Revista Brasileira de Ensino de Física, vol. 41, nº 4, e20180304 (2019)
www.scielo.br/rbef
DOI: http://dx.doi.org/10.1590/1806-9126-RBEF-2018-0304

Artigos Gerais

Uma interface de controle para a Fluidodinâmica Computacional

A control interface for Computational Fluid Dynamics

Thiago F. D. Dias Fernandes[*1], Nilton L. Moreira[2]

[1]Faculdade de Caldas Novas, Caldas Novas, GO, Brasil
[2]Universidade Federal de Catalão, Unidade Acadêmica de Física, Catalão, GO, Brasil

Recebido em 27 de Outubro, 2018. Revisado em 11 de Abril, 2019. Aceito em 13 de Maio, 2019.

Neste artigo apresentaremos uma proposta para a utilização da dinâmica de fluidos computacional sob uma perspectiva educacional, tendo como alvo as disciplinas de mecânica dos fluidos e fenômenos de transporte, nos períodos iniciais de cursos de Engenharia e Física. A importância da utilização da simulação na formação de base é amplamente corroborada pela literatura, contudo, pretende-se que o uso da CFD não seja somente demonstrativo, mas que coloque o usuário final como agente diretamente responsável por manipular e codificar dados e resultados das simulações propostas. Propõe-se então uma interface de manipulação e controle para o software *opensource OpenFOAM*, cujo objetivo principal é evitar a complexa edição e manipulação do código-fonte e *scripts* de controle típicos desse programa.
Palavras-chave: Fluidodinâmica Computacional, Mecânica dos Fluidos, Modelagem Computacional, Ensino de Física.

In this article we will present a proposal for the use of computational fluid dynamics under an educational perspective, targeting disciplines of fluid mechanics and transport phenomena, on initial periods of engineering and physics courses. The importance of the use of simulation in undergraduated education is widely supported by the literature, however, is proposing to work it so not only statement, but the final users as officer directly responsible for handling and encode data and results of simulations. Second, a manipulation and control interface for the opensource software *OpenFOAM*, in order to take out the users the responsibility of editing the source code and scripts of typical control of this program.
Keywords: Computational Fluid Dynamics, Fluid Mechanics, Computational Modeling, Physics Teaching.

1. Introdução

A utilização da simulação como forma de ensino, além de seu apelo visual, possui precisão suficiente para que sejam coletados dados concretos acerca das condições envolvidas no problema, dentro de certo grau de precisão e validade. Tal perspectiva acaba por motivar o aluno e o docente em focar o estudo no problema em vez de somente concentrar esforços na manipulação de equações do sistema. Heidemann [1] demonstra isso quando compara o enfoque do ensino, entre o tradicional e o uso de modelagem dos fenômenos físicos, concluindo que a modelagem se destaca frente a resolução de problemas teóricos por não utilizar "truques", de modo que o aprendizado ocorre pela experiência e não pela resolução de um grande número de problemas. Logo, a utilização didática de técnicas de simulação acaba por demonstrar que atividades que se diferenciem da abordagem tradicional, acabam por se tornar não só efetivas, mas também motivadoras [2].

Este trabalho propõe inicialmente contribuir para a discussão sobre esse tema e a implementação do uso da simulação computacional em fluidodinâmica como ferramenta de ensino. Na intenção desta, nota-se que a utilização da CFD (*Computational Fluid Dynamics*) exige, por vezes, experiência em programação básica e ainda relativa habituação a sistemas operacionais diferentes. Quando não se fazem presentes tais pré-requisitos, torna improdutivo a utilização de ferramentas como o CFD em cursos de um semestre, devido a necessidade de modificação de *scripts* e códigos fontes. Levando em conta estas possíveis dificuldades associadas à utilização destas técnicas, tanto no ambiente profissional quanto acadêmico, propõe-se também a construção de uma interface de simulação para tornar a utilização do pacote *OpenFOAM* [3] mais amigável ao usuário final.

A interface J.A.R.V.E.S. (acrônimo para *Just a Rather Very Educative Simulator* [1] - Figura 1), objeto da proposta, é um conjunto de *scripts* de controle e edição que, de forma direcionada, manipula *scripts* do software *OpenFOAM*. Na construção dessa ferramenta buscou-se

[*]Endereço de correspondência: profthiagomecflu@gmail.com.

[1]Apenas um Simulador Muito Educativo

Figura 1: Logomarca da Interface J.A.R.V.E.S.

uma melhor adaptabilidade às distribuições baseadas em GNU/LINUX, ofertando uma interface na qual seja possível sua utilização tanto no ambiente gráfico, quanto em modo texto, seja em *desktops* ou em terminais de comando de *clusters* de processamento paralelo.

Inicialmente na seção 2 apresentamos um aporte teórico das principais equações que regem o escoamento, tendo como foco a obtenção da Equação de Navier-Stokes. Na seção 3 discorremos sobre os métodos de modelagem da dinâmica de fluidos, com enfoque no Método dos Volumes Finitos e, ao longo da seção 4, discutimos sobre o *software OpenFOAM* e a caracterização completa das funcionalidades da interface de simulação J.A.R.V.E.S.

2. Segunda Lei de Newton aplicada ao escoamento

Define-se por escoamento toda movimentação de uma porção fluida em relação a um sistema referencial inercial adotado, onde sua dinâmica é regida por axiomas não-relativísticos, com quantidade de partículas suficientemente grande para que flutuações provocadas por descontinuidades sejam desconsideradas [4]. A Segunda Lei de Newton aplicada a um volume infinitesimal é descrita por

$$\frac{d}{dt}\int_{\mathcal{V}} \rho u_i dV = \int_{\mathcal{V}} \rho f_i dV + \int_{\mathcal{S}} t_i dS, \tag{1}$$

onde V e S são o volume e a superfice respectivamente do volume de controle considerado, ρ é a densidade do meio material em questão, u_i o campo de velocidades e f_i um campo externo qualquer, aqui considerado como sendo o campo gravitacional. O vetor tração t_i pode ser descrito em termos do tensor tensão σ_{ij} [5], na forma de

$$t_i = \hat{n}_j \sigma_{ji}, \tag{2}$$

e a transposta do tensor tensão na equação (2) indica que a atuação da força de tração t_j sobre uma superfície de normal $\hat{n}_i$ cria o estado de tensão traduzido pela matriz tensor tensão σ_{ij}, onde

$$\sigma_{ij} = \begin{vmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_{22} & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_{33} \end{vmatrix}.$$

Em simulações computacionais envolvendo difusões moleculares, um dos grandes problemas é a reprodução das forças intra e intermoleculares existentes entre os elementos do modelo. Na modelagem de fluidos, tais como a água, existem potenciais associados às forças intermoleculares, que modelam as forças de atração e repulsão existente nessas interações [6]. Seguindo esse raciocínio, o estado de stress interno existente é traduzido pela contribuição das forças de repulsão intermolecular, devido às forças de superfície que tendem a diminuir a distância existente entre estes elementos. Neste momento diferencia-se portanto, a pressão termodinâmica daquela advinda da própria interação intermolecular.

