Universidade Federal de Uberlândia
Laboratório de Mecânica dos fluidos
Programa de Pós-Graduação em Engenharia Mecânica

MFSIM:
MANUAL DO USUÁRIO

Uberlândia
June 13, 2023

Contents

1 Conhecendo o MFSim

1.1 Introdução

O MFSim é um software de simulação de fluidodinâmica (CFD) desenvolvido continuamente desde 2010, pelo Laboratório de Mecânica de Fluídos do programa de pós-graduação da Faculdade de Engenharia Mecânica (FEMEC) da Universidade Federal de Uberlândia.

1.2 Funcionalidades

As Funcionalidades do MFSim são:

- Discretização de volumes finitos, tridimensionais, *all Mach* de forma totalmente paralela (decomposição multi-domínio, MPI);
- Acoplamentos pressão-velocidade: SIMPLEE, SIMPLEC, Fractional Step;
- Discretização de espaço e tempo de segunda ordem em forma conservativa e não conservativa;
- Discretização implícita e semi-implícita, totalmente paramétrica e variável de *time step*;
- Métodos de discretização espacial: CDS, Up-wind, Weno de 5ł ordem, Cubista, Skew, Barton, QUICK e QUICKhayase;
- Modelos de turbulência: LES, DES, URANS, SAS;
- Fronteira imersa com geometrias deformáveis FSI (*Multi-direct forcing, Virtual Physical Model, Ghost fluid Method*);
- Modelos de combustão e transporte PDF: EBU, EDM, EDC como método euleriano;
- Mecanismos de combustão simplificados e detalhados via Cantera;
- Simulação realista de turbulência no fluxo para LES e DNS;
- Simulação multifase: *VoF, Front-Tracking, Level-set*;
- Modelos de evaporação tanto para fases discretas e contínuas;
- Abordagem Euler-Lagrange para transporte de partículas gás-sólido e gás-líquido nos regimes densos e diluído;
- Modelos de Erosão-Corrosão para FAC;
- Refinamento de simulação via malha adaptativa dinâmica;
- Execução particionada de simulações;
- Iniciliazação de simulações a partir de dados de outras simulações, com e sem alteração na decomposição de domínios;
- Suporte a Docker e Singularity

1.3 Pré-requisitos

É altamente recomendável que já tenha lido pelo menos os capítulos sobre "Filosofia de uso" e "Compilação e Instalação" do Manual de Instalação do MFSim. Também é requerido que já possua o MFSim instalado na sua máquina. Caso não possua o MFSim ainda ou não tenha lido os capítulos recomendados, acesse o site oficial do MFSim `https://www.mflab.mecanica.ufu.br/mfsim/` e obtenha o manual de instalação.

2 Setando Caso

Nesta seção veremos como setar casos no MFSim. Começaremos apresentando a estrutura de pastas, assim como para que serve cada uma e então passaremos para a apresentação dos arquivos de input e malha, que é onde você setará o caso. Ao final, apresentaremos um resumo para facilitar a memorização do que precisa para setar um caso no MFSim.

2.1 Estrutura de Pastas

2.1.1 A questão das pastas

No Manual de Instalação, o capítulo sobre a "Filosofia de uso" introduziu a noção básica sobre como o MFSim utiliza as pastas. Aqui não iremos repetir o que está lá, portanto se não leu este capítulo do Manual de Instalação, recomendo que o faça **antes** de prosseguir aqui.

Continuando, o MFSim utiliza uma pasta para compilação e outra para instalação. Para usar o MFSim, ele deve estar previamente instalado, então assumimos aqui que você já conseguiu instalar o MFSim, conforme indicado no Manual de Instalação. Feito isso, precisará criar 3 pastas, onde setará seu caso. **As 3 pastas são obrigatórias**, ou seja, mesmo que seu caso não utilize alguma das pastas, você terá que necessariamente a criar.

Então, a **primeira coisa a fazer** para setar um caso é criar uma pasta que conterá as 3 pastas de setagem de caso para o MFSim. **Como exemplificação**, vamos assumir que esta pasta se chamará `/home/USER/caso`. Alguns detalhes importantes sobre a pasta do caso:

- 1 - Ela pode estar em qualquer lugar do seu sistema de arquivos, desde que o MFSim tenha permissões de leitura nela
- 2 - É altamente recomendado que você não a coloque dentro da pasta com o fonte do MFSim
- 3 - Ela precisa, **obrigatoriamente**, conter as 3 pastas de setagem de caso.

Agora vamos criar as 3 pastas de setagem de caso: `input`, `geo` e `probes`. Considerando o exemplo teremos algo como:

```
/home/USER/caso
        /input
        /geo
        /probes
```

Com as pastas criadas, vamos entender para o que cada uma serve e o que cada uma contém:

- **input**
  Aqui estão os arquivos necessários para definir os dados de entrada do problema a ser estudado, ou seja, é nessa pasta que estão os arquivos que modificará para setar o seu caso no MFSim. Cada arquivo será abordado de forma detalhada posteriormente. Por agora, veremos uma listagem resumida dos arquivos e para que cada um deles serve:
  - *bc_functions.f90*: define as condições de contorno e iniciais dinâmicas para o caso;
  - *composedBC.amr3d*: ativa e configura as condições de contorno compostas, quando numa mesma parede é preciso aplicar condições de dirichlet e neumann;
  - *compressible.amr3d*: ativa a modelagem para escoamentos compressíveis e define parâmetros utilizados em tal modelagem;
  - *corner.amr3d*: define blocos de refinamento iniciais;
  - *dpm.amr3d*: define parâmetros utilizados na modelagem de escoamentos fluido-sólido ou na modelagem de gotas e bolhas como pontos lagrangianos;
  - *energy.amr3d*: define parâmetros utilizados quando a equação da energia é considerada na modelagem do problema;
  - *externalBC.amr3d*: configura o uso da execução particionada de simulação;
  - *ib.amr3d*: define parâmetros relacionados à modelagem de corpos imersos, ou seja, ao uso de fronteira imersa;
  - *input_file*: arquivo utilizado apenas para definir a quais arquivos da pasta *input* devem ser lidos pelo MFSim. **Esse arquivo não deve ser modificado em montagens de simulações**;
  - *input.amr3d*: define condições gerais da simulação, ativa módulos e integrações, configura propriedades e parâmetros específicos do problema, tais como definição de condições de contorno para velocidade e pressão e solvers;
  - *postproc.amr3d*: define quais campos do escoamento serão salvos para posterior visualização em pós-processamento;
  - *probes.amr3d*: arquivo para posicionamento de sondas em escoamentos com bolhas;
  - *probesSTL.amr3d*: define parâmetros relacionado ao uso de sondas para mapear propriedades do escoamento;

- *reaction.amr3d*: define parâmetros e propriedades para escoamentos reativos;
- *refinement_region.amr3d*: define blocos de refinamento que serão mantidos ao longo de toda a simulação. Para que o arquivo seja lido é preciso acionar o critério de remalhagem número 6;
- *scalar.amr3d*: define parâmetros e propriedades para as equações do escalar passivo, da variância do escalar e de concentrações;
- *structure.amr3d*: define parâmetros e propriedades relacionados ao uso de modelagem fluido-estrutura;
- *turbulence.amr3d*: define parâmetros e propriedades relacionados aos modelos de turbulência implementados no código;
- *vof.amr3d*: define parâmetros e propriedades relacionados ao método *Volume of Fluid*, utilizado na simulação de escoamentos multifásicos fluido-fluido;

- **geo**
  Nesta pasta são dispostos os arquivos da malha não estruturada usada nos métodos IB ou FT. Note que mesmo que seu caso não utilize fronteira imersa ou front-tracking, você ainda precisará criar essa pasta.

- **probes**
  Nesta pasta estão os arquivos que configuram as sondas a serem utilizadas durante a simualação. Assim como a pasta geo, você precisará obrigatoriamente, criar esta pasta.

2.2 Arquivos de Input

Nessa seção serão apresentados detalhes relativos aos arquivos contidos na pasta *input*, onde são definidas as principais propriedades do escoamento a ser simulado bem como parâmetros relacionados a solvers e construção da malha.

2.2.1 bc_functions.f90

Nesse arquivo são definidos os valores de condição de contorno e condição inicial para cada propriedade. **Somente as propriedades a serem resolvidas precisam ter as condições especificadas**.

Existem algumas situações específicas que podem exigir definições de propriedades não usuais nesse arquivo. O caso específico mais utilizado é a solução manufaturada, usada no processo de verificação computacional do código. Nesse processo é necessário que se defina valores para a propriedade e suas derivadas, além dos RHS's. Uma cópia do arquivo *bc_funtions.f90* utilizado na solução manufaturada pode ser encontrado em *tests/manuf/input/bc_functions.f90*. Um caso de solução manufaturada detalhado também é apresentado na seção 4.1.

O *bc_funtions.f90* segue um esquema padrão para auxiliar na identificação de qual propriedade deve ser definida ou modificada. Esse esquema consiste em uma sequência na seguinte ordem:

1. Valor exato da propriedade;
2. Valor da derivada da propriedade na direção x;
3. Valor da derivada da propriedade na direção y;
4. Valor da derivada da propriedade na direção z;
5. Valor da derivada temporal da propriedade;

Grande parte das propriedades segue esse esquema. Por exemplo, a definição para o valor da pressão está contida na função `exact_p`, o valor da derivada da pressão na direção x é definido na função `dpdx`, da derivada da pressão na direção y é definido na função `dpdy`, da derivada da pressão na direção zx é definido na função `dpdz` e da derivada temporal da pressão é definido na função `dpdt`.

O mesmo esquema é adotado para os três componentes do campo de velocidade (u, v e w), para o escalar (sca), para a variância (var), para as concentrações (yk), para a primeira equação dos modelos URANS (tur1), para a segunda equação dos modelos URANS (tur2), para a viscosidade dinâmica turbulenta (muturb) e para a temperatura (temp).

Exitem também algumas propriedades das quais somente é possível definir o valor da propriedade, não sendo possível definir o valor das derivadas. Esse é o caso da massa específica (rho), da viscosidade dinâmica molecular (mu), da fração volumétrica (vof), calor específico (cepe) e condutividade térmica (cond).

Uma observação muito importante a ser colocada é que, em grande parte dos casos, o valor de todas as derivadas é definido como nulo, levando à condição de contorno chamada Neumann homogênea. Por isso, **é muito importante se atentar ao fato de manter os valores corretos para essas propriedades**.

A definição de valores para as condições inicial e de contorno do problema não são tão complexas. A seguir serão apresentados alguns exemplos, aplicados à condição de velocidade, porém, as demais propriedades seguem a mesma esquemática.

2.2.1.1 Exemplo 1

No primeiro exemplo vamos supor um caso em que exista um perfil de velocidade constante de 2 m/s no instante inicial em todo o domínio e que o domínio tenha apenas a face x_{min} do tipo Dirichlet. Nessa face o fluido também irá entrar no domínio com um perfil de velocidade constante de 2 m/s. A definição de valor para velocidade no arquivo é realizada da seguinte forma:

```
1|  function exact_u(t,x,y,z)
2|      implicit none
3|      double precision :: t, x, y, z
4|      double precision :: exact_u
5|
6|      exact_u = 2.0d0
7|
8|  end function exact_u
```

Essa definição é suficiente para esse exemplo pois somente as faces do tipo Dirichlet irão acessar a função `exact_u(t,x,y,z)`.

2.2.1.2 Exemplo 2

No segundo exemplo vamos supor um caso muito similar ao exemplo 1, porém com as faces y_{min} e y_{max} definidas como parede. Ou seja, suponhamos que exista um perfil de velocidade constante de 2 m/s no instante inicial em todo o domínio e que o domínio tenha fluido entrando apenas na face x_{min}. Nessa face o fluido também irá entrar no domínio com um perfil de velocidade constante de 2 m/s. Além disso, como dito anteriormente, as faces y_{min} e y_{max} serão definidas como parede, ou seja, devem ser impostas condições de não deslizamento nessas duas faces, que quer dizer velocidade nula. A definição de valor para velocidade no arquivo é realizada da seguinte forma:

```
1|  function exact_u(t,x,y,z)
2|      implicit none
3|      double precision :: t, x, y, z
4|      double precision :: exact_u
5|
6|      exact_u = 2.0d0
7|
8|      !!! Definição da condição de não deslizamento
9|      if(y<=ga2 .or. y>=gb2)  exact_u = 0.0d0
10|
11|  end function exact_u
```

Conforme pode ser observado, as condições de contorno são baseadas nos limites do domínio `[a1,b1] X [a2,b2] X [a3,b3]`.

2.2.1.3 Exemplo 3

O terceiro exemplo consiste em um jato plano entrando na face x_{min} a 5 m/s. A condição inicial em todo o domínio é velocidade nula. Supondo que o jato tenha $0,013$ m de raio e esteja centrado em $yc = zc = 0,018$ no plano de entrada em x, a inicialização é realizada como:

```
1|  function exact_u(t,x,y,z)
2|      implicit none
3|      double precision :: t, x, y, z
4|      double precision :: exact_u
5|      double precision :: dist, ycent, zcent, raio, point
6|
7|      exact_u = 0.0d0
8|
9|      raio = 0.013d0
10|      ycent = 0.018d0
11|      zcent = 0.018d0
12|
13|      point = sqrt((y-ycent)**2.0d0 + (z-zcent)**2.0d0 )
14|      dist = raio - point
15|
16|      if(dist > 0.0d0 .and. x<=ga1)then
17|          exact_u = 5.0d0
18|      end if
19|
20|  end function exact_u
21|
```

2.2.1.4 Incluindo funções

Dependendo do caso a ser simulado pode ser necessário que uma função de condição de contorno dependa de outra função de condição de contorno. Há básicamente duas maneiras de se implementar isso:

Usando a função da qual se depende como ponteiro

Suponha que a condição de contorno para o valor da derivada da temperatura na direção X dependa do resultado da função `fc_temp`, que também está definida no arquivo *bc_funtions.f90*. Então precisamos que a função `dtempdx` precisa chamar `fc_temp`. Para isso, colocaremos `fc_temp` como um ponteiro dentro de `dtempdx`.

```
1|  function fc_temp(x, y, z)
2|      implicit none
3|      double precision :: x, y, z, fc_temp
4|
5|      fc_temp = 1.0d0
6|  end function fc_temp
7|
8|  function dtempdx(t,x,y,z)
9|      implicit none
10|      double precision :: t, x, y, z
11|      double precision :: dtempdx
12|      double precision, external :: fc_temp
13|
14|      dtempdx = fc_temp(x, y, z)
15|  end function dtempdx
16|
```

Usando a função da qual se depende como interface
Outra forma de implementar é criando uma interface para `fc_temp` dentro da `dtempdx`

```
1|  function fc_temp(x, y, z)
2|      implicit none
3|      double precision :: x, y, z, fc_temp
4|
5|      fc_temp = 1.0d0
6|  end function fc_temp
7|
8|  function dtempdx(t,x,y,z)
9|      implicit none
10|      double precision :: t, x, y, z
11|      double precision :: dtempdx
12|      double precision, external :: fc_temp
13|      interface
14|          function fc_temp(x, y, z)
15|              implicit none
16|              double precision, intent(in) :: x, y, z
17|              double precision :: fc_temp
18|          end function fc_temp
19|      end interface
20|
21|      dtempdx = fc_temp(x, y, z)
22|  end function dtempdx
23|
```

2.2.1.5 Incluindo objetos do MFSim

Pode também ser necessário que uma função de contorno utilize dados do MFSim em seu código. Nesse caso você incluir na função o módulo onde está o dado requerido. Ex: a condição de contorno da pressão no manuf precisa do valores pi, w1, w2, w3 e w4 que estão definidos no módulo `eul_functions`. Logo, a função que implementa a condição de contorno precisa incluir esse módulo:

```
1|  function exact_p(t,x,y,z)
2|      use eul_functions   ! <- observe que o módulo foi incluído aqui
3|      implicit none
4|      double precision :: t, x, y, z
5|      double precision :: exact_p
6|
7|      exact_p = cos(pi*w3*z+pi*w2*y+pi*w1*x+t*w4)
8|
9|  end function exact_p
```

Você pode incluir quando módulos precisar nas funções de condições de contorno.

2.2.1.6 Lista das funções e seus valores default

Função	Parâmetros	Valor padrão
exact_p	tempo e coordenadas (t,x,y,z)	0
dpdx	tempo e coordenadas (t,x,y,z)	0
dpdy	tempo e coordenadas (t,x,y,z)	0
dpdz	tempo e coordenadas (t,x,y,z)	0
dpdt	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
exact_u	tempo e coordenadas (t,x,y,z)	0
dudx	tempo e coordenadas (t,x,y,z)	0
dudy	tempo e coordenadas (t,x,y,z)	0
dudz	tempo e coordenadas (t,x,y,z)	0
dudt	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
exact_v	tempo e coordenadas (t,x,y,z)	0
dvdx	tempo e coordenadas (t,x,y,z)	0
dvdy	tempo e coordenadas (t,x,y,z)	0
dvdz	tempo e coordenadas (t,x,y,z)	0
dvdt	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
exact_w	tempo e coordenadas (t,x,y,z)	0
dwdx	tempo e coordenadas (t,x,y,z)	0
dwdy	tempo e coordenadas (t,x,y,z)	0
dwdz	tempo e coordenadas (t,x,y,z)	0
dwdt	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
exact_rho	tempo e coordenadas (t,x,y,z)	0
drhodx	tempo e coordenadas (t,x,y,z)	0
drhody	tempo e coordenadas (t,x,y,z)	0
drhodz	tempo e coordenadas (t,x,y,z)	0
drhodt	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
exact_sca	tempo e coordenadas (t,x,y,z)	0
dscadx	tempo e coordenadas (t,x,y,z)	0
dscady	tempo e coordenadas (t,x,y,z)	0
dscadz	tempo e coordenadas (t,x,y,z)	0
dscadt	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
exact_var	tempo e coordenadas (t,x,y,z)	0
dvardx	tempo e coordenadas (t,x,y,z)	0
dvardy	tempo e coordenadas (t,x,y,z)	0
dvardz	tempo e coordenadas (t,x,y,z)	0
dvardt	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
exact_temp	tempo e coordenadas (t,x,y,z)	0
dtempdx	tempo e coordenadas (t,x,y,z)	0
dtempdy	tempo e coordenadas (t,x,y,z)	0
dtempdz	tempo e coordenadas (t,x,y,z)	0
dtempdt	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
exact_h	tempo e coordenadas (t,x,y,z)	0
dhdx	tempo e coordenadas (t,x,y,z)	0
dhdy	tempo e coordenadas (t,x,y,z)	0
dhdz	tempo e coordenadas (t,x,y,z)	0
dhdt	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Va
exact_yk	tempo e coordenadas (t,x,y,z)	Constante: Enviro
dykdx	tempo e coordenadas (t,x,y,z)	
dykdy	tempo e coordenadas (t,x,y,z)	
dykdz	tempo e coordenadas (t,x,y,z)	
dykdt	tempo e coordenadas (t,x,y,z)	

Função	Parâmetros	Valor padrão
exact_tur1	tempo e coordenadas (t,x,y,z)	1.0d-8
dtur1dx	tempo e coordenadas (t,x,y,z)	0
dtur1dy	tempo e coordenadas (t,x,y,z)	0
dtur1dz	tempo e coordenadas (t,x,y,z)	0
dtur1dt	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
exact_tur2	tempo e coordenadas (t,x,y,z)	1.0d-2
dtur2dx	tempo e coordenadas (t,x,y,z)	0
dtur2dy	tempo e coordenadas (t,x,y,z)	0
dtur2dz	tempo e coordenadas (t,x,y,z)	0
dtur2dt	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
exact_vof	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
dist_vof	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
omega_term	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
exact_mu	tempo e coordenadas (t,x,y,z)	Constante: visc_phase_c

Função	Parâmetros	Valor padrão
exact_cepe	tempo e coordenadas (t,x,y,z)	Constante: cp_phase_c

Função	Parâmetros	Valor padrão
exact_cond	tempo e coordenadas (t,x,y,z)	Constante: k_phase_c

Função	Parâmetros	Valor padrão
omega_term	tempo e coordenadas (t,x,y,z)	Constante: diffusivitycoef

Função	Parâmetros	Valor padrão
omega_term_var	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
exact_muturb	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
dmuturbdx	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
dmuturbdy	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
dmuturbdz	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
dmuturbdt	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
rhs_values	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
rhssca_values	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
rhsvar_values	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
omega_m_term	tempo e coordenadas (t,x,y,z)	0

Função	Parâmetros	Valor padrão
rhsvel_uvalues	tempo, coordenadas e derivadas (t,dt,x,y,z,dx,dy,dz)	0

Função	Parâmetros	Valor padrão
rhsvel_vvalues	tempo, coordenadas e derivadas (t,dt,x,y,z,dx,dy,dz)	0

Função	Parâmetros	Valor padrão
rhsvel_wvalues	tempo, coordenadas e derivadas (t,dt,x,y,z,dx,dy,dz)	0

Função	Parâmetros	Valor padrão
rhs_mass	tempo, coordenadas e derivadas (t,dt,x,y,z,dx,dy,dz)	0

Função	Parâmetros	Valor padrão
iblocation	coordenadas iniciais e finais (xib, xi, xn, yib, yi, yn, zib, zi, zn)	Verdadeiro ou False

