

Iniciação Científica

Tutorial e Manual de Boas Práticas



Professor: Fernando Akira Kurokawa

Tema: Dinâmica dos Fluidos Computacional

São Paulo,
2022

Introdução

Esse documento tem como objetivo apresentar as principais funções existentes no software CFD OpenFoam. A partir deste, os alunos de iniciação científica do professor Fernando Kurokawa poderão entender melhor o funcionamento da ferramenta, além de dar os seus primeiros passos para a sua utilização. No tutorial, será exposto o caso do escoamento de fluidos em torno de um cilindro, o caso mais geral e mais comum que servirá de base para todos os posteriores.

Metodologia

O tutorial se divide em uma visão geral do OpenFoam, descrevendo o seu download, a linguagem utilizada, o terminal e as pastas gerais para a simulação. Após essa etapa, há um tutorial passo-a-passo de como criar a malha que será utilizada, assim como configurar os arquivos corretamente. Por fim, roda-se a simulação a fim de observar o comportamento dos ventos, além de obter os coeficientes necessários para essa análise, como o coeficiente de arrasto e o número de Strouhal.

1. Visão Geral

1.1 Download

Por ser um software em LINUX, o openFOAM não funciona como os softwares convencionais que você já utilizou no Windows. Assim, ele opera através de informações digitadas no terminal, inclusive para realizar o download da ferramenta. Para isso, basta acessar o site oficial do www.openfoam.com e pesquisar pelo download da versão 8 para Ubuntu, digitando as informações oferecidas no link no terminal do seu computador.

1.2 Sintaxe

Os arquivos de texto que editaremos para a formulação das simulações será baseado nos princípios gerais da linguagem de programação “C++”, dessa forma, pode-se perceber algumas características básicas como:

- Os arquivos tem um formato livre, ou seja, as linhas e colunas não tem um significado específico e não há necessidade de indicar continuação das linhas.
- Pode-se comentar linhas de código através da utilização de duas barras: (*// código*) no início ou compreendendo o texto da seguinte forma: */* código */*
- Os trechos de código que forem comentados serão completamente ignorados pelo openFOAM e, portanto, não irão alterar o desenrolar da simulação. Portanto, costuma-se utilizar os comentários de código para organizar as diferentes seções dos arquivos em suas respectivas funções.

Os arquivos do openFOAM funcionam através de “dicionários”: entidades que contém uma série de dados que são resgatados pelo aplicativo através de palavras-chave. A inserção de dados e palavras chaves seguem o seguinte padrão:

```
<Palavra-Chave> <Dado1>
```

A maioria dos arquivos de dados do openFOAM são, por si só, dicionários. Os dicionários tem como objetivo organizar as informações em categorias lógicas e podem ser classificados de forma hereditária de forma a possibilitar que um dicionário possua dentro dele outros dicionários

O formato padrão de um dicionário é o seguinte:

```
<Nome do Dicionário>
{
    Dados e Palavras chave
}
```

1.3 Pastas

Sabe-se que em LINUX não é possível ir montando a simulação diretamente na interface do software, sendo necessário alterar os arquivos das pastas da simulação para configurar as condições de contorno de acordo com o desejado. Assim, apenas no final da configuração de todas as pastas e de rodar a simulação, é possível visualizar o que está acontecendo, abrindo a interface gráfica do software através do comando “ParaFoam”.

1.3.1 Pasta “0”

Nessa pasta, configuram-se as condições iniciais do problema para cada face da malha. Existem dois arquivos chaves, o “p”, que configura a pressão, e o “u” que configura a velocidade do escoamento.

1.3.2 Pasta “system”

Nessa pasta, existem arquivos que configuram todo o sistema da simulação, ou seja, a forma com que ela será resolvida. Isso envolve o tempo e os intervalos de tempo que o computador irá resolver as equações (arquivo ControlDict), informações sobre o modelo de turbulência (existem diferentes tipos, como k-epsilon, k-omega) e também informações sobre a malha (BlockMeshDict)

1.3.3 Pasta “constant”

Após gerar a malha, será criada a pasta “PolyMesh”, a qual contém arquivos referentes à faces, células e componentes da malha, incluindo o arquivo “Boundary”, o qual é responsável pelos limites da simulação e é constantemente alterado. Além disso, a pasta “constant” possui os arquivos “transportProperties”, o qual comanda o número de Reynolds, e “turbulenceProperties”, responsável pelo modelo de turbulência.

2. Malhas

Sabendo que os softwares de CFD, incluindo o OpenFOAM, são regidos pelas equações diferenciais de Navier-Stokes. Essas equações não conseguem ser resolvidas para valores exatos, dessa forma, é necessário repartir em pequenos domínios o nosso campo de simulação, para que as equações sejam resolvidas, obtendo valores estimados, em cada uma dessas partições. Assim, surge a necessidade da construção de malhas para rodar a simulação. Nas malhas, geramos partições no domínio todo, de acordo com o nível de refinamento desejado pelo usuário. Quanto menores os quadradinhos, mais precisa será a nossa simulação. Analogamente, quanto maiores os quadradinhos, menos refinada e

precisa será a nossa simulação. Além disso, é possível fazer uma progressão, de forma que mais próximo de certa área, a malha será mais refinada e longe dela as partições vão aumentando de tamanho, seguindo uma progressão padrão escolhida pelo usuário. Vale ressaltar que a estrutura, o qual o escoamento irá ocorrer em torno dele, não será repartida. Ou seja, é como se ficasse um buraco na malha, o qual representa o nosso objeto. Assim, nesse buraco não serão resolvidas as equações e o software irá considerá-lo como um obstáculo. Para compreender melhor as informações apresentadas acima, observe a imagem a seguir:

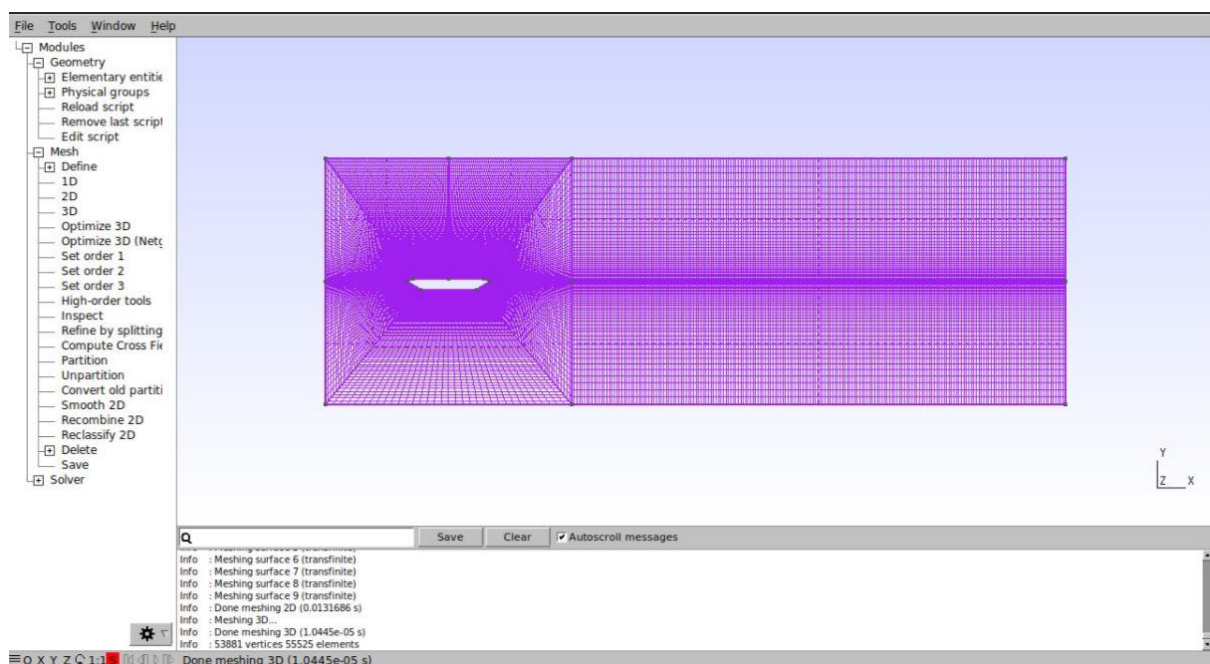


Figura X: Malha de uma seção de ponte feita utilizando o gmsh

Inicialmente, para a construção de malhas, utilizaremos o software gmsh, o qual pode ser facilmente baixado através do terminal (pesquisar na internet para realizar a sua instalação).

2.1 Tutorial para a construção da malha de um cilindro

O Gmsh pode ser utilizado de duas formas: através dos comandos da interface gráfica do programa ou editando o script, na forma do arquivo .geo, gerado pelo programa. Este tutorial abordará ambas as formas de uso do programa. Para abrir o script, vá para Modules→Geometry e clique em Edit script.