Assumindo que as tensões normais do fluido tem relação com a pressão hidrostática média no centro do volume, decompõe-se o tensor σ_{ij} na forma de

$$\underbrace{\sigma_{ij}}_{\text{Tensor Stress Total}} = \underbrace{-p\delta_{ij}}_{\text{Tensor Pressão Mecânica}} + \underbrace{\tau_{ij}}_{\text{Tensor Deviatórico}}. \tag{3}$$

onde o negativo do termo $p\delta_{ij}$ representa que na atuação da tração, a pressão é negativa em relação ao ponto analisado [7]. Considera-se também que

$$p\delta_{ij} = -\frac{1}{3}\sigma_{ii} = -\frac{1}{3}\left(\sigma_{11} + \sigma_{22} + \sigma_{33}\right). \tag{4}$$

Dentre as ferramentas utilizadas na descrição de sistemas contínuos, destaca-se o conceito de Derivada Material. O intuito principal da derivada material é estabelecer uma função de relação entre as abordagens lagrangiana e euleriana de uma grandeza qualquer analisada no escoamento de um fluido. Na notação aqui utilizada, é descrita por

$$\frac{Df}{Dt} = \partial_o f + u_j \partial_j f, \tag{5}$$

onde o primeiro termo do lado direito da equação (5) é a derivada local de uma grandeza f qualquer, e tal como no referencial euleriano, determina a variação desta ao longo do tempo na posição determinada. O segundo termo do lado direito é a derivada de transporte ou advectiva, indicando a variação instantânea da grandeza decorrente de uma variação de posição fruto do campo de velocidades u_i [8].

Na aplicação da derivada material no termo esquerdo da equação (1), juntamente com a substituição do Tensor

 DOI: http://dx.doi.org/10.1590/1806-9126-RBEF-2018-0304

de Cauchy, descrito em (2), chega-se na forma do Teorema de Transporte de Reynolds [9] aplicada à Segunda Lei de Newton para um volume de controle. Tais modificações resultam na forma primária do princípio do momento linear

$$\int_{\mathcal{M}} \left(\partial_o \rho u_i + \partial_j \left(\rho u_j u_i\right)\right) dV = \int_{\mathcal{M}} \rho f_i dV + \int_{\mathcal{S}} \hat{n}_j \sigma_{ji} dS. \tag{6}$$

Transformando o segundo termo do lado direto da equação (6) usando o Teorema de Gauss, reescreve-se na forma de

$$\int_{\mathcal{M}} \left(\partial_o \rho u_i + \partial_j \left(\rho u_j u_i\right) - \rho f_i - \partial_j \sigma_{ji}\right) dV = 0. \tag{7}$$

Dado que o diferencial dV é arbitrário, toma-se portanto que o integrando é nulo, escrevemos

$$\partial_o \rho u_i + \partial_j \left(\rho u_j u_i\right) = \rho f_i + \partial_j \sigma_{ji}. \tag{8}$$

A equação (8) é conhecida como Equação de Cauchy [10], válida para qualquer meio contínuo, onde o termo σ_{ij} carrega a informação da microestrutura, modelando suas consequência na macroestrutura. Como no caso analisado, $\sigma_{ji} = -p\delta_{ij} + \tau_{ji}$ temos

$$\partial_o \rho u_i + \partial_j \left(\rho u_j u_i\right) = \rho f_i + \partial_j \left(-p\delta_{ij} + \tau_{ji}\right) \tag{9}$$

$$\partial_o \rho u_i + \partial_j \left(\rho u_j u_i\right) = \rho f_i - \partial_j p + \partial_j \tau_{ji} \tag{10}$$

Apesar da equação (10) descrever matematicamente o formalismo do momento linear, sua relação com a microestrutura do fluido ainda não se faz presente. Entende-se por hora que o tensor τ_{ij} carrega a informação das tensões cisalhantes e, consequentemente, relação intrínseca com a viscosidade do fluido.

2.1. Equações de Navier-Stokes para fluidos isotrópicos

Na mecânica de meios contínuos, as equações constitutivas contribuem, assim como dito, na transposição do axioma matemático, em uma equação que consiga realmente definir a especifidade macroscópica do sistema material. Das equações principais que regem o escoamento, percebe-se uma forte relação entre a microestrutura e as forças externas. Tal situação emprega de modo prático uma equação constitutiva que se relaciona de modo direto com o tensor deviatórico e a interação viscosa do sistema material. Na tentativa de definir esse sistema, determinadas condições são impostas ao fluido, tais como:

a) o fluido é **homogêneo**, ou seja, não é função explícita da posição x_i;
b) o fluido é **isotrópico**, ou seja, não adota direção preferencial;
c) na ausência de tensões de deformação, o único stress considerado será o hidrostático e
d) o tensor deviatórico τ_{ij} é função linear e explícita do gradiente de velocidades.