2.2.2 composedBC.amr3d

```
1|  .false.     !composed boundary condition
2|  !BOUNDARY CONDITION FOR AN ESPCIFIC REGION
3|  !WEST FACE
4|  1                 !NUMBER OF SPECIFIC REGIONS AT WEST FACE
5|  2                 !KIND= SAQUARE (1) OR CIRCLE (2) OR STL FILE (3)
6|  0.025 0.0d0 0.0325d0 0.0325d0 !(SQUARE: XI, XN, YI, YN, ZI, ZN; CIRCLE: RAIO, XC, YC, ZC; STL_FILE:path and name of stl files)
7|  0                 !0 - none , 1 - white noise, 2 - smirnov noise
8|  0.0 0.0 0.0       !noise percentagen at u, v, w
9|  0.0 1.0 0.0       !ALFA BETA GAMMA FOR PRESSURE
10|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR U VELOCITY
11|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR V VELOCITY
12|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR W VELOCITY
13|  0.0 1.0 0.0       !ALFA BETA GAMMA FOR VOF
14|  0.0 1.0 0.0       !ALFA BETA GAMMA FOR SCALAR
15|  0.0 1.0 0.0       !ALFA BETA GAMMA FOR VARIANCE
16|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TEMPERATURE
17|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR Enthalpy
18|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TURBULENCE EQUATION 1
19|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TURBULENCE EQUATION 2
20|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR MASS FRAC
21|  0.0 1.0 0.0       !ALFA BETA GAMMA FOR DENSITY
22|
23|  !EAST FACE
24|  0                 !NUMBER OF SPECIFIC REGIONS AT EAST FACE
25|  2                 !KIND= SAQUARE (1) OR CIRCLE (2) OR STL FILE (3)
26|  0.0381d0 1.024d0 3.1142d0 0.128d0  !(SQUARE: XI, XN, YI, YN, ZI, ZN; CIRCLE: RAIO, XC, YC, ZC; STL_FILE:path and name of stl files)
27|  0                 !0 - none , 1 - white noise, 2 - smirnov noise
28|  1.0 0.0 0.0       !noise percentagen at u, v, w
29|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR PRESSURE
30|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR U VELOCITY
31|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR V VELOCITY
32|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR W VELOCITY
33|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR VOF
34|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR SCALAR
35|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR VARIANCE
36|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TEMPERATURE
37|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR Enthalpy
38|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TURBULENCE EQUATION 1
39|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TURBULENCE EQUATION 2
40|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR MASS FRAC
41|  0.0 1.0 0.0       !ALFA BETA GAMMA FOR DENSITY
42|
43|  !SOUTH FACE
44|  0                !NUMBER OF SPECIFIC REGIONS AT SOUTH FACE
45|  2                 !KIND= SQUARE (1) OR CIRCLE(2)
46|  0.0381 0.1861d0 1.28d0  0.128d0 !(SQUARE: XI, XN, YI, YN, ZI, ZN ; CIRCLE: RAIO, XC, YC, ZC; STL_FILE:path and name of stl files)
47|  0                 !0 - none , 1 - white noise, 2 - smirnov noise
48|  1.0 0.0 0.0       !noise percentagen at u, v, w
49|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR PRESSURE
50|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR U VELOCITY
51|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR V VELOCITY
52|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR W VELOCITY
53|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR VOF
54|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR SCALAR
55|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR VARIANCE
56|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TEMPERATURE
57|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR Enthalpy
58|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TURBULENCE EQUATION 1
59|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TURBULENCE EQUATION 2
60|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR MASS FRAC
61|  0.0 1.0 0.0       !ALFA BETA GAMMA FOR DENSITY
62|
63|  !NORTH FACE
64|  0                !NUMBER OF SPECIFIC REGIONS AT SOUTH FACE
65|  1                 !KIND= SQUARE (1) OR CIRCLE(2)
66|  0.0d0 440.0d-3 20.0d-3 20.0d-3 0.0d0 80.0d-5    !(SQUARE: XI, XN, YI, YN, ZI, ZN ; CIRCLE: RAIO, XC, YC, ZC; STL_FILE:path and name of stl files)
67|  0                 !0 - none , 1 - white noise, 2 - smirnov noise
68|  1.0 0.0 0.0       !noise percentagen at u, v, w
69|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR PRESSURE
70|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR U VELOCITY
71|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR V VELOCITY
72|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR W VELOCITY
73|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR VOF
74|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR SCALAR
75|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR VARIANCE
76|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TEMPERATURE
77|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR Enthalpy
78|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TURBULENCE EQUATION 1
79|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TURBULENCE EQUATION 2
80|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR MASS FRAC
81|  0.0 1.0 0.0       !ALFA BETA GAMMA FOR DENSITY
82|
83|
84|  !BOTTOM FACE
85|  0                 !NUMBER OF SPECIFIC REGIONS AT BOTTOM FACE
86|  1                 !KIND= SQUARE (1) OR CIRCLE(2)
87|  0.0d0 440.0d-3 0.0d0 20.0d-3 0.0d0 0.0d0   !(SQUARE: XI, XN, YI, YN, Zi, ZN; CIRCLE: RAIO, XC, YC, ZC; STL_FILE:path and name of stl files)
88|  0                 !0 - none , 1 - white noise, 2 - smirnov noise
89|  1.0 0.0 0.0       !noise percentagen at u, v, w
90|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR PRESSURE
91|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR U VELOCITY
92|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR V VELOCITY
93|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR W VELOCITY
94|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR VOF
95|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR SCALAR
96|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR VARIANCE
97|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TEMPERATURE
98|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR Enthalpy
99|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TURBULENCE EQUATION 1
100|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TURBULENCE EQUATION 2
101|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR MASS FRAC
102|  0.0 1.0 0.0       !ALFA BETA GAMMA FOR DENSITY
103|
104|  !TOP FACE
105|  0                 !NUMBER OF SPECIFIC REGIONS AT TOP FACE
106|  1                 !KIND= SAQUARE (1) OR CIRCLE(2)
107|  0.0d0 440.0d-3 0.0d0 20.0d-3 80.0d-5 80.0d-5 !(SQUARE: XI, XN, YI, YN, ZI, ZN; CIRCLE: RAIO, XC, YC, ZC; STL_FILE:path and name of stl files)
108|  0                 !0 - none , 1 - white noise, 2 - smirnov noise
109|  1.0 0.0 0.0       !noise percentagen at u, v, w
110|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR PRESSURE
111|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR U VELOCITY
112|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR V VELOCITY
113|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR W VELOCITY
114|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR VOF
115|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR SCALAR
116|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR VARIANCE
117|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR TEMPERATURE
118|  1.0 0.0 0.0       !ALFA BETA GAMMA FOR Enthalpy
```

```
119|   1.0 0.0 0.0       !ALFA BETA GAMMA FOR TURBULENCE EQUATION 1
120|   1.0 0.0 0.0       !ALFA BETA GAMMA FOR TURBULENCE EQUATION 2
121|   1.0 0.0 0.0       !ALFA BETA GAMMA FOR MASS FRAC
122|   0.0 1.0 0.0       !ALFA BETA GAMMA FOR DENSITY
```

2.2.3 compressible.amr3d

```
!!!****Compressible control parameters****!!!
.false.                      ! Simulate compressible flow (compresFlow)

.false.                      ! Continuous phase is compressible (cont_compr)
.false.                      ! Disperse phase is compressible   (disp_compr)

287.7d0                        ! Ideal gas constant for continuous phase (R_gas_c)
276.5d0                      ! Ideal gas constant for disperse phase (R_gas_d)

 1.4d0                        ! gamma of gas for continuous phase (gamma_compr_c)
 1.2d0                        ! gamma of gas for disperse phase (gamma_compr_c)

 0.00010227273               !Massa molar (m_molar)

 !!!!*****Boundary conditions*****!!!!
 .false.                      ! NRBC (NRBC_aux)
 .false.                      ! Wave transmissive B.C. (waveTrans_aux_west)
 .false.                      ! Wave transmissive B.C. (waveTrans_aux_east)
 .false.                      ! Wave transmissive B.C. (waveTrans_aux_south)
 .false.                      ! Wave transmissive B.C. (waveTrans_aux_north)
 .false.                      ! Wave transmissive B.C. (waveTrans_aux_bot)
 .false.                      ! Wave transmissive B.C. (waveTrans_aux_top)
```

2.2.4 corner.amr3d

```
1|  0
2|  1
3|  0.25  0.25  0.25 0.5 0.625 0.75
4|
5|
6|  1
7|  1
8|  0.5 0.5 0.25 0.75 0.75 0.75
```

Arquivo usado somente no processo de verificação computacional para posicionar blocos de refinamento. A estrutura de cada bloco é a seguinte:

```
numero do processo
numero de patchs no processo
xi, yi, zi, xf, yf, zf
```

A última linha do bloco corresponde às coordenadas mínimas e máximas do bloco de refinamento. Caso se deseje dois blocos de refinamento, a estrutura deve ser replicada duas vezes, conforme no exemplo.

2.2.5 dpm.amr3d

```
1|  !!!****DPM control parameters****!!!
2|  .false.                                          ! use_dpm [true|false]. Default: false
3|  .false.                                          ! Turn on sub-iterations for particles: .true. or .false (dpm_subiter)
4|  0                                                ! Set number of particles: 0 - by mass flow rate, 1 - by value (setnump)
5|  30.0d-3                                         ! Mass flow rate of real particles (dpm_mass_flow)
6|  1 9999                                          ! Number of computational particles to be injected at each step, number of time steps to inject particles
7|  1                                               ! Number of real particles in one computacional particle (nreal_part)
8|  2                                               ! Diameter distribution: 0 - constant, 1 - normal, 2 - log_normal (diamdp)
9|  300.0d-6  50.0d-6                                ! Diameter distribution parameters [diameter mean, diameter standard deviation] *Choose values in the same order of magnitude
10|  745.042756924d0                                    ! Specific mass (dpm_rho) *Or define expression in dpm_evap
11|  1.33065142465d-3                                        ! Dynamic viscosity (dpm_mu)
12|  1                                               ! Initial position distribution: 0 - coordinates, 1 - circular inlet, 2 - rectangular inlet, 3 - annular inlet (initposd)
13|  0.0d0   0.032d0   0.032d0                          ! Initial position by coordinates [x0, y0, z0]
14|  0.0d0   0.0d0   0.0d0  1.0d0  0.0d0   1.0d0       ! Circular/rectangular inlet plane [x1,x2] x [y1,y2] x [z1,z2] *Just planes parallel to xy, xz, yz ...
15|  0.032d0   0.032d0   1.5d-3  0.0d0                    ! Circular/rectangular inlet geometry: [center_pos1, center_pos2] / [diameter_ex/length_edge1, diameter_in/length_edge2]
16|  .false.  0  0.0d0   0.0d0   0.0d0   0.0d0          ! Initiate a volume of particles by value: .true. or .false / 0 - Cylinder [pos_1, pos_2], 1 - Sphere [xc, yc, zc, r], 2 - parallelepiped [center_posY, leng
17|  150.0d0   0.0d0   0.0d0                           ! Initial mean velocities [u0, v0, w0]
18|  0.0d0   5.0d0   5.0d0                            ! Initial standard deviation velocities [u0_, v0_, w0_]
19|  0.0d0   0.0d0   0.0d0                            ! Initial angular velocities [omega_x0, omega_y0, omega_z0]
20|  1   1   1   0  1                                 ! Considered forces: 0 - Deactivate, 1 - Activate [Weigth, Buoyancy, Shear Lift Force, Rotational Lift Force, virtual mass] *Drag always considered
21|  4                                               ! Coupling type: 1-way coupling, 2-way coupling, 4-way coupling (deterministic DPM model), 5-DEM and 6-Adaptative coupling type
22|
23|  !!!****Domain Boundaries control parameters****!!
24|  2   2   1   1   2   2                           ! Condition type for domain boundaries: 0 - Deletion, 1 - Reflection, 2 - Periodic, 3 - Free_slip at [WEST, EAST, SOUTH, NORTH, BOTTOM, TOP]
25|  1.0d0                                           ! Wall-particle coefficient of restitution  (dpm_wall_restitution_coeff)
26|  0.0d0                                           ! Wall-particle coefficient of dynamic friction (dpm_wall_friction_coeff)
27|  0  0.0d0                                        ! Roughness surface: 0 - no, 1 - yes / and standard deviation of roughness (std_roughness)
28|  0                                               ! Wall-particle collision with a immersed geometry: 0 - no, 1 - yes (dpm_ib_use)
29|  1                                               ! Particle position related to the immersed geometry: outside = -1, inside = 1 (dpm_imb_io)
30|
31|  !!!****Inter-particle collision model control parameters - 4-way coupling****!!
32|  1.0d0                                           ! Particles coefficient of restitution in 4-way coupling (dpm_restitution_coeff)
33|  0.0d0                                           ! Particles coefficient of dynamic friction in 4-way coupling (dpm_friction_coeff)
34|
35|  !!!****DEM control parameters****!!!
36|  1                                               ! 1-Linear DEM model and 2-Monlinear DEM model
37|  4.0d6                    ! Particle Young's modulus
38|  0.49d0                    ! Particle poisson ratio
39|  0.3d0                    ! Particle restitution coefficient
40|  0.8d0                    ! Particle static friction coefficient
41|  0.3d0                    ! Particle rolling friction coefficient
42|  0.3d0                    ! Particle rolling viscous damping ratio
43|  1.0d0                                           ! DEM CourantFriedrichsLewy condition (CFL)
44|
45|  !!!****Erosion model control parameters****!!!
46|  0        ! Erosion model: 0 - none, 1 - Oka et al. (2005)
47|
48|  !!!****Dense model control parameters****!!!
49|  0   0                                           ! [0-off/1-on, 0-dpm/1-VOF]
50|
51|  !!!****Droplet evaporation model control parameters****!!!
52|  1013250.0d0                                      ! Ambient pressure (dpm_p_amb)
53|  300.0d0                                         ! Initial droplet temperature (dpm_T0)
54|  2520.5d0                                        ! Liquid specific heat capacity (dpm_cp) *Or define expression in dpm_evap
55|  surrogate.cti                                   ! CTI file (dpm_cti_file)
56|  air                                             ! Cantera mechanism (dpm_cant_mech)
57|  3                                               ! Evaporation model: 0 - None, 1 - CEM, 2 - NEQ, 3 - ASM (dpm_evap_model)
58|  2                                               ! Heating model: 0 - ITC, 1 - FTC, 2 - ETC (dpm_heat_model)
59|  2                                               ! Diffusion model: 0 - None, 1 - FD, 2 - ED (dpm_diff_model)
60|  20                                              ! Number of terms in the series T(r,t) (dpm_nLambda_T)
61|  20                                              ! Number of terms in the series Yl,i(r,t) (dpm_nLambda_Yl)
62|
63|  !!!****Conversions Euler --> Lagrange parameters****!!!
64|  .false.                                         ! Use Conversions Euler --> Lagrange
65|  .false.  0.1d0                                  !Use forced conversion euler to lagrangian, position
66|  50                                              ! Number of time steps to convert Eulerian to Lagrangian
67|  12.d0                                           ! Cut volume - Eulerian to Lagrangian - cte multiplied by the finest dx*dy*dz
68|  .true.                                          ! Use shape criteria - Herrmann (2010)
69|  .true.                                          ! Use Berend filter criteria - Eulerian to Lagrangian
70|  1                                               ! 1 - Ling unperturbed field; 2 - Berend unperturbed field
71|  1.d0                                            ! Distance from IB NOT to convert - cte multiplied by the finest dx
72|
73|  !!!****Conversions Lagrange --> Euler parameters****!!!
74|  .false.                                         ! Use Conversions Lagrange --> Euler
75|  50                                              ! Number of time steps to convert Lagrangian to Eulerian
76|  -10.d0                                          ! Distance from IB to convert - cte multiplied by the finest dx
77|
78|  !!!****Droplet breakup model control parameters****!!!
79|  0.0261d0                                 ! Surface tension (dpm_sigma)
80|  0                                ! Secondary breakup model: 0 - None, 1 - TAB, 2 - KHRT, 3 - SSD (dpm_breakup_model)
```

Neste arquivo são definidos os parâmetros e as propriedades relacionados a abordagem Lagrangiana, utilizada em escoamentos do tipo fluido-sólido ou na modelagem Lagrangiana de bolhas e gotas.

Abaixo serão explicados os parâmetros linha a linha:

- linha 2 **(use_dpm)**: flag para ativar/desativar o módulo DPM
- linha 3 **(dpm_subiter)**: determina se sub passos de tempo serão utilizados para as partículas lagrangianas, sendo .true. para a utilização e .false. para a não utilização
- linha 4 **(setnump)**: escolhendo o número **0**, determina-se que uma vazão mássica de partículas será aplicada; ou escolhendo o número **1**, determina-se que um número fixo de partículas será considerado
- linha 5 **(dpm_mass_flow)**: caso a vazão mássica seja aplicada, colocar nessa linha o valor dessa vazão
- linha 6 **(dpm_nparticles, dpm_inject)**: caso um número fixo de partículas seja considerado, dois valores precisam ser definidos, sendo o primeiro deles o número de partículas, e o segundo a frequência de injeção desse número de partículas (de quanto em quanto passos de tempo esse número de partículas será injetado, ex.: **1** - caso seja injetado em todos os passos de tempo ou **9999999** - caso seja injetado apenas no primeiro passo de tempo)
- linha 7 **(nreal_part)**: se é desejado que apenas um número representativo de partículas seja simulado, deve-se colocar a quantidade de partículas reais que estão sendo representadas por uma partícula computacional

- linha 8 **(diamp)**: escolhendo o número **0**, determina-se que um valor constante será considerado para o diâmetro das partículas; escolhendo o número **1**, determina-se que uma distribuição normal será considerada para o diâmetro; ou escolhendo o número **2**, determina-se que uma distribuição log-normal será considerada para o diâmetro
- linha 9 **(dpm_mean_diam, dpm_std_diam)**: caso um valor constante seja considerado para o diâmetro das partículas, apenas o primeiro valor dessa linha deve ser alterado, que corresponde ao valor do diâmetro; caso uma das distribuições seja considerada para o diâmetro, o primeiro valor corresponde ao diâmetro médio e o segundo corresponde ao desvio padrão do diâmetro
- linha 10 **(dpm_rho)**: caso a massa específica da partícula seja constante basta definir o seu respectivo valor, senão uma expressão pode ser definida dentro da função *dpm_evap.c*
- linha 11 **(dpm_mu)**: precisa ser definido apenas quando a partícula é um fluido, caso a viscosidade dinâmica da partícula seja constante basta definir o seu respectivo valor nessa linha, senão uma expressão pode ser definida dentro da função *dpm_evap.c*
- linha 12 **(initposd)**: escolhendo o número **0**, determina-se que a posição inicial das partículas será definida pelas coordenadas x, y e z; escolhendo o número **1**, determina-se que as partículas serão posicionadas no interior de uma região circular; escolhendo o número **2**, determina-se que as partículas serão posicionadas no interior de uma região retangular; ou escolhendo o número **3**, determina-se que as partículas serão posicionadas no interior de uma região compreendida entre dois círculos
- linha 13 **(dpm_x0, dpm_y0, dpm_z0)**: caso tenha sido escolhido o número **0** na linha 12, os valores de x, y e z correspondentes à posição inicial das partículas devem ser definidos, respectivamente
- linha 14 **(inlet_x1, inlet_x2, inlet_y1, inlet_y2, inlet_z1, inlet_z2)**: caso tenha sido escolhido os números **1**, **2** ou **3** na linha 12, o plano para o qual será definido a região de injeção de partículas deve ser determinado, por exemplo, se **0.0d0 0.0d0 0.0d0 1.0d0 0.0d0 1.0d0** o plano é yz posicionado em $x = 0$, sendo então o primeiro e segundo valores relacionados com o eixo x, o terceiro e o quarto relacionados com o eixo y, e o quinto e o sexto relacionados com o eixo z
- linha 15 **(inlet_center_1, inlet_center_2, inlet_length_1, inlet_length_2)**: caso tenha sido escolhido os números **1**, **2** ou **3** na linha 12, a geometria da região de injeção de partículas deve ser determinada; para uma região circular, o primeiro e o segundo valor representam a posição do centro da círculo, o terceiro valor representa o diâmetro e o quarto não é necessário; para uma região retangular, o primeiro e o segundo valor representam a posição do centro do retângulo, e o terceiro e o quarto valor representa o tamanho do lado horizontal e vertical do retângulo; para uma região anular, o primeiro e o segundo valor representam a posição do centro dos dois círculos, o terceiro representa o diâmetro do círculo externo, e o quarto representa o diâmetro do círculo interno
- linha 16 **(dpm_initall, volume_type, volume_1, volume_2, volume_3, volume_r)**: caso seja iniciado um número fixo de partículas apenas no primeiro passo de tempo, e é desejado posicioná-las de forma aleatória dentro de uma região tridimensional deve-se colocar .true., senão colocar .false.; para o caso .true., os outros parâmetros dessa linha são importantes; se a região desejada é limitada por um cilindro com a face definida na linha 15, deve-se colocar o número **0**, em seguida, as posições inicial e final na terceira dimensão; se a região desejada é limitada por uma esfera, deve-se colocar o número **1**, em seguida, a posição do centro da esfera em x, y e z, e por último, o raio da esfera
- linha 17 **(dpm_u0, dpm_v0, dpm_w0)**: define as componentes da velocidade linear inicial das partículas nas direções x, y e z, respectivamente
- linha 18
- linha 19 **(dpm_omega_x0, dpm_omega_y0, dpm_omega_z0)**: define as componentes da velocidade angular inicial das partículas nas direções x, y e z, respectivamente
- linha 20 **(dpm_forces(1), dpm_forces(2), dpm_forces(3), dpm_forces(4), dpm_forces(5))**: define quais forças atuantes sobre as partículas devem estar ativadas ou não no código; estão disponíveis o peso, o empuxo, a força de cisalhamento devido à sustentação, a força de cisalhamento devido à rotação e a massa virtual, respectivamente; para as forças desativadas deve-se colocar o número **0**, enquanto que para as forças ativadas deve-se colocar o número **1**; a força de arrasto está sempre ativada no código
- linha 21 **(dpm_coupling)**: define o tipo de acoplamento utilizado; para o acoplamento de 1 via, deve-se colocar o número **1**; para o acoplamento de 2 vias, deve-se colocar o número **2**; e para o acoplamento de 4 vias (determinístico), deve-se colocar o número **4**

- linha 24 **(dpm_boundary(1), dpm_boundary(2), dpm_boundary(3), dpm_boundary(4), dpm_boundary(5), dpm_boundary(6))**: define as condições de contorno para as partículas lagrangianas; na ordem, estão as faces oeste, leste, sul, norte, fundo e topo; para as faces em que as partículas serão deletadas, deve-se colocar o número **0**; para as faces em que as partículas serão refletidas, deve-se colocar o número **1**; e para as faces com condição periódica, deve-se colocar o número **2**
- linha 25 **(dpm_restitution_coeff)**: caso o acoplamento de 4 vias esteja sendo utilizado, definir o coeficiente de restituição da colisão entre partículas
- linha 26 **(dpm_friction_coeff)**: caso o acoplamento de 4 vias esteja sendo utilizado, definir o coeficiente de fricção da colisão entre partículas
- linha 27 **(std_roughness)**: caso uma geometria imersa esteja sendo utilizada, definir se é desejado considerar a rugosidade da geometria; se sim, deve-se colocar o número **1** e, em seguida, definir o desvio padrão da rugosidade; senão, apenas deve-se colocar o número **0**
- linha 28 **(dpm_ib_use)**: caso esteja sendo utilizada uma geometria imersa no domínio, se for desejado ativar a colisão das partículas com a fronteira imersa, deve-se colocar o número **1**, senão deve-se colocar o número **0**
- linha 29 **(dpm_imb_io)**: se os vetores normais da fronteira imersa estiverem apontando para onde estão as partículas, deve-se colocar o número **1**, senão deve-se colocar o número **-1**
- linha 32 **(dpm_restitution_coeff)**: caso uma geometria imersa esteja sendo utilizada, definir o coeficiente de restituição da colisão entre partículas e fronteira imersa
- linha 33 **(dpm_friction_coeff)**: caso uma geometria imersa esteja sendo utilizada, definir o coeficiente de fricção da colisão entre partículas e fronteira imersa
- linha 37
- linha 38
- linha 39
- linha 40
- linha 41
- linha 42
- linha 43
- linha 46 **(dpm_ib_erosion)**: caso um modelo de erosão seja utilizado, definir o número correspondente; se nenhum modelo for utilizado, deve-se colocar o número **0**; para o modelo do Oka, deve-se colocar o número **1**
- linha 49 **(dpm_dense_model)**: para casos multifásicos dispersos em regime denso, deve-se colocar o número **1**; caso contrário, o número **0**
- linha 52 **(dpm_p_amb)**: define a pressão ambiente inicial em usada no cálculo das propriedades termodinâmicas e de transporte
- linha 53 **(dpm_T0)**: define a temperatura inicial das partículas injetadas
- linha 54 **(dpm_cp)**: caso o calor específico da partícula seja constante basta definir o seu respectivo valor, senão uma expressão pode ser definida dentro da função *dpm_evap.c*
- linha 55 **(dpm_cti_file)**: define o nome do arquivo de input do Cantera usado para o cálculo das propriedades termodinâmicas e de transporte; este arquivo do tipo .cti deve estar na pasta *mechanisms*
- linha 56 **(dpm_cant_mech)**: no início do arquivo de input do Cantera, cujo nome foi definido na linha 47, há a seguinte informação: *ideal_gas(name = "XXX"*; a *XXX* deve ser copiada aqui a fim de definir o mecanismo do Cantera usado
- linha 57 **(dpm_evap_model)**: define o modelo de evaporação utilizado; para que não ocorra evaporação, deve-se colocar o número **0**; para o modelo clássico, deve-se colocar o número **1**; para o modelo de não-equilíbrio, deve-se colocar o número **2**; e para o modelo de Abramzon-Sirignano, deve-se colocar o número **3**
- linha 58
- linha 59