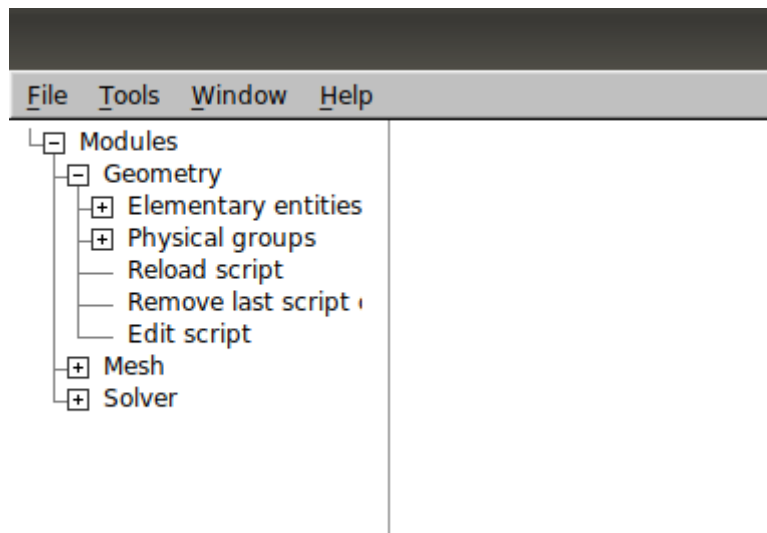


Figura X: interface do gmsh para editar o script

Para explicar o processo de desenvolvimento das malhas utilizando essa ferramenta, será usado o exemplo da malha para realizar o escoamento em torno de um cilindro, o caso mais simples e que é base para todos os outros.

- Definem-se os pontos:

O primeiro passo é a declaração das coordenadas da geometria. No arquivo .geo digite as coordenadas da geometria no formato "Point(1) = (0,0,0,1.0);", onde os primeiros três termos são as coordenadas x, y e z, respectivamente. Na nossa malha, definiremos um parâmetro d e um cosseno padrão, deixando todos os pontos com coordenadas em função deles. Assim, o arquivo deve ficar dessa forma:

```
d = DefineNumber[ 0.5, Name "Parameters/d" ];

cos = DefineNumber[ 0.707106782, Name "Parameters/cos" ];

Point(1) = {0, 0, 0, 1.0};

Point(2) = {d*cos, d*cos, 0, 1.0};
Point(3) = {-d*cos, d*cos, 0, 1.0};
Point(4) = {-d*cos, -d*cos, 0, 1.0};
Point(5) = {d*cos, -d*cos, 0, 1.0};

/*
Point(6) = {4*d*cos, 4*d*cos, 0, 1.0};
Point(7) = {-4*d*cos, 4*d*cos, 0, 1.0};
```

```
Point(8) = {-4*d*cos, -4*d*cos, 0, 1.0};
Point(9) = {4*d*cos, -4*d*cos, 0, 1.0};
*/
```

```
Point(10) = {-5.5, -5.5, 0, 1.0};
Point(11) = {-5.5, 5.5, 0, 1.0};
Point(12) = {19.5, -5.5, 0, 1.0};
Point(13) = {19.5, 5.5, 0, 1.0};
```

```
Point(14) = {5.5, -5.5, 0, 1.0};
Point(15) = {5.5, 5.5, 0, 1.0};
Point(16) = {19.5, 0, 0, 1.0};
```

```
Point(17) = {-5.5, 0, 0, 1.0};
Point(18) = {0, 5.5, 0, 1.0};
Point(19) = {5.5, 0, 0, 1.0};
Point(20) = {0, -5.5, 0, 1.0};
```

```
Point(21) = {-0.5, 0, 0, 1.0};
Point(22) = {0, 0.5, 0, 1.0};
Point(23) = {0.5, 0, 0, 1.0};
Point(24) = {0, -0.5, 0, 1.0};
```

Salve o arquivo. Na interface gráfica, clique em Modules→Geometry → Reload script (deverá ser pressionado sempre que for necessário aplicar as alterações feitas no.geo). O resultado é mostrado na figura abaixo.

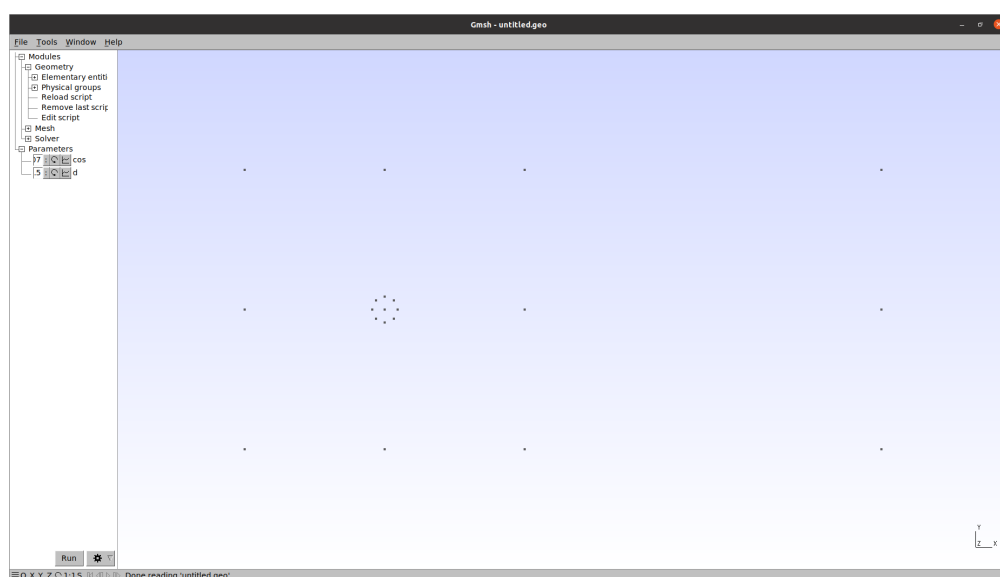


Figura X: visualização dos pontos da malha

Obs: Para criar os pontos utilizando a interface gráfica, vá em Modules→Geometry → Elementary entities → Add → Point.

Mantenha consistência na numeração dos pontos. Não pode haver dois pontos com o mesmo nome e a numeração dos pontos é utilizada para criação das demais partes da geometria. Logo, a organização é fundamental para facilitar o trabalho. Caso seja necessário, pode-se tornar o nome dos pontos visíveis na interface gráfica em Tools → Options → Geometry e marcar a opção Point labels.

- Definem-se as linhas:

As linhas retas que definem o domínio da simulação e os blocos da malha podem ser geradas via script adicionando o comando “Line(1) = {1,2};”, onde 1 e 2 são os pontos que deseja-se conectar. Repita o processo até conectar todos os devidos pontos. O resultado no .geo e na interface deve ser como mostrado abaixo:

Obs: para unir os pontos na interface gráfica, utilize a ferramenta Modules→Geometry → Elementary entities → Add → Line e selecione os pontos que devem ser conectados.

```
Line(5) = {17, 11};
Line(6) = {11, 18};
Line(7) = {18, 15};
Line(8) = {15, 13};
Line(9) = {13, 16};
Line(10) = {16, 12};
Line(11) = {12, 14};
Line(12) = {14, 20};
Line(13) = {20, 10};
Line(14) = {10, 17};
Line(15) = {17, 21};
Line(16) = {11, 3};
Line(17) = {18, 22};
Line(18) = {15, 2};
Line(19) = {23, 19};
Line(20) = {5, 14};
Line(21) = {24, 20};
Line(22) = {4, 10};
Line(23) = {19, 15};
Line(24) = {14, 19};
Line(25) = {19, 16};
```

```
Circle(26) = {21, 1, 3};
Circle(27) = {3, 1, 22};
```



```

Circle(28) = {22, 1, 2};
Circle(29) = {2, 1, 23};
Circle(30) = {23, 1, 5};
Circle(31) = {5, 1, 24};
Circle(32) = {24, 1, 4};
Circle(33) = {4, 1, 21};

```

O resultado pode ser observado abaixo:

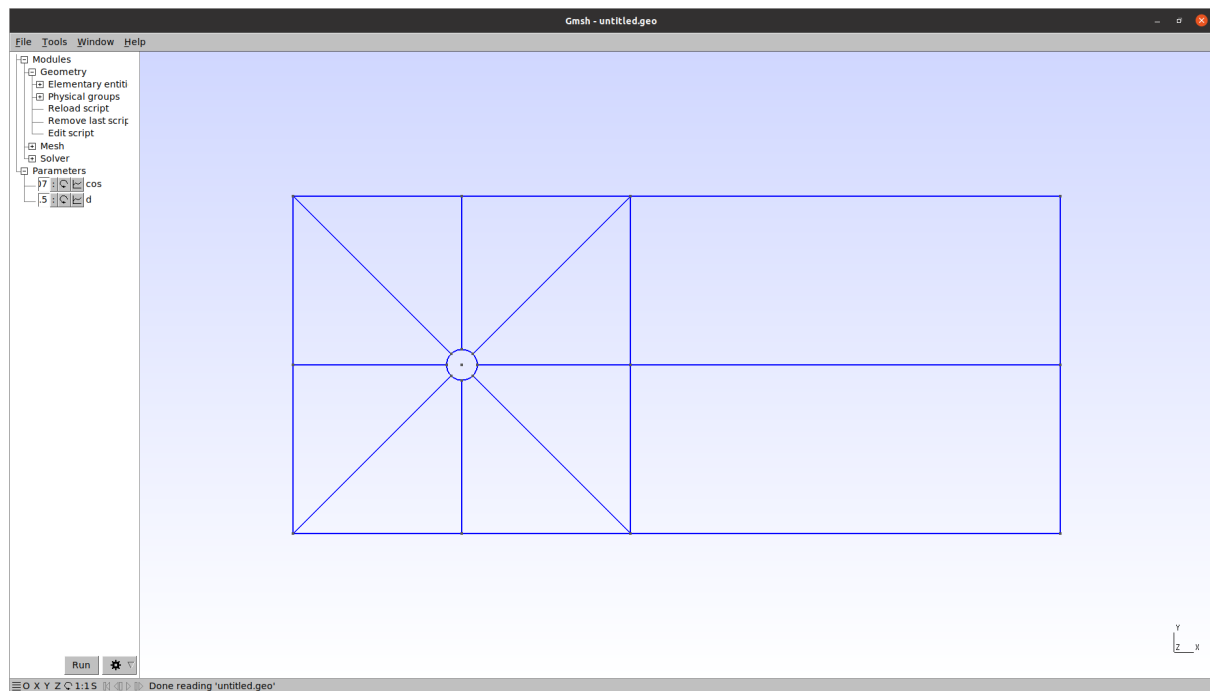


Figura X: visualização das linhas na malha

- Definem-se as superfícies

Em seguida, é hora de definir as superfícies. Para tal, selecione Modules → Geometry → Elementary entities → Add → Plane surface. Para definir uma superfície, quatro linhas devem ser selecionadas. Importante observar, que poderiam ser criadas superfícies entre três linhas ou mais. Contudo, como desejamos uma malha estruturada, as superfícies dos blocos da malha devem ser definidas desta forma. Isso explica as escolhas de repartição da geometria feitas anteriormente.

Selecione as quatro linhas que delimitam cada superfície e aperte a tecla “e” para confirmar. Se o processo for feito corretamente deverão aparecer linhas pontilhadas no meio das superfícies.

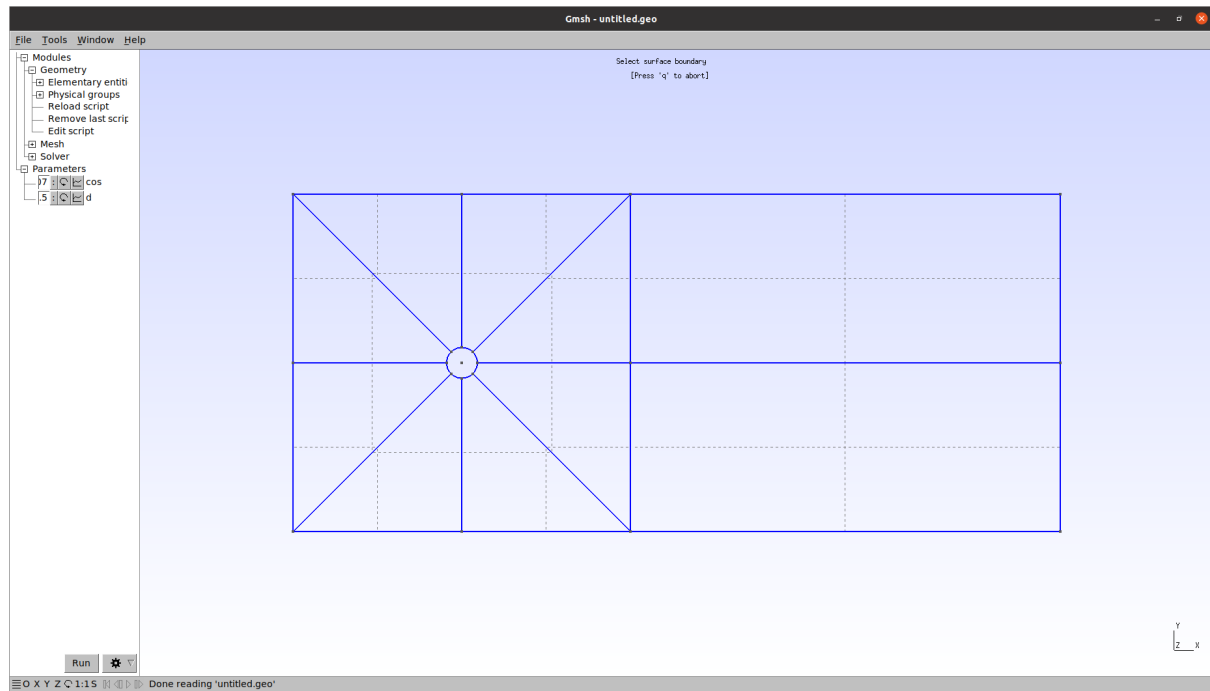


Figura X: visualização das superfícies na malha

No código, ficará da seguinte forma:

```
Curve Loop(1) = {5, 16, -26, -15};
Plane Surface(1) = {1};

Curve Loop(2) = {6, 17, -27, -16};
Plane Surface(2) = {2};

Curve Loop(3) = {7, 18, -28, -17};
Plane Surface(3) = {3};

Curve Loop(4) = {23, 18, 29, 19};
Plane Surface(4) = {4};

Curve Loop(5) = {24, -19, 30, 20};
Plane Surface(5) = {5};

Curve Loop(6) = {12, -21, -31, 20};
Plane Surface(6) = {6};

Curve Loop(7) = {13, -22, -32, 21};
```

```

Plane Surface(7) = {7};

Curve Loop(8) = {14, 15, -33, 22};

Plane Surface(8) = {8};

Curve Loop(9) = {23, 8, 9, -25};

Plane Surface(9) = {9};

Curve Loop(10) = {10, 11, 24, 25};

Plane Surface(10) = {10};

```

A função Curve Loop define as linhas que delimitam a superfície (os mesmos nomes das linhas anteriores) e a Plane Surface cria a superfície. Os sinais negativos no número de algumas linhas indicam as suas orientações, dependendo da ordem dos pontos que as definem.

- Transfinite

Para a criação de uma malha estruturada o Gmsh oferece a opção de Transfinite que deve ser aplicada em todas as linhas. Essa opção permitirá que você defina o número de volumes que sua malha terá, assim como a taxa de crescimento desses volumes ao longo de uma linha.

Clique em Modules → Mesh → Define → Transfinite → Curve. Será definido um Transfinite para cada linha, por isso é necessário que mantenha consistência na sua criação (pode-se deixar explícito o nome das linhas pelo mesmo caminho que foi utilizado para os pontos) e a utilização de comentários no .geo para que não haja confusão com os nomes. Selecione uma linha e aperte a tecla “e” para confirmar. No .geo, a função sefa definida assim:

```
// TRANSFINITE:
```

```
//+
```

```
Transfinite Curve {20} = 10 Using Progression 1;
```

O valor 10 indica o número de divisões que a malha terá naquela linha e o valor um, a razão de tamanhos entre dois volumes consecutivos. No código ficará da seguinte forma:

```
Recombine Surface {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
```

```
Transfinite Curve {22, 15, 16, 17, 18, 19, 20, 21} = 30 Using  
Progression 1; //lines around the cylinder
```

```
Transfinite Curve {23, 9} = 16 Using Progression 1;
```

```
Transfinite Curve {24, 10} = 16 Using Progression 1;
```

```
Transfinite Curve {6, 27} = 16 Using Progression 1;
```

```
Transfinite Curve {7, 28} = 16 Using Progression 1;
```

```
Transfinite Curve {23, 29} = 16 Using Progression 1;
```

```
Transfinite Curve {24, 30} = 16 Using Progression 1;
```

```
Transfinite Curve {12, 31} = 16 Using Progression 1;
```

```
Transfinite Curve {13, 32} = 16 Using Progression 1;
```

```
Transfinite Curve {14, 33} = 16 Using Progression 1;
```

```
Transfinite Curve {5, 26} = 16 Using Progression 1;
```

```
Transfinite Curve {11, 25, 8} = 32 Using Progression 1;
```

ATENÇÃO: linhas opostas em um mesmo bloco devem ter o mesmo número de divisões para que a malha resultante seja estruturada, por isso é indicado colocar a mesma variável para estas linhas. Para realizar uma progressão, é necessário que linhas paralelas, as quais estão em uma mesma superfície, estejam na mesma direção, se isso não acontecer, vai dar tudo errado.

Obs: se você desejar fazer uma malha mais refinada, ou seja, com mais elementos, basta mexer no número de divisões e na progressão, até atingir um resultado satisfatório.

