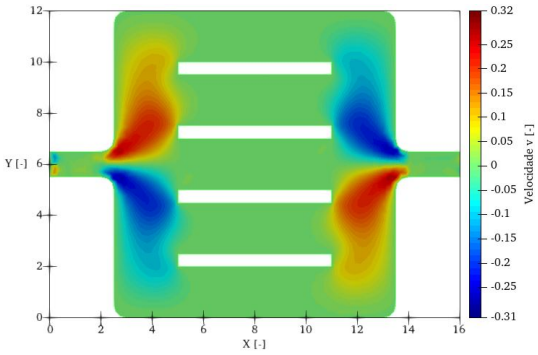


(b) Perfil de **U** obtido no estudo

Figura 6.13: Comparação, perfil da velocidade **U**

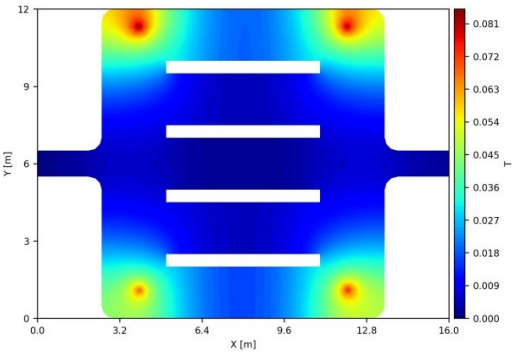


(a) Perfil de **V** obtido por *Florentino*

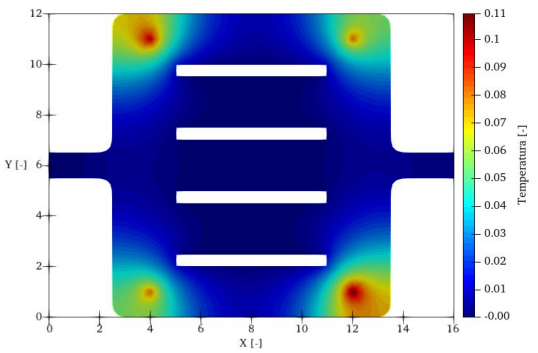


(b) Perfil de **V** obtido no estudo

Figura 6.14: Comparação, perfil da velocidade **V**



(a) Perfil de **T** obtido por *Florentino*



(b) Perfil de **T** obtido no estudo

Figura 6.15: Comparação, perfil da temperatura **T**

Conseguimos observar que visualmente o comportamento do fluido apresenta muitas semelhanças entre as duas simulações, mas que a faixa de valores obtidos se altera um pouco em suas extremidades, com o estudo do *Florentino* chegando a maiores valores nas velocidade **u**, e o algorítimo desse trabalho obtendo maiores valores em **T** e **v**.

Essas divergências fazem sentido dentro da parte teórica, visto que uma maior velocidade, leva a uma melhor distribuição de calor pelas partículas do fluido. Logo,

a divergência entre os códigos pode se dar pelas condições de contorno aplicadas, que não tem a definição do perfil de velocidades na saída. Também é possível verificar que há diferença na aplicação dos valores de contorno na entrada, que neste estudo da prioridade às condições impostas pelas paredes laterais do canal, enquanto no estudo do *Florentino* a entrada é quem ganha prioridade.

Tabela 6.4: Comparação entre os parâmetros obtidos com os observados no estudo de *Florentino*

Parâmetro	*Florentino*	Simulação
Velocidade **u** máxima [-]	1.503	1.489
Velocidade **v** máxima [-]	0.309	0.32
Temperatura **T** máxima [-]	0.084	0.11

Capítulo 7

Resultados

Esse trabalho visa estudar uma configuração de geometrias diferente da usual utilizada para arrefecimento liquido para computadores domésticos. Como tal, utilizaremos parâmetros usuais de bombas utilizadas na montagem de *desktops*, assim como utilizaremos água como o líquido refrigerante, visto que esse é uma alternativa comumente empregada e com bons parâmetros para o estudo.

Para o projeto da placa fria, foi estudado o design de um processador AMD 7700X, conforme os dados das especificações técnicas da AMD [19].

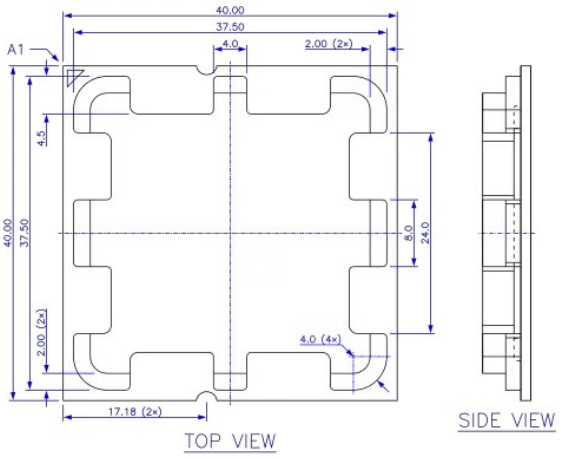


Figura 7.1: Especificações técnicas de design para chipsets AM5, dimensões em milímetros

Também foi estudada a distribuição de temperatura na superfície do chip, para poder entender como acontece o aquecimento da superfície do dissipador de calor integrado do processador. Para isso foi usado como fonte o estudo feito por *Wallossek* [20], dono do site Igorslab que estuda os mais diversos componentes eletrônicos presentes nos computadores.

Nesse estudo, foi aplicada uma pequena camada de material sobre o processador, o qual possui uma emissividade conhecida para obter leituras precisas pelo sensor

térmico. O estudo mostra a variação de temperatura em 5 pontos de interesse na superfície do *IHS* do chip. Tendo em seu maior regime de aquecimento uma distribuição quase uniforme de temperatura, com o centro do chip obtendo uma temperatura de 86.43^{o}C, apenas 4% maior que nas pontas do dispositivo.

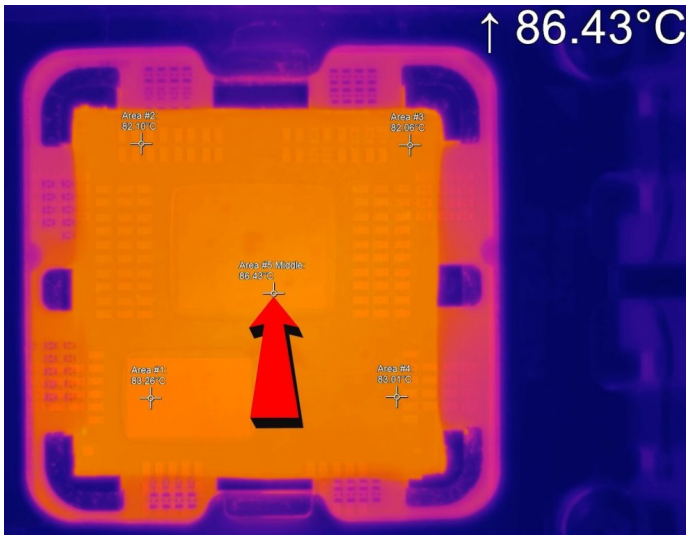


Figura 7.2: Distribuição de temperatura em um Ryzen 7700X
Fonte: IgorsLAB

Assim, foi pensado em um sistema composto por um canal de entrada, um canal maior ao centro com as dimensões da parte central do CPU e um canal de saída. Tendo o canal central subdividido em 7 canais menores e de comprimentos iguais.

A fim de simplificação, será considerado que a altura da placa fria é desprezível se comparado com o comprimento, podendo a placa fria ser reproduzida por uma geometria em duas dimensões, sendo elas em X e Y, e tendo como resultado a geometria proposta abaixo:

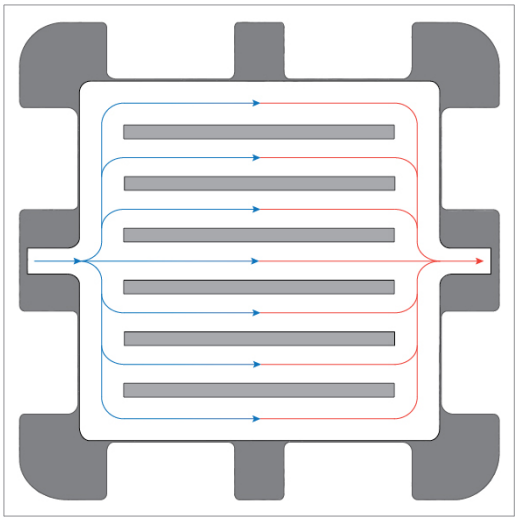


Figura 7.3: Projeto placa fria sobre o processador

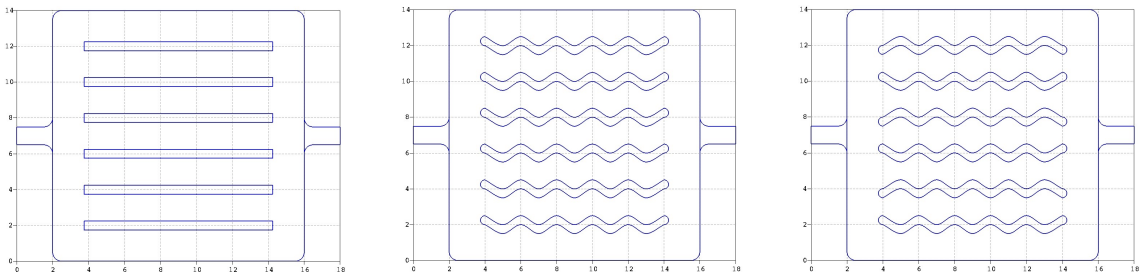
Essa geometria permite que os canais menores, que possuem as aletas que conduzem parte do calor para resfriamento, estejam em contato com a superfície do chip que mais produz calor. A geometria base da placa fria tem as seguintes especificações:

Tabela 7.1: Dimensões-base da placa fria

Altura dos canais de entrada e saída	2 mm
Altura total do canal central	28 mm
Raio de curvatura das quinas	1 mm
Comprimento dos canais internos	20 mm
Altura média dos canais internos	3 mm

Utilizando o trabalho de *Ramgadia* como base, foram escolhidas duas geometrias para estudo. Essas, formadas por aletas onduladas com função $y(x) = 0.25\sin(\pi x)$. Visando verificar se, para condições normais de operação de bombas de *water coolers* convencionais, é possível observar o descolamento da camada limite e a mistura e recirculação desta, proporcionando maior transferência de calor que as aletas retas comuns. Foram montados três sistemas, dois com geometrias de aletas onduladas e um de aletas retas, os quais estão expostos na figura 7.5 abaixo. As geometrias estudadas são:

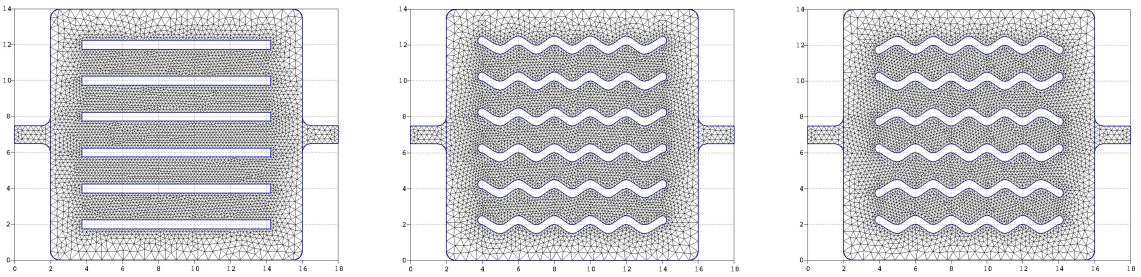
- Aletas retas: tradicionais na indústria e usadas como referência/padrão;
- Aletas onda 0^{o}: aletas com formato ondulado em fase entre os pares;
- Aletas onda 180^{o}: aletas onduladas com 180^{o} defasadas entre os pares.