Tais condições, modelam o que se conhece como fluido stokesiano [11]. Considerando, portanto, a quarta condição como início e que somente a parte simétrica do tensor gradiente de velocidades é responsável pela efetiva deformação do sistema material analisado, pode-se escrever

$$\tau_{ij} = A_{ijkl} \partial_{(k} u_{l)}. \tag{11}$$

A condição da isotropia do fluido permite resumir o tensor de proporcionalidade A_{ijkl}, escrito na forma do produto entre tensores de Levi-Civita ϵ_{ijk}

$$A_{ijkl} = \epsilon_{hij}\epsilon_{hkl} = C_1 \delta_{ij}\delta_{kl} + C_2 \delta_{ik}\delta_{jl} + C_3 \delta_{il}\delta_{jk}, \tag{12}$$

e consequentemente, na admissão da homogeneidade as, constantes C_2 e C_3 são iguais, resultando na forma final do tensor de proporcionalidade

$$A_{ijkl} = C_1 \delta_{ij}\delta_{kl} + C_2 \left(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk}\right). \tag{13}$$

Ao analisarmos o experimento proposto por Newton, contido na seção IX do livro 2 da obra *Philosophiae Naturalis Principia Mathematica* [12], onde discorre sobre a viscosidade de um fluido encerrado entre dois cilindros concêntricos, percebemos que os fatores C_1 e C_2 são escritos em termos da viscosidade absoluta μ. Ao final de todo trabalho de reconstrução do tensor de proporcionalidade, obtem-se que a forma final do tensor deviatórico é definida com

$$\tau_{ij} = 2\mu \partial_{(i} u_{i)} - \frac{2}{3}\mu \partial_{(k} u_{k)} \delta_{ij}, \tag{14}$$

onde observa-se que o segundo termo corresponde a chamada "segunda viscosidade", acontecendo exclusivamente para fluidos ditos compressíveis. Após esta determinação, pode-se finalmente obter o mais importante conjunto de equações da dinâmica dos fluidos, as Equações de Navier-Stokes. Inicialmente derivadas em 1822 por Claude Louis Marie Henri Navier [13], que utilizando a noção da interação intermolecular, propôs uma solução onde levava em consideração os efeitos viscosos do escoamento. Vinte e três anos após a publicação de Navier, George Gabriel Stokes rederivou em 1845 [14] o trabalho de Navier, apresentando-as na forma como conhecemos hoje.

Percebe-se de antemão que a operação do termo $\partial_j \tau_{ij}$ corresponde ao real interesse desse tópico, por conta de sua relação com os termos viscosos do escoamento. Para tanto, toma-se a derivada da equação (14)

$$\partial_j \tau_{ij} = \partial_j \left(2\mu \partial_{(i} u_{i)} - \frac{2}{3}\mu \partial_{(k} u_{k)} \delta_{ij}\right), \tag{15}$$

$$\partial_o \rho u_i + u_j \partial_j (\rho u_i) = \rho f_i - \partial_j p + 2\mu \left[\frac{1}{2} \partial_j (\partial_i u_j + \partial_j u_i) - \frac{1}{3} \partial_j (\partial_k u_k) \delta_{ij} \right] \quad (16)$$

$$\partial_o \rho u_i + u_j \partial_j (\rho u_i) = \rho f_i - \partial_j p + \mu \partial_j (\partial_i u_j + \partial_j u_i) \quad (17)$$

Vale ressaltar, no entanto, que a forma apresentada em (16) refere-se à fluidos newtonianos viscosos e compressíveis. No caso de fluidos incompressíveis, onde o traço do tensor gradiente de velocidade é nulo ($\partial_k u_k = 0$), temos a forma incompressível da equação de Navier-Stokes conforme é apresentada na (17). Considerando que $\partial_j (\partial_i u_j + \partial_j u_i) = \nabla^2 u_i$ e que $\frac{\mu}{\rho} = \nu$ apresenta-se as equações de Navier-Stokes em sua forma mais conhecida

$$\nabla \cdot u = 0 \quad (18)$$

$$\frac{\partial}{\partial t} u + u \nabla \cdot u = f - \frac{1}{\rho} \nabla p + \nu \nabla^2 u_i \quad (19)$$

Apesar de sua dedução ser trivial, quando se lança mão de termos e noções corretas, a não-linearidade da equação obtida impede uma solução analítica direta. Sua importância estende-se desde a análise de escoamentos na ótica da mecânica do contínuo até a movimentação de galáxias nos estudos de astronomia. A larga aplicabilidade dessas equações, no entanto, são feitas de modo extremamente restrito, sendo modeladas sequencialmente por equações constitutivas, colaborando com a construção de um modelo numérico restrito. As Equações de Navier-Stokes são consideradas um dos problemas do milênio pela *Clay Mathematics Institute*, oferecendo um prêmio de $1,000,000.00 baseado no seguinte enunciado:

Prove or give a counter-example of the following statement

In three space dimensions and time, given an initial velocity field, there exists a vector velocity and a scalar pressure field, which are both smooth and globally defined, that solve the Navier–Stokes equations.

Portanto, como se demonstra, a intensão é comprovar a existência e suavidade de soluções em $\mathbb{R}^3$ das equações de Navier-Stokes [15].

3. A Modelagem da Dinâmica de Fluidos

Existem hoje disponíveis no mercado uma grande quantidade de softwares de simulação em fluidodinâmica, onde se pode citar o *Fluent*, *Ansys*, *Flow3D* entre outros. Grande parte dos sistemas proprietários dessa classe oferecem soluções completas, desde interfaces de criação de malha e pré-processamento até softwares integrados de pós-processamento. Uma tendência hoje é o desenvolvimento de softwares *Open Source*, na qual a colaboração mútua de desenvolvedores em fóruns na internet permite uma melhor adaptabilidade do software, implicando porém em uma maior dificuldade de utilização, visto a não preocupação com a interface gráfica de utilização. Mesmo com toda a complexidade computacional agregada ao uso de simuladores, o conhecimento acerca da fenomenologia e da reologia do escoamento é imprescindível, visto que a imposição das condições de contorno em softwares mais completos ou adaptação das sub-rotinas em códigos via linha de comando são de responsabilidade do pesquisador.

3.1. Método dos Volumes Finitos

Enquanto o método das diferenças finitas [16] recorre à utilização de ferramentas como a Série de Taylor, o método dos volumes finitos discretiza uma equação integrando-a sobre toda a região $\mathcal{R}^3$ de interesse. Ao executar este primeiro passo, obtêm-se um conjunto semi-discretizado de equações que, por métodos de interpolação, transforma as relações algébricas em um conjunto de equações algébricas solucionáveis. Para o caso específico, considera-se a malha unidimensional, conforme ilustra a Figura 2.

Um ponto geral P tem em sua vizinhança dois pontos, W e E, com superfícies de fronteira w e e. As distâncias nodais são definidas por δx_{WP} e δx_{PE} e as distâncias nodo-fronteira por δx_{wP} e δx_{Pe}.