- Transfinite Surface

Em seguida, é preciso aplicar um Transfinite nas superfícies. Clique em Modules → Mesh → Define → Transfinite → Surface. Selecione cada superfície e aperte a letra “e” para confirmar. Em seguida vá em Modules → Mesh → Define → Recombine e selecione todas as superfícies e aperte “e” para confirmar. O resultado no .geo deverá ser o mostrado abaixo.

```
Transfinite Surface {1};  
Transfinite Surface {2};  
Transfinite Surface {3};  
Transfinite Surface {4};  
Transfinite Surface {5};  
Transfinite Surface {6};  
Transfinite Surface {7};  
Transfinite Surface {8};  
Transfinite Surface {9};  
Transfinite Surface {10};
```

Agora, nossa malha 2D está pronta, basta clicar em Modules → Mesh → 2D, e o resultado deve ser o mostrado a seguir:

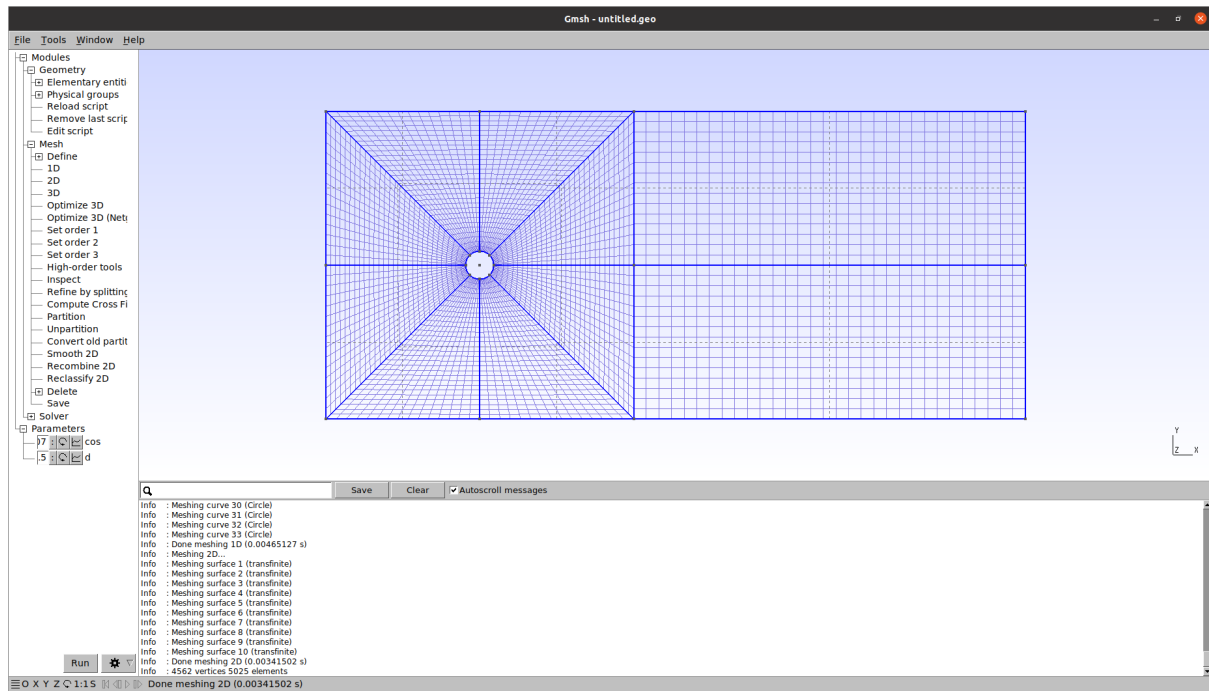


Figura X: malha 2d no gmsh

- Extrusão

Mesmo com o desejo de fazer uma simulação 2D, a malha precisa ser 3D para conseguir rodar. Para transformar a malha 2D em 3D, faz-se a extrusão da seguinte maneira no código:

```
Extrude {0, 0, 4} {
  Surface{1}; Surface{2}; Surface{3}; Surface{4}; Surface{5};
  Surface{6}; Surface{7}; Surface{8}; Surface{9}; Surface{10};
  Layers{2}; Recombine;
}
```

Na linha “Extrude {0, 0, 4}”, o número 4 significa a largura da extrusão na direção Z. Layers significa a subdivisão da extrusão, ou seja, os quadradinhos na direção Z. Agora, nossa malha irá ficar da seguinte forma:

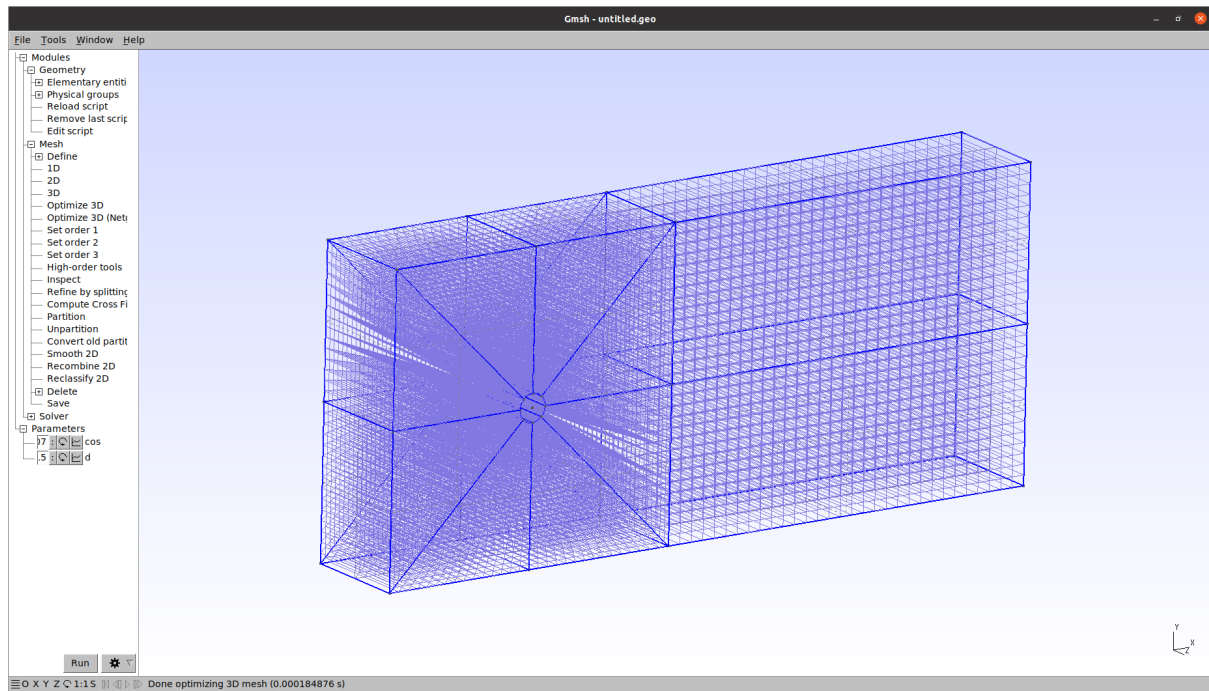


Figura X: malha 3d no gmsh

- Nomear as superfícies

Para realizar essa etapa, é melhor fazer pela interface gráfica do que pelo código.

- Exportar a malha

Após todas essas etapas, vá em Modules→Geometry → Reload script e crie a malha em 3D.

Por fim, é necessário exportar a malha indo em File →Export. Dê o nome do arquivo e escolha a pasta para guardá-lo e escolha a opção no canto inferior direito “Mesh-Gmsh MSH(*.msh)”.

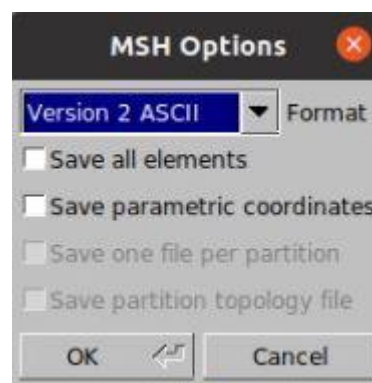


Figura X: opções para a exportação da malha

Agora sua malha estará pronta para rodar a simulação. Coloque o arquivo dentro da pasta da simulação:

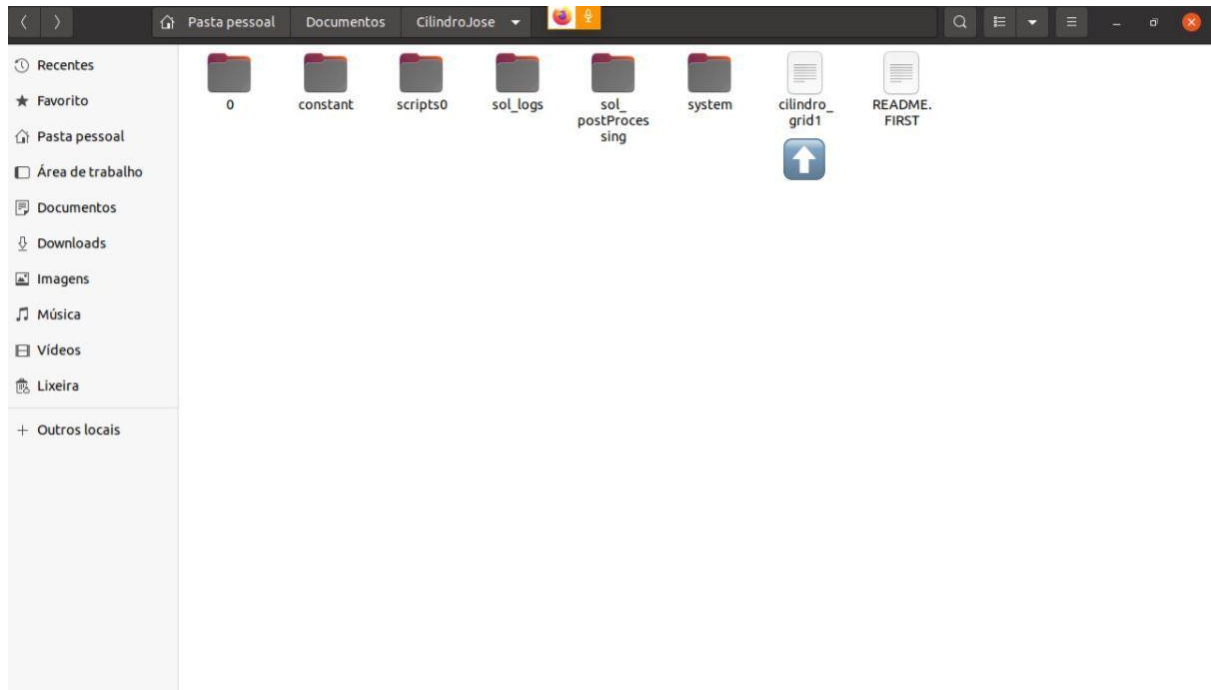


Figura X: arquivo da malha dentro da pasta de simulação

É essencial retirar os arquivos "TransportProperties" e "TurbulenceProperties" da pasta "constant" e deletá-la. Agora, abrimos o terminal do Linux (Ctrl + Alt + T) e digitamos da seguinte forma "gmshtofoam 'nome do arquivo adicionado referente à malha'", como pode ser representado abaixo:

```
tiago@tiago-Lenovo-IdeaPad-S145-15IWL: ~/c18_cilindro_grid1
tiago@tiago-Lenovo-IdeaPad-S145-15IWL:~$ cd c18_cilindro_grid1/
tiago@tiago-Lenovo-IdeaPad-S145-15IWL:~/c18_cilindro_grid1$ gmshtofoam cilindro_grid1
/*-----*/
=====
\\      / F ield      | OpenFOAM: The Open Source CFD Toolbox
\\      / O peration  | Website:  https://openfoam.org
\\      / A nd        | Version:   8
\\      / M anipulation|
/*-----*/
Build   : 8-1c9b5879390b
Exec    : gmshtofoam cilindro_grid1
Date    : Sep 02 2021
Time    : 14:27:23
Host    : "tiago-Lenovo-IdeaPad-S145-15IWL"
PID     : 10821
I/O     : uncollated
Case    : /home/tiago/c18_cilindro_grid1
nProcs  : 1
sigFpe  : Enabling floating point exception trapping (FOAM_SIGFPE).
fileModificationChecking : Monitoring run-time modified files using timeStampMaster (fileModificationSkew 10)
allowSystemOperations : Allowing user-supplied system call operations
```

Figura X: comando para a geração da malha

Por fim, escreva o comando “checkMesh” para ver se está tudo correto com a malha. Em caso de erro, o terminal dirá em que parte ele se encontra, facilitando sua solução.

```
cfdd05@cfdd05-XPS-8920: ~/TIAGO/c18_cilindro_grid1
faces: 40439
internal faces: 19789
cells: 10038
faces per cell: 6
boundary patches: 6
point zones: 0
face zones: 0
cell zones: 1

Overall number of cells of each type:
hexahedra: 10038
prisms: 0
wedges: 0
pyramids: 0
tet wedges: 0
tetrahedra: 0
polyhedra: 0

Checking topology...
Boundary definition OK.
Cell to face addressing OK.
Point usage OK.
Upper triangular ordering OK.
Face vertices OK.
Number of regions: 1 (OK).

Checking patch topology for multiply connected surfaces...
Patch      Faces    Points    Surface topology
frontAndBack 20076  20050    ok (non-closed singly connected)
cylinder     238    476      ok (non-closed singly connected)
inlet        59     118      ok (non-closed singly connected)
top          107    216      ok (non-closed singly connected)
bottom       107    216      ok (non-closed singly connected)
outlet       64     130      ok (non-closed singly connected)

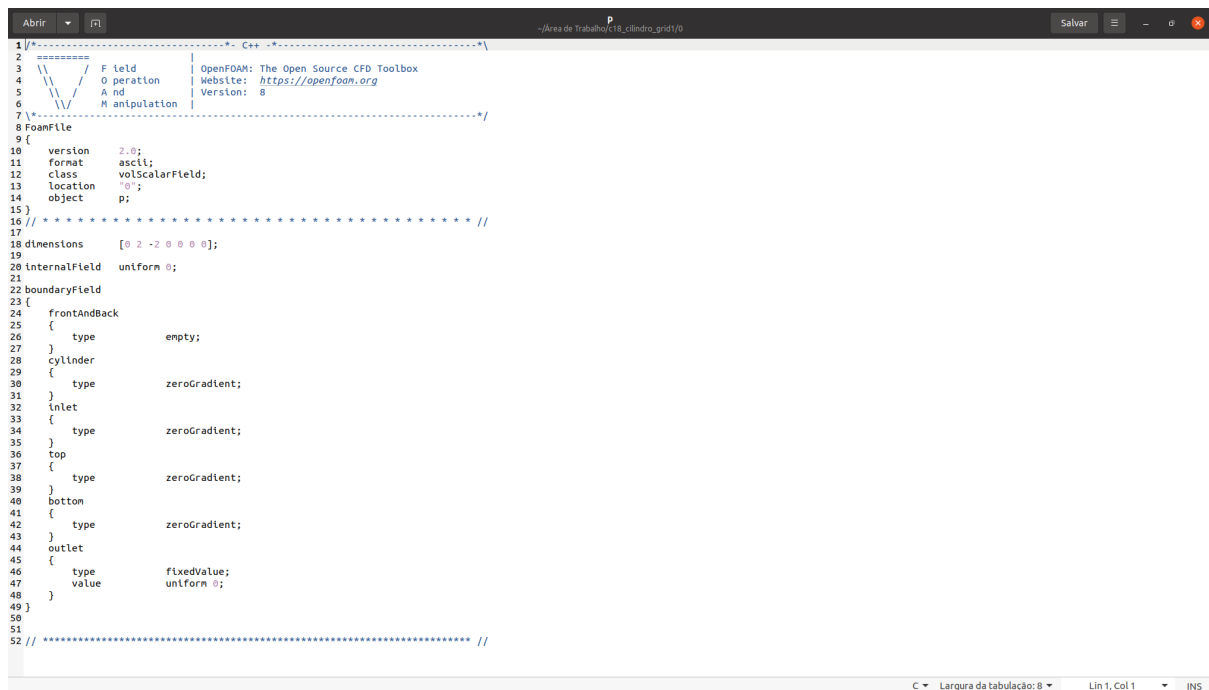
Checking geometry...
Overall domain bounding box (-5.5 -5.5 0) (19.5 5.5 1)
Mesh has 3 geometric (non-empty/wedge) directions (1 1 1)
Mesh has 3 solution (non-empty) directions (1 1 1)
Boundary openness (-8.5406539e-18 -2.8480846e-17 4.0063104e-19) OK.
Max cell openness = 2.4226647e-16 OK.
Max aspect ratio = 194.61262 OK.
Minimum face area = 0.0001412318. Maximum face area = 0.76379313. Face area magnitudes OK.
Min volume = 0.0001412318. Max volume = 0.17521515. Total volume = 274.21471. Cell volumes OK.
Mesh non-orthogonality Max: 44.188514 average: 16.621285
Non-orthogonality check OK.
Face pyramids OK.
Max skewness = 2.4582745 OK.
Coupled point location match (average 0) OK.

Mesh OK.
End
cfdd05@cfdd05-XPS-8920:~/TIAGO/c18_cilindro_grid1$
```

Figura X: comando checkMesh

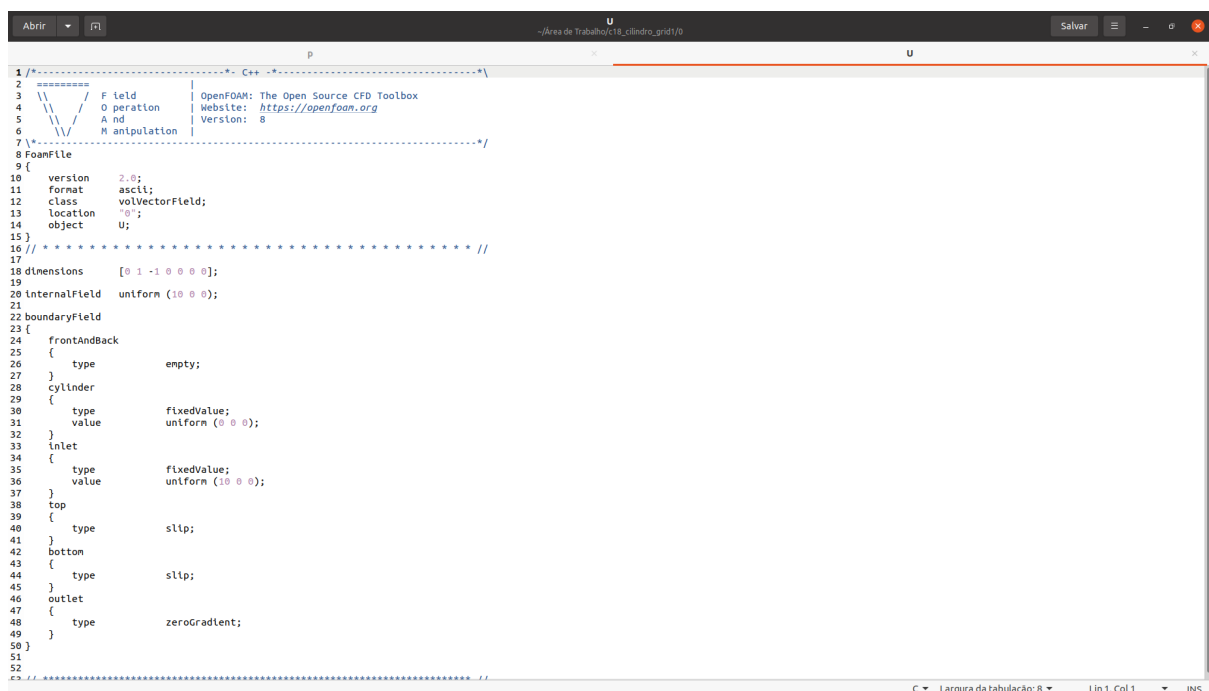
3. Configuração dos arquivos para rodar a simulação do cilindro