(a) Aletas retas (b) Aletas onda 0^{o} (c) Aletas onda 180^{o}

Figura 7.4: Geometrias de aletas

Para a malha foram utilizadas 3 densidades de pontos, a maior delas nas aletas, visto que essa região é a de maior interesse, uma intermediária nos canais de entrada e saída e a menor nas paredes exteriores da cavidade central, visto que essas são de menor interesse. Os estudos de convergência também foram realizados para as malhas do projeto, até chegar as malhas abaixo.



(a) Malha aletas retas (b) Malha aletas onda 0º (c) Malha aletas onda 180º

Figura 7.5: Malha de aletas

Tabela 7.2: Parâmetros das malhas

Aletas	Retas	Ondas 0	Ondas 180
Numero de nós	4801	5148	5047
Numero de elementos	8724	9334	9132

Com o intuito de facilitar o desenvolvimento teórico e computacional, foram adotadas as seguintes hipóteses simplificadoras:

- O fluido utilizado é newtoniano;
- Sua viscosidade é constante;
- O escoamento é incompressível;
- Condição de não deslizamento nas paredes;
- A geração de calor ocorre uniformemente por todo o canal central;
- As aletas possuem temperatura fixa (condição de contorno de Dirichlet).

Para garantir uma boa base de comparação, todos os parâmetros, tanto de fluido e escoamento como de simulação, foram mantidos em todas as geometrias, sendo a única diferença a quantidade de elementos e pontos das malhas, que foram mantidos o mais próximos possível, como observado na tabela 7.2 acima.

Tabela 7.3: Parâmetros do fluido e escoamento

Fluido utilizado	Água
Densidade	997 m^3/s
Viscosidade	0.001 Pa·s
Calor específico	4181 J/kgK
Condutividade térmica	0.58 W/mK
Velocidade de entrada	2 cm/s
Reynolds	30
Peclet	216.26

Os números de Reynolds e Péclet são calculados a partir das equações 3.40 e 3.53 respectivamente. O parâmetro utilizado para geração de calor será a *TDP* padrão do processador Ryzen 7700X, esse sendo distribuído uniformemente por toda a superfície de contato da placa fria com o *IHS* do chip.

Tabela 7.4: Parâmetros da simulação

Δt	0,002 s
Iterações	1500
Temperatura das paredes das aletas	40 °C
Geração de calor	105 W

Como visto no capítulo 3, e novamente no 4, as equações governantes utilizadas estão em sua forma adimensional. Assim, utilizaremos os números adimensionais de Reynolds e Péclet da tabela 7.3, juntamente com as variáveis adimensionais das condições de contorno do problema. Para manter a uniformidade do estudo entre as geometrias, as condições de contorno aplicadas para as três geometrias são iguais. Com $T_s = 1$, considerando que a temperatura da parede das aletas é igual à temperatura da superfície do chip utilizada como referência.

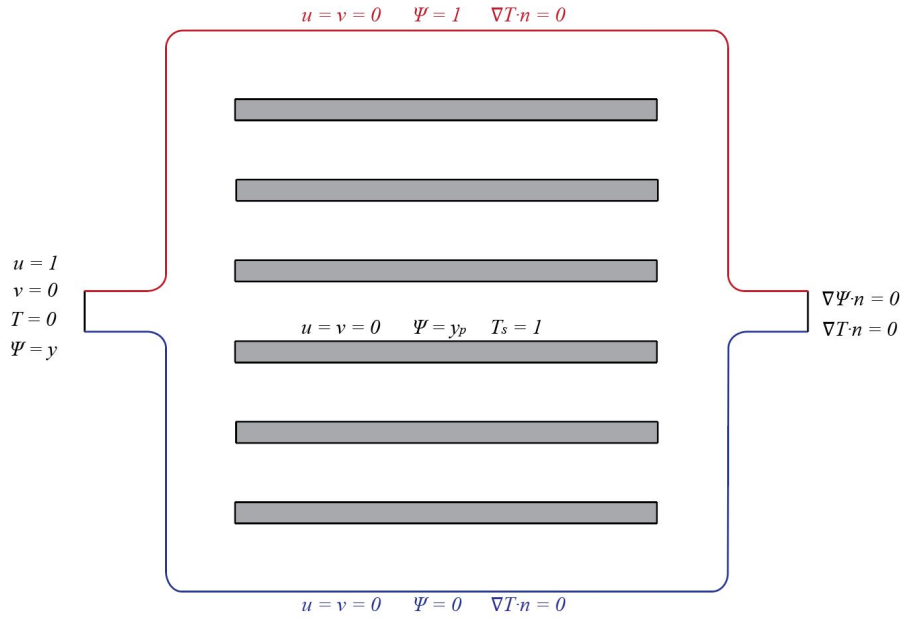


Figura 7.6: Condições de contorno

Como visto anteriormente a distribuição de calor pela superfície do processador se da quase que uniformemente e utilizando da hipótese simplificadora em que a altura é desprezível, podemos dizer que a geração de calor do processador é dada no mesmo plano da geometria. Assim, a condição de contorno utilizada para a geração de calor foi a distribuição uniforme da *TDP* de um Ryzen 7700X, por toda a área do canal central, conforme figura 7.7 abaixo.

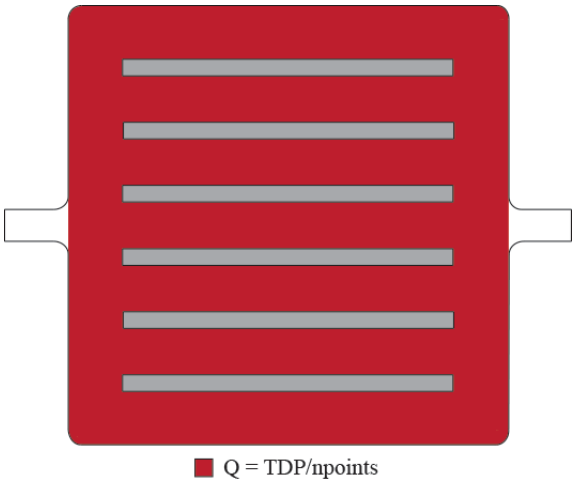
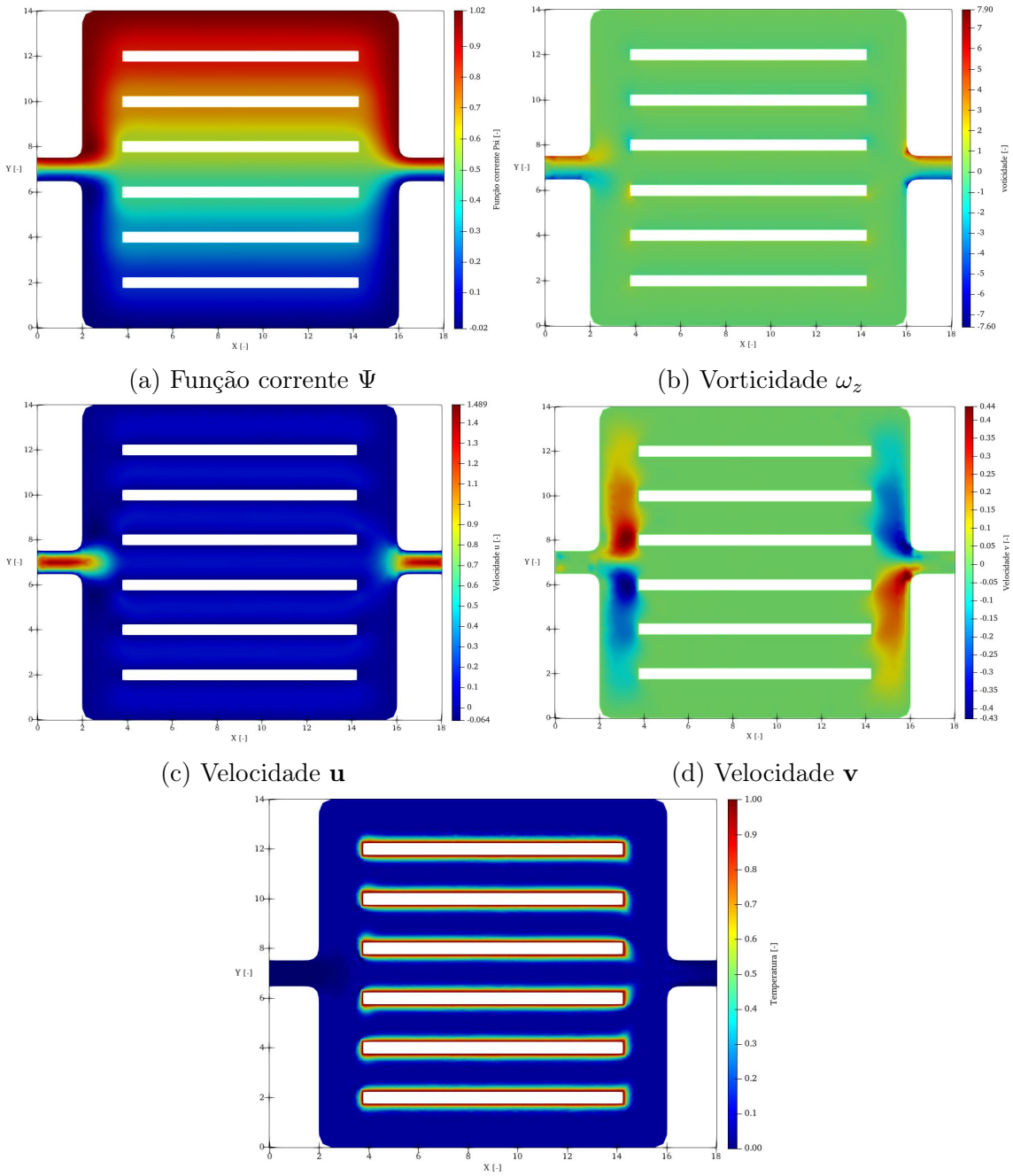


Figura 7.7: Condições de contorno

Com todos os parâmetros e hipóteses definidos, nas próximas sessões serão expostos os resultados obtidos a partir das simulações de cada geometria e por fim será comparado os perfis de temperatura e velocidade, assim como o resultado da temperatura média e máximas temperaturas observadas.