Considerando a equação geral de conservação, na sua forma permanente e exclusivamente difusiva [4], onde o negativo do divergente do vetor gradiente de ϕ é igual ao termo de fonte Q

$$\partial_i [\Gamma \partial_i \phi] + Q = 0, \quad (20)$$

onde ϕ é a grandeza analisada e Γ é a constante de difusão.

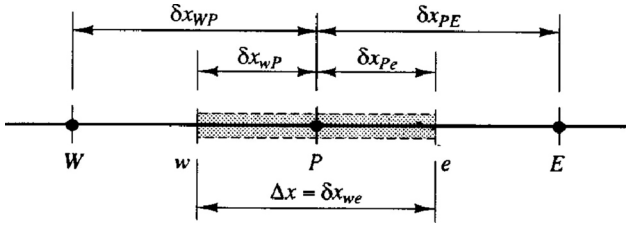

Figura 2: Malha unidimensional estruturada

 DOI: http://dx.doi.org/10.1590/1806-9126-RBEF-2018-0304

Integrando a equação (20) sobre todo o volume considerado e aplicando o Teorema de Gauss na primeira integral obtêm-se

$$\int_S \left[\Gamma \partial_i \phi\right] n_i dS + \int_V Q dV = 0. \tag{21}$$

Observando o exposto na Figura 2, onde w e e são as interfaces do volume de controle, nota-se que esta integral tem seu resultado definido por

$$\int_w^e \Gamma \partial_i \phi n_i dS = \left(\Gamma A \partial_i \phi\right)_w - \left(\Gamma A \partial_i \phi\right)_e, \tag{22}$$

onde (21) é reescrita como sendo

$$\left(\Gamma A \partial_i \phi\right)_w - \left(\Gamma A \partial_i \phi\right)_e + Q\Delta V = 0. \tag{23}$$

O termo fonte pode ser expresso pela soma de uma constante Q_u mais uma função da variável depentente $Q_p\phi_p$ na forma [17]

$$Q\Delta V = Q_u + Q_P \phi_P. \tag{24}$$

Admitindo-se que o fluxo convectivo de massa e a condutância difusiva ocorrem nas fronteiras dos volumes de controle, discretiza-se por interpolação linear o valor de Γ como sendo

$$\Gamma_w = \frac{\Gamma_W + \Gamma_P}{2} \qquad \Gamma_e = \frac{\Gamma_P + \Gamma_E}{2} \tag{25}$$

e para as derivadas de ϕ

$$\left(\partial_i \phi\right)_w = \left.\frac{d\phi}{dx_i}\right|_w = \left(\frac{\phi_P - \phi_W}{\delta x_{WP}}\right) \tag{26}$$

$$\left(\partial_i \phi\right)_e = \left.\frac{d\phi}{dx_i}\right|_e = \left(\frac{\phi_E - \phi_P}{\delta x_{PE}}\right) \tag{27}$$

Resultando na forma final da substituição da equação (24) na equação (23), levando em conta o demonstrado nas equações (25), (26) e (27).

$$a_P \phi_P = a_W \phi_W + a_E \phi_E + 2Q_u \tag{28}$$

onde a_i referem-se aos coeficientes de difusão efetivos, discretizados na malha em questão

$$a_W = \left(\frac{\Gamma_W + \Gamma_P}{\delta x_{WP}} A_w\right); \tag{29}$$

$$a_E = \left(\frac{\Gamma_P + \Gamma_E}{\delta x_{PE}} A_e\right); \tag{30}$$

$$a_P = a_W + a_E - 2Q_P. \tag{31}$$

Este resultado determina a discretização sob a perspectiva do método dos volumes finitos. Esta equação pode ser definida para cada nó da malha, a fim de solucionar o problema, quando apropriadamente adaptada as suas condições de contorno típicas. Seu resultado determina a distribuição da propriedade ϕ, discreta, sobre todo os nós da malha. A generalização para os casos bidimensional e tridimensional podem ser obtidos de formas semelhantes, de modo que a equação para o caso geral escreve-se na forma

$$a_P \phi_P = \sum a_n \phi_n + 2Q_p, \tag{32}$$

de modo que, para os casos especificados abaixo, os valores de n para as diferentes direções tratadas no problema

$$1D = a_W,\ a_E \tag{33}$$

$$2D = a_W,\ a_E,\ a_N,\ a_S \tag{34}$$

$$3D = a_W,\ a_E,\ a_N,\ a_S,\ a_F,\ a_B \tag{35}$$

Observa-se também que a generalização para o tratamento de casos advectivos/convectivos também pode ser obtido de forma semelhante, tornando esse método uma ferramenta amplamente usada na abordagem da dinâmica de fluidos computacional.

3.2. O algoritmo SIMPLE / PISO

Dado a construção do conjunto de equações discretizadas, faz-se necessário estabelecer a rotina de funcionamento deste modelo. Analisando o obtido na discretização da equação geral de conservação, poderia em primeiro momento concluir-se que a equação da continuidade e o princípio da conservação da quantidade de movimento são casos específicos desta, demonstrada sem os termos transiente e convectivo na equação (20). Mesmo que tal definição induza a pensar na simplicidade da solução, o desconhecimento *a priori* do campo de velocidades que efetivamente realiza o transporte da grandeza ϕ já induziria uma significativa complexidade, visto que este faz parte das equações a serem resolvidas. Tal fato gera termos não lineares como a derivada de $\rho u_i u_j$, o que torna ainda mais complexa a obtenção da solução numérica desejada.