- linha 60
- linha 61
- linha 64
- linha 65
- linha 66
- linha 67
- linha 68
- linha 69
- linha 70
- linha 71
- linha 74
- linha 75
- linha 76
- linha 79
- linha 80

O módulo da modelagem lagrangiana teve a sua base desenvolvida pela Msc. Jessica Guarato e a metodologia utilizada pode ser encontrada com riqueza de detalhes em sua dissertação. A parte de modelagem da evaporação Lagrangiana foi desenvolvida pela Msc. Abgail pinheiro e mais detalhes a respeito podem ser encontrados em sua dissertação de mestrado, disponível em `https://repositorio.ufu.br/bitstream/123456789/22510/1/LagrangianModelingDroplet.pdf`

2.2.6 energy.amr3d

```
1|  !!!****Energy control parameters****!!!
2|  .false.                 !Use energy equation for temperature(energyEquation)
3|  .false.                 !h_form (chemical + sensible enthalpy)
4|  2                      !Energy solver: 1 PetsC , 2 Multigrid
5|  1                      !Temporal discretisation of energy (1: semi-implicit, 2: implicit)
6|  "cubista"             !advection model temperature:  IMPLICIT: upwind1O, upwind2O, CDS, QUICK, QUICKhayase;    SEMI-IMPLICIT: upwind1O, upwind2O, Barton, MSOU, CDS, cubista, QUICK, QUICKhayase;
7|  .false.                !Use divergente form of Energy Equation
8|  1                      !Strategy of the fluxes construction: 1-Traditional, 2-Malalasekera
9|
10|  .false.                !Use density calculation based on temperature
11|
12|  !#############Boussinesq aproximation ##########################
13|  .false.                 !Use de Boussinesq approximation (boussinesqApprox)
14|  300.0d0                 !Temperature of reference of Boussinesq term(referTemp)
15|  .false.                 !Phase change
16|  .false.                 !Phase change with multicomponent
17|  0                       !eulerian phase change model- 0: split, 1: source
18|  3000.0d0                !latent heat
19|  0.0d0                   !saturation temperature
20|  0.9d0                   ! turbulent prandtl number
21|
22|  !########### Physical properties ##############
23|  1295.0d0      4194.0d0                      !specific heat at continuous phase and disperse phase (cp_phase_c, cp_phase_d)
24|  2.37d-2       6.53d-1                      !thermal conductivity at continuous phase and disperse phase (k_phase_c, k_phase_d)
25|
26|  !########### Temperature boundary conditions ##############
27|  0.0d0  1.0d0  0.0d0                       !temp boundary condition at WEST face (alfa, beta, gamma, bc_kind, g)
28|  0.0d0  1.0d0  0.0d0                       !temp boundary condition at EAST face (alfa, beta, gamma, bc_kind, g)
29|  0.0d0  1.0d0  0.0d0                       !temp boundary condition at SOUTH face (alfa, beta, gamma, bc_kind, g)
30|  0.0d0  1.0d0  0.0d0                       !temp boundary condition at NORTH face (alfa, beta, gamma, bc_kind, g)
31|  0.0d0  1.0d0  0.0d0                       !temp boundary condition at BOTTOM face (alfa, beta, gamma, bc_kind, g)
32|  0.0d0  1.0d0  0.0d0                       !temp boundary condition at TOP face (alfa, beta, gamma, bc_kind, g)
33|  !########### Enthalpy boundary conditions ##############
34|  0.0d0  1.0d0  0.0d0                       !h boundary condition at WEST face (alfa, beta, gamma, bc_kind, g)
35|  0.0d0  1.0d0  0.0d0                       !h boundary condition at EAST face (alfa, beta, gamma, bc_kind, g)
36|  0.0d0  1.0d0  0.0d0                       !h boundary condition at SOUTH face (alfa, beta, gamma, bc_kind, g)
37|  0.0d0  1.0d0  0.0d0                       !h boundary condition at NORTH face (alfa, beta, gamma, bc_kind, g)
38|  0.0d0  1.0d0  0.0d0                       !h boundary condition at BOTTOM face (alfa, beta, gamma, bc_kind, g)
39|  0.0d0  1.0d0  0.0d0                       !h boundary condition at TOP face (alfa, beta, gamma, bc_kind, g)
```

Neste arquivo são definidos os parâmetros e as propriedades relacionados a equação de transporte da temperatura ou energia.

Abaixo serão explicados os parâmetros linha a linha:

- linha 2: flag para ativar a solução da equação de transporte da energia.
- linha 3: escolhe o tipo de solver a ser utilizado para a equação de transporte da energia (1 para PETSc e 2 para Multigrid). Cabe ressaltar que o Multigrid geralmente apresenta melhor custo-benefício. Parâmetro válido somente quando se ativa a solução da equação da energia.
- linha 4: define se a equação de transporte da energia será tratada de forma semi-implicita (1) ou completamente implicita (2). Parâmetro válido somente quando se ativa a solução da equação da energia.
- linha 5: define o modelo utilizado no tratamento do termo advectivo da equação de transporte da energia . Parâmetro válido somente quando se ativa a solução da equação da energia.
- linha 6: flag para escolher entre a forma conservativa (*.true.*) ou não-conservativa *.false.* para a equação de transporte da energia . Parâmetro válido somente quando se ativa a solução da equação da energia.
- linha 7: parâmetro para escolher a forma de discretização dos fluxos da equação de transporte da energia (1: tradicional, 2: Malalasekera). Parâmetro válido somente quando se ativa a solução da equação da energia.
- linha 9: flag para ativar o cálculo da densidade baseado na temperatura. Essa flag deve ser ativada para simulações de escoamentos compressíveis.
- linha 12: flag para ativar o uso da aproximação de Boussinesq no termo de peso-empuxo da equação da quatidade de movimento linear. Mais detalhes a respeito dessa aproximação podem ser encontrados no trabalho de Bernardo [4]
- linha 13: temperatura de referência utilizada na aproximação de Boussinesq para o termo de peso-empuxo da equação da quatidade de movimento linear.
- linha 14: flag para ativar a modelagem de mudança de fase euleriana. Detalhes sobre essa modelagem podem ser encontrados no trabalho de Bernardo [5].
- linha 15: flag para ativar a modelagem de mudança de fase euleriana com multi componentes.
- linha 16: definição do modelo de mudança de fase.
- linha 17: definição do calor latente utilizado na modelagem euleriana de mudança de fase.
- linha 18: definição da temperatura de saturação utilizada na modelagem euleriana de mudança de fase.
- linha 21: definição do valor para o calor específico (c_p) na fase contínua e na fase dispersa, respectivamente. O valor na fase dispersa só será utilizado caso a modelagem VoF seja ativada.

- linha 22: definição do valor para a condutividade térmica (k) na fase contínua e na fase dispersa, respectivamente. O valor na fase dispersa só será utilizado caso a modelagem VoF seja ativada.
- linhas 25 a 30: definição do tipo de condição de contorno para a temperatura na equação da energia. As condições de contorno são definidas por zero ou um, sendo que apenas uma das colunas deve ser preenchida com o valor um. A primeira coluna definida como um indica o uso de condição de contorno do tipo Dirichlet, já a segunda coluna com o valor um, indica condição de contorno do tipo Neumann. A terceira coluna não funciona para as equações dos escalares e nunca deve conter o valor $1.0d0$. Parâmetro válido somente quando se ativa a solução da equação da energia.

OBS 1: a definição das condições de contorno segue sempre a mesma estrutura. São seis linhas correspondentes, respectivamente, às faces x_{min}, x_{max}, y_{min}, y_{max}, z_{min} e z_{max}.

2.2.7 externalBC.amr3d

```
!!!****External input files****!!!
.false.                                  !use external input file for boundary condition? - VELOCITIES (EULER-PHASE)
"externalBC/vel.txt"                            !filepath
41                                      !n - number of cells in face for each direction (face results in n*n)

.false.                                 !use external input file for boundary condition? - TEMPERATURE
"externalBC/temperature.txt"                    !filepath
41                                      !n - number of cells in face for each direction (face results in n*n)

 .false.                                 !use external input file for boundary condition? - MASS FRACTION
 "externalBC/yk.txt"                             !filepath
 41                                      !n - number of cells in face for each direction (face results in n*n)

 .false.                                  !use external input file for boundary condition? - DPM_DIAMETER
 "externalBC/dpdiam.txt"                         !filepath
 41                                      !n - number of cells in face for each direction (face results in n*n)

 .false.                                  !use external input file for boundary condition? - DPM_VEL
 "externalBC/dpvel.txt"                          !filepath
 41                                      !n - number of cells in face for each direction (face results in n*n)

 .false.                                  !use external input file for boundary condition? - DPM_VEL
 "externalBC/dpm_flow.txt"                       !filepath
 41
```

2.2.8 ib.amr3d

```
1|  !!!!*****Immersed Boundary parameters****!!!
2|  200          1.0d-4                    !maximum numbem of cycles in the direct forcing scheme (maxc), and eslon of directing force (rigid_epslon) - for rigid body
3|  1                                      !kind of file: '1-stl ascii file',  '2-stl2amr file', '3-manual file' (kindIBFile)
4|  1                                      !number of immersed boundaries (ib_n_surfaces)
5|  "teste1.stl"                          !path and name of immersed boundary files
6|  .false.                                !move_IB: .true. if the IB is in movement or .false. if isn't in movement
7|  0.d0 0.d0 0.d0                         !Input linear velocity
8|  0.d0 0.d0 0.d0                         !Input angular velocity
9|  0.d0 0.d0 0.d0                         !Reference position for rotation
10|  1                                     !ib level
11|  1                                     !kind of the interp./distrib. function: 1-hat, 2-Gauss, 3-cubic, 4-Roma/Peskin/Berger (kindIntIB)
12|  1                                     !strategy for the construction of the lagrangian volume [1 -> (L^2*sqrt(3)/4)*L = dx^3]; [2 -> (L^2*sqrt(3)/4)*dx = dx^3] (stratLagVol)
13|  .false.                               ! print pvd, pvtp, and vtp first time step.
14|  .false.                               ! print pvd, pvtp, and vtp EVERY time step.
15|
16|  !!!******IB indicator funcyion - normal must be pointed at inner region of  immersed boundary
17|  .false.               !active the IB indictaor function that is 1 in the inner region and zero outter
18|  .false.               ! Is the flow external to the IBs?
19|
20|  !!!!***** Compressible parameters****!!!
21|  .false.               !use free slip boundary condition (freeSlipIB)
22|
23|  !!!!*****Scalar Immersed Boundary parameters****!!!
24|  .false.                 !Use scalar immersed boundary (termicalIB)
25|  .false.                 !Use scalar immersed boundary (scalarIB)
26|  1     320.0d0            !Type of Temperature BC (1-dirichlet,2-neumann,3-robin) and impose volue
27|  1     0.0d0             !Type of Scalar BC (1-dirichlet,2-neumann,3-robin) and impose volue
```

Neste arquivo estão contidos os parâmetros relacionados à modelagem de corpos imersos, ou seja, ao uso de fronteira imersa. Os únicos parâmetros relacionados à fronteira imersa que não se encontra neste arquivo é o número máximo de ciclos do *Multi Direct Forcing* e a tolerância pretendida. Esses dois parâmetros serão definidos no arquivo *input.amr3d*.

Abaixo serão explicados os parâmetros linha a linha:

- linha 2: definição do número máximo de ciclos e da tolerância do método *Multi-Direct Forcing*, utilizado na modelagem de escoamentos sobre corpos imersos.
- linha 3 (kind of file): nesta linha é definido o formato do arquivo de entrada dos pontos lagrangianos correspondentes à fronteira imersa. Atualmente são possíveis três formatos de entrada: *stl*, *stl2amr3d* e *.dat*. O formato *stl* é obtido através do *gmsh* e de outros softwares geradores de malha. O formato *stl2amr3d* é um padrão antigo presente ainda no MFSim para fins de compatibilidade com simulações antigas. Por fim, o formato *.dat* é utilizado quando a fronteira imersa é construida de forma manual. É recomendável que a entrada seja feita no formato *stl*.
- linha 4 (number of IBs): aqui é definido o número de corpos que serão modelados com essa metodologia. Caso o valor seja diferente de um, as linhas 6 a 10 devem ser replicadas para possibilitar definições dos demais corpos, o mesmo deve ser feito para as linhas 22 a 25. Por exemplo, para a simulação do escoamento sobre dois cilindros, esta linha deve conter o valor 2 e os blocos com as linhas 6 a 10 e 22 a 25 devem ser replicados duas vezes.
- linha 5: nome do arquivo que contém os dados da fronteira imersa no formato definido na linha 4. Esse arquivo deve estar na pasta `geo`
- linha 6: flag para definir se a fronteira imersa será móvel (*.true.*) ou estática (*.false.*).
- linha 7: valor para cada componente da velocidade linear. Só funciona se a fronteira for definida como móvel na linha 7;
- linha 8: valor para cada componente da velocidade angular. Só funciona se a fronteira for definida como móvel na linha 7;
- linha 9: definição do centro de rotação para o corpo imerso. Só funciona se a fronteira for definida como móvel na linha 7;
- linha 10: aqui é determinado em qual nível a fronteira imersa deve ser colocada. Geralmente é igual ao valor ngc do arquivo input.amr3d
- linha 11: define o tipo de interpolação e distribuição que serão utilizados no *Multi Direct Forcing*. Os diferentes tipos de interpolação apresentam diferentes níveis de custo, acurácia e robustez.
- linha 12: informa a estratégia utilizada para calcular o tamanho característico dos elementos da fronteira imersa. A malha lagrangiana deve possuir triângulos com tamanhos característicos próximos a malha euleriana, para isso podem ser utilizadas as duas formas de cálculo apresentadas, onde dx é o espaçamento da malha euleriana e L é o tamanho característico do elemento triangular a ser calculado.
- linha 13: flag para ativar gravação de um arquivo no formato *pvtp* no primeiro passo de tempo. A construção desse arquivo pode auxiliar no pós-processamento.
- linha 14: flag para ativar gravação de um arquivo no formato *pvtp* na frequencia de gravação de resultados definida na linha 56 do arquivo *input.amr3d*. Essa flag deve ser definida como *.true.* para simulações com fronteiras móveis ou iteração fluido estrutura.

- linha 17: flag para ativar o mapeamento da região interna e externa à fronteira imersa. Esse recurso é utilizado, por exemplo, para anular a velocidade de advecção da interface (VoF) no escoamento complementar.
- linha 18: define se o escoamento é externo à fronteira imersa.
- linha 21:
- linha 24: flag para ativar o uso da modelagem de fronteira imersa para a temperatura.
- linha 25: flag para ativar o uso da modelagem de fronteira imersa para o escalar.
- linha 26: definição do tipo de forçagem (1: valor imposto, 2: derivada imposta, 3: condição de terceira espécie) na fronteira e o valor a ser forçado. Valores relativos à temperatura.
- linha 27: definição do tipo de forçagem (1: valor imposto, 2: derivada imposta, 3: condição de terceira espécie) na fronteira e o valor a ser forçado. Valores relativos ao escalar.

OBS: para a utilização da modelagem de fronteira imersa com os modelos de turbulência $k - \epsilon$, as normais da malha devem estar direcionadas para o escoamento de interesse.

Para verificar a direção das normais da malha, abra o arquivo *stl* no *gmsh*, clique no botão O, localizado no canto inferior esquerdo da janela, e escolha a opção *All mesh options*. Na janela que irá se abrir existem dois campos, nomeados *Normals and tangents*, definidos como zero. Basta alterar o valor do primeiro espaço, correspondente às normais para um valor em torno de 50, ou ajustá-lo de modo que a visualização das normais se torne possível.

2.2.9 input.amr3d

```
!!!****mesh control parameters****!!!
0.0d0    1.0d0   0.0d0   1.0d0   0.0d0  1.0d0  ![a1,b1]x[a2,b2]x[a3,b3]: domain (the largest parallelepiped)
2 2 2                              !refinement ratio               (r)
1                                  !number of ghost cells          (ngc)
2                                 !number of physical levels       (npl)
32 32 32                            !number of cells in lbot level
0.99d0                             !(cuttof) - goes from 0 to 1, and define how slim is the patch
2                                  !buffer zone between levels      (bzbl)
2                                  !buffer zone to flagged points   (bzfp)
 .false.                           !remesh in the initial configuration: .true. or .false.
 .false.                           !.true./.false. allow the user to choose if he wants an adaptive mesh (dynamic_mesh)
500                                !number of time step of regriding (rfn)
1                                  !Remesh type: 1- Standard Interp+Project, 2- Popinet Div-free Interp
.true.                             !interface is covered by the finest level (interface_finest_level)
25    0.05d0   0                   !options of refinement: 1- density, 2- ib points, 3- mach, 4- turb visc e 5- vorticity and Control parameters of the refinement criteria 5 (eps_input) and vorticity level l
1                                 !0: set the initial patch by coordinates, 1: read the initial patch from corner.amr3d file
2                                  !number of initial patches (number_of_initial_patchs)
0.2d0    0.5d0   0.2d0   0.6d0   0.2d0  0.8d0 2
0.5d0    0.8d0   0.5d0   0.8d0   0.2d0  0.8d0 2

!!!!*****Pressure velocity coupling and discretization models*****!!!!
1     39     1.0d-5                !kind_pressure_velocity_coupling(1-F.S., 2-SIMPLE, 3-SIMPLEC); maximum number of iterations(SIMPLE, SIMPLEC);   maximum divergent(SIMPLE, SIMPLEC)
0.7d0     0.3d0      0.1d0   0.7d0  !Relaxation coefficient of velocity(SIMPLE, SIMPLEC); Relaxation coefficient of pressure(SIMPLE); !Relaxation coefficient of temperature(SIMPLE, SIMPLEC); !Relaxation coeffi
1                              !temporal discretization model: 1 semi-implicit , 2 implicit
"sbdf"                             !time discretization -"sbdf","mcnab","abcn","cnlf" (implicit_disc)
2                                !Conservative form: 1, not conservative form: 2
"CDS"                             !advection model:  IMPLICIT: upwind1O, upwind2O, CDS, QUICK, QUICKhayase;   SEMI-IMPLICIT: upwind1O, upwind2O, godunov, Barton, MSOU, CDS, skew, cubista
1                                  !flux discretization at momentum equation: 1- malalasekera, 2- traditional
0.5d0                              !CourantFriedrichsLewy condition, CFL (cflGlobal)
.false. 1d-4                       ! Flag dt constant (use_dtcte): true|false; dt value if use_dtcte = true
.false. 0.5 0.5                            !Pseudo transient; Relaxation Factors for velocity and scalars

!!!****multigrid control parameters****!!!
50                                 !maximum number of v-cycles or w-cycles (nvcmax)
1            !smoothers options: 1- gsrb, 2- msip, 3- sip, 4- cg, 5- bicgstab (smoothers_pressure)
.false.                            !red black smooth at pressure multigrid (redblack)
1.0d0                              !over relaxation parameter at pressure multigrid (omeg)
1.3d0                              !over relaxation parameter at velocity multigrid (omegv)
1.0d0                              !over relaxation parameter at scalar multigrid (omegs)
1                                  !solver pressure :1- v_cycle, 2 - PETScVI
1                                  !solver velocity :1- v_cycle, 2 - PETSc, 3 - SIP
.true.                             !PETSc update matrix ?
.false. 0                          !PETSc at lbase level??, wich level?? (petsc_level=lbot-petsc_level)
5 5 2                              !number of smoothness in the 2x2x2 level (relax_base), in down process (relax_down) and in up process (relax_up)

!!!****two-phase control parameters****!!!
1.0d0    1.0d0                     !density at continuous phase and disperse phase (dens_phase_c,dens_phase_d)
1.0d-1   1.0d-1              !viscosity at continuous phase and disperse phase (visc_phase_c,visc_phase_d)
0.0d0                               !surface tension (surface_tension)
0.0d0   0.0d0  0.0d0               !gravity acceleration (gravity_x, gravity_y, gravity_z)

!!!****simulations control parameters****!!!
.false.                            !.true./.false. allow the user to choose if he wants to restart the code (re_start)
10000                               !save the data each xxx time step to restart the code (ctrst)
3999                               !code restart point (rct). Please check the folder "restart/". (USE THE SAME EXPRESSION AFTER "ct_")
100                               !number of time steps to print the results (prt)
100000000                          !maximum number of time steps (ctmax)
0.2d0                              !final time of simulation (final_t)
"hdf5"                             !printing format: "tecplot"; "paraview"; "hdf5"(printing)

!!!****boundary conditions control parameter ****!!!
0   0   0                          !periodic boundary conditions: (0 -> no periodic, 1 -> periodic)
!!!**** alfa u + beta du/dn + gamma du/dt = g****!!!!!
!########### U velocity boundary conditions ############
1.0d0  0.0d0  0.0d0                        !u boundary condition at WEST face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !u boundary condition at EAST face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !u boundary condition at SOUTH face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !u boundary condition at NORTH face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !u boundary condition at BOTTOM face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !u boundary condition at TOP face (alfa, beta, gamma)
!########## V velocity boundary conditions ############
1.0d0  0.0d0  0.0d0                        !v boundary condition at WEST face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !v boundary condition at EAST face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !v boundary condition at SOUTH face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !v boundary condition at NORTH face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !v boundary condition at BOTTOM face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !v boundary condition at TOP face (alfa, beta, gamma)
!########## W velocity boundary conditions ############
1.0d0  0.0d0  0.0d0                        !w boundary condition at WEST face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !w boundary condition at EAST face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !w boundary condition at SOUTH face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !w boundary condition at NORTH face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !w boundary condition at BOTTOM face (alfa, beta, gamma)
1.0d0  0.0d0  0.0d0                        !w boundary condition at TOP face (alfa, beta, gamma)
!########### pressure boundary conditions ##############
0.0d0  1.0d0  0.0d0                        !p boundary condition at WEST face(alfa,beta,gamma)
0.0d0  1.0d0  0.0d0                        !p boundary condition at EAST face(alfa,beta,gamma)
0.0d0  1.0d0  0.0d0                        !p boundary condition at SOUTH face(alfa, beta, gamma)
0.0d0  1.0d0  0.0d0                        !p boundary condition at NORTH face(alfa, beta, gamma)
0.0d0  1.0d0  0.0d0                        !p boundary condition at BOTTOM face(alfa, beta, gamma)
0.0d0  1.0d0  0.0d0                        !p boundary condition at TOP face(alfa, beta, gamma)
!########### density boundary conditions ##############
0.0d0  1.0d0  0.0d0                        !p boundary condition at WEST face(alfa,beta,gamma)
0.0d0  1.0d0  0.0d0                        !p boundary condition at EAST face(alfa,beta,gamma)
0.0d0  1.0d0  0.0d0                        !p boundary condition at SOUTH face(alfa, beta, gamma)
0.0d0  1.0d0  0.0d0                        !p boundary condition at NORTH face(alfa, beta, gamma)
0.0d0  1.0d0  0.0d0                        !p boundary condition at BOTTOM face(alfa, beta, gamma)
0.0d0  1.0d0  0.0d0                        !p boundary condition at TOP face(alfa, beta, gamma)

 !!!****initial partition topology****!!!
 2 1 1                                  !number of processors in x,y,z axis

 !!!****Perform load balance
 0                                      !0: dont perform load balance; 1: perform load balance

 !!!****Start by HDF5
 false                                   !use start by hdf5
 "start.hdf5"                           !hdf5 file

 !!!**** Method
 1               ! method: 1 - Noone; 2 - IB; 3 - VoF; 4 - FT; 5 - VoF+IB; 6 - FT+IB

 !!!**** EWF
 .false.                    ! use_ewf [true|false]. Default: false

 !!!**** Libs
 .true.             ! use_petsc
 .false.             ! use_slepc
```

```
119|  .false.            ! use_zoltan
120|
121|  !!!**** Manuf
122|  .true.            | use_manuf [true|false]. Default: false
123|
124|  !!!**** PETSc control
125|  none                  ! pestc_sort_mode: none => no sort, ND => Nested Dissection,1WD => One-way Dissection,RCM => Reverse Cuthill-McKee,QMD => Quotient Minimum Degree
126|  .false.               ! petsc_print_matrix: generate matrix image
127|  .false.               ! petsc_debug_mode: write petsc debug data on proc_*.log files
128|  8  9                  ! petsc_load_treshold_(min/max): load treshold (in thousands) allowed per proccess for PETSc matrices
129|
130|  !!!**** Error subsystem control
131|  .true.               ! mfsim_override: allow override for non critical errors
```

Neste arquivo são definidos os parâmetros relacionados a domínio, malha e controle da simulação, as propriedades dos fluidos, método e integrações com bibliotecas. Após a explicação linha a linha do arquivo serão apresentadas algumas explicações que podem ser muito importantes no entendimento dos parâmetros do arquivo. Portanto, o leitor pode começar a leitura por essas aplicações e, posteriormente, para a explicação linha a linha.