7.1 Aletas retas

A simulação das aletas retas foi realizada como controle, para ter uma base a qual comparar e analisar as eficiências das outras geometrias. Na figura 7.8 a seguir, é possível verificar os resultados dos perfis dos vetores Ψ, ω_z, $\mathbf{u}$, $\mathbf{v}$, $\mathbf{T}$ após 1500 iterações, ou 3 segundos:



(a) Função corrente Ψ

(b) Vorticidade ω_z

(c) Velocidade $\mathbf{u}$

(d) Velocidade $\mathbf{v}$

(e) Temperatura $\mathbf{T}$

Figura 7.8: Solução da simulação em 1500 iterações

O primeiro ponto a se notar é a distribuição da função corrente $\mathbf{\Psi}$ na figura 7.8a. Nela é possível observar que as linhas de corrente oscilam logo após sair do canal de entrada até se estabilizarem, essa oscilação se deve ao aumento súbito de espaço disponível para o escoamento e gera uma pequena zona de recirculação. Essa zona também pode ser notada ao observar o perfil de velocidade $\mathbf{u}$ na figura 7.8c, que explicita a região com um azul mais escuro denotado a uma velocidade negativa de pequena magnitude.

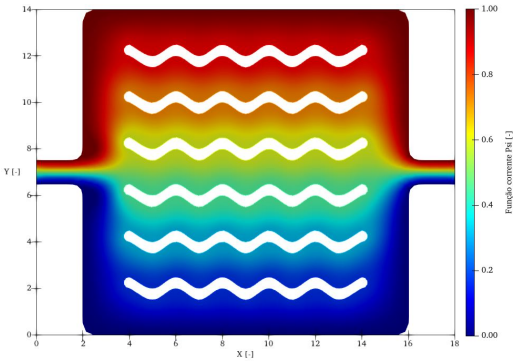
Na região de saída do fluido as linhas de corrente constringem de forma mais linear, resultando em um escoamento com menor gradiente e sem recirculações. Tal fato se reflete nos gráficos dos vetores 7.8c e 7.9d, que apresentam padrões diferentes entre as regiões de entrada e saída do canal central. Na entrada ao canal central, o vetor $\mathbf{u}$ mantém maiores velocidades em uma região maior se comparado com a saída em que a velocidade aumenta de maneira gradual conforme se aproxima da abertura do canal de saída.

O vetor $\mathbf{v}$ tem seu comportamento explicitado pela imagem 7.9d, nela podemos observar que o mesmo possui seus maiores valores nas regiões de entrada e saída do canal central, relacionados a expansão e constrição súbitas. Apesar de possuir valores baixos pela maior parte do escoamento, nessas regiões é possível notar que o mesmo atinge seus maiores valores para que o fluido ocupe e desocupe o espaço extra que o canal central dispõe.

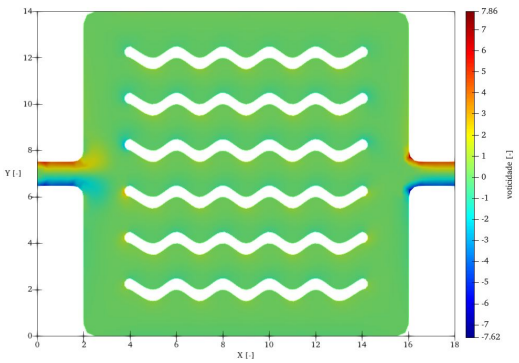
A figura 7.8e expõe a solução do vetor T para o sistema, nela é possível notar pequenas ondulações na parte da solução referente às entradas das aletas centrais. Essas ondulações são instabilidades associadas à simulação e acontecem pelo alto valor de Péclet do escoamento, mas ao entrar nos canais menores essas flutuações quase desaparecem, ficando muito pequenas e em sua maior parte associadas aos canais das extremidades. Também é possível observar o deslocamento da temperatura conforme o fluxo do fluido, com destaque à diferença de temperatura entre o canal de entrada e de saída, que demonstra a transferência de calor entre das aletas para o fluido. A temperatura dentro dos canais também possuem um perfil de parábola achatada, causada pelo aquecimento dado pela geração de calor adicionada pelo código e associada a geração térmica do chip.

7.2 Aletas onduladas 0º

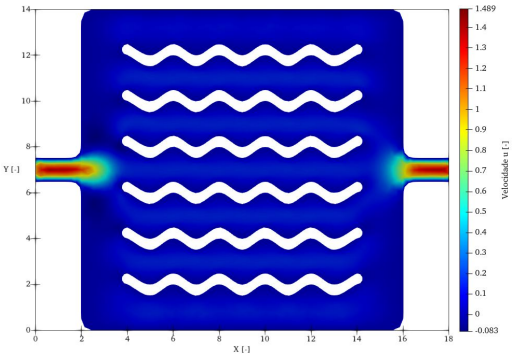
A primeira geometria ondulada estudada é a de aletas onduladas com 0º de defasagem entre elas, na figura 7.9 é possível ver os perfis das soluções obtidas para os vetores Ψ, ω_z, $\mathbf{u}$, $\mathbf{v}$, $\mathbf{T}$:



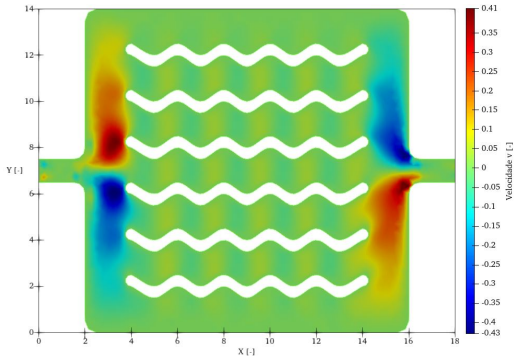
(a) Função corrente Ψ



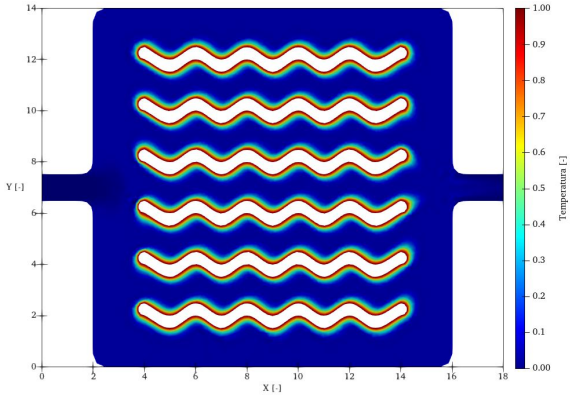
(b) Vorticidade ω_z



(c) Velocidade $\mathbf{u}$



(d) Velocidade $\mathbf{v}$



(e) Temperatura $\mathbf{T}$

Figura 7.9: Solução da simulação em 1500 iterações

A mudança de geometrias traz mudanças no perfil de $\mathbf{\Psi}$ mostrado na figura 7.9a, por não possuir simetria horizontal, a geometria causa um movimento diferente nas linhas de corrente que vão para a parte superior do canal em relação as que vão para a parte inferior. As linhas de corrente que seguem para cima tem uma oscilação maior que as que se direcionam para baixo, isso é devido à quarta aleta (contando de baixo para cima) dar inicialmente mais espaço para o fluido na horizontal, fazendo-o recuar depois quando ocorre a constrição. As linhas de corrente acompanham as ondulações da geometria e depois se unem em um gradiente mais suave antes do estrangulamento para o canal de saída.