Primeiro passo consiste em entrar na pasta Constant →PolyMesh →Boundary. Na pasta boundary, faça as seguintes alterações:

A screenshot of a text editor window showing the configuration file 'p'. The window title is 'p' and the path is '~\Área de Trabalho\c18_cilindro_grid1\0'. The code is in C++ and defines a volScalarField. It sets dimensions to [0 2 -2 0 0 0], internal field to uniform 0, and boundary field with frontAndBack (empty), cylinder (zeroGradient), inlet (zeroGradient), top (zeroGradient), bottom (zeroGradient), and outlet (fixedValue uniform 0).

```
1 /*-----* C++ -*-----*/
2 =====
3 \ \ \ \ \ F i e l d      | OpenFOAM: The Open Source CFD Toolbox
4 \ \ \ \ \ O peration    | Website: https://openfoam.org
5 \ \ \ \ \ A n d         | Version: 8
6 \ \ \ \ \ M anipulation  |
7 /*-----*
8 FoamFile
9 {
10     version      2.0;
11     format       ascii;
12     class        volScalarField;
13     location     "0";
14     object       p;
15 }
16 // *****
17
18 dimensions      [0 2 -2 0 0 0];
19
20 internalField    uniform 0;
21
22 boundaryField
23 {
24     frontAndBack
25     {
26         type      empty;
27     }
28     cylinder
29     {
30         type      zeroGradient;
31     }
32     inlet
33     {
34         type      zeroGradient;
35     }
36     top
37     {
38         type      zeroGradient;
39     }
40     bottom
41     {
42         type      zeroGradient;
43     }
44     outlet
45     {
46         type      fixedValue;
47         value      uniform 0;
48     }
49 }
50
51
52 // *****
```

Figura X: configurando arquivo “p”

A screenshot of a text editor window showing the configuration file 'u'. The window title is 'u' and the path is '~\Área de Trabalho\c18_cilindro_grid1\0'. The code is in C++ and defines a volVectorField. It sets dimensions to [0 1 -1 0 0 0], internal field to uniform (1 0 0), and boundary field with frontAndBack (empty), cylinder (fixedValue uniform (0 0 0)), inlet (fixedValue uniform (1 0 0)), top (slip), bottom (slip), and outlet (zeroGradient).

```
1 /*-----* C++ -*-----*/
2 =====
3 \ \ \ \ \ F i e l d      | OpenFOAM: The Open Source CFD Toolbox
4 \ \ \ \ \ O peration    | Website: https://openfoam.org
5 \ \ \ \ \ A n d         | Version: 8
6 \ \ \ \ \ M anipulation  |
7 /*-----*
8 FoamFile
9 {
10     version      2.0;
11     format       ascii;
12     class        volVectorField;
13     location     "0";
14     object       u;
15 }
16 // *****
17
18 dimensions      [0 1 -1 0 0 0];
19
20 internalField    uniform (1 0 0);
21
22 boundaryField
23 {
24     frontAndBack
25     {
26         type      empty;
27     }
28     cylinder
29     {
30         type      fixedValue;
31         value      uniform (0 0 0);
32     }
33     inlet
34     {
35         type      fixedValue;
36         value      uniform (1 0 0);
37     }
38     top
39     {
40         type      slip;
41     }
42     bottom
43     {
44         type      slip;
45     }
46     outlet
47     {
48         type      zeroGradient;
49     }
50 }
51
52
```

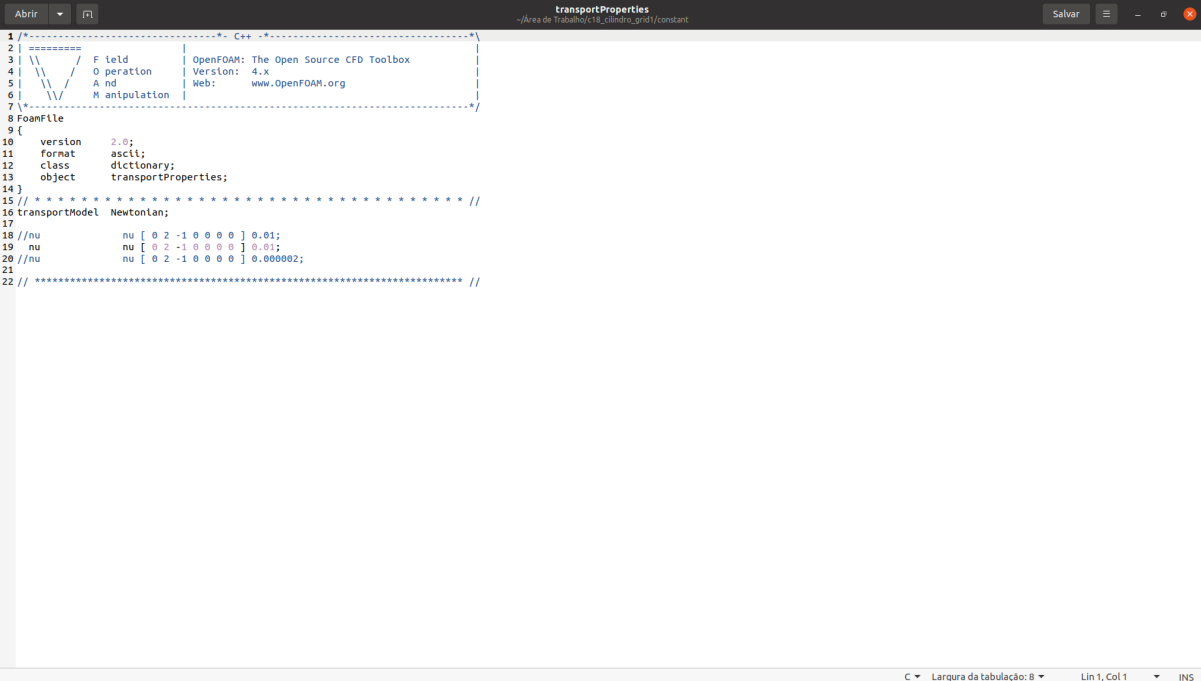
Figura X: configurando arquivo “u”

(EXPLICAR ALTERAÇÕES)

No documento "TransportProperties", o qual se localiza dentro da pasta “constant”, podemos alterar a viscosidade dinâmica do fluido. Alterando esse parâmetro, alteramos o coeficiente de Reynolds da simulação. Observe as imagens abaixo:

$$Re = \frac{\rho V L}{\mu}$$

Fórmula X: número de Reynolds



```

1 // ***** C++ *****
2 // =====
3 // \   \   F i e l d       OpenFOAM: The Open Source CFD Toolbox
4 //  \   /   O peration     Version: 4.x
5 //   \ /    A nd           Web: www.OpenFOAM.org
6 //    /     M anipulation
7 // =====
8 FoamFile
9 {
10     version      2.0;
11     format       ascii;
12     class        dictionary;
13     object       transportProperties;
14 }
15 // *****
16 transportModel  Newtonian;
17
18 // nu [ 0 2 -1 0 0 0 ] 0.01;
19 nu [ 0 2 -1 0 0 0 ] 0.01;
20 // nu [ 0 2 -1 0 0 0 ] 0.000002;
21
22 // *****

```

Figura X: configurando o arquivo “transportProperties”

Por fim, vá em system → ControlDict. Esse arquivo é responsável por controlar os parâmetros da simulação, como o tempo inicial e final, o intervalo de tempo em que a simulação vai guardar as informações em pastas, e o deltaT, intervalo de tempo em que serão resolvidas as equações. Quanto menor o writeInterval e o deltaT, maior será o detalhamento da simulação, demorando mais tempo para ser concluída. Vale ressaltar que é importante ajustar o deltaT para que o Número de Courant, o qual é uma condição necessária para a convergência enquanto se resolve numerosas equações diferenciais parciais numericamente (equações de Navier-Stokes), não assuma valores elevados e a simulação pare de rodar. Observe a foto abaixo:

```

1 /*===== C++ =====*/
2
3 // Field      OpenFOAM: The Open Source CFD Toolbox
4 // Operation  Version:  4.x
5 // And        Web:      www.OpenFOAM.org
6 // Manipulation
7
8 FoamFile
9 {
10     version      2.0; grupo vento
11     format        ascii;
12     class         dictionary;
13     object        controlDict;
14 }
15 // ***** //
16
17 application      pisoFoam;
18
19 startFrom        startTime;
20 //startFrom      latestTime;
21
22 startTime        0;
23
24 stopAt           endTime;
25
26 endTime          30;
27
28 deltaT           0.00010;
29 //deltaT         0.0001;
30 //deltaT         0.0005;
31 //deltaT         0.0010;
32 //deltaT         0.0015;
33 //deltaT         0.0020;
34 //deltaT         0.0025;
35 //deltaT         0.0030;
36
37 writeControl      runTime;
38 /*
39 adjustableRunTime
40 clockTime
41 cpuTime
42 runTime
43 timeStep
44 */
45 writeInterval     0.1
46
47 /*
48 secondaryWriteControl  cpuTime;
49 secondaryWriteInterval  1000;
50 secondaryPurgeWrite    1;
51 */
52
53 purgeWrite        0;
54
55

```

Figura X: configurando arquivo “controlDict”

Agora, após editar todas as pastas, vá no terminal e digite o comando “renumberMesh -overwrite” para ver se as edições foram feitas corretamente. Observe a seguir:

(IMAGEM)

Quando der tudo certo, coloque o comando “pisoFoam” e espere a simulação rodar por completo, algo que vai demorar um certo tempo. Para visualizá-la, digite “paraFoam” no terminal.