Abaixo serão explicados os parâmetros linha a linha:

- linha 2: definição das dimensões do domínio retangular do problema. As coordenadas indicam, respectivamente, x_{min}, x_{max}, y_{min}, y_{max}, z_{min} e z_{max}, conforme mostra a figura abaixo:

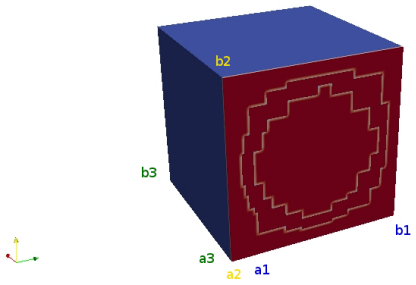


Figure 1: Exemplo de domínio. Os vértices a1 e b1 são associados ao eixo X, a2 e b2 ao eixo Y e a3 e b3 ao eixo Z

OBS: recomenda-se evitar o uso de coordenadas negativas na definição dos limites, ou seja, utilizar somente coordenadas maiores que zero. O algorítimo de construção de malhas pode apresentar algumas instabilidades com dimensões negativas.

- linha 3: define a razão de refinamento em cada direção e pode assumir o valor 1, indicando que a malha não será refinada nessa região ou 2, indicando que a malha será refinada nessa região.
- linha 4 *number of ghost cells*: Depois define-se o número de camadas de células fantasmas que se deseja introduzir ao redor de cada patch. Em geral emprega-se uma camada somente, porém em problemas advectivos de segunda ordem é necessario alterar para duas camadas.
- linha 5 *number of physical levels*: Define-se o número máximo de níveis físicos a serem adotados na malha. Por exemplo, um domínio com $4 \times 4 \times 4$ elementos coberto inteiramente por 3 níveis seria coberto por uma malha de $16 \times 16 \times 16$. A conta para descobrir o número de células no nível mais fino (l_{top}) é dada por $n_{ltop} = n_{lbot} \cdot 2^{(npl-1)}$.
- linha 6 *number of cells in lbot level*: define-se a malha base do problema por meio do número de células para cada eixo X,Y e Z.
- linha 7 *cuttof*: Essa propriedade que pode ter valor entre 0 e 1. Quanto mais próxima de 1, mais delgado será o patch a ser definido.
- linha 8: define o número de células da malha do nível anterior a ser utilizado na transição entre os níveis. Por exemplo, supondo uma malha de 3 níveis. Este parâmetro define quantas células do nível 1 serão cobertas por células do nível 2 na região de transição para o nível 3. Geralmente definido como 2.
- linha 9: número de células posicionadas além dos *bad points* marcados. Essas células funcionam como uma zona de segurança mantendo as regiões de interesse na malha fina até a próxima remalhagem.
- linha 10: flag utilizada para solicitar a reconstrução da malha no primeiro passo de tempo. Geralmente esse recurso não é necessário quando são posicionados blocos de malha fina na configuração inicial da malha.
- linha 11: flag para ativar a remalhagem dinâmica, ou seja, para ativar a reconstrução da malha baseada nos critérios definidos na linha 15.

- linha 12: número de passos de tempo para realizar a reconstrução da malha, ou seja, esse parâmetro define a frequência de remalhagem.
- linha 13: define o tipo de remalhagem dentre 1- Interpolação Padrão + Projeção ou 2- Interpolação *divergent-free* de Popinet. A primeira estratégia é a tradicional, utilizada desde as primeiras versões do código, já a segunda foi implementada recentemente de acordo com o trabalho de 14.
- linha 14: flag para definir se a interface entre fluidos para o método VoF estará no nível mais fino da malha (*.true.*) ou não (*.false.*). Recomenda-se manter sempre a interface no nível mais fino.
- linha 15: escolha das propriedades a serem utilizadas para definir as regiões nas quais a malha será refinada. Atualmente, é possível escolher entre os seguintes critérios:
  - **1 - densidade**: promove o refinamento baseado no gradiente de massa específica. Esse critério é bastante utilizado em simulações de escoamentos bifásicos fluido-fluido com o método VoF.
  - **2 - ib points**: promove o refinamento de regiões onde existam corpos imersos, ou seja, fronteiras imersas.
  - **3 - curvature**: promove o refinamento baseado na curvatura da interface entre dois fluidos. Esse critério só funciona em simulações de escoamentos bifásicos fluido-fluido com o método VoF.
  - **4 - vorticidade_1**: promove o refinamento de regiões de alta vorticidade, ou seja, onde existam recirculações.
  - **5 - vorticity_2**: promove o refinamento de regiões de alta vorticidade, ou seja, onde existam recirculações. Critério semelhante ao anterior, porém, define as regiões que devem ser remalhadas baseado em uma metodologia diferente do critério 4. Esse critério atualmente é mais utilizado que o critério 4.
  - **6 - região específica**: esse critério mantém um refinamento fixo em determinada região do domínio ao longo de toda a simulação. Essa região específica é definida no arquivo *refinement region.amr3d*.
  - **7 - variância do escalar passivo**: promove o refinamento de regiões com alta variância do escalar. Esse critério é mais utilizado em casos reativos modelados com o método da PDF.
  - **8 - gradiente normalizado do escalar passivo**: critério a ser utilizado somente em simulações de escoamentos com transporte de escalar passivo. Promove o refinamento de regiões onde exista um alto gradiente da quantidade do escalar. Esse critério normaliza todo o campo, levando a valores entre 0 e 1 e marca para remalhar regiões onde exista determinada porcentagem do gradiente calculado.
  - **a - gradiente normalizado do número de partículas**: critério a ser utilizado somente em simulações de escoamentos do tipo Euler-Lagrange. Promove o refinamento de regiões onde exista um alto gradiente do número de partículas. Esse critério segue a mesma metodologia do critério 8.
  - **b - gradiente normalizado de ϵ ou ω**: critério a ser utilizado somente em simulações de escoamentos turbulentos utilizando modelos de fechamento a duas equações da classe URANS. Promove o refinamento de regiões onde exista um alto gradiente do número de partículas. Esse critério segue a mesma metodologia do critério 8. O critério "c" é mais adequado que este critério.
  - **c - gradiente normalizado da viscosidade turbulenta**: critério a ser utilizado somente em simulações de escoamentos turbulentos utilizando modelos de fechamento da classe URANS. Promove o refinamento de regiões onde exista um alto gradiente da viscosidade turbulenta. Esse critério segue a mesma metodologia do critério 8.
  - **d - gradiente normalizado do número de Mach**: critério a ser utilizado somente em simulações de escoamentos compressíveis. Promove o refinamento de regiões onde exista um alto gradiente do número de Mach. Esse critério segue a mesma metodologia do critério 8.

  A definição dos parâmetros é realizada da seguinte forma: no primeiro parâmetro da linha concatene os números ou letras correspondentes a todos os critérios que deseja utilizar, por exemplo, `25c` para utilizar a remalhagem baseada em fronteira imersa, vorticidade e gradiente normalizado da viscosidade turbulenta. O segundo e o terceiro parâmetro dessa linha são os limites mínimo e máximo utilizados pelo critério 4, o quarto parâmetro é uma tolerância de controle do critério 5. O último parâmetro da linha é um número inteiro que indica quantos níveis abaixo do nível mais fino será construída a malha dos critérios que não são fronteira imersa. Por exemplo, se a sua simulação tem 5 níveis a fronteira imersa deve ficar sempre no nível mais fino (5) mas você pode remalhar com base na vorticidade, colocando malha com o refinamento do nível 3, para isso defina esse parâmetro como 2, pois $5 - 2 = 3$.
- linha 16: definição do método de criação de blocos de malha fina na configuração inicial de malha. O mais usual é definir esses blocos por coordenadas a partir da linha 18. Para isso, escolha a opção 0.
- linha 17: definição do número inicial de blocos de malha fina na configuração inicial de malha.
- linhas 18 e 19: coordenadas x_{min}, x_{max}, y_{min}, y_{max}, z_{min} e z_{max} para blocos de refinamento iniciais. Esses blocos devem cobrir as regiões de interesse na configuração inicial de malha. Caso se esteja utilizando fronteiras imersas, esses blocos devem cobrir todo o corpo inicialmente para bom funcionamento do código. Devem existir o número de linhas correspondente ao número definido na linha anterior (17).

- linha 22: o primeiro parâmetro da linha define o método de acoplamento pressão-velocidade utilizado, podendo se escolher entre passo fracionado (1), SIMPLE (2) ou SIMPLEC (3). O segundo e o terceiro parâmetro da linha definem, respectivamente, o número máximo de iterações e a tolerância para os métodos SIMPLE e SIMPLEC. Cabe ressaltar que o SIMPLE e o SIMPLEC são métodos iterativos, diferentemente do passo fracionado. O SIMPLE e o SIMPLEC vem sendo utilizados nas simulações de escoamentos compressíveis, nos demais tipos de simulação o passo fracionado é mais usual. Detalhes a respeito dos diferentes métodos de acoplamento podem ser encontrados no livro de Ferziger [7] e, principalmente, de Maliska [11].

- linha 23: estes parâmetros definem, respectivamente, o coeficiente de relaxação para a solução da velocidade usando o SIMPLE ou o SIMPLEC, o coeficiente de relaxação para a solução da pressão usando o SIMPLE, o coeficiente de relaxação para a solução da temperatura usando o SIMPLE ou o SIMPLEC e o coeficiente de relaxação para a solução das concentrações usando o SIMPLE ou o SIMPLEC.

- linha 24: define se as equações da quantidade de movimento linear serão tratadas de forma semi-implicita (1) ou completamente implicita (2).

- linha 25: define o modelo utilizado no tratamento do termo temporal das equações da quantidade de movimento linear. É possível escolher entre "euler", "sbdf", "mcnab", "abcn" ou "cnlf". Os detalhes a respeito da discretização temporal e dos métodos disponíveis podem ser encontrados na seção 4.4 da tese do Dr. Marcelo M. Damasceno [2], disponivel em `https://repositorio.ufu.br/handle/123456789/21483`.

- linha 26: flag para escolher entre a forma conservativa (*1*) ou não-conservativa *2* para as equações da quantidade de movimento linear.

- linha 27: define o modelo utilizado no tratamento do termo advectivo das equações da quantidade de movimento linear. É importante observar que existem os modelos disponíveis para a abordagem semi-implicita (upwind1O, upwind2O, godunov, Barton, MSOU, CDS, skew, cubista) e para a abordagem implícita (upwind1O, upwind2O, CDS, QUICK, QUICKhayase)

- linha 18: definição da discretização do fluxo na equação do momento.

- linha 29: definição do valor do CFL. O CFL é um coeficiente utilizado no cálculo do passo de tempo, sendo que quanto menor o CFL, menor será o passo de tempo. Maiores explicações de como o CFL é levado em conta no cálculo do passo de tempo podem ser encontrados na seção 4.4 da tese do Dr. Marcelo M. Damasceno [2], disponivel em `https://repositorio.ufu.br/handle/123456789/21483`.

- linha 30: flag para indicar se o dt será ou não constante e caso seja, qual o valor do dt.

- linha 34: definição do número máximo de ciclos a serem executados pelo Multigrid, quando o mesmo é utilizado na solução de alguma das equações resolvidas. Maiores detalhes a respeito do método Multigrid-multinivel utilizado no código podem ser encontrados na seção 4.4 da tese do Dr. Rafael Romão [12], disponível em `https://repositorio.ufu.br/bitstream/123456789/18667/1/ModelagemSimulacaoEscoamentos.pdf`.

- linha 35: escolha do solver da pressão. É possível que se escolha entre 5 solvers diferentes, sendo Gauss-Seidel (1) o mais barato e menos robusto e o Gradiente BiConjugado (5) o mais robusto e com maior custo.

- linha 36: flag para ativar o método de relação da equação do resíduo para o método Multigrid.

- linha 37 *omeg*: define o coeficiente de sobre-relaxação para a pressão, usado pelo método Multigrid.

- linha 38 *omegv*: define o coeficiente de sobre-relaxação para a velocidade, usado pelo método Multigrid.

- linha 39 *omegs*: define o coeficiente de sobre-relaxação para as equações escalares, usado pelo método Multigrid.

- linha 40: escolhe o solver para a equação da pressão. É possível escolher entre Multigrid (1) e PETSC (2). Vale ressaltar que o Multigrid é um solver mais barato tanto em termos de tempo computacional, quanto em uso de memória. Porém, o PETSc, apesar de mais caro, é mais robusto em alguns casos como altas razões de massa específica e domínios com grande razão de aspecto.

- linha 41: escolhe o solver para as equações de quantidade de movimento linear. É possível escolher entre Multigrid (1), PETSC (2) e SIP (3). Vale ressaltar que o Multigrid e o SIP são um solvers mais baratos tanto em termos de tempo computacional, quanto em uso de memória. Não é comum a utilização do PETSc na solução das velocidades. Em casos inadequados para o multigrid é comum a utilização do SIP.

- linha 42: flag para ativar a atualização da matriz do PETSc, quando o mesmo é utilizado. A atualização da matriz é um procedimento caro e necessário apenas quando existirem variações nas propriedades físicas do fluido.

- linha 43: flag para ativar a atualização da matriz do PETSc no nivel base e caso verdadeiro em qual nível.
- linha 44: definição de quantas vezes o método escolhido na linha 34 será executado no nível base (relax_base), em cada nível no processo de descida (relax_down) e em cada nível no processo de subida (relax_up) do Multigrid.
- linha 47: definição da massa específica (ρ) da fase contínua e da fase dispersa. Caso a modelagem VoF não seja utilizada, somente a massa específica da fase contínua será considerada.
- linha 48: definição da viscosidade dinâmica molecular (μ) da fase contínua e da fase dispersa. Caso a modelagem VoF não seja utilizada, somente a viscosidade dinâmica molecular da fase contínua será considerada.
- linha 49: definição do valor de tensão interfacial entre dois fluidos. Essa propriedade é utilizada somente em escoamentos multifásicos do tipo fluido-fluido e seu valor depende do par de fluidos.
- linha 50: definição do valor de cada componente do campo gravitacional. Cabe lembrar que se a direção da componente for contrária à do eixo, deve-se acrescentar um sinal de negativo.
- linha 53: flag utilizada para informar a utilização de um ponto de restart, ou seja, reiniciar um caso que caiu por algum motivo ou foi finalizado.
- linha 54: frequência para a gravação de pontos de restart.
- linha 55: especificação do ponto de restart a ser utilizado para reiniciar o caso. Basta verificar a última pasta salva na pasta *restart* e copiar o seu número. Por exemplo, se na pasta *restart*, a pasta mais atual for a *ct_000599*, esta linha deve ser definida como 599.
- linha 56: parâmetro utilizado para definir o intervalo para gravação dos arquivos de pós-processamento. Cabe ressaltar que, geralmente, o passo de tempo da maioria dos problemas está abaixo de 1 ms, portanto, não é necessário gravar os resultados com uma frequência baixa.
- linha 57: define o número máximo de iterações a serem executadas. Esse parâmetro é utilizado com mais frequência em atividades de debugar o código.
- linha 58: definição do tempo físico total de simulação.
- linha 59: linha onde se escolhe o formato dos arquivos de saída. Recomenda-se a escolha do formato *hdf5*, que é compatível tanto com o Visit quanto com o Paraview, que são os dois softwares de pós-processamento mais utilizados no MFLab.
- linha 62: imposição de condição de contorno periódica em alguma direção. A condição de contorno periódica replica o que acontece na face $_{max}$ na face $_{min}$. Por exemplo, para se considera-se periodicidade na direção Z, a linha deve ser definida como `0 0 1` e tudo o que acontece na face z_{max} é replicado na face z_{min}, funcionando como um domínio infinito. É importante ressaltar que quando se impõe periodicidade as condições de contorno na direção são ignoradas pelo código.
- linhas 65 a 70, linhas 72 a 77 e linhas 79 a 84: condições de contorno para U, V e W, respectivamente, que são as componentes do campo de velocidade nas direções X,Y e Z. As condições de contorno são definidas por zero ou um, sendo que apenas uma das colunas deve ser preenchida com o valor um. A primeira coluna definida como um indica o uso de condição de contorno do tipo Dirichlet e a segunda coluna com o valor um, indica condição de contorno do tipo Neumann. A terceira coluna impõe condição do tipo advectiva, onde uma equação da advecção é resolvida no contorno. Essa condição é capaz de trabalhar com recirculações que chegam no contorno e, geralmente, é utilizada na face de saida (*outlet*). Para definir condições de contorno do tipo simetria na face, somente a componente de velocidade perpendicular à face deve ser do tipo Dirichlet e com velocidade nula, as componentes nas outras duas direções devem ser do tipo Neumann. Para se definir simetria, por exemplo, na face y_{max}, U e W devem ser do tipo Neumann e V do tipo Dirichlet, considerando velocidade nula.
- linhas 86 a 91: definição do tipo de condição de contorno para a pressão. As condições de contorno são definidas por zero ou um, sendo que apenas uma das colunas deve ser preenchida com o valor um. A primeira coluna definida como um indica o uso de condição de contorno do tipo Dirichlet e a segunda coluna com o valor um, indica condição de contorno do tipo Neumann. A terceira coluna não funciona para a pressão e nunca deve conter o valor 1.0*d*0. A definição das condições de contorno da pressão devem ser realizadas com base nas condições da velocidade. Para condição do tipo Dirichlet, Simetria ou Advectiva para a velocidade, a condição de contorno para a pressão deve ser do tipo Neumann. Caso a condição de contorno para a velocidade seja do tipo Neumann, a condição de contorno da pressão deve ser do tipo Dirichlet.
- linhas 93 a 98: definição do tipo de condição de contorno para a densidade

- linha 101: definição do número de processos a serem utilizados em cada diração. Cabe lembrar que o número de elementos por processo deve ser um inteiro e, caso se esteja utilizando o multigrid, o número de elementos por processo deve ser um inteiro par. Por exemplo, não é possivel dividir a direção X em 3 processos caso existam 100 elementos nessa direção, já que 100 não é divisível por 3. É importante ressaltar que uma das formas mais eficientes de se paralelizar, é dividindo o domínio em apenas uma direção. O número total de processadores é dado pela multiplicação da quantidade de processadores em cada direção. No arquivo exemplo, o número total de processadores seria $2 \times 1 \times 1 = 2$.
- linha 104: flag para utilizar (1) ou não utilizar (0) o balanço de carga na simulação. O balanço de carga é responsável por realizar uma divisão mais adequada dos processos e de forma dinâmica ao longo da simulação.
- linha 107: flag para ativar/desativar o boot de caso via arquivo hdf5
- linha 108: nome do arquivo hdf5 a ser utilizado para bootar o caso. O arquivo deve estar localizado na pasta `input`.
- linha 111: método utilizado na simulação
- linha 114: flag para ativar/desativar o uso de EWF
- linha 117: flag para ativar/desativar o uso de PETSc
- linha 118: flag para ativar/desativar o uso de SLEPc
- linha 119: flag para ativar desativar o uso de Zoltan
- linha 122: flag para ativar/desativar as condições de normais do manuf
- linha 125: método de ordenação da matriz PETSc
- linha 126: flag para indicar se a matriz PETSc deve ser renderizada em imagem
- linha 127: flag para ativar a escrita de dados de debug do PETSc nos arquivos de log do MFSim.
- linha 128: configuração do treshold (mínimo, máximo) para as matrizes PETSc por processo
- linha 131: flag para indicar se o MFSim deve ou não parar as simulações por conta de erros não críticos

2.2.9.1 Explicações importantes para entender os parâmetros do arquivo *input.amr3d*

Conforme explicado anteriormente, o MFSim é um código de malha bloco-estruturada com refinamento dinâmico. Isso quer dizer que o código trabalha, a princípio apenas em domínios retangulares. A simulação de geometrias não-cartesianas é possível através da modelagem de fronteira imersa.

A definição principal de malha é relativa ao nível mais grosso, chamado de *lbot*. O domínio é então discretizado com a malha escolhida e, caso se deseje refinar determinadas regiões do domínio, malhas mais finas são sobrepostas a essa malha grosseira. O nível mais fino da malha é chamado *ltop*.

O MFsim exige que certos parâmetros da malha sejam estabelecidos. A seguir ilustra-se um esquema de malha onde há 3 níveis físicos.

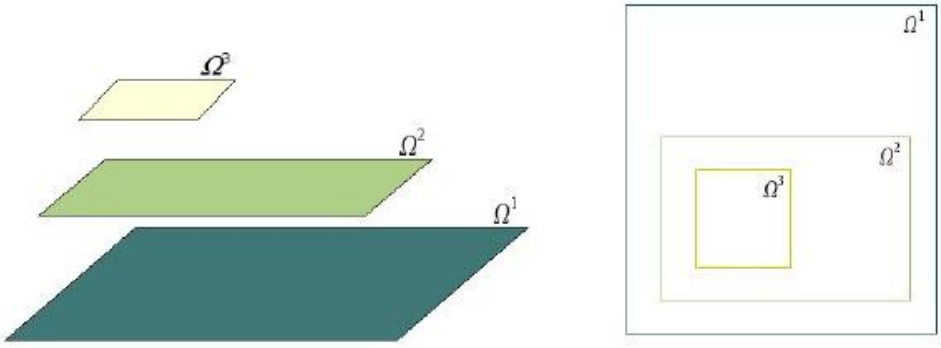


Figure 2: Esquema de malha do AMR, onde há 3 níveis físicos

Pela figura acima, pode-se concluir que estes níveis são sucessivamente mais refinados, sendo que o nível l=3 é o mais refinado. O algoritmo do AMR obedece a regras de aninhamento entre níveis (explicadas em detalhes no capítulo 2), sendo os dois principais pontos:

1. Os cantos de uma malha mais fina coincidem com os cantos das células pertencentes a um nível imediatamente abaixo.

2. Malhas mais finas devem estar no interior da união das malhas do nível abaixo, exceto quando tocam as bordas do nível físico.