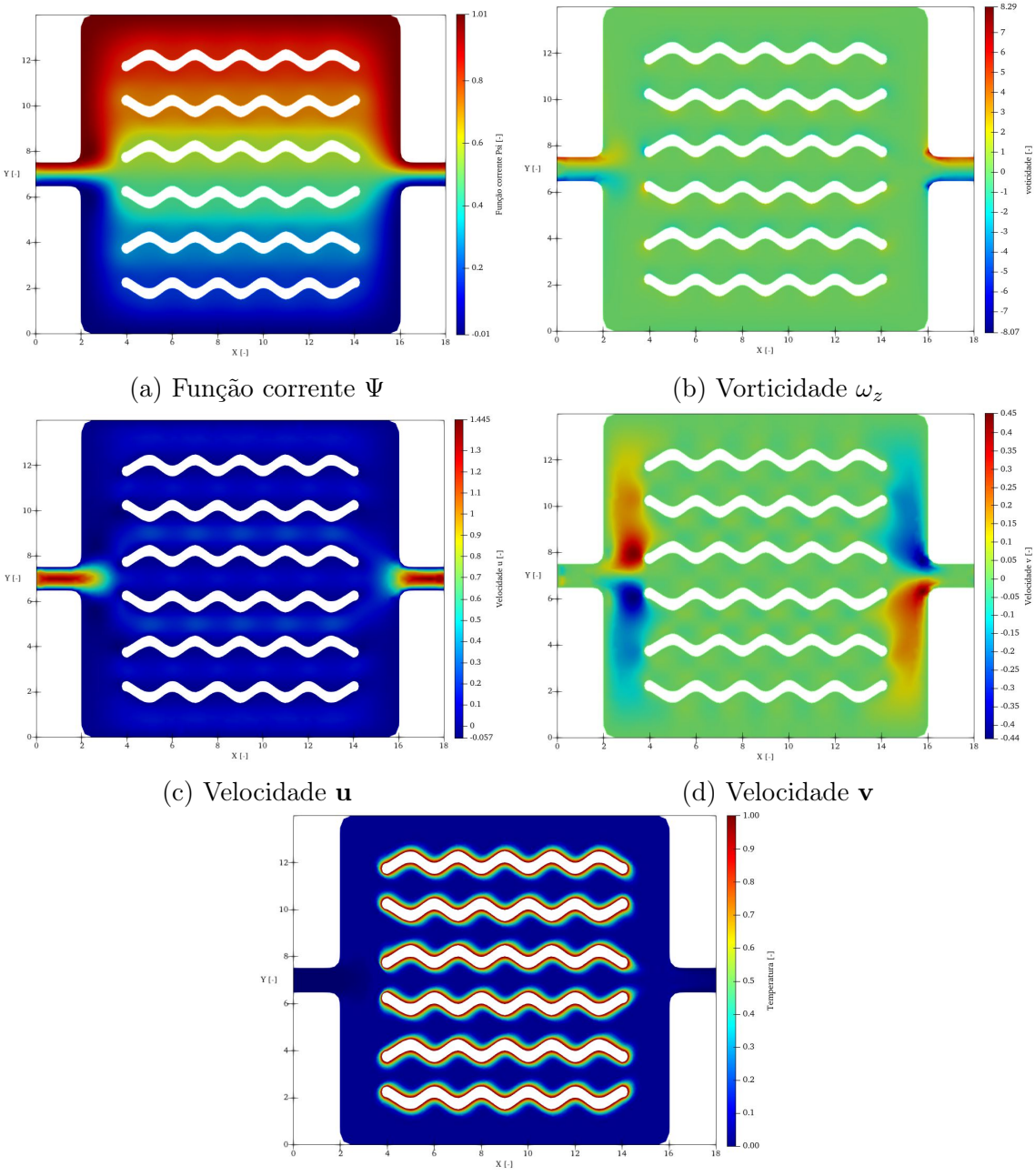
Essa diferença fica mais evidente ao observar os gradientes do vetor $\mathbf{v}$ no gráfico 7.9d, que apresentam padrões diferentes para o fluido que escoa para cima em relação ao que escoa para baixo no eixo Y. Também é interessante notar que nos subcanais intermediários o vetor $\mathbf{v}$ alterna entre valores positivos e negativos, fato que ocorre devido à nova geometria de aleta. Na figura 7.9b é possível observar que os pontos onde há essa mudança de sentido de escoamento, também há uma pequena vorticidade associada, muito menos que as observadas nos canais de entrada e saída que ocorrem pela mudança súbita dos vetores $\mathbf{v}$ e $\mathbf{u}$.

O vetor $\mathbf{u}$ apresenta um comportamento parecido no gráfico da figura 7.9c, tendo como maior diferença pequenas zonas mais escuras nos vales das ondas das aletas, em que há uma desaceleração horizontal do fluido, em detrimento da aceleração vertical para mudança de sentido do escoamento.

A solução da temperatura também apresenta oscilações, como é possível notar na imagem 7.9e, mas dessa vez as oscilações são menores. Assim como visto anteriormente, também é possível verificar o fluido acresce de temperatura conforme se desloca da entrada para a saída do sistema estudado, com a região de entrada apresentando temperaturas menores que a de saída.

7.3 Aletas onduladas 180º

Por último será estudada a geometria de aletas onduladas com 180º de defasagem entre si, diferente da geometria de aletas onduladas 0º, essa possui simetria horizontal. Na figura 7.10 é possível ver os perfis das soluções encontradas para os vetores Ψ, ω_z, $\mathbf{u}$, $\mathbf{v}$, $\mathbf{T}$:



(a) Função corrente Ψ

(b) Vorticidade ω_z

(c) Velocidade $\mathbf{u}$

(d) Velocidade $\mathbf{v}$

(e) Temperatura $\mathbf{T}$

Figura 7.10: Solução da simulação em 1500 iterações

Pelo fato de possuir simetria horizontal, as linhas de corrente, apresentadas pelo vetor $\mathbf{\Psi}$, se distribuem por igual ao sair do canal de entrada, e assim como observado nas geometrias anteriores, também gera uma pequena zona de recirculação e com um gradiente menor antes do estrangulamento do canal de saída. Nos canais intermediários as linhas de corrente mantém o valor médio ao centro dos canais, tendo suas linhas esticadas e contraídas conforme a geometria dos canais oscila.

Essa variação nas linhas de corrente dos subcanais causa uma mudança nos perfis de $\mathbf{v}$ e $\mathbf{u}$, se comparados com as geometrias anteriores. A figura 7.10d mostra como o fluido se expande quando há espaço extra no subcanal e se comprime quando há menos espaço, podendo ser observada uma maior magnitude nos valores de $\mathbf{u}$ durante os estrangulamentos e menor durante as expansões, conforme o gráfico 7.10c. Novamente é possível observar a vorticidade tendo valores mais expressivos nas curvas das ondas, onde ha a inversão dos sentidos do vetor $\mathbf{v}$.

A temperatura apresenta poucas oscilações na sua solução, mostrada na figura 7.10e, mas, assim como nas aletas onda 0^{o}, é possível notar pequenas oscilações na paredes das aletas que formam o canal central. Também é possível observar que a conforme o fluido progride no fluxo estabelecido pelo sistema, sua temperatura aumenta, mostrando que está carregando calor para fora do sistema.

7.4 Comparação entre as geometrias

Após a análise qualitativa feita pelas imagens, que nos dão o perfil de desenvolvimento do escoamento e uma ideia geral do seu comportamento, é importante também ser realizada uma análise quantitativa. Para isso o código foi desenvolvido para coletar dados de temperatura (máxima e média) e velocidade (máxima e média). Como a temperatura no contorno das aletas é fixada em 1 [-], o código para cálculo da temperatura máxima e média exclui esses valores de contorno, para assim nos dar um resultado mais preciso e que não seja tendencioso à geometria que possuir maior número de pontos de contorno em aletas. Os dados obtidos estão expostos na tabela 7.5.

Tabela 7.5: Parâmetros do fluido e escoamento

Geometrias	Aletas retas	Aletas onda 0°	Aletas onda 180°
Temperatura máxima [-]	0.6827	0.7886	0.8024
Temperatura média [-]	0.1674	0.1769	0.1773
Velocidade u máxima [-]	1.421	1.421	1.421
Velocidade u média [-]	0.088	0.086	0.085

Pelos dados da temperatura média de cada geometria, é possível observar que ambas as aletas ondulas possuem uma maior temperatura, com apenas 0,2% de diferença entre si, se comparadas com a aleta reta. A temperatura máxima também

é maior nas aletas onda 180º, com isso é possível notar que a geometria ondulada resulta em cerca de 6% no aumento de temperatura, que reflete uma quantidade maior de calor absorvido do sistema pelo fluido.

As aletas onda possuem menor velocidade u média se comparado com as aletas retas, isso se deve ao fato da geometria mudar a direção de parte do vetor velocidade durante a passagem do fluido pelos subcanais intermediários do canal central. Apesar da velocidade menor, elas são apenas cerca de 2 a 4% menores que da aleta reta, o que mostra que a diferença de geometria não gera muitos impactos para o fluxo do fluido.

O software *Paraview* nos permite traçar um gráfico de linhas de uma seção de interesse da solução plotada, a partir desse recurso é possível analisar como os perfis de cada vetor se comporta dentro da passagem dos canais. Tomando a velocidade **u** e **T** como os perfis de maior interesse, foi traçada a seção dos três subcanais centrais para análise e a partir dai é possível verificar a diferença entre cada perfil para o estudo.

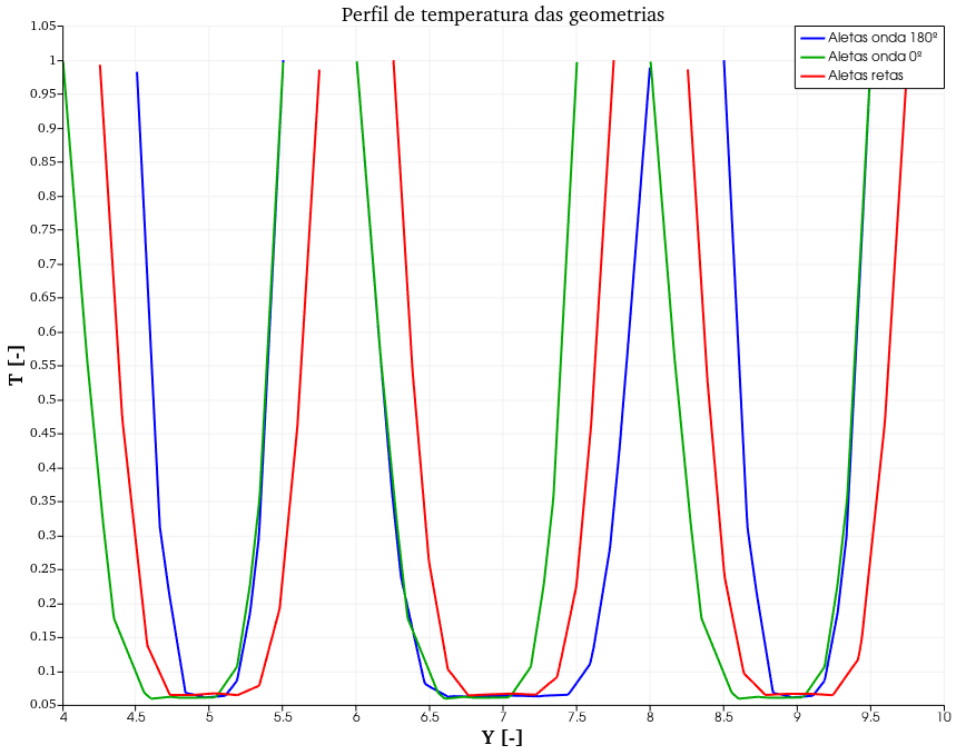


Figura 7.11: Comparação entre os perfis de temperatura em x = 9 [-]

No gráfico 7.11 apresentado acima, é possível observar o plote dos perfis de temperatura tomados ao centro de cada um dos 3 subcanais centrais.

As temperaturas dentro dos canais são todas representadas por parábolas achatadas, esse "achatamento"das parábolas se deve à geração de calor inserida no sistema, que eleva a temperatura base da parábola e mesmo as temperaturas tendo perfis diferentes conforme suas geometrias, todas possuem o mesmo mínimo ao centro de

seus respectivos canais.

Tomando mais de perto cada perfil em separado, é possível observar como a velocidade e as linhas de corrente se correlacionam com a distribuição desses perfis de temperatura. Para isso foram plotados um gráfico para cada geometria, com os perfis de velocidade, linhas de corrente e temperatura da mesma sessão que compreende os 3 subcanais centrais.