5.GNUPLOT

5.1 Visualização durante a simulação

Caso estejam sendo usados os tutoriais do wolfdynamics como base para simulações, é possível visualizar os coeficientes de arrasto e sustentação **durante** uma simulação usando, em uma nova aba do terminal, o comando:

gnuplot scripts0/plot_coeffs

Esse comando irá plotar os coeficientes a partir do aplicativo GNUPLOT, usando uma linha de comando pré-definida que está salva em um arquivo de texto na pasta *scripts0*. A linha de comando, com os comentários do significado de cada linha, é a seguinte:

```
set term x11 persist
```

```
set multiplot layout 1,2 // Plota 2 janelas
```

```
set size 1, 0.5 // Tamanho da primeira janela
```

```
set xlabel 'time' // Nome do eixo x
```

```
set ylabel 'cd' // Nome do eixo y
```

```
plot [0:][0:2] 'postProcessing/forceCoeffs_object/0/forceCoeffs.dat' u 1:3 w l  
notitle
```

// vai plotar um gráfico [plot] com o intervalo em x de 0 ao final dos dados ([0:]) e em y de 0 a 2 ([0:2]). Na sequência é dado o nome do arquivo onde os dados estão salvos e a localização desse arquivo a partir da pasta onde o terminal foi aberto, entre aspas. Posteriormente falamos quais colunas vamos usar desse arquivo (use 1 e 3 = u 1:3) e ainda pedimos para serem usadas linhas para ligar os pontos (w l = with lines) e sem título o gráfico.

```
set size 1, 0.5 //Tamanho da segunda janela
```

```
set origin 0, 0.5 //Onde começa a segunda janela
```

```
set xlabel 'time' // Nome do eixo x
```

```
set ylabel 'cl' // Nome do eixo y
```

```
plot [0:][-2:2] 'postProcessing/forceCoeffs_object/0/forceCoeffs.dat' u 1:4 w l  
notitle
```

// vai plotar um gráfico [plot] com o intervalo em x de 0 ao final dos dados ([0:]) e em y de -2 a 2 ([2:2]). Na sequência é dado o nome do arquivo onde os dados estão salvos e a localização desse arquivo a partir da pasta onde o terminal foi aberto, entre aspas. Posteriormente falamos quais colunas vamos usar desse arquivo (use 1 e 4 = u 1:3) e ainda pedimos para serem usadas linhas para ligar os pontos (w l = with lines) e sem título o gráfico.

```
unset multiplot
```

```
pause 2
```

```
reread
```

// Esse código é relido em intervalos de tempo para ler no arquivo os novos dados que estão sendo salvos na simulação.

Esse é um código pronto para plotar os gráficos com os dados gerados na simulação. Ele usa o seguinte arquivo, de onde são extraídos os dados:

postProcessing/forceCoeffs_object/0/forceCoeffs.dat

O formato do arquivo é o seguinte:

```
1# Force coefficients
2# liftDir      : (0.0000000e+00 1.0000000e+00 0.0000000e+00)
3# dragDir      : (1.0000000e+00 0.0000000e+00 0.0000000e+00)
4# pitchAxis    : (0.0000000e+00 0.0000000e+00 1.0000000e+00)
5# magUInf      : 1.0000000e+01
6# lRef         : 1.0000000e+00
7# Aref         : 5.0000000e+00
8# CofR         : (0.0000000e+00 0.0000000e+00 0.0000000e+00)
9# Time
10 0
11 0.00311504
12 0.0168375
13 0.0335074
14 0.0499992
15 0.0677534
16 0.0859235
17 0.103806
18 0.121402
19 0.13901
20 0.156734
21 0.174642
22 0.193051
23 0.211813
24 0.231136
25 0.250749
26 0.270575
```

	Cm	Cd	Cl	Cl(f)	Cl(r)
10 0	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00	0.0000000e+00
11 0.00311504	9.54994614e+00	-1.73605258e+00	3.03660316e-04	9.55009797e+00	-9.54979431e+00
12 0.0168375	-2.22869688e+00	4.04897767e-01	-3.20416571e-04	-2.22885709e+00	2.22853667e+00
13 0.0335074	-3.06718375e+00	5.57400058e-01	-2.69145094e-04	-3.06731832e+00	3.06704918e+00
14 0.0499992	-3.57686775e+00	6.50250015e-01	-8.92685760e-05	-3.57691238e+00	3.57682311e+00
15 0.0677534	-4.19257180e+00	7.62200219e-01	-8.53768635e-05	-4.19261449e+00	4.19252912e+00
16 0.0859235	-4.81101755e+00	8.74746998e-01	1.68048733e-05	-4.81100915e+00	4.81102596e+00
17 0.103806	-5.42892448e+00	9.87193403e-01	1.16566137e-04	-5.42886620e+00	5.42898276e+00
18 0.121402	-5.98998012e+00	1.08924717e+00	1.59303020e-04	-5.98990046e+00	5.99005977e+00
19 0.13901	-6.44304423e+00	1.17166104e+00	1.98123674e-04	-6.44294517e+00	6.44314329e+00
20 0.156734	-6.62569714e+00	1.20503612e+00	3.64311967e-04	-6.62551498e+00	6.62587929e+00
21 0.174642	-6.59360678e+00	1.19930856e+00	4.72145538e-04	-6.59337071e+00	6.59384285e+00
22 0.193051	-6.44186539e+00	1.17171797e+00	4.70759859e-04	-6.44163001e+00	6.44210077e+00
23 0.211813	-6.27758120e+00	1.14188285e+00	5.05065635e-04	-6.27732866e+00	6.27783373e+00
24 0.231136	-6.13563735e+00	1.11620136e+00	6.31673716e-04	-6.13532152e+00	6.13595319e+00
25 0.250749	-6.01308478e+00	1.09393571e+00	6.47903492e-04	-6.01276083e+00	6.01340874e+00
26 0.270575	-5.89992482e+00	1.07347987e+00	7.66460908e-04	-5.89954159e+00	5.90030806e+00

Em uma primeira parte temos alguns inputs que são usados para calcular os coeficientes: direção da sustentação, arrasto, eixo para o coeficiente de momento, velocidade de inlet, comprimento de referência para coeficiente de momento, área de referência para cálculo do arrasto e sustentação e coeficiente de referência(?). Esses valores podem ser alterados, antes de começar a simulação, no arquivo *controlDict* na pasta *System*, a partir da linha 229, no item *forceCoeffs_object*.

```
Abrir  controlDict
~/OpenFOAM/cfd05-9/run/BRENO/006/system

228
229 forceCoeffs_object
230 {
231     // rhoInf - reference density
232     // CofR - Centre of rotation
233     // dragDir - Direction of drag coefficient
234     // liftDir - Direction of lift coefficient
235     // pitchAxis - Pitching moment axis
236     // magUInf - free stream velocity magnitude
237     // lRef - reference length
238     // Aref - reference area
239     type forceCoeffs;
240     functionObjectLibs ("libforces.so");
241     //patches ("body1" "body2" "body3");
242     patches ("obstacle");
243
244     pName p;
245     Uname U;
246     rho rhoInf;
247     rhoInf 1.0;
248
249     /// Dump to file
250     log true;
251
252     CofR (0.0 0 0);
253     liftDir (0 1 0);
254     dragDir (1 0 0);
255     pitchAxis (0 0 1);
256     magUInf 10.0;
257     lRef 1.0; // reference length
258     Aref 5.0; // reference area 1 for 2d
259
260     writeControl timeStep;
261     writeInterval 10;
262 }
263
264 ///////////////////////////////////////////////////////////////////
265
```

Na segunda parte do arquivo são apresentados os valores obtidos na simulação. São essas colunas que selecionamos quando mandamos plotar um gráfico no GNUPLOT. Usualmente colocamos no eixo X a coluna 1, que tem os valores do tempo, e no eixo y o coeficiente de arrasto (coluna 3) ou o de sustentação (coluna 4), para isso usando o comando *u 1:3* ou *u 1:4*, respectivamente.