Discretizando a equação da continuidade, para o caso de um fluido invíscido, obtem-se

$$a_e u_e = \sum a_n u_n - \left(p_P - p_E\right) A_e + 2S_u \tag{36}$$

Na análise da equação (36) percebe-se um acoplamento da solução da equação da quantidade de movimento com a determinação do campo de pressão. Uma abordagem a ser utilizada nesse problema é a solução acoplada das equações de momento, conjuntamente com um correlator do campo de pressão. Tal rotina caracteriza o algoritmo SIMPLE (acrônimo para *Semi-Implicit Method for Pressure-Linked Equations*)

O algoritmo SIMPLE [18] baseia-se no princípio que a equação do momento linear somente irá satisfazer a equação da continuidade se, e somente se, o campo de pressões correto for determinado [19]. Nesse algoritmo

existe a necessidade de corrigir-se tanto o campo de pressão como o de velocidades

$$u_{nf} = u^*_{nf} + u^{'}_f \qquad p_\alpha = p^*_\alpha + p^{'}_\alpha \tag{37}$$

onde de modo direto, relaciona-se a correção da velocidade e a correção do campo de pressão na forma de

$$u^{'}_f = d_f \left(p^{'}_\alpha - p^{'}_{\alpha+1} \right). \tag{38}$$

O algoritmo SIMPLE possui a seguinte sequência de execução

1. determinar um campo de pressões p^*;
2. resolver a equação do momento linear, afim de obter os valores para u^* (consequentemente para v^* e w^* no caso tridimensional);
3. resolver a equação de $p^{'}$;
4. determinar o valor exato do campo p;
5. resolver a equação de correção para determinar o valor de u (consequentemente para v e w no caso tridimensional);
6. usar o novo campo de pressões p para determinar um novo campo de pressões p^* e repetir novamente todo o processo até a obtenção da solução.

O algoritmo SIMPLE mostra-se eficiente na solução de diversos tipos de escoamentos desde que em seu estado estacionário, sejam eles laminares ou turbulentos. Para regimes de escoamento transientes utiliza-se o algoritmo PISO (acrônimo para *Pressure Implicit with Splitting of Operator*) [20], sendo uma extensão do algoritmo SIMPLE, no qual são gerados dois níveis de correção, na intensão de melhor caracterizar o regime transiente simulado.

4. A Simulação Computacional e a Interface de Controle

O *OpenFOAM* [3] é um conjunto de bibliotecas escritas em $C++$ desenvolvidas para simulações de sistemas na qual a fenomenologia é escrita em termos de campos tensoriais. Dentro dessas bibliotecas, existem vários modelos nas mais diversas áreas, onde por consequência uma das aplicabilidades mais usadas e com maior suporte online [2] é exatamente a Dinâmica de Fluidos Computacional. Inicialmente desenvolvido em 1993 por Henry Weller e Hrvoje Jasak, enquanto estudantes do Imperial College, o FOAM (acrônimo de *Field Operation and Manipulation*) foi uma proposta frente a inacessibilidade aos softwares de simulação disponíveis na época. Escrito em $C++$, o *OpenFOAM* é um software de simulação em CFD, que pode ser obtido via repositório da SourceForge, Github e da própria NABLA, detentora da licença oficial de desenvolvimento e distribuição do FOAM até o ano de 2005 e que posteriormente o liberou sob licença pública GNU.

[2] existe diversos fóruns de discussão e comunidades na internet que dispõe de FAQ's e conteúdos voltados à utilização do *OpenFOAM*

Hoje o código *OpenFOAM* é mantido e desenvolvido pela OpenCFD [21] e paralelamente a Wikki mantém e desenvolve o FoamExtended, similar ao *OpenFOAM*.

A estrutura de linguagem do *OpenFOAM* foi desenvolvida para aproximar de forma eficiente a linguagem matemática da linguagem de programação. A Tabela 1 demonstra alguns operadores diferenciais que estão disponíveis dentro da biblioteca de códigos do *OpenFOAM*.

Na manipulação das dimensões das grandezas representadas, o *OpenFOAM* permite operações algébricas nessas propriedades desde que devidamente representadas. Como forma de prevenir a implementação errada de determinadas rotinas, o código executa o cálculo dimensional em todas as operações tensoriais efetuadas. Para representá-las, o usuário tem disponível a seguinte linha de comando

```
dimensions      [0 2 -3 0 0 0 0];
//               | |  | | | | |
//               1 2  3 4 5 6 7
```

De modo que cada entrada entre os colchetes representa a dimensão abaixo especificada
De acordo com a Tabela 2, percebe-se que na linha de comando exemplificada, a dimensão da grandeza é em $\frac{m^2}{t^3}$. Em relação a sintaxe do programa, onde utilizando como exemplo a equação de transporte

$$\partial_o \rho T + \partial_i(\rho \phi T) - \partial_i(dt \partial_i T) = 0, \tag{39}$$

em que dt é o passo temporal, observa-se as seguintes regras:

```
solve
(
      fvm::ddt(T)
    + fvm::div(phi, T)
    - fvm::laplacian(dT, T)
);
```

Tabela 1: Tabela de equivalência das operações e a sitaxe utilizada no *software OpenFOAM*

peração	Expressão	Sintáxe
Derivada Temporal	∂_o	`ddt`
Gradiente	∇	`grad`
Divergente	$\nabla\cdot$	`div`
Laplaciano	∇^2	`laplacian`

Tabela 2: Tabela de entrada das dimensões suportadas pelo *software OpenFOAM*

N°	Propriedade	Unidade SI
1	Massa	quilograma Kg
2	Comprimento	metro m
3	Tempo	segundo t
4	Temperatura	Kelvin K
5	Quantidade molar	mol mol
6	Corrente elétrica	Àmpere A
7	Luminosidade	candela cd

 DOI: http://dx.doi.org/10.1590/1806-9126-RBEF-2018-0304

Convém notar que a variável ϕ (phi) presente no código descrito acima, corresponde ao campo de velocidades linearmente interpolado para as fronteiras do volume de controle, conforme a equação abaixo

$$\phi = \int_{\mathcal{A}} u_i n_i dA. \tag{40}$$

O *OpenFOAM* tem implementado o Método dos Volumes Finitos, como operação para discretização das EDP's a serem calculadas ao longo da simulação. Para tanto o software possui duas bibliotecas específicas, a `FVM.H` e a `FVC.H`. A primeira biblioteca corresponde ao conjunto de operações de discretização típicas do método dos volumes finitos. Os resultados fornecidos são armazenados em um objeto tipo `fvMatrix<Type>`, de modo a serem armazenados em uma matriz que será solucionada pelo próprio método. A segunda biblioteca traz o ferramental para manipular os termos da EDP de modo explícito não guardando-os porém, dentro de uma matriz solucionável pelo sistema.