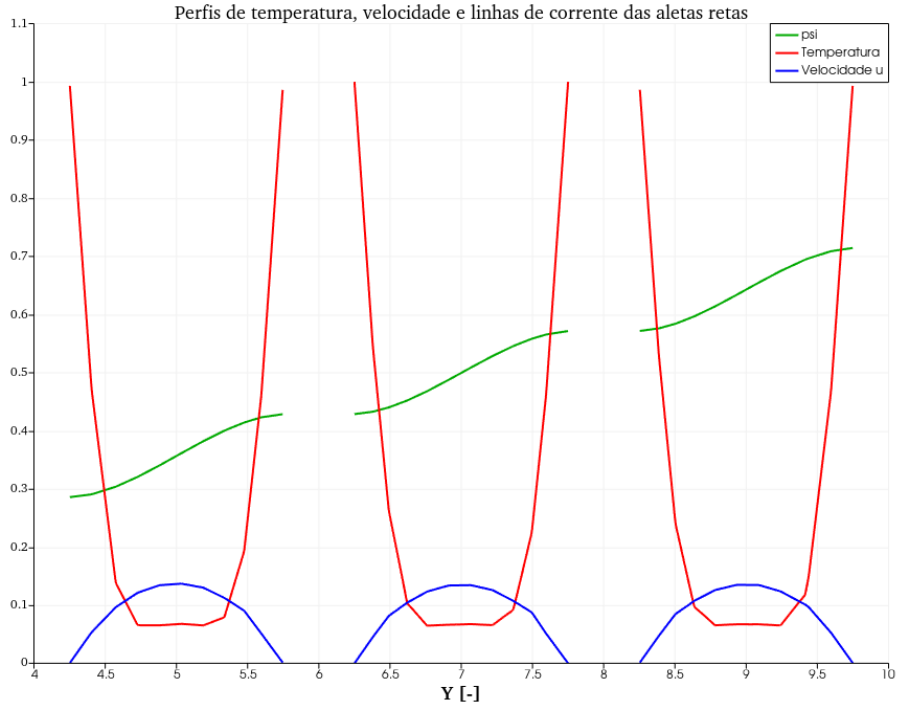


Figura 7.12: Comparação entre os perfis de velocidade, linhas de corrente e temperatura em x = 9 [-] para as aletas onda 0^o

A figura 7.12 mostra que aletas retas apresentam perfis bem simétricos dados pela sua geometria simples, sendo possível verificar que a velocidade e a temperatura apresentam perfis parabólicos bem definidos, assumindo uma velocidade máxima de cerca de 0.15 [-] e uma temperatura mínima de cerca de 0.06 [-]. A função corrente apresenta uma variação em S, como esperado de um fluido desenvolvido e velocidade 0 nas paredes.

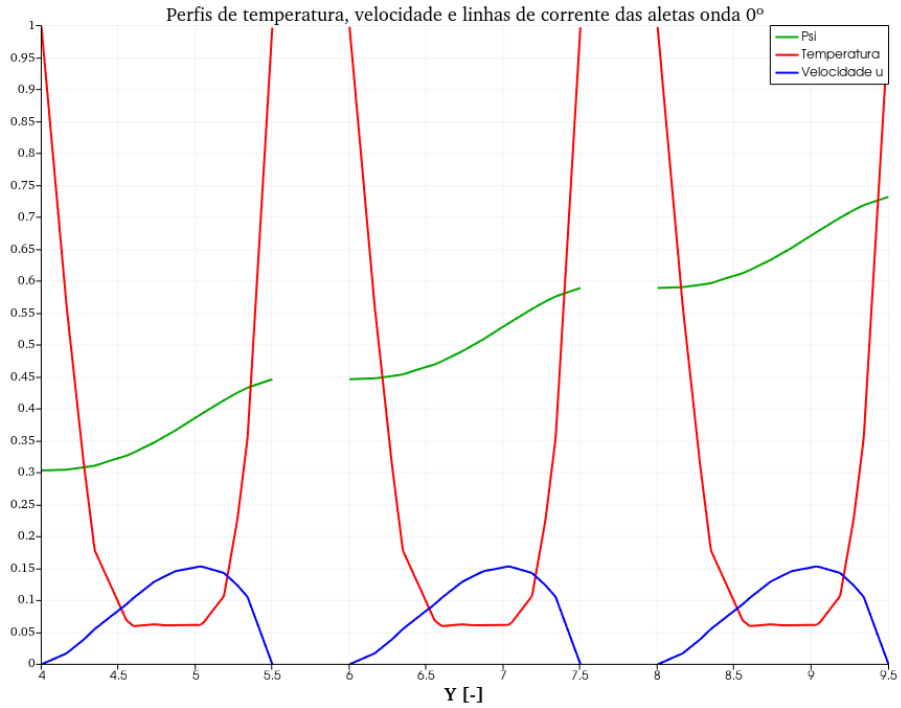


Figura 7.13: Comparação entre os perfis de velocidade, linhas de corrente e temperatura em x = 9 [-] para as aletas retas

Na figura 7.13, as aletas retas 0^o apresentam perfis assimétricos, é possível notar que a função corrente cresce mais rapidamente perto das cristas e mais lentamente perto dos vales.

Isso faz com que a velocidade reflita esse comportamento, possuindo maiores velocidades perto das cristas e menor nos vales, dando a impressão que a curva da velocidade está tombada para um lado. É interessante notar que mesmo com um perfil diferente das aletas retas, a velocidade e temperaturas atingem um pico e mínimo similar ao das aletas retas, cerca de 0.15 [-] e 0.06 [-] respectivamente.

A parábola do perfil de temperatura apresenta uma curvatura mais acentuada na direita, que na parte esquerda, assim como a velocidade. Dessa forma, explicitando que a região dos vales possui uma distribuição com temperatura maiores que na região das cristas e associando esse aumento a uma menor velocidade do fluido naquela região.

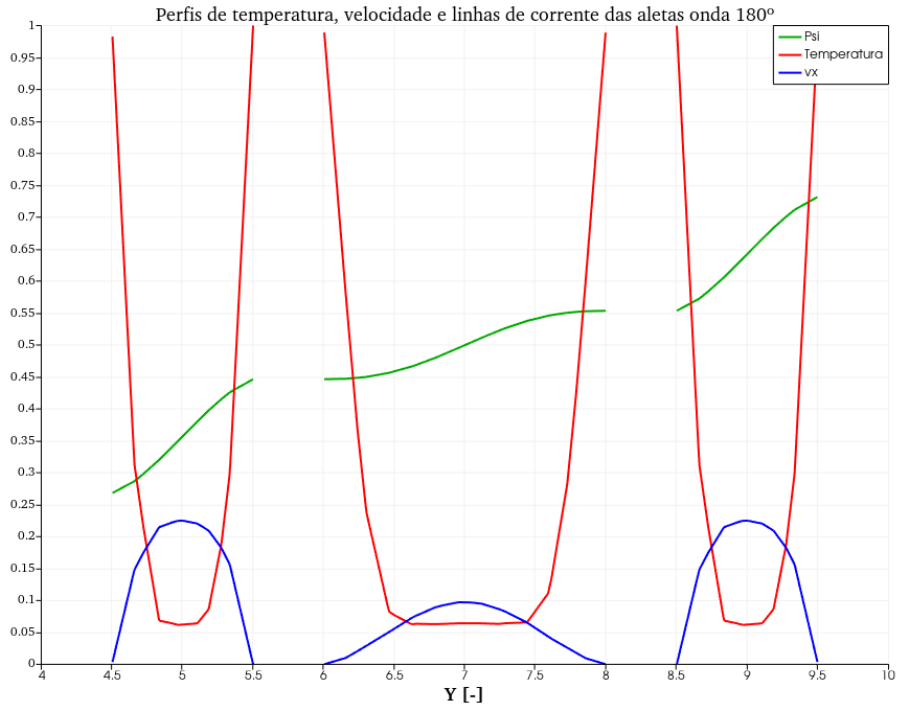


Figura 7.14: Comparação entre os perfis de velocidade, linhas de corrente e temperatura em x = 9 [-] para as aletas onda 180º

A geometria de aletas onda 180º apresentada na figura 7.14 expõe dois diferentes conjuntos de perfis, isso porque a seção escolhida corta nas cristas das ondas que formam a geometria dos canais mais externos e nos vales das ondas que formam o canal mais ao centro.

Começando pelos perfis relacionados às cristas, é possível verificar que diferente do apresentado nas geometrias anteriores, que possuíam a velocidade máxima do perfil por volta de 0.15 [-], esses perfis nos mostra uma velocidade máxima por volta de 2.25 [-] e uma linha distribuição da função corrente muito mais inclinada. Isso se deve pelo estrangulamento, que força a mesma quantidade de fluido passar por uma área menor, e para isso há um acréscimo de velocidade ao mesmo tempo que faz com que a função corrente tenha que percorrer um $\Delta\Psi$ em um espaço menor.

Já os perfis relacionados aos valores mostram uma função corrente em um "S"bem alongado e horizontal, visto que o mesmo tem mais espaço para desenvolver o mesmo $\Delta\Psi$ de cada canal. Isso faz com que a velocidade possua um perfil mais achatado, com velocidade máxima menor que as verificadas anteriormente, cerca de 0.1 [-], e um perfil de temperatura um seu vale também bem achatado na mesma temperatura mínima observada para todos os escoamentos.

Pela inclinação das curvas dos dois perfis de temperatura distintos, assim como na geometria de aletas onda 0º, é possível observar que os perfis relacionados às cristas possuem uma inclinação de parábola muito maior que o perfil associado aos vales, relacionando mais uma vez o vale com um acréscimo de temperatura do fluido.

Capítulo 8

Conclusões

O trabalho procura desenvolver os conceitos teóricos de mecânica dos fluidos e transferência de calor para aplicá-los em uma solução numérica a fim de realizar uma simulação de dinâmica dos fluidos computacional (CFD) a partir de um algorítimo próprio. O objeto de estudo utilizado como motivador foram as geometrias de aletas em placas frias, em particular as utilizadas pelo mercado de *water coolers* para *desktops* residenciais. A partir dele foram estudadas outras geometrias que proporcionassem uma melhor transferência de calor para aplicá-las ao estudo. Com as geometrias escolhidas, foi estudado como o algorítimo desempenhava em problemas clássicos e presentes na literatura, para então aplicá-lo no projeto de placa frio proposto para cada tipo de aleta escolhido para o trabalho.