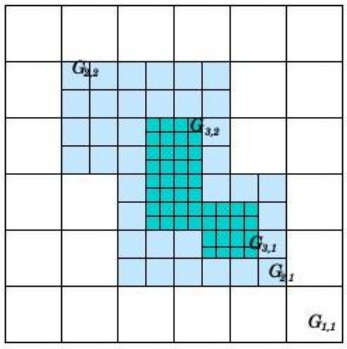


Figure 3: Exemplo de células corretamente alinhadas

Os detalhes principais a respeito das metodologias básicas utilizadas no código em termos de discretização e métodos podem ser encontradas de forma bastante detalhada na tese do Dr. Franco Barbi [1] disponível em `https://repositorio.ufu.br/bitstream/123456789/18155/1/ExperimentacaoNumericaBolhas.pdf`, na tese do Dr. Rafael Romão [12], disponível em `https://repositorio.ufu.br/bitstream/123456789/18667/1/ModelagemSimulacaoEscoamentos.pdf` e na tese do Dr. Marcelo M. Damasceno [2], disponível em `https://repositorio.ufu.br/handle/123456789/21483`. O objetivo do manual é apresentar definições sucintas dos parâmetros de entrada, porém, essas três teses abordam de maneira completa a teoria que permeia as metodologias utilizadas no código.

A respeito dos métodos de solução utilizados tanto para pressão e velocidade quanto para os escalares, o Multigrid é diretamente implementado no código e apresenta menor custo computacional e consumo de memória. Já o PETSc (*Portable, Extensible Toolkit for Scientific Computation*) é um conjunto de estruturas e rotinas de dados para a solução escalável (paralela) de aplicativos científicos modelada por equações diferenciais parciais. Sua documentação está disponível em `https://petsc.org/release/`. Atualmente, são utilizados três conjuntos de pré-condicionadores e solvers do PETSc dentro do MFSim que, apesar do maior tempo computacional e consumo de memória pode ser mais robusto em certos problemas.

Esse é o arquivo de entrada mais extenso da pasta *input* e contém os parâmetros que podem ser considerados o coração do código.

2.2.10 postproc.amr3d

```
!!!!****List of variables to be recorded****!!!
.true.    # U-velocity
.true.    # V-velocity
.true.    # W-velocity
.false.   # Q-criterion
.false.   # X-vorticity
.false.   # Y-vorticity
.false.   # Z-vorticity
.false.   # Vorticity magnitude
 .true.    # Pressure
 .false.   # Pressure correction
 .true.    # Viscosity
 .true.    # Density
 .false.   # Volumetric fraction            ###### input.amr3d method must be 3 (VoF) or 5 (VoF+IB)
 .false.   # Front Tracking                 ###### input.amr3d method must be 4 (FT) or  6 (FT+IB)
 .false.   # Turbulent equation number 1    ###### logical variable "kw_model.or.sst_model.or.sa_model" must be set on
 .false.    # Turbulent equation number 2   ###### logical variable "kw_model.or.sst_model" must be set on
 .false.    # Turbulent Viscosity           ###### logical variable "kw_model.or.sst_model.or.sa_model" must be set on
 .false.    # Temperature                   ###### logical variable "energyEquation" must be set on or it will be set as the one obtained with combustion models
 .false.    # Enthalpy                      ###### logical variable "energyEquation" must be set on or it will be set as the one obtained with combustion models
 .false.    # cp_mix
 .false.    # k_mix
 .false.    # Scalar                        ###### logical variable "transpEquation" must be set on
 .false.    # Scalar variance               ###### logical variable "varianceEquation" must be set on
 .false.    # dpm_npart                     ###### logical variabel "use_dpm" must be set on (dpm.amr3d)
 .false.   # dpm_Dnpart
 .false.   # dpm_diam_part
 .false.   # dpm_mass_part
 .false.   # dpm_Dmass_part
 .false.   # dpm_u
 .false.   # dpm_v
 .false.   # dpm_w
 .false.   # dpm_ufluct
 .false.   # dpm_vfluct
 .false.   # dpm_wfluct
 .false.   # dpm_temp_part
 .false.   # pdf_mean                     ###### logical variable "use_comb" must be set on (reaction.amr3d)
 .false.   # pdf_var
 .false.   # pdf_u
 .false.   # pdf_v
 .false.   # pdf_w
 .false.   # pdf_npvc
 .false.   # pdf_wdot
 .false.   # pdf_weight
 .false.   # pdf_density
 .false.   # pdf_kinenergy
 .false.   # pdf_mass
 .false.   # Qsas
 .false.   # dwall
 .false.   # ib_indicator
 .false.   # divergent
 .false.   # gamma
 .false.   # Mach
 .false.   # rhs_Energy
 .false.   # std_h
 .false.   # std_cp
 .false.   # therm_pressure
 .false.   # rhs_var_comp2
 .false.   # rhs_var_comp
 .false.   # rhs_comb
 .false.   # react_var_total (sum of mass fractions)
 .false.   # lewis_k                       ###### logical variables "eul_cantera_props" and "transpEquation" must be set on
 .false.   # wdot_k                       ###### logical variables "eul_cantera_props" and "transpEquation" must be set on
 .false.   # diff_k                       ###### logical variables "eul_cantera_props" and "transpEquation" must be set on
 .false.   # h_k                          ###### logical variables "eul_cantera_props" and "transpEquation" must be set on
 .false.   # rhs_arrhenius_spc            ###### logical variables "eul_cantera_props" and "transpEquation" must be set on
 .false.   # rhs_EDM_spc                  ###### logical variables "eul_cantera_props" and "transpEquation" must be set on
 .false.   # rhs_comb_spc                 ###### logical variables "eul_cantera_props" and "transpEquation" must be set on
 .false.   # Reactive scalars             ###### logical variables "eul_cantera_props" and "transpEquation" must be set on
```

A função deste arquivo é simplesmente definir quais propriedades terão o campo gravado para visualização no pós-processamento. As variáveis definidas como *.true.* terão o seu campo gravado para pós-processamento no Visit ou Paraview.

2.2.11 probes.amr3d

```
1|  !!#### probes setup ############################
2|  .false.                                          !!! Enable probes
3|  0          !!! Number of probes
4|  .false.                                         !!! Probes follow bubble's centroid (edit positions in ../src_amr/eul_probes), if FALSE, edit positions in ./amr3d_probes
5|  .false.                                        !!! Treat probes as badpoints
6|  !!#### Probes's Coordinates : X, Y, Z ############################
```

Nesse arquivo são definidos parâmetros para sondas, que são pontos de gravação de resultados, em casos de colunas de bolhas.

- linha 2: flag para ativar a gravação de dados através de sondas que seguem bolhas.
- linha 3: definição do número de probes a serem utilizadas.
- linha 4: flag para ativar o mecanismo em que as sondas seguem as bolhas.
- linha 5: flag para ativar o tratamento das bolhas como *bad points*. Isso quer dizer que a cada remalhagem a malha mais fina será posicionada sobre as probes.
- linha 7 adiante: devem ser fornecidas as coordenadas de cada sonda. O número de linhas deve ser igual ao parâmetro definido na linha 3.

2.2.12 probesSTL.amr3d

```
1|  !!!!*****Probe Files and Parameters****!!!
2|  .false.                    !use stl probes (use_probes_lag)
3|  1                          !Polynomial order for interpolation: 1-Linear, 2-Quadratic, 3-Cubic (Probe_Interp)
4|  1                          !Frequency in time steps to probe variables (probe_interval)
5|  1        !Number of files of stl probes (repeat save format and path for each file)
6|  1                          !Save format (1-file_by_point ; 2-file_by_t)
7|  "../test.dat"          !path and name of stl probes files
8|
9|  !!!!*****Probe Variables****!!!
10|  .true.          !Save iteration (ct - ct_out_ctl)
11|  .true.          !Save time(t - t_out_ctl)
12|  .true.         !Save time step (dt - dt_out_ctl)
13|  .true.          !Save x position (xc - xc_out_ctl)
14|  .true.          !Save y position(yc - yc_out_ctl)
15|  .true.          !Save z position(zc - zc_out_ctl)
16|  .true.          !Save velocity u (u - u_out_ctl)
17|  .true.          !Save velocity v (v - v_out_ctl)
18|  .true.          !Save velocity w (w - w_out_ctl)
19|  .false.          !Save pressure (P - P_out_ctl)
20|  .false.         !Save scalar (sca - sca_out_ctl)
21|  .false.         !Save density (rho - rho_out_ctl)
22|  .true.          !Save viscosity (mu - mu_out_ctl)
23|  .true.           !Save temperature (temperature - temp_out_ctl)
24|  .false.          !Save turbulence variable 1 (tur1 - tur1_out_ctl)
25|  .false.          !Save turbulence variable 2 (tur2 - tur2_out_ctl)
26|  .false.          !Save particles number (dpm_npart - dpm_npart_out_ctl)
27|  .false.          !Save particle diameter (dpm_diam_part - dpm_diam_part_out_ctl)
28|  .false.          !Save particle mass (dpm_mass_part - dpm_mass_part_out_ctl)
29|  .false.          !Save particle velocity u (dpm_upart - dpm_upart_out_ctl)
30|  .false.          !Save particle velocity v (dpm_vpart - dpm_vpart_out_ctl)
31|  .false.          !Save particle velocity w (dpm_wpart - dpm_wpart_out_ctl)
32|  .false.          !Save dpm velocity fluctuation u' (dpm_ufluct - dpm_ufluct_out_ctl)
33|  .false.          !Save dpm velocity fluctuation v' (dpm_vfluct - dpm_vfluct_out_ctl)
34|  .false.          !Save dpm velocity fluctuation w' (dpm_wfluct - dpm_wfluct_out_ctl)
35|  .false.          !Save particle temperature (dpm_temp_part - dpm_temp_part_out_ctl)
36|  .false.          !Save volfrac (Volume fraction VoF - volfrac_out_ctl)
37|  .true.           !Save Yfuel (Fuel Mass fraction from concentrations)
38|  .true.           !Save YCO2 (CO2 Mass fraction from concentrations)
39|  .true.           !Save YCO (CO Mass fraction from concentrations)
40|  .true.           !Save YOH (OH Mass fraction from concentrations)
41|  .true.           !Save YN2 (N2 Mass fraction from concentrations)
42|  .true.           !Save YNO (NO Mass fraction from concentrations)
```

Neste arquivo, são definidos os parâmetros utilizados para gravação de dados através de sondas. As sondas são pontos escolhidos no domínio para que informações sobre o escoamento sejam gravadas. É possível levantar perfis de propriedades ao longo do tempo ou realizar médias temporais e espaciais por meio dos dados gravados pelas sondas.

Abaixo serão explicadas as linhas contidas nesse arquivo:

- linha 2: flag para ativar a gravação de dados por meio de sondas. Para utilizar esse recurso esse parâmetro deve ser definido como *.true.*
- linha 3: por meio deste parâmetro é possível escolher a ordem da interpolação utilizada para obter o valor da informação na coordenada da sonda. Por exemplo, o escalar passivo é uma variável centrada. Caso a sonda esteja em uma posição do domínio que não seja coincidente com um centro de célula, é necessário interpolar o valor do escalar para que se obtenha o valor da propriedade no ponto em que a probe foi posicionada.
- linha 4: define a frequência, em termos de passo de tempo, com que as informações da sonda serão gravados no arquivo de saída.
- linha 5: número de arquivos de entrada das probes. Por exemplo, suponha-se que, para avaliar o comportamento do escoamento seja necessário posicionar duas linhas de sondas em regiões diferentes, é possível fornecer as coordenadas de cada linha de sonda em um arquivo diferente (*linha1.dat* e *linha2.dat*, por exemplo). Caso se tenha mais de um arquivo de entrada para as probes, as linhas 6 e 7 devem ser repetidas para cada arquivo.
- linha 6: parâmetro para definir o formato dos arquivos de saída das probes. É possível que os arquivos de saída sejam escritos por sonda (1) ou por tempo (2). Para exemplificar, considere um caso em que serão executados 100 passos de tempo, com 10 sondas posicionadas no domínio para gravação de dados gravando as informações a todo passo de tempo (parâmetro da linha 4 definido como 1). Caso se escolha a primeira opção nessa linha (*1-file_by_point*), serão obtidos 10 arquivos com 100 linhas cada. Porém, caso se escolha a segunga opção nessa linha (*2-file_by_t*), serão obtidos 100 arquivos com 10 linhas cada. Para a obtenção de médias temporais, a opção 1 é mais indicada, já para a obtenção de perfis em escoamentos médios (modelos URANS), a opção 2 pode ser mais conveniente.
- linha 7: definição do nome do arquivo de entrada com as coordenadas de cada probe. Esse arquivo **deve** estar na pasta `probes`. Sua primeira linha deve conter o número total de sondas que serão informadas no arquivo, a seguir, cada linha deve seguir o seguinte padrão:

  `número_da_probe coordenada_x coordenada_y coordenada_z`

  Sendo assim, o número total de linhas do arquivo será $n_{probes} + 1$, onde n_{probes} é o número de sondas definido na primeira linha.
- linhas 10 a 35: informam quais informações serão gravadas em cada sonda. Apenas as propriedades definidas como *.true.* seão gravadas no arquivo, sendo uma em cada coluna. Ou seja, caso se deseje gravar 8 informações do ponto, o arquivo conterá 8 colunas.

2.2.13 reaction.amr3d

```
!!!****Control flags****!!!
.false.                                  !use_cantera
.false.                                  ! use_comb
.false.                                  ! use pdf - transpdf - [true|false]. Default = false

!!!****Chemical reaction control parameters****!!!
.false.                                  !cantera_logical
.false.                                  !evap_logical
C3H8_2step_VOLVO.cti methane_air_global.cti 2S_CH4_BFER.cti     !CTI file
 c3h8 ch4mec CH4_BFER_mix                 !cant_mechanism for cantera
 "C3H8:0.04007046,O2:0.22366331,N2:0.73626623"  "CH4:0.055,O2:0.220,N2:0.725,CO2:0.000,H2O:0.000"           !composition in main jet or unburnt jet
 "C3H8:0.04007046,O2:0.22366331,N2:0.73626623" "C3H8:6.19081272e-67,O2:7.83130907e-02,N2:7.36266232e-01,CO:5.41542808e-05,CO2:1.19886395e-01,H2O:6.54801285e-02"  "CH4:0.000,O2:0.000,N2:0.725,CO2:0.151,H2O:0.124
 "C3H8:0.04007046,O2:0.22366331,N2:0.73626623"   !composition in pilot jet
 288.0d0 288.00d0 288.00d0  101000  353.1718d0  288.0d0 1791.11d0 353.1718d0                !T_main_jet, T_coflow, Po or T_unburnt, T_burnt, Po
 C3H8                                    !Main_fuel
 .false.                                 !mix_frac (mixture fraction formulation - for diffusive flames and passive scalar)
 .false.                                 !prog_var (progress variable formulation - for premixed flames and passive scalar)
 .false.                                 !species diffusivities

 !########### Combustion modeling ##############
 .false.                                 !pure_mixing
 .false.                                 !burke_schumann (only for diffusive flames)
 .false.                                 !arrhenius
 .false.                                 !zimont turbulent flame speed closure model
 .false.                                 !ebu
 .false.                                 !bml
 .false.                                 !edm
 .false.                                 !edc
 .false.                                 !hybrid_arrhenius_edc
 .false. 2d-5 5.0d10                     !ignite the mixturd- valid for concentrations with edm, edc or arrhenius; ignite,T_ignt,Pwr_ignt

 !########### ISAT ##############
 .false.                                 !turn ISAT ON (.true.) / OFF (.false.)
 1                                       !turn binary search tree restart ON (1) / OFF (0)
 2009                                    !ISAT version (2009 or 1997)
 200000                                  !maximum number of leaves in the binary tree (a very large number can explode the memory!!)
 5                                       !size of the MRU list                    (Default value: 5      )
 30                                      !size of the MFU list                    (Default value: 30     )
 2                                       !affine subspace dimension               (Default value: 5      )
 1.0d-3                                  !ISAT error tolerance - etol             (Default value: 1.0d-04)
 1.0d0                                   !error scaling matrix absolute tolerance (Default value: 1.0d00 )
 0.0d0                                   !error scaling matrix relative tolerance (Default value: 0.0d00 )
 1.0d-12  14                             !Jacobian solver absolute tolerance      (Default value: 1.0d-12)
 1.0e-6                                  !Jacobian solver relative tolerance      (Default value: 1.0d-06)

 !########### PDF input data ##############
 11                                      !ITENS - MPI NOTIONAL TYPE: number of itens describing the mpi notional type
 1.0d-8                                  !COMB_TOLERANCE
 10                                      !PARTICLES_PER_CELL
 100                                     !NUMBER_OF_PARTICLES: UNIFORM INIT in the whole domain - if not defined PARTICLES_PER_CELL
 1.0d7                                   !K_NOTIONAL - It is just a factor which is cancelled after operations

 ! Cloning and annihilation parameters
 0.8d0                                   !npc_min factor      PER LEAF MINIMUM = npc_min factor*PARTICLES_PER_CELL  - It must be greater than 0.5*PARTICLES_PER_CELL
 3.0d0                                   !npc_max factor      PER LEAF MAXIMUM = npc_max factor*PARTICLES_PER_CELL
 1                                       !annihilation_type - TYPE 1 - without any preference; TYPE 2 - zeta ordering

 !!!!TODO set in the userFunctions.c file
 1.0d0                                   !ZETA1_MIX
 0.0d0                                   !ZETA2_MIX
 0.5d0                                   !TARG_MIX
 5.0d0                                   !MIXLGTH

 ! SCALAR TRANSPORT PARAMETERS
 11                                      !n_Yi
 2.0d0                                   !COMEGA
 1.0d0                                   !RHO
 1.0d0                                   !SC


 @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
 #equilibrio para grimech phi=1 t_ini=300k
 Tequi=2225.5245834770044
 "H2:2.64919135d-04,H:1.43445171d-05,O:1.25797318d-04,O2:5.39245427d-03,OH:1.78293841d-03,H2O:1.20503195d-01,HO2:6.00394886d-07,H2O2:5.67141950d-08,C:9.71189456d-18,CH:1.59556233d-18,CH2:5.02571540d-18,CH2(S):3
 @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

 @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
 #equilibrio para YangPope1998 phi=1 t_ini=300k
 Tequi=2231.279223055053
 "CH4:1.50398552d-17,O2:6.20898995d-03,N2:7.24665002d-01,OH:1.88935930d-03,CO2:1.37382282d-01,CO:8.91634528d-03,H2O:1.20527126d-01,H2:2.55730688d-04,H:1.45470397d-05,O:1.39874252d-04,HO2:6.80083324d-07,H2O2:6.1
 @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
```

2.2.14 refinement_region.amr3d

```
1|  1                                  !number of regions
2|  3                                  !levels it should be covered
3|  5.0d0 7.0d0 4.0d0 6.0d0 0.0d0 6.0d0     !paralelepiped coordinates (xo, xmax, yo, ymax, zo, zmax)
```

Nesse arquivo são informados os parâmetros utilizados no critério de refinamento 6 (refinamento fixo em uma região do domínio). A estrutura desse arquivo é simple e cada linha será explicada abaixo.

- linha 1: informa o número de regiões com refinamento fixo ao longo de toda a simulação que existirão no domínio. As duas linhas seguintes devem ser replicadas de acordo com o número de regiões informadas nessa linha.
- linha 2: informa qual o nível estará cobrindo a região informada. Por exemplo, suponha-se uma simulação onde a malha possua 5 níveis, porém, deseja-se manter uma região com refinamento fixo da malha correspondente ao nível 3. Para esse caso, o valor a ser informado nessa linha é 3.
- linha 3: informa as coordenadas do bloco de refinamento a ser posicionado, da seguinte forma:

  `x_min x_max y_min y_max z_min z_max`

Para um exemplo em que deseja-se posicionar dois blocos fixos de refinamento o arquivo seria definido como:

```
1|  2                          !number of regions
2|  3                          !levels it should be covered
3|  5.0 7.0 4.0 6.0 0.0 6.0     !paralelepiped coordinates (xo, xmax, yo, ymax, zo, zmax)
4|  3                          !levels it should be covered
5|  2.0 3.0 2.0 3.0 0.0 2.0     !paralelepiped coordinates (xo, xmax, yo, ymax, zo, zmax)
```

2.2.15 scalar.amr3d

```
1|  !!!****Scalar control parameters****!!!
2|
3|  !---- Scalar equation ---------------------------
4|  .true.                                  !Use transport equation of a generic scalar(transpEquation)
5|  2                                       !Scalar solver: 1 PetsC , 2 Multigrid
6|  1                                       !temporal discretization model for scalar (1: semi-implicit, 2: implicit)
7|  "CDS"                                  !advection model for scalar:  IMPLICIT: CDS, upwind1O, upwind2O, cubista.  SEMI-IMPLICIT: upwind1O, upwind2O, Barton, MSOU, CDS, cubista, QUICK, QUICKhayase;
8|  .false.                                 !Use divergente form of Scalar Equation
9|  1                                       !Strategy of the fluxes construction: 1-Traditional, 2-Malalasekera
10|  .false.                                !update_sca (update rho and mu based on a passive scalar)
11|
12|  !---- Variance equation -------------------------
13|  .false.                                !Use transport equation of variance(varianceEquation)
14|  2                                       !Variance solver: 1 PetsC , 2 Multigrid
15|  1                                       !temporal discretization model for variance (1: semi-implicit, 2: implicit)
16|  "cubista"                               !advection model for variance:  IMPLICIT: upwind1O, upwind2O, CDS, QUICK, QUICKhayase;    SEMI-IMPLICIT: upwind1O, upwind2O, Barton, MSOU, CDS, cubista, QUICK, QUICKhaya
17|  .false.                                 !Use divergente form of Variance Equation
18|  1                                       !Strategy of the fluxes construction: 1-Traditional, 2-Malalasekera
19|
20|  !---- Concentrations equations ------------------
21|  .false.                                !Use transport equation of concentrations (multiple_scalars)
22|  2                                       !Concentrations solver: 1 PetsC , 2 Multigrid
23|  1                                       !temporal discretization model for concentrations: (1: semi-implicit, 2: implicit)
24|  "cubista"                               !advection model for concentrations:  IMPLICIT: upwind1O, upwind2O, CDS, QUICK, QUICKhayase;    SEMI-IMPLICIT: upwind1O, upwind2O, Barton, MSOU, CDS, cubista, QUICK, QUI
25|  .false.                                 !Use divergente form of Concentrations Equations
26|  1                                       !Strategy of the fluxes construction: 1-Traditional, 2-Malalasekera
27|
28|
29|  0.0001d0                               !(alfaRefer) mi/pr
30|  2.0d0                                  !(h_IB)
31|  1.0d0                                  !(ks_IB)
32|  300.0d0                                !(Tinf_IB)
33|  4.76d-5                                 !diffusivity coefficient
34|  0.70d0                                 !Schmit number
35|  .false.                                 !Use Corrosion Model
36|
37|  !########### Scalar boundary conditions ##############
38|  1.0d0  0.0d0  0.0d0                      !sca boundary condition at WEST   face (alfa, beta, gamma)
39|  1.0d0  0.0d0  0.0d0                      !sca boundary condition at EAST   face (alfa, beta, gamma)
40|  1.0d0  0.0d0  0.0d0                      !sca boundary condition at SOUTH  face (alfa, beta, gamma)
41|  1.0d0  0.0d0  0.0d0                      !sca boundary condition at NORTH  face (alfa, beta, gamma)
42|  1.0d0  0.0d0  0.0d0                      !sca boundary condition at BOTTOM face (alfa, beta, gamma)
43|  1.0d0  0.0d0  0.0d0                      !sca boundary condition at TOP    face (alfa, beta, gamma)
44|  !########### Variance boundary conditions ##############
45|  0.0d0  1.0d0  0.0d0                      !vari boundary condition at WEST   face (alfa, beta, gamma)
46|  0.0d0  1.0d0  0.0d0                      !vari boundary condition at EAST   face (alfa, beta, gamma)
47|  0.0d0  1.0d0  0.0d0                      !vari boundary condition at SOUTH  face (alfa, beta, gamma)
48|  0.0d0  1.0d0  0.0d0                      !vari boundary condition at NORTH  face (alfa, beta, gamma)
49|  0.0d0  1.0d0  0.0d0                      !vari boundary condition at BOTTOM face (alfa, beta, gamma)
50|  0.0d0  1.0d0  0.0d0                      !vari boundary condition at TOP    face (alfa, beta, gamma)
51|  !########### Concentrations boundary conditions ##############!
52|  1.0d0  0.0d0  0.0d0                      !concentration boundary condition at WEST   face (alfa, beta, gamma)
53|  1.0d0  0.0d0  0.0d0                      !concentration boundary condition at EAST   face (alfa, beta, gamma)
54|  1.0d0  0.0d0  0.0d0                      !concentration boundary condition at SOUTH  face (alfa, beta, gamma)
55|  1.0d0  0.0d0  0.0d0                      !concentration boundary condition at NORTH  face (alfa, beta, gamma)
56|  1.0d0  0.0d0  0.0d0                      !concentration boundary condition at BOTTOM face (alfa, beta, gamma)
57|  1.0d0  0.0d0  0.0d0                      !concentration boundary condition at TOP    face (alfa, beta, gamma)
```

Neste arquivo são definidos os parâmetros e as propriedades relacionados a equação de transporte do escalar passivo, a equação de transporte da variância do escalar passivo e a equação de transporte das concentrações.