5. Estrutura de Funcionamento

A interface J.A.R.V.E.S. foi construída para disponibilizar uma biblioteca de simulações pré definidas, baseadas nos exemplos mais utilizados nos cursos de mecânica dos fluidos, tais como:

a) escoamento laminar externo sobre um cilindro com dois módulos de Reynolds diferentes: $Re = 30$ e $Re = 195$;
b) escoamento turbulento interno em um alargamento tipo degrau (reprodução do experimento de Robert W. Pitz e John W. Daily de 1980) [22];
c) escoamento multi fase com dois fluidos imiscíveis de viscosidades absolutas distintas;
d) escoamento laminar interno em uma curva 90° com injetor;
e) simulação da esteira de Von-Karman [23];
f) escoamento supersônico de um fluido compressível sobre um estreitamento tipo degrau ($T = 3K$ e $P = 1N/m^2$);
g) escoamento de transição em um alargamento tipo degrau.

A interface J.A.R.V.E.S. está baseado em três ferramentas nativas ao Linux, onde o funcionamento conjunto auxilia a utilização, controle e edição dos *scripts* do OpenFOAM.

5.1. Linguagem *ShellScript*

A escolha da linguagem de extensão *Shell Script* veio da necessidade de adaptação e controle do OpenFOAM. Como a intenção principal na construção da interface J.A.R.V.E.S é controlar a forma como se trata a prémontagem fragmentada da simulação no OpenFOAM. Outras linguagens como `Python` e `C++`, nativa do próprio OpenFOAM, não atenderam a necessidade de controle direto das ações no computador, tal como a linguagem nativa do próprio Shell consegue executar.

O primeiro shell foi criado pelo próprio desenvolvedor do sistema OS UNIX [24], Kenneth Lane Thompson (1943), e foi distribuído nas versões 1 a 6 do UNIX, entre os anos 1971 a 1975 e, embora em desuso, o *Thompson Shell* foi o precursor de outros sistemas [25]. O *Bash*, interpretador Shell nativo da grande maioria das distribuições baseadas no GNU/LINUX foi desenvolvido por Brian Fox em 1988, a pedido de Richard Stallman, fundador da *Free Software Foundation*. Stallman desejava um shell livre e ao mesmo tempo eficiente, sendo um dos únicos projetos financiados pela FSF. O *Bash* foi mantido por Fox até 1994, sendo hoje assegurado por Chet Ramey, também da FSF.

Trabalhando em conjunto com o *Bash*, a linguagem *Shell Script* dá ação o comando escrito em código fonte, em um arquivo com extensão `.sh`, permitindo a construção de um programa ordenado, onde é possível estruturas que contenham:

a) estruturas de decisão (*if*);
b) estruturas de repetição (*for* ou *while*);
c) funções e argumentos;
d) definições de variáveis e formalismo de funcionamento.

A primeira linha de um *ShellScript* começa obrigatoriamente com um comando #! (Vale ressaltar no entanto que o indicativo de comentário no *Shell Bash* é o comando "#", sendo diferente do par utilizando no início do arquivo, que em inglês é pronunciado como "*shebang*"), informando diretamente ao núcleo qual shell ele deverá usar, sendo escrito tal como:

```
#!/bin/bash
```

Tal comando determina que o shell *Bash*, localizado no diretório `/bin` execute o script. Após isso, todos os comandos devem vir em linhas separadas.

5.2. Ferramenta SED

A ferramenta SED, acrônimo para *Stream Editor* ou editor de fluxo, que cumpre o papel de edição dos scripts típicos da simulação em OpenFOAM [26]. Como a interface precisava manipular e editar dados ou seções inteiras dentro do OpenFOAM, foi escolhido a ferramenta SED em função da praticidade de inserir, apagar, substituir e editar partes de um documento qualquer. A ferramenta SED foi criada por Lee E. McMahon entre os anos 1973 e 1974, quando era programador da BellLabs. Foi inicialmente disponibilizado na versão 7 do UNIX, sendo considerado um dos primeiros comandos para edição de texto via linha de comando. Vale ressaltar porém que existe uma grande diferença entre a ferramenta SED e editores nativos ao terminal como o `VI` e o `VI-M`, pois

o SED edita por orientação de linha, não necessitando de "abrir e editar" o arquivo, trabalhando-o dentro do espaço padrão (*Pattern space*).

A ferramenta SED usa a seguinte seguinte sintaxe básica:

```
sed [-opções] [comando] [<arquivo(s)>]
```

As opções possíveis para a ferramenta SED estão disponíveis no comando `~$ sed --help`, onde sua saída no terminal de comando lê-se, de modo traduzido como: Um exemplo prático da uso do comando SED é visto no script abaixo, retirado da interface J.A.R.V.E.S:

```
#!/bin/bash
#/-------------------------------------------------------------------------\
#|        J.A.R.V.E.S. - Just A Rather Very Educactive Simulator           |
#|-------------------------------------------------------------------------|
#| Versão do software:1.0                                                  |
#| Versao do DOC: 1.0                                                      |
#| Desenvolvedores: Thiago Felipe Domingos Dias Fernandes                  |
#|                  Nilton Luis Moreira                                    |
#| Contato: profthiagomecflu@gmail.com                                      |
#\-------------------------------------------------------------------------/
#LINHA 20
# ------------------------------------------------------------------------
sed -i '20 d' /$HOME/JARVES_1.0/SOLVERS/IN_COM_TUR/PITZDAILY/pitzDailynew/0/k
# ------------------------------------------------------------------------
sed -i "20s/^/internalField   uniform $num;\n/" /$HOME/JARVES_1.0/SOLVERS/
IN_COM_TUR/PITZDAILY/pitzDailynew/0/k
# ------------------------------------------------------------------------
```

Este comando trabalha diretamente na linha 20 do arquivo *k*, localizado no diretório original. O comando descrito na linha 15 executa a remoção da linha 20, através da opção "`d = delete`" e o comando da linha 17 executa a inserção do valor `internalField uniform $num;` nesta mesma linha através da opção "`s = substitute`" e o comando " `/^/    \`", onde o espaço entre o *slash* e o *backslash* é considerado um valor e não mais um comando, captando a string `$num` do stream anterior e substituindo-a pelo seu valor determinado. Apesar de aparentemente trabalhosa, a ação "apagar – substituir" mostrou-se bastante útil, visto que elimina a exigência da interferência do usuário na edição do script do OpenFOAM, automatizando significativamente a interface J.A.R.V.E.S.