O estudo começa com o estudo da mecânica dos fluidos e de transferência de calor, buscando as equações governantes que possam descrever o comportamento de um fluido de forma acurada. Para isso foi utilizada a formulação função-corrente - vorticidade, visto que ela nos permite resolver o problema de dinâmica dos fluidos em duas dimensões com o desacoplamento das variáveis de pressão e para o acoplamento da solução térmica foi utilizada a equação geral de calor para duas dimensões.

Com as equações estabelecidas, foi então desenvolvida a adimensionalidade das equações de governo, para ser possível analisar o fluido de maneira simplificada, de maneira a analisar não somente suas variáveis individuais, mas como suas características globais se relacionam com o problema proposto. Após a adimensionalização foi feita a elaboração das equações governantes em sua forma discretizada para implementação computacional, para isso foi escolhido a metodologia Galerking de métodos de elementos finitos. Dada as geometrias onduladas apresentarem muitas curvas, foi escolhido o elemento triangular, visto que esse possui mais flexibilidade para compor geometrias curvas que o elemento quadrangular. Foram verificadas as equações e matrizes locais que descrevem o elemento triangular para o método galerkin.

Com a parte teórica fundamentada, foi então escrito o código da simulação. O

algorítimo foi escrito utilizando a linguagem de programação *Python*, visto que essa possui uma gama de bibliotecas com funções necessárias aos funcionamento do algorítimo. Para deixar o código principal de solução mais claro foram criados dois módulos. O primeiro de leitura e registro dos dados em planilhas A.1, para ser possível alterar os dados de simulação de maneira ordenada e clara e também salva-los de maneira a automática e precisa. O segundo para o processo de assembling das matrizes globais A.2, já que é um processo que ocupa muitas linhas e padronizado para toda solução MEF. Assim, o código pode focar nas particularidades das simulações e pode ser mais facilmente debugado quando a simulação encontrava inconsistências ou novos parâmetros eram introduzidos. Também foi mais fácil realizar mudanças na metodologia de solução, a qual foi refinada muitas vezes durante o trabalho.

Depois do desenvolvimento do algorítimo foi necessário validá-lo, para garantir que a implementação numérica das equações de governo tenha sido feita da forma correta e para isso foi escolhido a princípio o problema de escoamento de Poiseuille entre dua placas paralelas. Para essa validação, foi utilizada a solução analítica dada por *Pontes e Mangiavacchi* [18], e partir dela foi realizada uma comparação entre as soluções analíticas da temperatura e velocidade, assim como feita uma análise dos erros entre esses, sendo obtido um resultado muito satisfatório com erros absolutos menores que 0.015 [-] para a velocidade $\mathbf{u}$ e 0.0065 [-] para a temperatura. Com isso o algorítimo foi aplicado ao mesmo problema proposto por *Florentino*[2] e os resultados observados foram muito semelhantes aos obtidos por *Florentino* em seu trabalho.

Após validar o código, foi realizado o estudo do aquecimento de um processador para saber em quais áreas da geometria haveria o aquecimento, tendo visto que o aquecimento ocorre de forma quase uniforme por toda a área da parte central da placa fria. Assim, foram estabelecidas as condições de contorno que seriam aplicadas à simulação, essas se mostraram como um grande desafio, visto que se aplicadas de maneira errada, a simulação diverge ou apresenta resultados fora da realidade.

Para as geometrias de aletas estudadas, foram escolhidas as aletas retas como referência e a partir dos estudos de *Ramgadia* [14] foram escolhidas duas geometrias de aletas onduladas promissoras para a melhora da transferência de calor. Em seus estudos, *Ramgadia* mostra que, para altos números de Reynolds, a transferência de calor melhora em até 3 vezes. Como foi observado que quanto maior o Reynolds, melhor a transferência de calor, a ideia é verificar se para o número de Reynolds de uma placa fria aplicada a desktop apresentaria uma melhora de desempenho.

Assim foram propostos três sistemas com aletas diferentes, modelados de forma simplificada e utilizando de hipóteses simplificadoras para facilitar a solução do problema. Mesmo assim foram encontrados desafios no estudo dessas geometrias, o primeiro sendo o número de Reynold do fluxo, que foi estabelecido em 30, apesar de

ser um valor baixo, visto que ao aumentá-lo, o número de Péclet também aumentaria e esse já estava em 216. o problema se dá, pois com altos números de Péclet (para a solução transferência de calor) e altos números de Reynolds (para a solução de mecânica dos fluidos) a solução galerkin do MEF se torna instável.

O segundo grande obstáculo foi relacionado ao refino da malha e número de iterações. Conforme o progresso do estudo, foi observado que o mesmo precisaria de cerca de 1500 iterações, utilizando um Δt de 0.002 segundos para obter um desenvolvimento satisfatório do escoamento. Assim, foi utilizado de três densidades de malhas, para tentar focar o poder computacional na região entre as aletas e mesmo assim o código demorou cerca de 1:30 hora para rodas cada simulação. Fazendo o custo computacional e a carga horária aumentarem muito com cada refino maior da malha.

Mesmo com os desafios encontrados, foi possível observar como o fluido se comporta ao escoar por essas três geometrias. Os resultados obtidos foram bem animadores, uma vez que mostraram um aumento na temperatura do fluido nas geometrias de aletas. Tendo as duas geometrias de aletas performado de maneira similar, ambas apresentaram uma temperatura média 6% maior que a das aletas retas, mostrando que o fluido foi capaz de puxar 6% a mais de calor do sistema em que as aletas onduladas estavam presentes. Infelizmente não foi possível verificar as zonas de recirculação nos vales das ondas, efeito dito como o responsável pelo acréscimo de temperatura. Isso pode ter como causa um baixo refino da malha, ou uma necessidade de aumento do número de Reynolds, visto que os vórtices de recirculação são crescem conforme o número de Reynolds.

8.1 Sugestões para trabalhos futuros

Esse trabalho possui código aberto e com isso a possibilidade de ser utilizado como base para novos trabalhos, assim como feito a partir do projeto do *Florentino*, ou melhoramento do código base para novos métodos, tecnologia e afins. Assim foram listadas algumas sugestões para trabalhos futuros:

- Realizar a simulação utilizando a condição de contorno de Neumann nas aletas;
- Implementar do método Taylor-Galerkin, ou outros métodos que garantam uma maior estabilidade do número de Péclet;
- Realizar a portabilidade do código para utilização de uma placa gráfica dedicada pelo algorítimo, garantindo maior poder de processamento, podendo aumentar o nível de refino de malha e o número de iterações;

- Aplicação de modelos de *Deep Learning* ao desenvolvimento de novas geometrias que propiciem a transferência de calor;
- Implementação de outros métodos numéricos, como diferenças finitas e volumes finitos, para estudo das geometrias propostas;
- Realizar a validação experimental do problema.

Referências Bibliográficas

[1] KHARE, M. "How to squeeze billions of transistors onto a computer chip". Disponível em: <https://www.ibm.com/thought-leadership/innovation-explanations/mukesh-khare-on-smaller-transistors-analytics>.

[2] FLORENTINO, Y. S. *ESTUDO DE ESCOAMENTOS PARA ARREFECIMENTO DE ELETRÔNICOS ATRAVÉS DE MÉTODO DE ELEMENTOS FINITOS*. Dissertação de bacharelado, Universidade Federal do Rio de Janeiro, Rio de Janeiro, RJ, 2022.

[3] GEUZAINE, C., REMACLE., J.-F. "Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities." *International Journal for Numerical Methods in Engineering*, v. 79(11), pp. 1309–1331, 2009.

[4] AHRENS, J., GEVECI, B., LAW, C. "ParaView: An End-User Tool for Large Data Visualization". In: *Visualization Handbook*, Elesvier, 2005. ISBN 978-0123875822.

[5] INTEL. "Thermal Design Power (TDP) in Intel® Processors". . Disponível em: <https://www.intel.com/content/www/us/en/support/articles/000055611/processors.html>.

[6] INTEL. "CPU Cooler: Liquid Cooling Vs. Air Cooling". . Disponível em: <https://www.intel.com.br/content/www/br/pt/gaming/resources/cpu-cooler-liquid-cooling-vs-air-cooling.html>.

[7] BOYD. "The Best Heat Transfer Fluids for Liquid Cooling". Disponível em: <https://www.boydcorp.com/resources/temperature-control/best-heat-transfer-fluids.html>.

[8] ÇENGEL, Y. A. *Thermodynamics : an engineering approach*. 5 ed. Boston, McGraw-Hill Higher Education, 2008.

[9] BERGMAN, T. L., INCROPERA, F. P. *Fundamentals of Heat and Mass Transfer*. 7th ed. Hoboken, NJ, Wiley, 2011.

[10] PRITCHARD, P. J., . M. J. W. *Fox and McDonald's Introduction to Fluid Mechanics*. 9th ed. New York, NY, John Wiley Sons., 2015.

[11] JACOB FISH, T. B. *A First Course in Finite Elements*. Hoboken, NJ, USA, John Wiley Sons, Inc., 2007.

[12] ANJOS, G. R. *Computação Científica para Engenheiros*. UFRJ, 2022.

[13] CUNHA, L. H. C. *Formulação Corrente-Vorticidade em Problemas de Transferência de Calor Conjugado usando MEF*. Dissertação de bacharelado, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, RJ, 2019.

[14] RAMGADIA, A. G., SAHA, A. K. "Numerical study of fully developed unsteady flow and heat transfer in asymmetric wavy channels", *International Journal of Heat and Mass Transfer*, v. 102, pp. 98–112, 2016. ISSN: 0017-9310. doi: https://doi.org/10.1016/j.ijheatmasstransfer.2016.05.131. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0017931015301277>.

[15] GOLDSTEIN, LEONARDO, J., SPARROW, E. M. "Heat/Mass Transfer Characteristics for Flow in a Corrugated Wall Channel", *Journal of Heat Transfer*, v. 99, n. 2, pp. 187–195, 05 1977. ISSN: 0022-1481. doi: 10.1115/1.3450667. Disponível em: <https://doi.org/10.1115/1.3450667>.

[16] NISHIMURA, T., OHORI, Y., KAWAMURA, Y. "FLOW CHARACTERISTICS IN A CHANNEL WITH SYMMETRIC WAVY WALL FOR STEADY FLOW", *Journal of Chemical Engineering of Japan*, v. 17, n. 5, pp. 466–471, 1984. doi: 10.1252/jcej.17.466.