Abaixo serão explicados os parâmetros linha a linha:

- linha 4: flag para ativar a solução da equação de transporte do escalar passivo.
- linha 5: escolhe o tipo de solver a ser utilizado para a equação de transporte do escalar passivo (1 para PETSc e 2 para Multigrid). Cabe ressaltar que o Multigrid geralmente apresenta melhor custo-benefício. Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linha 6: define se a equação de transporte do escalar passivo será tratada de forma semi-implicita (1) ou completamente implicita (2). Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linha 7: define o modelo utilizado no tratamento do termo advectivo da equação de transporte do escalar passivo. Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linha 8: flag para escolher entre a forma conservativa (*.true.*) ou não-conservativa *.false.* para a equação de transporte do escalar passivo. Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linha 9: parâmetro para escolher a forma de discretização dos fluxos da equação de transporte do escalar passivo (1: tradicional, 2: Malalasekera). Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linha 10: flag para ativar a atualização da massa específica (ρ) e da viscosidade dinâmica molecular (μ) com base nos valores do escalar passivo.
- linha 13: flag para ativar a solução da equação de transporte da variância do escalar passivo.
- linha 14: escolhe o tipo de solver a ser utilizado para a equação de transporte da variância do escalar passivo (1 para PETSc e 2 para Multigrid). Cabe ressaltar que o Multigrid geralmente apresenta melhor custo-benefício. Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linha 15: define se a equação de transporte da variância do escalar passivo será tratada de forma semi-implicita (1) ou completamente implicita (2). Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.

- linha 16: define o modelo utilizado no tratamento do termo advectivo da equação de transporte da variância do escalar passivo. Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linha 17: flag para escolher entre a forma conservativa (*.true.*) ou não-conservativa *.false.* para a equação de transporte da variância do escalar passivo. Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linha 18: parâmetro para escolher a forma de discretização dos fluxos da equação de transporte da variância do escalar passivo (1: tradicional, 2: Malalasekera). Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linha 21: flag para ativar a solução da equação de transporte das concentrações.
- linha 22: escolhe o tipo de solver a ser utilizado para a equação de transporte das concentrações (1 para PETSc e 2 para Multigrid). Cabe ressaltar que o Multigrid geralmente apresenta melhor custo-benefício. Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linha 23: define se a equação de transporte das concentrações será tratada de forma semi-implicita (1) ou completamente implicita (2). Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linha 24: define o modelo utilizado no tratamento do termo advectivo da equação de transporte das concentrações. Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linha 25: flag para escolher entre a forma conservativa (*.true.*) ou não-conservativa *.false.* para a equação de transporte das concentrações. Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linha 26: parâmetro para escolher a forma de discretização dos fluxos da equação de transporte das concentrações (1: tradicional, 2: Malalasekera). Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linha 29: parâmetro utilizado somente na forçagem direta de terceira espécie para temperatura (Robin).
- linha 30: definição do valor para o coeficiente de transferência térmica do material relacionado ao corpo imerso.
- linha 31: definição do valor para a condutividade térmica do material relacionado ao corpo imerso.
- linha 32: definição do valor da temperatura a ser imposta no corpo imerso caso se esteja utilizando o método de fronteira imersa em conjunto com a equação da energia.
- linha 33: coeficiente de difusão utilizado no termo difusivo das equações da concentração e do escalar passivo caso o cantera não esteja sendo utilizado.
- linha 34: flag para ativar o uso do modelo de corrosão.
- linhas 37 a 42: definição do tipo de condição de contorno para a equação de transporte do escalar passivo. As condições de contorno são definidas por zero ou um, sendo que apenas uma das colunas deve ser preenchida com o valor um. A primeira coluna definida como um indica o uso de condição de contorno do tipo Dirichlet, já a segunda coluna com o valor um, indica condição de contorno do tipo Neumann. A terceira coluna não funciona para as equações dos escalares e nunca deve conter o valor $1.0d0$. Parâmetro válido somente quando se ativa a solução da equação do escalar passivo.
- linhas 44 a 49: definição do tipo de condição de contorno para a equação de variância do escalar passivo. As condições de contorno são definidas por zero ou um, sendo que apenas uma das colunas deve ser preenchida com o valor um. A primeira coluna definida como um indica o uso de condição de contorno do tipo Dirichlet, já a segunda coluna com o valor um, indica condição de contorno do tipo Neumann. A terceira coluna não funciona para as equações dos escalares e nunca deve conter o valor $1.0d0$. Parâmetro válido somente quando se ativa a solução da equação do escalar passivo e a solução da equação da variância.
- linhas 51 a 56: definição do tipo de condição de contorno para a equação de transporte de concentrações. As condições de contorno são definidas por zero ou um, sendo que apenas uma das colunas deve ser preenchida com o valor um. A primeira coluna definida como um indica o uso de condição de contorno do tipo Dirichlet, já a segunda coluna com o valor um, indica condição de contorno do tipo Neumann. A terceira coluna não funciona para as equações dos escalares e nunca deve conter o valor $1.0d0$. Parâmetro válido somente quando se ativa a solução da equação para as concentrações.

OBS 1: a definição das condições de contorno segue sempre a mesma estrutura. São seis linhas correspondentes, respectivamente, às faces x_{min}, x_{max}, y_{min}, y_{max}, z_{min} e z_{max}.

2.2.16 structure.amr3d

```
1|  .false.                 ! Use Beam FSI modeling (use_beam_fsi)
2|  .false.                 ! Use Plate FSI modeling (use_plate_fsi)
3|  .false.                  ! Use Solid FSI modeling (use_solid_fsi)
4|
5|  !########## FSI user's parameters ##############
6|
7|  0.0d0                   ! FSI calculations Start time (t_fsi)
8|  1                       ! time steps to do FSI calculations
9|  100                      ! Number of time steps to do FSI
10|
11|  !########## FSI Coupling ##############
12|
13|  3                       ! Time discretization method. 1: Runge-Kutta 4th order (rk_44); 2: Runge-Kutta-Fehlberg method (rkf45); 3: Runge-Kutta-DormandPrince method (rkdp54)
14|  .false.                 ! Use Strong Coupling (is_strong)
15|  19                      ! Max. number of iterations on Strong Coupling loop
16|  1.0d-5                  ! Residue for Strong Coupling convergence
17|
18|  !############ Pluck FSI ################
19|
20|  .false.                  ! use or not Pluck test (use_pluck_test)
21|  1.0d4                   ! Pluck Force magnitude (F_pluck)
22|  0.0d0 -1.0d0 0.0d0       ! Pluck Force direction (F_pluck_dir)
23|  0                       ! FSI iteration to start force aplication (itp_Start)
24|  10                      ! Number of time steps during Force aplication. Uses a cossin curve. (itp_delta)
25|  340                     ! Structural element to apply the force (ele_pluck)
26|
27|  !############ Beam FSI ################
28|
29|  0                       !Number of Beam structures
30|  "Li_Lee_8Onodes_W.bdf"  ! Beam NASTRAN file name 'geo/' (mesh_file)
31|  1                       ! Associated Immersed Boundary Number (ib_number)
32|  0.064                   ! Caracteristic Lenght for IB Mapping (h)
33|  11                       ! Beam Theory: 11-Euler-Bernoulli, 12-Timoshenko Circular, 13-Timoshenko Rectangular (Theory)
34|  1.d-3    1.d-7          ! Damping matrix constants (alpha, beta)
35|  6                       ! Number of modes to simulate (nmodes)
36|  0.d0                    ! Axial Force (Faxial)
37|  0.d0  0.d0              ! Internal Mass [Kg/m] and Mass Moment of inertia (R1^2 + R2^2) [m^2] to implicit mass correction (inc_mass,inc_moment)
38|  .false.                 ! Use explicit mass correction (use_expl_mass)
39|  0.d0  0.d0              ! Internal Mass [Kg/m] and Mass Moment of inertia (R1^2 + R2^2) [m^2] to explicit mass correction (corr_mass,corr_moment)
40|  .true.                  ! Choose if initial displacement have own weight (Weight_flag)
41|  2                       ! Number of nodes with Boundary conditions (n_bcs)
42|  1.d9  1.d9  1.d9        ! Linear Stiffness of the flexible support (Kx, Ky, Kz)
43|  1.d9  1.d9  1.d9        ! Rotational Stiffness (Krx, Kry, Krz)
44|  1.d2  1.d2  1.d2        ! Linear damping of the flexible support (Dx, Dy, Dz)
45|  1.d2  1.d2  1.d2        ! Rotational damping of the flexible support (Drx, Dry, Drz)
46|  1.d9  1.d9  1.d9        ! Linear Stiffness of the flexible support (Kx, Ky, Kz)
47|  1.d9  1.d9  1.d9        ! Rotational Stiffness (Krx, Kry, Krz)
48|  1.d2  1.d2  1.d2        ! Linear damping of the flexible support (Dx, Dy, Dz)
49|  1.d2  1.d2  1.d2        ! Rotational damping of the flexible support (Drx, Dry, Drz)
50|
51|  !############ Plate FSI ################
52|
53|  0                       ! Number of Plate structures
54|  "Nastran.bdf"           ! Plate NASTRAN file name (mesh_file)
55|  1                       ! Associated Immersed Boundary Number (ib_number)
56|  21                       ! Plate Theory: 21-Kirchoff, 22-Mindlin, 23-Shell
57|  1.000d-3                ! Caracteristic Lenght for IB Mapping (h)
58|  0.d0    0.d0            ! Damping matrix constants (alpha, beta)
59|  5                       ! Number of modes to simulate (nmodes)
60|  .false.                 ! Choose if initial displacement have own weight (Weight_flag)
61|
62|  !############ Solid FSI ################
63|
64|  1                       ! Number of Solid structures
65|  1                       ! Mesh file format: 1-.bdf NASTRAN; 2-.cfx5 Fluent
66|  "line_full_concrete_nnd.bdf"  ! Mesh file name (mesh_file)
67|  1                       ! Associated Immersed Boundary Number (ib_number)
68|  32                       ! Type of element: 31: Hexa-8 standard element; 32: Hexa-8 With extra shape function
69|  10.000d-3                ! Caracteristic Lenght for IB Mapping (h)
70|  0.d0    0.d0            ! Damping matrix constants (alpha, beta)
71|  20                      ! Number of modes to simulate (nmodes)
72|  .true.                  ! Choose if initial displacement have own weight (Weight_flag)
```

2.2.17 turbulence.amr3d

```
1|  !!!****turbulence control parameters****!!!
2|
3|  !---------------- Large Eddy Simulation models (LES) ----------------!
4|  .false.               ! Smagorinsky turbulence model (smag_model)
5|  .false.               ! Vreman turbulence model (vreman_model)
6|  0.1d0                 ! Smagorinsky constant value (cte_smagorinsky)
7|  .false.                ! Smagorinsky turbulence model + Van Driest damping function (smag_vandr_model)
8|  .false.               ! Dynamic smagorinsky turbulence model (dyn_model)
9|  .false.                ! use explicit filter for LES models -> Smag, Smag+VanD or Dyna (explicit_LES_filter)
10|
11|  !---------------- Non-linear Large Eddy Simulation models (LES) ----------------!
12|  .false.               ! Verstappen turbulence model (verst_model)
13|
14|  !---------------- Reynolds-averaged NavierStokes equations models (RANS) ----------------!
15|  .false.               ! Spalart_Allmaras's model (sa_model). ATTENTION: Configure [turb. eq. 1 b.c. for \tilde{nu}]
16|  .false.               ! Wilcox's k-omega model (kw_model). ATTENTION: Configure [turb. eq. 1 b.c. for k], and [turb. eq. 2 b.c. for omega]
17|  .false.               ! Menter's k-omegaSST model (sst_model). ATTENTION: Configure [turb. eq. 1 b.c. for k], and [turb. eq. 2 b.c. for omega]
18|  .false.               ! Standard k-epsilon model (ke_model). ATTENTION: Configure [turb. eq. 1 b.c. for k], and [turb. eq. 2 b.c. for omega]
19|  .false.               ! Realizable k-epsilon model (rke_model). ATTENTION: Configure [turb. eq. 1 b.c. for k], and [turb. eq. 2 b.c. for omega]
20|  .false.               ! Use two-layer model
21|  .false.               ! Use standard wall law
22|
23|  !---------------------------- Detached Eddy Simulation (DES) ----------------------------!
24|  .false.               ! Apply DES model on URANS model?
25|
26|  !---------------------------- Scale-Adaptive Simulation (SAS) ----------------------------!
27|  .false.               ! Apply SAS model on URANS model?
28|
29|  2                     ! URANS solver: 1 PetsC , 2 Multigrid
30|  1                     ! Temporal discretisation of turbulence variables (1: semi-implicit, 2: implicit)
31|  "cubista"                ! advection model for turbulence:  IMPLICIT: upwind1O, upwind2O, CDS, QUICK, QUICKhayase;    SEMI-IMPLICIT: upwind1O, upwind2O, Barton, MSOU, CDS, cubista, QUICK, QUICKhayase;
32|  .true.               ! Use divergente form of Turbulence Equation
33|  1                     !Strategy of the fluxes construction: 1-Traditional, 2-Malalasekera
34|
35|  !########### Turbulence equation number 1 boundary conditions ##############
36|  0.0d0  1.0d0  0.0d0                        !turb1 boundary condition at WEST face (alfa, beta, gamma, bc_kind, g)
37|  0.0d0  1.0d0  0.0d0                        !turb1 boundary condition at EAST face (alfa, beta, gamma, bc_kind, g)
38|  0.0d0  1.0d0  0.0d0                        !turb1 boundary condition at SOUTH face (alfa, beta, gamma, bc_kind, g)
39|  0.0d0  1.0d0  0.0d0                        !turb1 boundary condition at NORTH face (alfa, beta, gamma, bc_kind, g)
40|  0.0d0  1.0d0  0.0d0                        !turb1 boundary condition at BOTTOM face (alfa, beta, gamma, bc_kind, g)
41|  0.0d0  1.0d0  0.0d0                        !turb1 boundary condition at TOP face (alfa, beta, gamma, bc_kind, g)
42|  !########### Turbulence equation number 2 boundary conditions ##############
43|  0.0d0  1.0d0  0.0d0                        !turb2 boundary condition at WEST face (alfa, beta, gamma, bc_kind, g)
44|  0.0d0  1.0d0  0.0d0                        !turb2 boundary condition at EAST face (alfa, beta, gamma, bc_kind, g)
45|  0.0d0  1.0d0  0.0d0                        !turb2 boundary condition at SOUTH face (alfa, beta, gamma, bc_kind, g)
46|  0.0d0  1.0d0  0.0d0                        !turb2 boundary condition at NORTH face (alfa, beta, gamma, bc_kind, g)
47|  0.0d0  1.0d0  0.0d0                        !turb2 boundary condition at BOTTOM face (alfa, beta, gamma, bc_kind, g)
48|  0.0d0  1.0d0  0.0d0                        !turb2 boundary condition at TOP face (alfa, beta, gamma, bc_kind, g)
49|
50|  1.0d3          ! Maximum ratio between turbulent viscosity and molecular viscosity - Configure for the URANS models (ratioVisc)
51|
52|  !!!**** Noise global parameters ****!!!
53|  1                         ! Noise type (1- White Noise ; 2- Smirnov Noise)
54|  1                         ! S_type (1 - isotropico 2 anisotropico)
55|  4.60868d-5                ! turbtime turbulent time scale
56|  0.0d0 0.0d0 0.0d0        ! Noise Percentages in U,V,W
57|
58|  !!!**** Noise BC control parameters ****!!!
59|  .false. !Noise imposition on West BC
60|  .false. !Noise imposition on East BC
61|  .false. !Noise imposition on South BC
62|  .false. !Noise imposition on North BC
63|  .false. !Noise imposition on Bottom BC
64|  .false. !Noise imposition on Top BC
65|
66|  !!!****Noise Volume control parameters****!!!
67|  .false. !Noise imposition on domain interior
```

Neste arquivo são definidos os parâmetros e as propriedades relacionados aos modelos de fechamento da turbulência implementados no código. Cabe ressaltar que apenas um modelo de turbulência pode ser acionado por vez e que é importante estudar e conhecer as características de cada modelo de fechamento disponível para que se utilize o mais adequado na simulação.

Abaixo serão explicados os parâmetros linha a linha:

- linha 4: ativa o modelo clássico ou constante de Smagorinsky [18]. Esse é um modelo da classe LES.
- linha 5:
- linha 6: define o valor da constante utilizada no modelo clássico de Smagorinsky.
- linha 7: ativa o modelo clássico ou constante de Smagorinsky com a função de amortecimento de Van Driest. O uso dessa função é recomendado para simulações que envolvam paredes, já que ela leva a viscosidade turbulenta para zero em regiões de parede.
- linha 8: ativa o modelo dinâmico de Smagorinsky [8, 10]. Esse é um modelo da classe LES.
- linha 9: ativa o uso da filtragem explícita para os modelos da classe LES. Em suma, esse método de filtragem é mais rigoroso pois utiliza como parâmetro o comprimento característico do nível mais fino da malha (*ltop*).
- linha 12: ativa o uso da modelaqem não-linear da classe LES, baseada no modelo de Verstappen [17].
- linha 15: ativa o uso do modelo de fechamento Spallart-Almaras [19]. Este é um modelo de fechamento a uma equação da classe URANS.
- linha 16: ativa o uso do modelo $k-\omega$ padrão [20]. Este é um modelo de fechamento a duas equações da classe URANS.

- linha 17: ativa o uso do modelo $k-\omega$ SST [13], que é uma hibridação dos modelos $k-\epsilon$ padrão e $k-\omega$ padrão. Este é um modelo de fechamento a duas equações da classe URANS.
- linha 18: ativa o uso do modelo $k-\epsilon$ padrão [9]. Este é um modelo de fechamento a duas equações da classe URANS.
- linha 19: ativa o uso do modelo $k-\epsilon$ realizável [16]. Este é um modelo de fechamento a duas equações da classe URANS.
- linha 20: flag para ativar o modelo de tratamento de parede a duas camadas (*two-layer wall treatment*). Esse modelo deve ser utilizado sempre que existirem paredes em simulações realizadas com os modelos $k-\epsilon$. Atenção: esse tratamento funciona apenas para os modelos da classe $k-\epsilon$.
- linha 21: ativa o uso de uma lei de parede padrão para modelos $k-\epsilon$. Atualmente esta parte do código se encontra em manutenção para correção de problemas encontrados. Caso seja necessário o uso de um tratamento de parede em simulações utilizando os modelos $k-\epsilon$, utilize o two-layer.
- linha 24: aplica a modelagem híbrida do tipo DES no modelo URANS escolhido. Esta opção está disponível para os modelos Spalart-Allmaras, k-ω SST e k-ϵ padrão. Ressaltando que a flag só funciona quando um dos modelos URANS citados também é ativado (*.true.*). Mais detalhes a respeito da modelagem podem ser encontrados na dissertação de mestrado do aluno Alex José Elias [6].
- linha 27: aplica a modelagem híbrida do tipo SAS no modelo URANS escolhido. Esta opção está disponível para os modelos Spalart-Allmaras, k-ω SST e k-ϵ padrão. Ressaltando que a flag só funciona quando um dos modelos URANS citados também é ativado (*.true.*). Mais detalhes a respeito da modelagem podem ser encontrados na dissertação de mestrado do aluno Alex José Elias [6].
- linha 29: escolhe o tipo de solver a ser utilizado nas equações de transporte adicionais dos modelos URANS (1 para PETSc e 2 para Multigrid). Cabe ressaltar que o Multigrid geralmente apresenta melhor custo-benefício. Parâmetro válido somente para modelos URANS.
- linha 30: define se as equações de transporte adicionais dos modelos URANS serão tratadas de forma semi-implicita (1) ou completamente implicita (2). Parâmetro válido somente para modelos URANS.
- linha 31: define o modelo utilizado no tratamento do termo advectivo das equações de transporte adicionais dos modelos URANS. Parâmetro válido somente para modelos URANS.
- linha 32: flag para escolher entre a forma conservativa (*.true.*) ou não-conservativa *.false.* para as equações de transporte adicionais dos modelos URANS. Parâmetro válido somente para modelos URANS.
- linha 33: parâmetro para escolher a forma de discretização dos fluxos das equações de transporte adicionais dos modelos URANS (1: tradicional, 2: Malalasekera). Parâmetro válido somente para modelos URANS.
- linhas 36 a 41: definição do tipo de condição de contorno para a primeira equação dos modelos URANS (k ou $\tilde{\nu}$). As condições de contorno são definidas por zero ou um, sendo que apenas uma das colunas deve ser preenchida com o valor um. A primeira coluna definida como um indica o uso de condição de contorno do tipo Dirichlet e a segunda coluna com o valor um, indica condição de contorno do tipo Neumann. A terceira coluna não funciona para as equações dos escalares e nunca deve conter o valor $1.0d0$. Parâmetro válido somente para modelos URANS.
- linhas 43 a 48: definição do tipo de condição de contorno para a segunda equação dos modelos URANS (ϵ ou ω). As condições de contorno são definidas por zero ou um, sendo que apenas uma das colunas deve ser preenchida com o valor um. A primeira coluna definida como um indica o uso de condição de contorno do tipo Dirichlet e a segunda coluna com o valor um, indica condição de contorno do tipo Neumann. A terceira coluna não funciona para as equações dos escalares e nunca deve conter o valor $1.0d0$. Parâmetro válido somente para modelos URANS.
- linha 50: valor máximo da razão μ_t/μ. Utilizado para limitar a ação dos modelos URANS. Quanto mais alto o valor desse parâmetro, mais difusivo o modelo se tornará. O valor *default* desse parâmetro no código ANSYS Fluent é definido como $1.0d5$.
- linha 53: definição do tipo de ruído que se quer aplicar na simulação. Esse parâmetro só funciona se algum dos parâmetros das linhas 60 a 65 ou da linha 68 forem definidos como *.true.*. O número 1 ativa a aplicação de um ruído branco, já o número 2 ativa o ruído do tipo Smirnov. Recomenda-se estudar e conhecer as características de cada tipo de ruído disponível para que se utilize o mais adequado na simulação.
- linha 54: define o tipo do ruído Smirnov a ser aplicado, podendo ser do tipo isotrópico (1) ou anisotrópico (2). O uso do ruído Smirnov isotrópico exige apenas a definição da escala de tempo turbulenta na linha seguinte, já o ruído anisotrópico exige adequações nos arquivos da pasta *src_turb/noise*. Mais detalhes a respeito do uso do ruído Smirnov podem ser fornecidos pelo Dr. Marcelo M. Damasceno ou pelo professor Dr. João Marcelo.