5.3. A Interface DIALOG

Tendo estruturado o cerne de operação da interface J.A.R.V.E.S., faz-se necessário agora determinar as dependências da interface com o usuário. Das muitas opções disponíveis para as distribuições LINUX, a interface DIALOG foi a que melhor adaptou-se na proposta. A interface DIALOG foi criada por Savio Lam, sendo sua primeira versão lançada em dezembro de 1993, ainda na versão 0.1. Após 1999, o código-fonte do DIALOG está sendo mantido por Thomas E. Dickey, conforme destacado no site `https://invisible-island.net/dialog/` onde preserva o log de alterações do código fonte [27].

A interface de usuário é uma das partes mais importantes quando se trata da construção e funcionalidade de um software. Através dela, o usuário realiza tarefas, determina comandos e altera condições na estrutura de dados ou de processamento do computador. Seu surgimento deu-se exatamente para simplificar e tornar menos hostil a interação usuário-computador, disponibilizando elementos gráficos que, de modo intuitivo, guiem e correlacionem entradas e saídas de informação. Desde os primórdios da computação até meados da década de 80, a maioria dos computadores tinha sua interface baseada em texto (*Text-based User Interface* – TUI), onde a entrada dos comandos necessários é feita prioritariamente pelo teclado. As interfaces baseadas em texto caracterizam-se por disponibilizar exclusivamente texto e admitir entrada exclusivamente de texto, onde o principal benefício é a baixa exigência de processamento durante o uso e principal desvantagem é a pouca interatividade da interface.

A escolha de uma interface baseada em texto para o J.A.R.V.E.S. deu-se devido a possibilidade de sua utilização em terminais de comando, sem a exigência de ter-se instalado uma interface gráfica na distribuição LINUX, e a lista dos tipos de caixas disponíveis é dado pelo comando `dialog -- help`.

```
FLOW=$(                                                                    \
dialog                                                                     \
  --backtitle 'J.A.R.V.E.S. - Just A Rather Very Educative Simulator'      \
  --stdout                                                                 \
  --nocancel                                                               \
  --title 'J.A.R.V.E.S.'                                                   \
  --menu 'O ESCOAMENTO É INTERNO OU EXTERNO?'                              \
  10 60 0                                                                  \
  INTERNO   ''                                                             \
  EXTERNO   ''                                                             \
)

[ $? -ne 0 ] && bash /$HOME/JARVES_1.0/NODES/NODES2/retornarnode2.sh

case "$FLOW" in
  INTERNO)
    echo "$FLOW" > /tmp/JARVES/TIPO.tmp
    bash /$HOME/JARVES_1.0/NODES/NODES2/CLASSE.sh;;
  EXTERNO)
    echo "$FLOW" > /tmp/JARVES/TIPO.tmp
    bash /$HOME/JARVES_1.0/NODES/NODES2/CLASSE.sh;;
esac
```

Esta caixa baseada em DIALOG é disponibilizada para determinar qual tipo de escoamento será simulado. A string `FLOW` é relacionada à saída da caixa tipo menu, onde existem duas opções disponibilizadas, nas linhas 24 (`INTERNO`) e 25 (`EXTERNO`). A encadeamento de opções utilizadas determina que a caixa tenha uma mensagem de fundo (linha 18, opção `--backtitle`), que a saída seja interpretada como direta (linha 19, opção `--stdout`), sem a presença do botão CANCEL (linha 20, opção `--nocancel`) e que o título da caixa seja visto como J.A.R.V.E.S. (linha 21, opção `--title`). É criado um direcionamento caso o botão ESC do teclado seja pressionado (valor ”0” na saída), direcionando o terminal a uma tela de confirmação. Capturando a saída da caixa, o condicional `case` escrito das linhas 30 a 40 direciona para a execução de outro dois nós de direcionamentos descritos nas linhas 34 e 38. A saída do script, quando executado, é visualizado na Figura 3.

A junção destas poderosas ferramentas possibilitou a criação da interface automatizada, permitindo que o usuário concentre-se na modelagem básica do problema, eximindo um largo ”*know-how*” em manipulação de scripts. Apesar do detrimento da forma em função da usabilidade, a interface J.A.R.V.E.S. busca cumprir o papel de simplificar e popularizar o uso educacional da CFD, mais precisamente descomplexificar o formalismo do OpenFOAM.

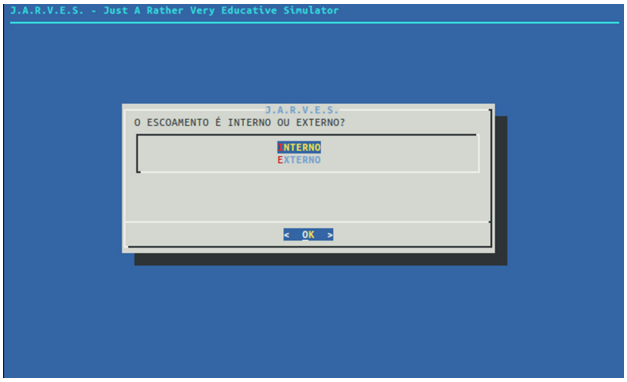

Figura 3: Caixa de texto criada pelo comando SED

5.4. Instalação e Manuais

O código-fonte da interface, juntamente com o pacote OpenFOAM adaptado está hospedado no site https://sourceforge.net/projects/jarves-1-0/, estando inicialmente compactado em protocolo `.tar.gz`. Juntamente a este, encontra-se o arquivo `MANUAL.pdf` onde está disposto um revisional sobre a dinâmica de fluidos do ponto de vista teórico e computacional, o manual de instalação, toda a rotina de simulação com os *scripts* associados discutidos e os objetivos e perspectivas do ponto de vista educacional.

6. Considerações Finais

A computação como ferramenta de ensino permite que fenômenos inteiros sejam modelados, de modo que além de sua ampla utilização no campo da pesquisa, a visualização e a manipulação de dados e condições da simulação contribuem para a construção do conhecimento [28]. No caso específico da sua utilização na educação superior, demonstra uma potencialidade ainda maior [29]. Em disciplinas abstratas como o caso da Mecânica dos Fluidos, onde apesar da larga utilização de ferramentas experimentais, determinadas características do escoamento acabam sendo assimiladas exclusivamente através em esquemas ou ilustrações. Dado que grande parte do estudo dedicar-se a dinâmica do escoamento, tais modos de representação acabam por serem falhos.