[17] WANG, G., VANKA, S. "Convective heat transfer in periodic wavy passages", *International Journal of Heat and Mass Transfer*, v. 38, n. 17, pp. 3219–3230, 1995. ISSN: 0017-9310. doi: https://doi.org/10.1016/0017-9310(95)00051-A. Disponível em: <https://www.sciencedirect.com/science/article/pii/001793109500051A>.

[18] PONTES, J.; MANGIAVACCHI, N. *Fenômenos de Transferência*. 1. ed. ed. [S.l.: s.n.], 2010.

[19] WIKICHIP. "Socket AM5 - Packages - AMD". Disponível em: <https://en.wikichip.org/wiki/amd/packages/socket_am5>.

[20] WALLOSEK, I. “AMD Ryzen 7000 heatspreader and cooling analysis – temperatures, hotspots and problems”. Disponível em: <https://www.igorslab.de/en/amd-ryzen-7000-heatspreader-and-cooling-analysis-temperatures-hotspots-and-3/>.

Apêndice A

Algoritimos

A.1 Ler/Gravar Planilha

```
import pandas as pd

def entrada(nome, geo):
    projeto = pd.ExcelFile(nome)
    input = pd.read_excel(projeto, 'inputs')


    row = input.loc[input['nome geometria'] == geo]

    inputs = row.values.tolist()


    return(inputs[0])

def gravar(nome, geo, npoints, nelem, tempo, Tmax, Tmed, Umax, Umed, Re, Pe, date):

    respostas = [geo, npoints, nelem, tempo, Tmax, Tmed, Umax, Umed, Re, Pe, date]

    resultados = pd.read_excel('resultados.xlsx', sheet_name="resultados")

    resultados.loc[len(resultados)] = respostas

    resultados.to_excel('resultados.xlsx', sheet_name="resultados", index=False)
```

A.2 Módulo de Assembly

```
import numpy as np

def assembly(X, Y, IEN, ne, K, M, Gx, Gy):

    for e in range(0,ne):
        #Vetores i,j,k
        vi, vj, vk = IEN[e]

        #area do triangulo
        area = 0.5*np.linalg.det([[1.0, X[vi], Y[vi]],
```

```
                                [1.0, X[vj], Y[vj]],
                                [1.0, X[vk], Y[vk]]])

        #Coenficiente b e c
        bi = Y[vj] - Y[vk]
        bj = Y[vk] - Y[vi]
        bk = Y[vi] - Y[vj]

        ci = X[vk] - X[vj]
        cj = X[vi] - X[vk]
        ck = X[vj] - X[vi]


        kxelem = (1.0/(4.0*area))*np.array([[ bi*bi, bi*bj, bi*bk],
                                            [ bj*bi, bj*bj, bj*bk],
                                            [ bk*bi, bk*bj, bk*bk]]) #Matriz de rigidez x

        kyelem = (1.0/(4.0*area))*np.array([[ ci*ci, ci*cj, ci*ck],
                                            [ cj*ci, cj*cj, cj*ck],
                                            [ ck*ci, ck*cj, ck*ck]]) #Matriz de rigidez y

        kelem = kxelem+kyelem #Matriz do laplaciano de cada elemento

        melem = (area/12)*np.array([[ 2.0, 1.0, 1.0],
                                    [ 1.0, 2.0, 1.0],
                                    [ 1.0, 1.0, 2.0]])

        gxelem = (1.0/6.0)*np.array([[bi, bj, bk],
                                     [bi, bj, bk],
                                     [bi, bj, bk]])

        gyelem = (1.0/6.0)*np.array([[ci, cj, ck],
                                     [ci, cj, ck],
                                     [ci, cj, ck]])

        for ilocal in range(0,3):
            iglobal = IEN[e, ilocal]
            for jlocal in range(0,3):
                jglobal = IEN[e, jlocal]

                K[iglobal, jglobal] += kelem[ilocal, jlocal]
                M[iglobal, jglobal] += melem[ilocal, jlocal]
                Gx[iglobal, jglobal] += gxelem[ilocal, jlocal]
                Gy[iglobal, jglobal] += gyelem[ilocal, jlocal]

    return(K, M, Gx, Gy)
```