5.1 Visualização depois da simulação, sem utilizar comando do scripts0

Para simplificar ir até a pasta:

postProcessing/forceCoeffs_object/0/

Abrir o terminal.

Abrir o GNUPLOT.

```
cfd05@cfd05-XPS-8920: ~/OpenFOAM/cfd05-9/run/BRENO/006/postProcessing/forceCo...
cfd05@cfd05-XPS-8920:~/OpenFOAM/cfd05-9/run/BRENO/006/postProcessing/forceCoeffs_object/0$ gnuplot

G N U P L O T
Version 5.2 patchlevel 8   last modified 2019-12-01

Copyright (C) 1986-1993, 1998, 2004, 2007-2019
Thomas Williams, Colin Kelley and many others

gnuplot home:      http://www.gnuplot.info
faq, bugs, etc:    type "help FAQ"
immediate help:    type "help" (plot window: hit 'h')

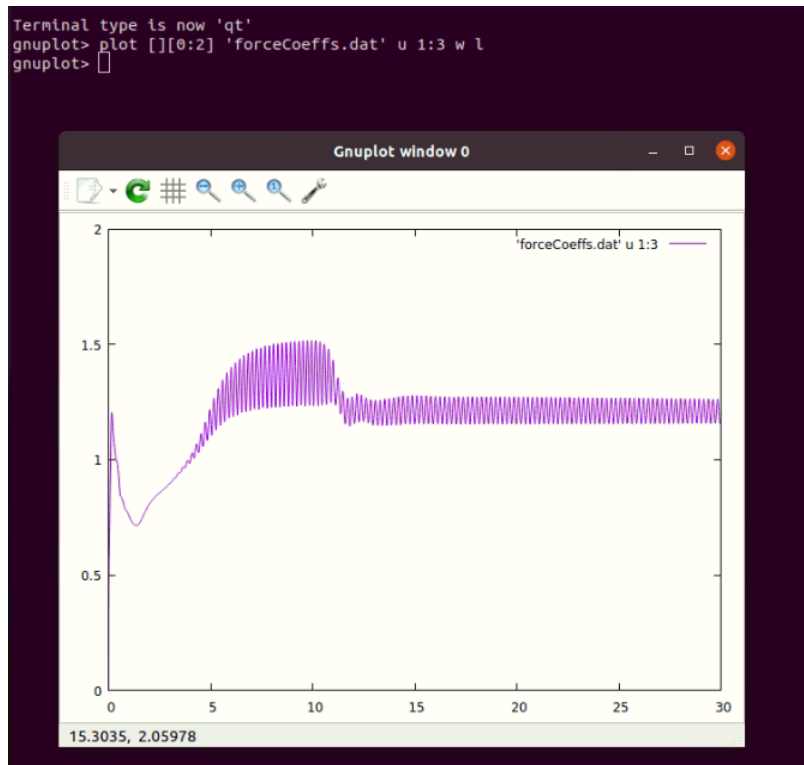
Terminal type is now 'qt'
gnuplot>
```

Serão apresentados dois comandos úteis: *plot* e *stats*

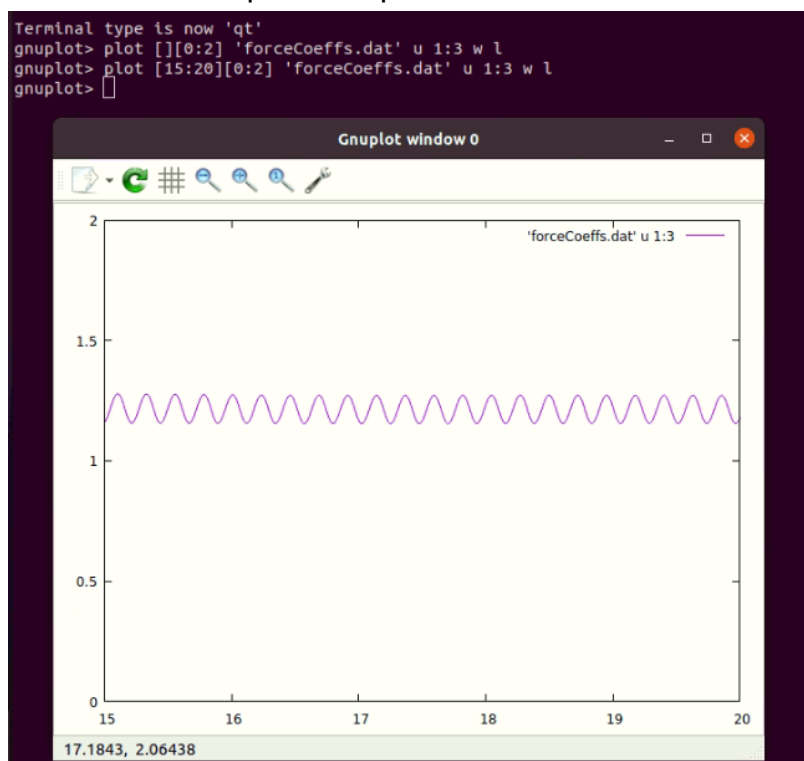
A forma de dar entrada no plot para obter um gráfico já foi apresentada:

```
plot [][0:2] 'forceCoeffs.dat' u 1:3 w l
```

Nesse caso, por exemplo, não foi dada nenhuma restrição para o eixo x e não foi indicado o caminho do arquivo pois o terminal foi aberto na pasta onde o arquivo está. O resultado é o seguinte:



Mudando o intervalo do eixo x por exemplo:



Para explorar mais possibilidades do aplicativo ver: <http://www.gnuplot.info/documentation.html>.

Outra função útil é a *stats*, que tem a mesma estrutura, devendo ser omitida a parte gráfica. Ela retornará estatísticas do intervalo selecionado:

stats [::] 'forceCoeffs.dat' u 1:3

```
gnuplot> stats [::] 'forceCoeffs.dat' u 1:3

* FILE:
Records:      1601
Out of range: 0
Invalid:      0
Column headers: 0
Blank:        0
Data Blocks:  1

* COLUMNS:
Mean:          15.3858      1.1981
Std Dev:       8.4961      0.1691
Sample StdDev: 8.4988      0.1692
Skewness:      -0.0335     -4.2591
Kurtosis:      1.8418      61.5897
Avg Dev:       7.3330      0.1003
Sum:           24632.6084   1918.0787
Sum Sq.:       494558.9607  2343.7512

Mean Err.:     0.2123      0.0042
Std Dev Err.:  0.1501      0.0030
Skewness Err.: 0.0612      0.0612
Kurtosis Err.: 0.1224      0.1224

Minimum:       0.0000 [ 0]   -1.7361 [ 1]
Maximum:      29.9911 [1600]  1.5167 [ 495]
Quartile:      8.2317      1.1637
Median:        15.4224      1.2198
Quartile:      22.7186      1.2642

Linear Model:   y = 0.005545 x + 1.113
Slope:         0.005545 +- 0.0004781
Intercept:     1.113 +- 0.008403
Correlation:    r = 0.2786
Sum xy:        3.015e+04

gnuplot> 
```

Essa função é útil, por exemplo, para avaliar a média dos coeficientes quando estes atingem o regime estacionário. No caso a seguir, na segunda coluna dos outputs, temos a média do coeficiente de arrasto no intervalo [20:30]s

stats [20:] 'forceCoeffs.dat' u 1:3

```
gnuplot> stats [20:] 'forceCoeffs.dat' u 1:3

* FILE:
Records:      550
Out of range: 1051
Invalid:      0
Column headers: 0
Blank:        0
Data Blocks:  1

* COLUMNS:
Mean:          25.0000      1.2130
Std Dev:       2.8879      0.0398
Sample StdDev: 2.8906      0.0398
Skewness:      -0.0014     -0.0396
Kurtosis:      1.8000      1.5149
Avg Dev:       2.5010      0.0358
Sum:           13750.0122   667.1529
Sum Sq.:       348337.6757  810.1309

Mean Err.:     0.1231      0.0017
Std Dev Err.:  0.0871      0.0012
Skewness Err.: 0.1044      0.1044
Kurtosis Err.: 0.2089      0.2089

Minimum:       20.0045 [ 0]   1.1544 [ 60]
Maximum:      29.9911 [549]  1.2718 [ 29]
Quartile:      22.5006      1.1738
Median:        25.0032      1.2136
Quartile:      27.5031      1.2521

Linear Model:   y = -0.0005622 x + 1.227
Slope:         -0.0005622 +- 0.0005881
Intercept:     1.227 +- 0.0148
Correlation:    r = -0.0408
Sum xy:        1.668e+04
```

Fontes:

<https://www.openfoam.com/documentation/user-guide/2-openfoam-cases/2-2-basic-inputoutput-file-format>

<https://aerospacemodel.paginas.ufsc.br/tutoriais-e-materiais/tutoriais-em-cfd/tutorial-de-geracao-de-malha-em-bocal-axissimetrico/>

Sites úteis

Tutoriais:

<http://www.wolfdynamics.com/tutorials.html?id=126>

Plotar gráficos durante ou após a simulação com GNUPLOT:

<http://www.gnuplot.info/documentation.html>