A pré existência de ferramentas computacionais em dinâmica dos fluidos colaborou significativamente, motivando a pesquisa sobre os métodos relacionados a este campo de estudo. A simulação computacional como ciência busca, dentre outros intuitos, ser uma ferramenta na análise de cenários. Modelar um problema computacionalmente é não só buscar reproduzi-lo como representação da situação real, mas também buscar possíveis cenários onde resultados colaborem com a predição de acontecimentos em escala real, demonstrando não somente a perspectiva de visualização, mas também a predição de acontecimentos, limitados claramente em validade e precisão. Sendo portanto uma ferramenta com aplicabilidade tão vasta em análise, predição e *design*, uma perspectiva educacional pode ser almejada, desde que adaptada apropriadamente para a situação. Como já enunciado anteriormente, a dificuldade intrínseca na manipulação da estrutura de algoritmo e o necessário *know-how* pré-existente em programação acaba por desmotivar docentes em inserir em suas disciplinas, estruturas de simulação de sistemas dinâmicos, abrindo portanto uma interessante demanda. A solução proposta na criação da interface J.A.R.V.E.S vem de encontro exatamente com esta necessidade, realizando em primeira instância, o papel de "programador-base" da simulação a ser apresentada educacionalmente.

Apesar de tratar problemas específicos a simulação computacional como ferramenta de solução de modelos matemáticos colabora para o entendimento acerca do fenômeno com um todo. Sua especificidade fica clara quando, ainda citando Heidemann [1], demonstra que: "Os modelos computacionais são 'recortes' da realidade, ou seja, são implementações computacionais de modelos específicos, e, como tais, desprezam diversos aspectos do sistema real, a fim de focar a atenção em certos aspectos particulares da natureza, o que facilita a compreensão do fenômeno físico". Durante a construção da interface, buscou-se não somente criar mais uma ferramenta de controle, mas sim um aparato baseado em teorias educacionais que efetivamente corroborasse com os preceitos teóricos pré-definidos, buscando gerar ao discente que, imerso em suas dependências, consiga alcançar um aprendizado significativo. Tal intensão vai de encontro com o apresentado por Araújo [29] quando diz "Não se melhora o ensino simplesmente produzindo novos e sofisticados recursos instrucionais. O desenvolvimento instrucional deve estar acoplado à pesquisa em ensino ou, pelo menos, levar em conta o conhecimento produzido pela pesquisa em ensino e os enfoques teóricos sobre aprendizagem compartilhados pela comunidade de educadores e pesquisadores em Ensino de Física".

Agradecimentos

Thiago F. D. Dias Fernandes agradece ao FAPEG pela bolsa de aperfeiçoamento concedida, ao Mestrado Nacional Profissional em Ensino de Física – MNPEF, Polo UFG/Catalão e à Faculdade de Caldas Novas pelas instalações para o desenvolvimento da pesquisa.

Referências

[1] L.A. Heideman, I.S. Araujo e E.A. Veit, Cad. Bras. Ensino Física **29**, 965 (2012).
[2] R. Tavares, Rev. Online Cie. Cog. **13**, 99 (2008).
[3] C.J. Greenshields *Openfoam user guide* (OpenFOAM Foundation, London, 2015).
[4] R.D. Blevins, *Applied fluid dynamics handbook* (Van Nostrand Reinhold, New York, 1984).
[5] D.R. Smith, *An Introduction to Continuum Mechanics — after Truesdell and Noll* (Springer, Dordrecht, 1993), p. 143.
[6] R. Laprise, Mont. Weather Rev. **120**, 197 (1992).
[7] P.M. Dixit e U.S. Dixit, *Modeling of metal forming and machining processes: by finite element and soft computing methods* (Springer-Verlag, London, 2008).
[8] J. Sokolowski e J.P. Zolesio, *Introduction to shape optimization* (Springer, Berlin, 1992).
[9] B.R. Munson, T.H. Okiishi, W.W. Huebsch e A.P. Rothmayer, *Fluid mechanics* (Wiley, Singapore, 2013).
[10] A.J. Chorin e J.E. Marsden, *A mathematical introduction to fluid mechanics* (Springer, New York, 1990).
[11] J. Serrin, J. Math. Mec. **4**, 459 (1959).
[12] I. Newton *Philosophiae Naturalis Principia Mathematica* (Cambridge University Press, London, 1972).
[13] C.L.M.H. Navier, Ann. Chim. Phys **19**, 234 (1822).
[14] G.G. Stokes, Trans. Cambridge Philos. Soc. **8**, 287 (1845).
[15] L. Tartar, *An introduction to Navier-Stokes equation and oceanography* (Springer-Verlag, Berlin, 2006).
[16] A.R. Mitchell e D.F. Griffiths, *The finite difference method in partial differential equations* (Wiley, New York, 1980).
[17] H.K. Versteeg e W. Malalasekera, *An introduction to computational fluid dynamics: the finite volume method* (Longman Scientific & Techincal, Essex, 2007).
[18] S. Patankar *Numerical heat transfer and fluid flow* (Hemisphere, Washington, 1980).
[19] J.P. Van Doormaal e G.D. Raithby, Num. Heat Trans. **2**, 147 (1984).

[20] D.S. Jang, R. Jetli e S. Acharya, Num. Heat Trans. **3**, 209 (1986).
[21] `http://www.openfoam.com`, acessado em 08/08/2018.
[22] R.W. Pitz e J.W. Daily, AIAA Journal **11**, 1565 (1983).
[23] T. Von Kármán e L. Edson, *The wind and beyond* (Little, Brown & Company, Boston, 1967).
[24] M.J. Bach. *The design of the UNIX operating system* (Prentice-Hall, Englewood Cliffs, 1986).
[25] S.R. Bourne, Bell System Tec. J. **6**, 1971 (1978).
[26] E. Siever, S. Figgins, R. Love e A. Robins, *Linux in a Nutshell* (O'Reilly Media, Sebastopol, 2005).
[27] J.C. Neves *Programação Shell Linux* (Brasport, Rio de Janeiro, 2008).
[28] A. Medeiros e C.F. de Medeiros, Rev. Bras. Ensino Física **2**, 77 (2002).
[29] I.S. Araujo e E.A. Veit, Rev. Bras. Pesq. Educação Ciências **3**, (2011).