A.3 Simulação de aletas

```
import planilha2 as pl
import assembly as ass
import meshio
import numpy as np
from tqdm import tqdm
from timeit import default_timer as timer
```

```
from datetime import date

start = timer()

# Indicar qual geometria simular e qual planilha mae com os parametros desejados
geo = "aletas retas"
planilha = 'projeto.xlsx'

print("Iniciando simulacao de " + geo)

# Carregando parametros da simulacao da planilha mae
inputs = pl.entrada(planilha, geo)
nome_geo, Inter, dt, L, T0, U0, D0, CPU, TDP, nome_fluido, rho0, mu0, k0, cp0 = inputs

# Constantes calculadas
ni0 = mu0/rho0 #[m2/s]
alpha0 = k0/(rho0*cp0) #[m2/s]


# Numeros adimensionais
Re = (U0*D0)/ni0 #[-]
Pr = ni0/alpha0 #[-]
Pe = Re*Pr #[-]

# variável auxiliar
nq = 0
nt = 0
beta = 0.5 #crank-nicholson
auxT = []

msh = meshio.read('./geometrias/' + geo + '/' + geo + '.msh')
X = msh.points[:,0]
Y = msh.points[:,1]
IEN = msh.cells[1].data #1 para triangulo
IENbound = msh.cells[0].data #0 para line
IENbound2D = msh.cells[1].data #0 para superficie
IENboundTypeElem = list(msh.cell_data["gmsh:physical"][0] - 1)
IENboundTypeElem2D = list(msh.cell_data["gmsh:physical"][1] - 1)
boundNames = list(msh.field_data.keys())
boundNames2D = list(msh.field_data.keys())
IENboundElem = [boundNames[elem] for elem in IENboundTypeElem]
IENboundElem2D = [boundNames2D[elem] for elem in IENboundTypeElem2D]
npoints = len(X)
ne = IEN.shape[0]

# Lista com nos com as condicoes de contorno 1D e 2D
cc = np.unique(IENbound.reshape(IENbound.size))
cc2D = np.unique(IENbound2D.reshape(IENbound2D.size))
nodes = np.unique(IEN.reshape(IEN.size))

ccName = [[] for i in range( len(X) )]
ccName2D = [[] for i in range( len(X) )]

# Prioridade 10
for elem in range(0,len(IENbound)):
    if IENboundElem[elem] == 'aleta6':
        ccName[ IENbound[elem][0] ] = IENboundElem[elem]
        ccName[ IENbound[elem][1] ] = IENboundElem[elem]
```

```
# Prioridade 9
for elem in range(0,len(IENbound)):
    if IENboundElem[elem] == 'aleta5':
        ccName[ IENbound[elem][0] ] = IENboundElem[elem]
        ccName[ IENbound[elem][1] ] = IENboundElem[elem]

# Prioridade 8
for elem in range(0,len(IENbound)):
    if IENboundElem[elem] == 'aleta4':
        ccName[ IENbound[elem][0] ] = IENboundElem[elem]
        ccName[ IENbound[elem][1] ] = IENboundElem[elem]

# Prioridade 7
for elem in range(0,len(IENbound)):
    if IENboundElem[elem] == 'aleta3':
        ccName[ IENbound[elem][0] ] = IENboundElem[elem]
        ccName[ IENbound[elem][1] ] = IENboundElem[elem]

# Prioridade 6
for elem in range(0,len(IENbound)):
    if IENboundElem[elem] == 'aleta2':
        ccName[ IENbound[elem][0] ] = IENboundElem[elem]
        ccName[ IENbound[elem][1] ] = IENboundElem[elem]

# Prioridade 5
for elem in range(0,len(IENbound)):
    if IENboundElem[elem] == 'aleta1':
        ccName[ IENbound[elem][0] ] = IENboundElem[elem]
        ccName[ IENbound[elem][1] ] = IENboundElem[elem]

# Prioridade 4
for elem in range(0,len(IENbound)):
    if IENboundElem[elem] == 'saida':
        ccName[ IENbound[elem][0] ] = IENboundElem[elem]
        ccName[ IENbound[elem][1] ] = IENboundElem[elem]

# Prioridade 3
for elem in range(0,len(IENbound)):
    if IENboundElem[elem] == 'entrada':
        ccName[ IENbound[elem][0] ] = IENboundElem[elem]
        ccName[ IENbound[elem][1] ] = IENboundElem[elem]

# Prioridade 2
for elem in range(0,len(IENbound)):
    if IENboundElem[elem] == 'superior':
        ccName[ IENbound[elem][0] ] = IENboundElem[elem]
        ccName[ IENbound[elem][1] ] = IENboundElem[elem]

# Prioridade 1
for elem in range(0,len(IENbound)):
    if IENboundElem[elem] == 'inferior':
        ccName[ IENbound[elem][0] ] = IENboundElem[elem]
        ccName[ IENbound[elem][1] ] = IENboundElem[elem]

#Criando CCName 2D
# Prioridade 2
for elem in range(0,len(IENbound2D)):
    if IENboundElem2D[elem] == 'canal':
        ccName2D[ IENbound2D[elem][0] ] = IENboundElem2D[elem]
```

```
        ccName2D[ IENbound2D[elem][1] ] = IENboundElem2D[elem]
        ccName2D[ IENbound2D[elem][2] ] = IENboundElem2D[elem]
# Prioridade 1
for elem in range(0,len(IENbound2D)):
    if IENboundElem2D[elem] == 'superficie':
        ccName2D[ IENbound2D[elem][0] ] = IENboundElem2D[elem]
        ccName2D[ IENbound2D[elem][1] ] = IENboundElem2D[elem]
        ccName2D[ IENbound2D[elem][2] ] = IENboundElem2D[elem]

# Inicialização dos vetores
vx_cc = np.zeros((npoints), dtype = 'float')
vy_cc = np.zeros((npoints), dtype = 'float')
Psi_cc = np.zeros((npoints), dtype = 'float')
T_cc = np.zeros((npoints), dtype = 'float')
Q_cc = np.zeros((npoints), dtype = 'float')

# Definição das condições de contorno: $\nu_x$, $\nu_y$, $\omega_z$, $\Psi$ e $T$
for i in cc:
    if ccName[i] == 'inferior':
        vx_cc[i]  = 0.0 # [-]
        vy_cc[i]  = 0.0 # [-]
        Psi_cc[i] = 0.0 # [-]


    if ccName[i] == 'superior':
        vx_cc[i]  = 0.0 # [-]
        vy_cc[i]  = 0.0 # [-]
        Psi_cc[i] = 1.0 # [-]


    if ccName[i] == 'entrada':
        vx_cc[i]  = 1.0 # [-]
        vy_cc[i]  = 0.0 # [-]
        Psi_cc[i] = Y[i]- 6.5 # [-]
        T_cc[i]   = 0.0 # [-]

    if ccName[i] == 'aleta1':
        vx_cc[i]  = 0.0 # [-]
        vy_cc[i]  = 0.0 # [-]
        Psi_cc[i] = 2/14.0 # [-]
        T_cc[i]   = 1.0 # [-]

    if ccName[i] == 'aleta2':
        vx_cc[i]  = 0.0 # [-]
        vy_cc[i]  = 0.0 # [-]
        Psi_cc[i] = 4/14.0 # [-]
        T_cc[i]   = 1.0 # [-]

    if ccName[i] == 'aleta3':
        vx_cc[i]  = 0.0 # [-]
        vy_cc[i]  = 0.0 # [-]
        Psi_cc[i] = 6/14.0 # [-]
        T_cc[i]   = 1.0 # [-]

    if ccName[i] == 'aleta4':
        vx_cc[i]  = 0.0 # [-]
        vy_cc[i]  = 0.0 # [-]
        Psi_cc[i] = 8/14.0 # [-]
        T_cc[i]   = 1.0 # [-]
```

```
    if ccName[i] == 'aleta5':
        vx_cc[i]  = 0.0 # [-]
        vy_cc[i]  = 0.0 # [-]
        Psi_cc[i] = 10/14.0 # [-]
        T_cc[i]   = 1.0 # [-]

    if ccName[i] == 'aleta6':
        vx_cc[i]  = 0.0 # [-]
        vy_cc[i]  = 0.0 # [-]
        Psi_cc[i] = 12/14.0 # [-]
        T_cc[i]   = 1.0 # [-]

# Contando quantos pontos existem na superfície
for i in cc2D:
    if ccName2D[i] == 'superficie':
        nq += 1

# Distribuindo igualmente a portência dada pela TDP por cada ponto
for i in cc2D:
    if ccName2D[i] == 'superficie':
        Q_cc[i] = 1 # [W]

# Definindo St conforme a distribuição de pontos
St = TDP/nq

# Montagem das matrizes globais $K_x$, $K_y$, $K_{xy}$, $M$, $G_y$ e $G_x$
K    = np.zeros( (npoints, npoints), dtype = 'float')
M    = np.zeros( (npoints, npoints), dtype = 'float')
Gx   = np.zeros( (npoints, npoints), dtype = 'float')
Gy   = np.zeros( (npoints, npoints), dtype = 'float')

# Montagem dos vetores variaveis
vx  = np.zeros((npoints), dtype = 'float')
vy  = np.zeros((npoints), dtype = 'float')
Psi = np.zeros((npoints), dtype = 'float')
T   = np.zeros((npoints), dtype = 'float')

for i in cc:
    vx[i] = vx_cc[i]
    vy[i] = vy_cc[i]

# Montando as matrizes globais a partir do modulo assembly
K, M, Gx, Gy = ass.assembly(X, Y, IEN, ne, K, M, Gx, Gy)

omega_z_cc = np.linalg.solve(M, (Gx@vy_cc - Gy@vx_cc))

point_data = {'psi_cc' : Psi_cc}
data_vx_cc = {'vx_cc' : vx_cc}
data_vy_cc = {'vy_cc' : vy_cc}
data_omega_z_cc = {'omega_z_cc' : omega_z_cc}
data_T_cc = {'T_cc' : T_cc}
data_Q_cc = {'Q_cc' : Q_cc}
point_data.update(data_vx_cc)
point_data.update(data_vy_cc)
point_data.update(data_omega_z_cc)
point_data.update(data_Q_cc)
point_data.update(data_T_cc)
meshio.write_points_cells('./geometrias/' + geo + '/solucoes/CondicaoDeContorno.vtk',
```

```
                            msh.points,
                            msh.cells,
                            point_data=point_data,
                            )

# Cálculo da condicao inicial de omega_z, Psi e T
omega_z = omega_z_cc.copy()

for i in cc:
    T[i] = T_cc[i]
    Psi[i] = Psi_cc[i]

point_data = {'psi' : Psi}
data_vx = {'vx' : vx}
data_vy = {'vy' : vy}
data_omega_z = {'omega_z' : omega_z}
data_T = {'T' : T}
point_data.update(data_vx)
point_data.update(data_vy)
point_data.update(data_omega_z)
point_data.update(data_T)
meshio.write_points_cells('./geometrias/' + geo + '/solucoes/CondicaoInicial.vtk',
                            msh.points,
                            msh.cells,
                            point_data=point_data,
                            )

#Iniciando as matrizes A antes do loop para poupar processamento
A = (1/dt)*M + (1/Re)*K #Vorticidade
A_T = (1/dt)*M + (1/Pe)*K #Temperatura
A_psi = K.copy() #Psi

#Aplciando as CCs em omega_z e Psi
for i in cc:
    if ccName[i] == 'inferior' or \
        ccName[i] == 'superior' or \
        ccName[i] == 'aleta1' or \
        ccName[i] == 'aleta2' or \
        ccName[i] == 'aleta3' or \
        ccName[i] == 'aleta4' or \
        ccName[i] == 'aleta5' or \
        ccName[i] == 'aleta6' or \
        ccName[i] == 'entrada':
        A[i,:] = 0.0 #Zerando a linha A
        A[i,i] = 1.0 #Colocando 1 na diagonal, para garantir que na linha da C.C. só ela seja levada em conta A
        A_psi[i,:] = 0.0 #Zerando a linha A_psi
        A_psi[i,i] = 1.0 #Colocando 1 na diagonal, para garantir que na linha da C.C. só ela seja levada em conta A_psi

# Aplicando as CCs em T
for i in cc:
    if ccName[i] == 'aleta1' or \
        ccName[i] == 'aleta2' or \
        ccName[i] == 'aleta3' or \
        ccName[i] == 'aleta4' or \
        ccName[i] == 'aleta5' or \
        ccName[i] == 'aleta6' or \
        ccName[i] == 'entrada':
        A_T[i,:] = 0.0 #Zerando a linha A_T
        A_T[i,i] = 1.0 #Colocando 1 na diagonal, para garantir que na linha da C.C. só ela seja levada em conta A_T
```

```
        auxT.append(i)

# Loop dos passos de tempo
for n in tqdm(range (0, Inter)):
    #Cálculo da condição de contorno
    omega_z_cc  = np.linalg.solve(M, (np.dot(Gx,vy) - np.dot(Gy,vx)))
    #Montagem da matriz A
    vx_diag = np.diag(vx)
    vy_diag = np.diag(vy)

    #Montagem do vetor b de vorticidade e temperatura
    vg = np.dot(vx_diag,Gx) + np.dot(vy_diag,Gy)
    b = (1/dt)*np.dot(M,omega_z) - np.dot(vg,omega_z)
    b_T = (1/dt)*np.dot(M,T) + St*np.dot(M,Q_cc) - np.dot(vg,T) #- np.dot(Kest,T)

    #Condição de contorno para o sistema Ax = b de omega_z
    for i in cc:
        if ccName[i] == 'inferior' or \
           ccName[i] == 'superior' or \
           ccName[i] == 'aleta1' or \
           ccName[i] == 'aleta2' or \
           ccName[i] == 'aleta3' or \
           ccName[i] == 'aleta4' or \
           ccName[i] == 'aleta5' or \
           ccName[i] == 'aleta6' or \
           ccName[i] == 'entrada':
            b[i]   = omega_z_cc[i]

    # Solução do sistema linear para Omega_z (Solução da equação de transporte de vorticidade)
    omega_z = np.linalg.solve(A,b)

    # Solução da equação função corrente
    b_psi = np.dot(M,omega_z)

    for i in cc:
        if ccName[i] == 'inferior' or \
           ccName[i] == 'superior' or \
           ccName[i] == 'aleta1' or \
           ccName[i] == 'aleta2' or \
           ccName[i] == 'aleta3' or \
           ccName[i] == 'aleta4' or \
           ccName[i] == 'aleta5' or \
           ccName[i] == 'aleta6' or \
           ccName[i] == 'entrada':
            b_psi[i]   = Psi_cc[i]

    #Solução do sistema linear para Psi
    Psi = np.linalg.solve(A_psi,b_psi)

    #Condição de contorno para o sistema Ax = b de T
    for i in cc:
        if ccName[i] == 'entrada'or \
           ccName[i] == 'aleta1' or \
           ccName[i] == 'aleta2' or \
           ccName[i] == 'aleta3' or \
           ccName[i] == 'aleta4' or \
           ccName[i] == 'aleta5' or \
           ccName[i] == 'aleta6':
            b_T[i]   = T_cc[i]
```

```
    #Solução do sistema linear para T
    T = np.linalg.solve(A_T,b_T)

    #Cálculo do campo de velocidades:
    vx =  np.linalg.solve(M,np.dot(Gy,Psi))
    vy = -np.linalg.solve(M,np.dot(Gx,Psi))

    for i in cc:
        if ccName[i] == 'inferior' or \
           ccName[i] == 'superior' or \
           ccName[i] == 'aleta1' or \
           ccName[i] == 'aleta2' or \
           ccName[i] == 'aleta3' or \
           ccName[i] == 'aleta4' or \
           ccName[i] == 'aleta5' or \
           ccName[i] == 'aleta6' or \
           ccName[i] == 'entrada':
            vx[i] = vx_cc[i]
            vy[i] = vy_cc[i]

    data_psi = {'psi' : Psi}
    data_vx = {'vx' : vx}
    data_vy = {'vy' : vy}
    data_omega_z = {'omega_z' : omega_z}
    data_T = {'T' : T}
    data_Q_cc = {'Q_cc' : Q_cc}
    point_data.update(data_psi)
    point_data.update(data_vx)
    point_data.update(data_vy)
    point_data.update(data_omega_z)
    point_data.update(data_T)
    point_data.update(data_Q_cc)
    meshio.write_points_cells('./geometrias/' + geo + '/solucoes/solucao-'+str(n+1)+'.vtk',
                              msh.points,
                              msh.cells,
                              point_data=point_data,
                              )

Tcalc = np.delete(T, auxT)

Tmax = max(Tcalc)
Tmed = sum(Tcalc) / len(Tcalc)
Umax = max(vx)
Umed = sum(vx) / len(vx)

end = timer()

tempo = round(end - start, 2)

date = date.today()

# salvando os resultados na planilha
pl.gravar(planilha, geo, npoints, ne, tempo, Tmax, Tmed, Umax, Umed, Re, Pe, date)
```