- linha 55: definição da escala de tempo turbulenta, utilizada apenas no ruído Smirnov isotrópico.
- linha 56: definição da intensidade de ruído aplicado em cada direção dada em porcentagem. Por exemplo, a definição desse parâmetro como $0.1d0$ indica 10%.
- linhas 59 a 64: flags para aplicação de ruído nas diferentes faces do domínio a cada passo de tempo.
- linha 67: flag para aplicação de ruído em todo o domínio no primeiro passo de tempo, é o chamado ruído volumétrico.

OBS 1: a definição das condições de contorno segue sempre a mesma estrutura. São seis linhas correspondentes, respectivamente, às faces x_{min}, x_{max}, y_{min}, y_{max}, z_{min} e z_{max}.

OBS 2: mais detalhes a respeito da teoria envolvida nos modelos de fechamento da turbulência apresentados podem ser encontrados no livro do professor Dr. Aristeu da Silveira Neto.

OBS 3: mais informações a respeito das aplicações de cada modelo de turbulência e das equações utilizadas podem ser encontrados no manual do Fluent (ANSYS Fluent Theory Guide).

OBS 4: mais informações a respeito dos ruídos podem ser encontradas no artigo publicado pelo laboratório a respeito desse tema [3].

2.2.18 vof.amr3d

```
1|  !############ VOF CL boundary conditions #############
2|  0                        !Contact Line at WEST   face (0= Don't use; 1= Use)
3|  0                        !Contact Line at EAST   face (0= Don't use; 1= Use)
4|  0                        !Contact Line at SOUTH  face (0= Don't use; 1= Use)
5|  0                        !Contact Line at NORTH  face (0= Don't use; 1= Use)
6|  0                        !Contact Line at BOTTOM face (0= Don't use; 1= Use)
7|  0                        !Contact Line at TOP    face (0= Don't use; 1= Use)
8|  !############ VOF boundary conditions #############
9|  0.0d0  1.0d0  0.0d0                          !vof boundary condition at WEST face (alfa, beta, gamma, bc_kind, g)
10|  0.0d0  1.0d0  0.0d0                          !vof boundary condition at EAST face (alfa, beta, gamma, bc_kind, g)
11|  0.0d0  1.0d0  0.0d0                          !vof boundary condition at SOUTH face (alfa, beta, gamma, bc_kind, g)
12|  0.0d0  1.0d0  0.0d0                          !vof boundary condition at NORTH face (alfa, beta, gamma, bc_kind, g)
13|  0.0d0  1.0d0  0.0d0                          !vof boundary condition at BOTTOM face (alfa, beta, gamma, bc_kind, g)
14|  0.0d0  1.0d0  0.0d0                          !vof boundary condition at TOP face (alfa, beta, gamma, bc_kind, g)
15|
16|  !!!****VOF control parameters****!!!
17|  .false.                       !vof twice fine model
18|  .false.                      !mass-momentum compatibility
19|  "CSF"      !CSF or SSF
20|  2             !vof curvature model: "height_function=1; "shirani=2"; "twentysevencells=3"; "paraboloid=4"; "least_square=5"; "exact=6"
21|  .true.      !use the vof smooth to evaluate the density and viscosity: .true. or .false. (vof_smooth)
22|  1      !number of bodys to initialize
23|  1.50d0  1.50d0  2.95d0   0.10d0           !xc,yc,zc,raio
24|  4                                          !What Surface: sphere=0,ellipsoid=1,torus=2,tank=3,halfSphere=4
25|  30 0 1                                    !Contact angle; Contact Angle model: 0=Static; 1=Yokoi; 2=Shikmurzaev; gammaMax
26|  1                           ! vof_init_model: 1=sussman, 2=points, 3=vofi
```

Nesse arquivo são definidos parâmetros e propriedades para escoamentos multifásicos do tipo fluido-fluido, utilizando o método VoF. Abaixo os parâmetros serão descritos linha a linha.

- linhas 2 a 7: parâmetros para ativar a modelagem de contato triplo em cada face do domínio. A explicação teórica e numérica da metodologia é apresentada na tese de doutorado do Dr. Rodrigo Lizita [15], disponível em

  `https://repositorio.ufu.br/bitstream/123456789/21285/1/UnderstandingDynamicsGas-liquid-solid.pdf`

- linhas 9 a 14: definição do tipo de condição de contorno para a fração volumétrica do método VoF. As condições de contorno são definidas por zero ou um, sendo que apenas uma das colunas deve ser preenchida com o valor um. A primeira coluna definida como um indica o uso de condição de contorno do tipo Dirichlet e a segunda coluna com o valor um, indica condição de contorno do tipo Neumann. A terceira coluna não funciona para a fração volumétrica do método VoF e nunca deve conter o valor $1.0d0$. O usual, na maior parte dos problemas, é manter as condições do tipo Neumann em todas as faces do domínio.

- linha 17: flag para ativar a modelagem que utiliza uma malha não-física correspondente a um nível hipotético ltop+1 sobre a interface (*twice fine model*)

- linha 18: flag para ativar a compatibilidade massa-momentum. Esse recurso é utilizado para mudar a forma de avaliação da massa específica de uma média linear para um cálculo do tipo upwind. Esse parâmetro é utilizado somente na forma conservativa e auxilia em simulações de transição entre regimes como, por exemplo, estratificado-slug.

- linha 19: parâmetro para realizar a escolha do tratamento de tensão superficial na interface. Atualmente é possivel se utilizar o modelo CSF ou o modelo SSF, sendo mais usual o CSF.

- linha 20: parâmetro utilizado para escolher o modelo a ser utilizado no cálculo de curvatura da interface. É possível escolher entre seis modelos atualmente, sendo a shirani (2) mais barata computacionalmente e menos acurada. Também é possível utilizar combinações, por exemplo, definindo esse parâmetro como 12, é utilizada uma combinação dos modelos função altura (1) e shirani (2). **OBS:** uma configuração bastante utilizada na literatura é 14 (função altura + parabolóide)

- linha 21: flag para ativar uma forma de suavização da massa específica (ρ) com o objetivo de melhorar a solução da pressão.

- linha 22: define o número de corpos modelados com VoF a serem inicializados.

- linha 23: define propriedades utilizadas na inicialização da interface (coordenadas do centro e raio).

- linha 24: parâmetro que informa o tipo de corpo a ser inicializado.

- linha 25: definição de parâmetros relacionados ao uso da modelagem de ponto triplo de contato (gás-líquido-parede). A explicação teórica e numérica da metodologia é apresentada na tese de doutorado do Dr. Rodrigo Lizita [15], disponível em

  `https://repositorio.ufu.br/bitstream/123456789/21285/1/UnderstandingDynamicsGas-liquid-solid.pdf`

- linha 26: flag para indicar o modelo vof: 1 - sussman, 2 - points, 3 - vofi

OBS 1: Cabe ressaltar que, dependendo da forma inicial da interface entre os fluidos, é necessário alterar a definição na subrotina *selectdist*, que fica no arquivo *src_vof/src_vof_initialization/vof_sussman_initialization.f90*;

2.3 Arquivos de Malha

A última parte que você precisa considerar ao setar é um caso é o arquivo de malha. Nem todos os métodos requerem o uso de um arquivo de malha, porém os que requerem, terão tal arquivo fornecido no formato *.stl*. Esse tipo de arquivo é gerado através de softwares geradores de malha como o *gmsh*. Na seção 4 há casos de exemplo que usam arquivos de malha e servem como tutorial de como usar esses arquivos.

2.4 Arquivo CFG

Agora que já setou o caso, precisará apenas montar o arquivo de configuração da simulação do MFSim, que serve para o informar onde estão as pastas de entrada e saída de dados. Esse é o **arquivo cfg** e seu conteúdo precisa conter o seguinte:

```
input_path: ""
geo_path: ""
output_path: ""
restart_path: ""
probes_path: ""
```

Onde

1. **input_path**: indica o caminho da pasta que contém os arquivos de input descritos nas seção 2.2
2. **geo_path**: indica o caminho da pasta que contém os `*.stl` usados para as malhas nos métodos FT e IB
3. **output_path**: indica o caminho da pasta onde o MFSim gerará os arquivos de saída de dados `*.hdf5` entre outros
4. **restart_path**: indica o caminho da pasta onde o MFSim gerará os dados de reinício de caso
5. **probes_path**: indica o caminho da pasta onde estão as definições de sonda

Exemplificando, consideremos que o caso está setado na pasta `/home/USER/caso` e queremos gravar os resultados na pasta `/home/USER/caso-saida`. O **arquivo cfg** ficará assim:

```
input_path: "/home/USER/caso/input"
geo_path: "/home/USER/caso/geo"
output_path: "/home/USER/caso-saida/output"
restart_path: "/home/USER/caso-saida/restart"
probes_path: "/home/USER/caso/probes"
```

Suponhamos agora que não queremos usar duas pastas e queremos manter tudo num local só. Então o caso inteiro, arquivos de entrada e saída devem ficar em `/home/USER/caso`. Nesse cenário o cfg ficará assim:

```
input_path: "/home/USER/caso/input"
geo_path: "/home/USER/caso/geo"
output_path: "/home/USER/caso/output"
restart_path: "/home/USER/caso/restart"
probes_path: "/home/USER/caso/probes"
```

Mais detalhes de como montar o arquivo cfg, considerando os detalhes de cada *stack*, então na seção 3.

2.5 Repositório de casos

Com o uso do arquivo cfg é possível setar casos literalmente em qualquer lugar do sistema de arquivos. Desde que o MFSim tenha acesso ao local onde está o caso e os caminhos das pastas estejam devidamente configurados no arquivo cfg, o MFSim conseguirá achar o caso como deveria.

Isso é muito importante porque desacopla o código fonte do MFSim dos casos de uso. Ao mesmo tempo, permite o uso de repositórios de caso.

Nesse sentido, o MFLab possui um repositório de casos no seu git. Se você possui as credenciais de acesso, pode clonar o repositório que está em `https://www.mflab.mecanica.ufu.br/gitea/root/MFSim-cases.git`.

Esse repositório está organizado em 3 pastas principais:

- **dev**: onde estão os casos usados para o desenvolvimento e teste de novas funcionalidades
- **master**: onde estão os casos validados com o branch master do MFSim-cmake
- **to_be_ported**: onde estão casos antigos ainda pendentes de serem portados e testados no MFSim

Os casos que você está desenvolvendo, **devem obrigatoriamente ir para a pasta dev**. Ou seja, para adicionar um novo caso ao repositório, MFSim-cases, acesse a pasta `dev` e dentro dela crie uma pasta que conterá o seu caso.

Outra **regra** no uso do repositório de casos é **não subir** arquivos de malha no mesmo, visto que esses arquivos são binários e podem chegar a dezenas de megabytes, prejudicando o desempenho de todo o repositório.

Por este motivo, se o seu caso for utilizar arquivos de malha (os que vão dentro da pasta `geo`), **você precisa hospedar esses arquivos em outro repositório**. Se você possui acesso a filegator do MFLab, você pode usá-lo para isso. Caso contrário, precisará hospedar esse (e qualquer outro arquivo binário gigante que for usar) em outro servidor e deixar instruções no caso de como obter esses arquivos depois.

Agora que as regras de uso do MFSim-cases estão explicadas, vamos exemplificar um pouco o uso do mesmo, para fixarmos a ideia.

Suponha que deseja guardar no repositório de casos, o casoX que você está desenvolvendo e que não usa arquivos de malha. Nessa situação, você criará dentro da pasta `dev` do repositório de casos, uma pasta chamanda casoX contendo as pastas `input` e `probes` e comitará essas modificações. Considere nesse exemplo que o MFSim-cases está em `/home/USER/MFSim-cases`. Os comandos para executar a sequência descrita (com explicações indicadas por ¡-) são:

```
$ cd /home/USER/MFSim-cases     <- acessa a pasta onde está o MFSim-cases
$ cd dev                        <- acessa a pasta dev
$ mkdir casoX && cd casoX       <- cria e acessa a pasta do caso
$ mkdir input                   <- cria a pasta para os arquivos *.amr3d
$ mkdir probes                  <- cria a pasta de probes
```

Agora é só criar e configuar os arquivos de input descritos na seção 2.2 e configurar as sondas (probes) que precisar.

Vamos mudar um pouco o exemplo anterior e adicionar arquivos de malha. Agora o casoX precisará utilizar fronteira imersa e por conta disso, terá objetos em que a malha será fornecida o MFSim via arquivo *.stl.

Para essa situação, o processo de criação do caso será praticamente o mesmo: acessa o MFSim-cases ¿ acessa a pasta dev ¿ cria as pastas `input` e `probes`. Porém, precisamos da pasta `geo`, que o MFSim-cases **não rastreia***. Logo, iremos criá-la também e depois subir essa pasta no filegator para que outros consigam acessar os arquivos *.stl que estamos trabalhando:

```
$ cd /home/USER/MFSim-cases     <- acessa a pasta onde está o MFSim-cases
$ cd dev                        <- acessa a pasta dev
$ mkdir casoX && cd casoX       <- cria e acessa a pasta do caso
$ mkdir input                   <- cria a pasta para os arquivos *.amr3d
$ mkdir probes                  <- cria a pasta de probes
$ mkdir geo                     <- cria a pasta para os *.stl
```

Antes de continuar vamos entender o que significa "o MFSim-cases não rastreia a pasta geo". O MFSim-cases, assim como o MFSim-cmake (onde está o código fonte do MFSim) são ambos repositórios git, que é um software de gestão de código fonte (assim como o svn). Esse tipo de software não foi feito para armazenar arquivos binários e especialmente, arquivos muito grandes, sendo que sua performance cai consideravelmente quando essa regra "não escrita" é ignorada.

É por conta disso que separamos os arquivos *.stl do repositório de casos. O MFLab dispõe de um servidor de dados que utiliza um software feito especialmente para troca de arquivos, independente de seu tamanho, o filegator. Logo, se o seu caso precisar utilizar arquivos *.stl ou qualquer outro arquivo que ultrapasse os 5 MB's, **você deve subir esse arquivo fora do MFSim-cases**, seja no filegator, se tiver acesso, seja em outro local (Google Drive, Dropbox, etc). Evidentemente, precisará informar isso no caso, visto que outras pessoas depois poderão testar o seu caso e precisarão saber onde estão os arquivos de malha ou outros arquivos externos que porventura seu caso precise.

Agora que entendemos isso, vamos continuar o exemplo. Suponha que seu caso utilizará o arquivo `malha.stl`. Você precisará então colocar esse arquivo na pasta geo que criamos anteriormente

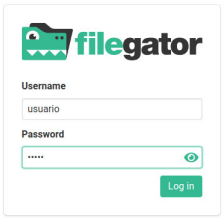
```
/home/USER/MFSim-cases/dev/casoX
                            /input
                            /geo
                                /malha.stl
                            /probes
```

O MFSim-cases ignorará a pasta `geo` inteira, quando comitar o caso, porém o MFSim conseguirá acessar a pasta e utilizar o arquivo `malha.stl` para rodar a simulação.

2.5.1 Tutorial filegator

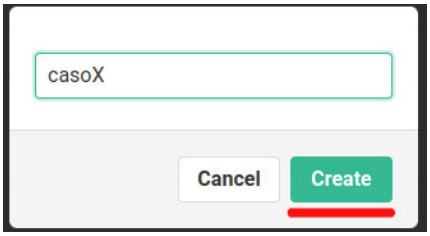
Falta agora enviar a pasta `geo` para o filegator, de forma a permitir que outros também consigam rodar seu caso posteriormente. Para isso siga a sequência

1. Acesse o filegator em seu site `https://www.mflab.mecanica.ufu.br/filegator/#/` e faça o login

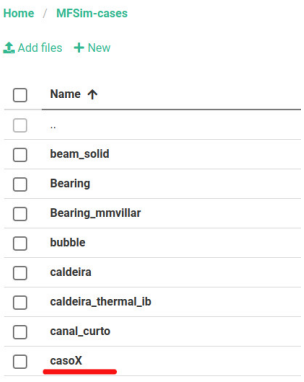


2. Acessa a pasta MFSim-cases

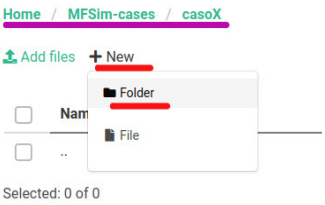
3. Crie uma pasta com o nome do seu caso



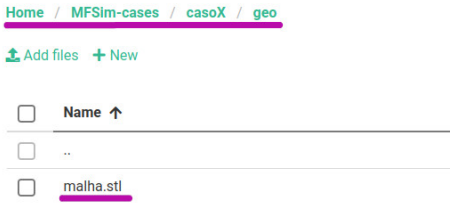
4. Acesse a pasta do seu caso



5. Crie a pasta geo



6. Acesse a pasta geo e envie o arquivo de malha



Pronto, seus arquivos de malha estão devidamente guardados no filegator para uso posterior. Agora você só precisa informar na pasta do caso onde estão os arquivos.

Isso pode ser feito adicionando um arquivo de texto à pasta do caso. Uma boa prática é adicionar um arquivo `README.md` a pasta. Considerando o exemplo que estamos trabalhando, o arquivo ficará em

```
/home/USER/MFSim-cases/dev/casoX
                        /input
                        /geo
                            /malha.stl
                        /probes
                        /README.md    <- o arquivo vai dentro da pasta do caso
```

Dentro do `README.md` você pode descrever onde estão os arquivos de malha, como também pode colocar outras instruções que considerar pertinentes ao seu caso. Exemplo de arquivo `README.md`:

```
## Arquivos de malha

Os arquivos de malha do casoX estão no filegator, na pasta MFSim-cases/casoX/geo

## Instruções para execução

O casoX não roda bem com múltiplos processadores no eixo Z, então evite mudar a distribuição dos
↪  processos incrementando nesse eixo

## Base teórica

O casoX foi gerado a partir do artigo Fulano, do autor Ciclano, com modificações nos parâmetros A, B,
↪  C por conta disso, disso e daquilo
```

Enfim, coloque no `README.md` **quaisquer informações** que considerar importantes outras pessoas saberem sobre seu caso.

2.5.2 Regras MFSimcases

Por fim, vamos sumarizar as regras do repositório de casos:

1. **não é necessário criar branchs**. Por ser apenas um repositório de casos, basta criar uma pasta dentro da pasta `dev` para o seu caso e comitar no branch master
2. pastas com saída de dados, como a `output` e `restart`, **não devem ser adicionadas a esse repositório**. A ideia aqui é apenas guardar a setagem de casos
3. A pasta `geo` não é rastreada pelo repositório. Então embora você possa a utilizar tranquilamente em sua máquina, qualquer arquivo que colocar nela deve ser hospedado fora, quer no filegator, quer em outro servidor.

2.6 Como realmente seto o meu caso?

O procedimento básico para setar um caso no MFSim consiste em:

1. Clonar o MFSim-cases, caso já não o tenha feito;
2. Preencher o arquivo de entrada de dados *input.amr3d*, definindo os parâmetros da simulação;
3. Preencher os arquivos da pasta *input* que serão utilizados na simulação. Por exemplo, se o seu caso utilizará fronteiras imersas, você precisará editar o arquivo *ib.amr3d*, também na pasta *input*, configurando as fronteiras. Caso a simulação seja não isotérmica, as configurações relativas à equação da energia devem ser realizadas no arquivo *energy.amr3d*, e assim com todos os parâmetros necessários à sua simulação;
4. Caso haja interesse de se posicionar sondas no domínio, deve-se alterar o arquivo *probesSTL.amr3d* na pasta *input* informando onde estão os arquivos de configuração das sondas, que por sua vez irão na pasta *probes*;
5. Caso sua simulação utilize arquivos de malha você deve inseri-los na pasta *geo* em um dos formatos suportados e informar ao arquivo no arquivo de configuração do método (*ib.amr3d* por exemplo), o nome do arquivo de malha. Não se esqueça de disponibilizar esses arquivos de malha em algum repositório externo ao MFSim-cases, como o filegator ou outro repositório em que tiver acesso;
6. Definir as condições de contorno e condições iniciais no arquivo *bc_functions.f90* na pasta *input*;
7. Definir o arquivo cfg com a configuração de pastas de entrada e saída de dados;

Uma vez que tenha passado por este procedimento, está pronto para rodar a simulação. Esse é o assunto da próxima seção.

3 Rodando Caso

3.1 Definições

Para executar o MFSim é necessário que o mesmo já esteja instalado em sua máquina. Caso não tenha feito isso ainda, sugiro que pegue o manual de instalação no site oficial do MFSim em `https://www.mflab.mecanica.ufu.br/mfsim/` e realize a instalação do mesmo antes de continuar.

3.1.1 Executando o MFSim

Com o caso setado e cfg montado, só nos resta agora executar o MFSim. Nesse sentido precisamos chamar o binário do MFSim, que estará na pasta `bin` de onde ele foi instalado, pelo lançador do MPI, informando a quantidade de processos que a simualação precisa e o arquivo cfg.

O MPI é utilizado para permitir a execução paralela do MFSim e todos os ambientes descritos no manual de instalação contém uma versão de MPI disponível. Embora não seja necessário é **altamente recomendável utilizar o MPI até para casos seriais**.

No geral, a execução do MFSim se dá numa sequência parecida com:

```
$ cd PASTA_DE_INSTALAÇÃO/bin
$ mpirun -n (número de processos) ./amr3d arquivo_cfg
```

Evidentemente há mais detalhes pertinentes a cada ambiente. Por isso nas duas seções seguintes iremos expor como compilar e executar o MFSim nos dois ambientes que escolhemos para exemplificação nesse manual.

3.2 Windows: Docker

Como apresentado no manual de instalação, o único ambiente suportado para o Microsof Windows é o **Docker**. Nele há duas formas de uso: a interativa e em lotes.

A execução interativa permite a interferência do usuário na execução, sendo particularmente útil quando é necessário acompanhar passo a passo a execução do MFSim (numa execução de teste por exemplo). Já a execução em lotes não permite a interação do usuário, ou seja, você dispara a execução e deixa ela rodando sozinha até terminar ou parar por outro motivo.

Antes de prosseguir, vamos entender como o **Docker** trabalha. De maneira simples, o **Docker** funciona instalando uma **imagem**, que é um pacotão com bibliotecas e softwares na sua máquina, e depois executando essa imagem dentro de um ambiente controlado, que ele chama de **container**.

Logo, para usar o MFSim no Windows você precisará ter uma **imagem** Docker devidamente instalada em sua máquina (caso tenha perdido essa parte, veja no manual de instalação no site oficial: `https://www.mflab.mecanica.ufu.br/mfsim/`), que então será utilizada para gerar os containers, onde conseguirá executar o MFSim. Essa parte de gerar e usar os containers é o que mostraremos nessa seção.

Uma última observação a se fazer é que utilizaremos dois terminais diferentes. Um é o PowerShell que utilizaremos no Windows (não o CMD!) e o outro é Bash, que utilizaremos dentro do container. Para fazer diferença entre ambos adotaremos marcadores no início da linha. **Quando o marcado for ¿ (sinal de maior) é um comando PowerShell**. Já quando o marcado for um **# (cerquilha/sharp/jogo da velha) é um comando Bash**.

3.2.1 Forma interativa

Vamos começar vendo como gerar um container de forma interativa, ou seja, que você poderá interagir executando comandos. Para isso precisaremos instanciar o **container** docker, a partir da imagem MFLab, informando onde está o código do MFSim e onde ele será executado. Faremos isso executando um comando parecido com o mostrado a seguir:

```
docker run -it -v PATH_CASO:/mflab -v PATH_INSTALACAO:/mfsim mflab_image /bin/bash
```

Esse comando gerará o container e nos entregará o terminal, para executarmos o MFSim, o que será algo parecido com:

```
$ mpirun -n (número de processos) --allow-run-as-root ./amr3d arquivo_cfg 2>err
```

Agora, vamos aplicar o que vimos nessa seção a um exemplo

3.2.1.1 Executando uma simulação

Suponha que sua precisará de 8 processos, que o caso está setado em `C:\Users\mflab\Desktop\MFSIM-caso\caso1` e o MFSim instalado em `C:\Users\mflab\Desktop\MFSIM\install`.

Adaptando o comando de geração de container para este exemplo, executaremos **no PowerShell do Windows** o seguinte:

```
[PowerShell] > docker run -it -v C:\Users\mflab\Desktop\MFSIM-caso\caso1:/mflab -v
↪  C:\Users\mflab\Desktop\MFSIM\install:/mfsim mflab_image /bin/bash
```

Explicando o comando acima:

- *docker*: chama o docker diretamente;
- *run*: comando do docker que indica que quer executar um container;
- *-it*: opção para indicar que irá executar em modo interativo, ou seja, irá interagir com o container diretamente (há outras formas de execução);
- *-v*: indica que irá montar um volumer virtual, uma forma de possibilitar que o container docker acesse um diretório da sua máquina;
- `C:\Users\mflab\Desktop\MFSIM-caso\caso1`: esse é caminho do diretório na sua máquina onde o caso foi setado;
- */mflab*: esse é o ponto de montagem do diretório do código fonte do MFSim dentro do container docker;
- `C:\Users\mflab\Desktop\MFSIM\install`: esse é caminho do diretório onde o MFSim está instalado e por onde o executará;
- */mfsim*: esse é o ponto de montagem do diretório de execução do MFSim dentro do container docker;
- *mflab_image*: nome da imagem docker a partir da qual está gerando o container;
- */bin/bash*: caminho do terminal que usará dentro do container;

Agora que temos o *container* instanciado. Vamos configurar o arquivo cfg.

Visto que rodaremos o MFSim dentro do *container* e não fora dele, todos os caminhos devem indicar caminhos do docker. Além disso, guardaremos os resultados na mesma pasta onde setamos o caso. Sendo assim, o arquivo cfg ficará:

```
input_path: "/mflab/input"
geo_path: "/mflab/geo"
output_path: "/mfsim/output"
restart_path: "/mfsim/restart"
probes_path: "/mflab/probes"
```

É **muito importante** entender essa questão dos caminhos dentro e fora do *container*. Da forma como o instanciamos, o caminho interno `/mflab` aponta para o caminho externo `C:\Users\mflab\Desktop\MFSIM-caso\caso1`, assim como o `/mfsim` aponta para `C:\Users\mflab\Desktop\MFSIM\install`.

Sendo assim, durante a execução, que ocorre obrigatoriamente **dentro do container**, o MFSim executado em `/mfsim` é na verdade o MFSim que está em `C:\Users\mflab\Desktop\MFSIM\install` e que lerá as pastas `input`, `geo` e `probes` dentro de `/mflab` ao mesmo tempo que escreverá nas pastas `output` e `restart` também dentro de `/mflab`. Contudo, `/mflab` é na verdade `C:\Users\mflab\Desktop\MFSIM-caso\caso1`.

Resumindo:

```
/mflab -> C:\Users\mflab\Desktop\MFSIM-caso\caso1
/mfsim -> C:\Users\mflab\Desktop\MFSIM\install
```

Prosseguindo ...

Por fim precisamos agora apenas disparar o MFSim. Fazemos isso chamando o binário dentro de `/mfsim/bin`:

```
[bash] # cd /mfsim/bin
[bash] # mpirun -n 8 --allow-run-as-root ./amr3d /mflab/caso1.cfg 2>err
```

O primeiro comando da sequência anterior é auto explicável. Estamos acessando a pasta de execução. Já o comando seguinte, o de disparo da execução, vamos detalhar:

mpirun executa o lançador do MPI. O MFSim utiliza MPI para permitir execução paralela, como é o nosso caso aqui

-n 8 indica o número de processos (ou *cores*) que nossa simulação precisa.

–**allow-run-as-root** o Docker no Windows só tem o usuário root. Por isso precisamos dessa flag

./amr3d executa o binário do MFSim propriamente dito.

./mflab/caso1.cfg é caminho **interno do container** para onde está o arquivo cfg do caso

2¿err cria um arquivo *err* para conter o log de erros

3.2.2 Forma não interativa

A segunda forma de execução é a não interativa/em lotes. Essa maneira é muito útil para rodar simulações que demandam muito tempo para terminar, pois você pode disparar a execução e só verificar depois a conclusão da mesma.

Contudo, como não irá interagir com a execução, precisará de meios para saber se ela terminou ou não. Um bom meio, é redirecionar os dados que o MFSim exibe durante sua execução para um arquivo e ir acompanhando esse arquivo de tempos em tempos.

Também precisará mudar a forma de usar o Docker. Ao invés de executá-lo em modo interativo, como fez na seção anterior, precisará executá-lo em modo **não interativo**. Isso implica em instanciar o **container** de forma um pouco diferente. Novamente, vamos abstrair os detalhes de sistema operacional e considerar o comando básico:

```
docker run -v PATH_CASO:/mflab -v PATH_INSTALACAO:/mfsim mflab_image /bin/bash -c
↪  "COMANDO_DE_EXECUÇÃO"
```

O comando acima executará o **container** em modo não interativo e disparará, já em seu início, tudo o que estiver dentro do `-c "COMANDO_DE_EXECUÇÃO"`. Vamos elaborar esse **COMANDO_DE_EXECUÇÃO**.

Conforme o que aprendemos, para executarmos o MFSim precisamos logo após instanciar o **container**, ir para a pasta onde está o binário do MFSim. Como aqui estamos instanciando o **container** e executando o MFSim de uma só vez, então o **COMANDO_DE_EXECUÇÃO** precisará incluir, essa parte de ir até onde o binário do MFSim está:

```
docker run -v PATH_CASO:/mflab -v PATH_INSTALACAO:/mfsim mflab_image /bin/bash -c "cd /mfsim/bin"
```

Além de ir para a pasta `bin`, precisamos também disparar a execução da simulação com o `mpirun`. Vimos na execução interativa, que o comando que faz isso é uma variação de:

```
mpirun -n (número de processos) --allow-run-as-root ./amr3d arquivo_cfg 2>err
```

Então, vamos adicionar esse comando, ao que já tínhamos antes:

```
docker run -v PATH_CASO:/mflab -v PATH_INSTALACAO:/mfsim mflab_image /bin/bash -c "cd /mfsim/bin &&
↪  mpirun -n (número de processos) --allow-run-as-root ./amr3d arquivo_cfg 2>err"
```

Falta agora, redirecionar a saída do MFSim para um arquivo, por onde poderemos acompanhar a execução, sem necessariamente interagir com ela. Como o docker usa o bash e nesse terminal o desvio de fluxo é feito com o operador `>` (sinal de maior), vamos então alterar o comando anterior para incluir essa possibilidade:

```
docker run -v PATH_CASO:/mflab -v PATH_INSTALACAO:/mfsim mflab_image /bin/bash -c "cd /mfsim/bin &&
↪  mpirun -n (número de processos) --allow-run-as-root ./amr3d arquivo_cfg 2>err > out.dat"
```

O arquivo `out.dat` será gerado na mesma pasta em que está o binário do MFSim.

Pronto. Agora temos como executar o MFSim no docker de forma não interativa.

3.2.3 Nomeando containers

É possível atribuir nomes específicos para os *containers*, quer interativos, quer não interativos. Isso é muito útil para a o reaproveitamento de containers, ou seja, ao invés de criar um *container* a cada vez que for rodar uma simulaçaõ no MFSim, você cria um só e o utiliza toda vez que precisar. Esse é o assunto da próxima seção. O desta é como nomear o *container*.

Para nomear o *container*, durante o comando de criação, inclua a instrução `--name NOME_DO_CONTAINER`.

Exemplo interativo:

```
docker run -it -v PATH_CASO:/mflab -v PATH_INSTALACAO:/mfsim --name mfsim_compile mflab_image
↪  /bin/bash
```

Exemplo não interativo:

```
docker run -v PATH_CASO:/mflab -v PATH_INSTALACAO:/mfsim --name mfsim_run mflab_image /bin/bash -c "cd
↪  /mfsim/bin && mpirun -n (número de processos) --allow-run-as-root ./amr3d arquivo_cfg 2>err >
↪  out.dat"
```

3.2.4 Reaproveitando containers

A cada vez que se executa o comando `docker run` é um criado um **container** e espaço em disco é alocado para ele. Quando a tarefa do **container**, quer interativo quer não interativo, terminar, o **espaço alocado para o container não será liberado**. Eventualmente isso pode criar um problema de falta de espaço para a criação de novos containers e inviabilizar o uso do MFSim no docker.

A fim de evitar esse problema, o próprio docker oferece a possibilidade de reaproveitar um **container**. É isso que veremos nessa seção.

Primeiro, vamos verificar quais containers temos disponíveis no docker com o comando:

```
docker ps -a
```

O resultado será parecido com:

```
CONTAINER ID   IMAGE         COMMAND       CREATED       STATUS    PORTS     NAMES
686739c8b631   mflab_image "/bin/bash"   2 days ago    Exited        mflab_container
```

Lembrando que os dados acima são de exemplo e provavelmente estarão diferentes na sua máquina.

Caso o comando não retorne algum resultado, então não há containers a serem reaproveitados no docker e logo, pode pular essa seção. Caso contrário, continue nesta seção.

As informações que nos interessam no resultado do comando são o **STATUS** e o **NAMES**. Um indica se o **container** está executando, parado ou com problemas. O outro é o nome dele. No caso do exemplo, o **container** está parado.

Para reaproveitar o **container**, primeiro precisamos "reativá-lo". Fazemos isso com o comando

```
docker start NOME_DO_CONTAINER
```

Pegando o exemplo anterior, o comando seria:

```
docker start mflab_container
```

O docker irá imprimir o nome do **container** como resultado dessa operação.

Podemos agora verificar o status do **container**, com o `docker ps -a` novamente:

```
docker ps -a
CONTAINER ID   IMAGE         COMMAND       CREATED       STATUS    PORTS     NAMES
686739c8b631   mflab_image "/bin/bash"   2 days ago    Up            mflab_container
```

Temos acima uma saída possível indicando que o **container** está ativo.

Para interagir com o **container**, usamos:

```
docker exec -it NOME_DO_CONTAINER bash
```

Uma vez dentro do **container**, o usamos da mesma maneira que quando criamos o **container** com o `docker run`.

Agora, o **container** em questão pode ser um não interativo. Nesse caso é ainda mais interessante, porque quando o **container** é criado com um comando para execução em lotes, ele fica "travado" com esse comando o tempo todo. Assim, ao executar o `docker exec` simplesmente re-executamos a tarefa para a qual o **container** estava previamente programado. Exemplificando:

Vamos considerar o **container** não interativo usado nos exemplos anteriores:

```
docker run -v PATH_CASO:/mflab -v PATH_INSTALACAO:/mfsim --name mfsim_batch mflab_image /bin/bash -c
↪  "cd /mfsim/bin && mpirun -n (número de processos) --allow-run-as-root ./amr3d arquivo_cfg 2>err >
↪  out.dat"
```

Como pode observar pelo comando de criação, esse container já foi criado com o nome mfsim_batch.
Para executarmos novamente o MFSim de forma não interativa, como está programado no mfsim_batch, executamos:

```
docker start mfsim_batch
```

Pronto. Basta isso para dispararmos a execução do MFSim sem precisarmos fazer qualquer outra coisa.
Por último, falta falarmos como **parar um container em execução**. Essa é a parte mais fácil, porque basta o comando:

```
docker stop NOME_DO_CONTAINER
```

Ao usar esse comando, o seu **container** será parado, independente dele ser interativo ou não interativo.
Agora que vimos as operações de uso do docker, vamos aos detalhes do sistema operacional.

3.2.5 Facilitando ainda mais a vida

Há ainda duas coisas que podemos fazer no caso da execução **não interativa** no Windows. Como o PowerShell está integrado com a interface gráfica do Windows, podemos escrever dois scripts para executar a simulação apenas com dois cliques.
Para isso precisaremos

1. Criar um script PowerShell que ativa e dispara um *container* docker **não interativo** previamente criado;
2. Criar um script CMD que pode ser chamado através da interface gráfica do Windows e que executa o script criado no item anterior;

Vamos chamar esses dois scripts de `start.ps1` e `start.bat`.
Vamos também salvá-los da seguinte maneira:

```
C:\Users\mflab\Desktop\MFSIM\start.bat
C:\Users\mflab\Desktop\MFSIM\start.ps1
```

O script `start.ps1` é o script PowerShell que executará o docker. Vamos considerar que o *container* tem como nome "mfsim_run". Sendo assim, o conteúdo de `start.ps1` será:

```
docker start mfsim_run
```

Agora, vamos criar o script `start.bat` que executará o script acima, através do clique duplo na interface:

```
@ECHO OFF
ECHO Iniciando o docker ...
PowerShell -NoProfile -ExecutionPolicy Bypass -Command "& 'C:\Users\mflab\Desktop\MFSIM\start.ps1'"
PAUSE
```

Pronto. Agora temos como disparar a execução da simulação através de cliques diretamente na interface gráfica do Windows.
Não se esqueça de alterar os diretórios e o nome do *container* docker como precisar.

3.3 Linux: Singularity

Uma das grandes vantagens do singularity é que ele se integra com facilidade ao sistema de arquivo da máquina hospedeira, além de gerar a imagem com os softwares e bibliotecas nessários ao MFSim em um arquivo que pode ser colocado em qualquer local do sistema de arquivos da sua máquina.
Logo, desde que você consiga acessar a imagem, pode fazer o que quiser. Por isso, para **fins de exemplificação** adotaremos um conjunto de *caminhos* aqui que evidentemente precisarão ser adaptados para a sua máquina ao executar os comandos.
Por fim, é importante ressaltar que assim como o docker, o singularity possui tanto a forma interativa, como a não interativa.

3.3.1 Forma interativa

Para executar sua simulação no singularity usando a forma interativa, instancie o **container** singularity a partir da imagem do MFLab:

```
$ singularity shell IMAGEM_SINGULARITY
```

E dentro do **container** instanciado, rode sua simulação.
Vejamos um exemplo de fixação

3.3.1.1 Executando uma simulação

Suponha que sua simulação utilizará 16 processos, que o MFSim está instalado em `~/MFLab/MFSim/install`, que o caso está setado em `~/MFLab/MFSim-casos/caso1` e que a imagem singularity está em `~/MFLab/Singularity/mflab_image.sif`.

O arquivo cfg do caso está em `~/MFLab/MFSim-casos/caso1/caso.cfg` e possui o segunte conteúdo:

```
input_path: "~/MFLab/MFSim-casos/caso1/input"
geo_path: "~/MFLab/MFSim-casos/caso1/geo"
output_path: "~/MFLab/MFSim-casos/caso1/output"
restart_path: "~/MFLab/MFSim-casos/caso1/restart"
probes_path: "~/MFLab/MFSim-casos/caso1/probes"
```

Vamos então instanciar o *container*:

```
$ cd ~/MFLab/MFSim/install
$ singularity shell ~/MFLab/Singularity/mflab_image.sif
```

O primeiro comando da sequência é tranquilo de entender. Estamos acessando a pasta onde o MFSim está instalado. Agora vamos detalhar o comando singularity para instanciar o *container*:

- *singularity*: chama o singularity diretamente;
- *shell*: indica ao singularity que deseja usar o modo interativo;
- `~/MFLab/Singularity/mflab_image.sif`: caminho (PATH) para a imagem do singularity;

Agora que temos o *container* singularity ativo, vamos proceder com a execução do MFSim:

```
Singularity> cd bin
Singularity> source ../etc/mfsim-env.sh
Singularity> mpirun -n 16 ./amr3d ~/MFLab/MFSim-casos/caso1/caso.cfg 2>err
```

O primeiro comando da sequência anterior é auto explicável. Estamos acessando a pasta onde está o binário do MFSim. Já os comandos seguintes, vamos detalhar:

source executa o utilitário de configuração de variáveis de ambiente do bash

../etc/mfsim-env.sh caminho para o script que configura as variáveis de ambiente do MFSim

mpirun executa o lançador do MPI. O MFSim utiliza MPI para permitir execução paralela, como é o nosso caso aqui

-n 16 indica o número de processos (ou *cores*) que nossa simulação precisa.

./amr3d executa o binário do MFSim propriamente dito.

/MFLab/MFSim-casos/caso1/caso.cfg é o caminho para o arquivo cfg do caso

2¿err cria um arquivo *err* para conter o log de erros

3.3.2 Forma não interativa

Agora suponha que precisa rodar uma simulação muito grande. Ela vai precisar de 64 cores e rodará por muito tempo e você quer verificar o resultado de tempos em tempos ao invés de a todo momento. Esse é o cenário ideal para a forma não interativa, ou seja, execução sem a sua interferência.

Novamente vamos utilizar os mesmos caminhos do exemplo interativo, ou seja, o MFSim está instalado em `~/MFLab/MFSim/insta` o caso está setado em `~/MFLab/MFSim-casos/caso1` e a imagem singularity está em `~/MFLab/Singularity/mflab_image.sif`.

Continuando, a sequência de comandos agora será um pouco diferente:

```
$ cd ~/MFLab/MFSim/install/bin
$ source ../etc/mfsim-env.sh
$ mpirun -n 64 singularity exec ~/MFLab/Singularity/mflab_image.sif ./amr3d ~/MFLab/MFSim-casos/caso1/caso.c
```

Essa forma de uso do singularity é particularmente útil. Não apenas porque permite o uso sem interferência, mas porque você pode acabar tendo que rodar o MFSim em um cluster e isto, dependendo do local, será via singularity.

3.4 Linux: Lmod

Embora a *stack* lmod seja a mais complexa de instalar e usar é também a mais versátil. Por ela você consegue facilmente configurar um conjunto de variáveis de ambiente que te permitem utilizar várias versões das mesmas bibliotecas no mesmo sistema operacional. É possível por exemplo, abrir um terminal e utilizar um compilador intel e em outro terminal, ao mesmo tempo, utilizar um compilador gnu.

Evidentemente isso vem a um custo de **carregar os módulos** antes de utilizar. No âmbito do MFSim, serão necessários 4 conjuntos de módulos:

- 1 - compilador C/C++ e fortran, para acesso a libc, libfortra e libstdc++;
- 2 - uma biblioteca MPI;
- 3 - HDF5;
- 4 - Cantera, GSL, zoltan, slepc;

Sendo assim, a carga de módulos será mais ou menos assim:

```
$ ml gnu openmpi hdf5 cantera gsl slepc zoltan
```

É importante ressaltar que o nome e versão dos módulos pode variar de acordo com o ambiente. Por isso é **fortemente recomendado** que você verifique os módulos disponíveis antes de rodar. Isso pode ser feito pelo comando `module avail`.

Uma vez que os módulos estão instanciados, só precisamos disparar o MFSim, o que é feito chamando o MPI e passando o binário do MFSim como parâmetro para ele:

```
$ mpirun -n (número de processos) ./amr3d arquivo_cfg 2>err
```

Novamente, recorreremos ao uso de um exemplo.

3.4.0.1 Executando uma simulação

Suponha que sua simulação utilizará 128 processos, que o MFSim está instalado em `~/MFLab/MFSim/install`, que o caso está setado em `~/MFLab/MFSim-casos/caso1`. O arquivo cfg do caso está em `~/MFLab/MFSim-casos/caso1/caso.cfg` e possui o segunte conteúdo:

```
input_path: "~/MFLab/MFSim-casos/caso1/input"
geo_path: "~/MFLab/MFSim-casos/caso1/geo"
output_path: "~/MFLab/MFSim-casos/caso1/output"
restart_path: "~/MFLab/MFSim-casos/caso1/restart"
probes_path: "~/MFLab/MFSim-casos/caso1/probes"
```

Executando o MFSim

```
$ cd ~/MFLab/MFSim/install/bin
$ source ../etc/mfsim-env.sh
$ mpirun -n 128 ./amr3d ~/MFLab/MFSim-casos/caso1/caso.cfg 2>err > out.dat
```

3.5 Clusters MFLab

Se você tem acesso aos clusters do MFLab, pode ser necessário executar o MFSim em um dos clusters. A execução para esse caso segue praticamente a mesma sequência mostrada para a execução no Linux com lmod mostrada na seção 3.4. Você precisará:

1. carregar os módulos
2. compilar o MFSim caso já não o tenha feito
3. instalar o MFSim caso já não o tenha feito
4. setar o caso e criar o arquivo cfg

A diferença vem em como executar o MFSim. Ao invés de executar o `mpirun` diretamente, você precisará utilizar o PBS para executar o MFSim.

O PBS é um sistema de execução em lotes, ou seja, ele executa programas **sem a interferência do usuário**. Para isso ele realiza todo o trabalho de distribuição do programa ao longo do cluster, executa o programa para onde ele foi alocado e colhe os dados de saída e erros.

Um programa em execução no PBS é chamado de **JOB**. Lembre-se bem desse conceito pois ele será importante em seu uso dos clusters.

Como estamos falando de clusters, ou seja, grupos de vários computadores interconectados via rede de forma a compartilhar recursos, então a execução do MFSim nesse ambiente precisa considerar esse importante detalhe. Nesse sentido, para saber como pedir recursos ao PBS em primeiro lugar, você precisará entender quais recursos o cluster tem disponíveis.

Isso pode ser feito executando o comando `pbsnodes -a` que lista quais computadores (**nós**) existem no cluster e quais os recursos cada um possui. A seguir temos um exemplo da execução desse comando em um dos clusters do MFLab

```
node01
     Mom = node01.cluster.mflab.mecanica.ufu.br
     ntype = PBS
     state = job-busy
     pcpus = 64
     jobs = 8483.euler/0, 8483.euler/1, 8483.euler/2, 8483.euler/3,
       8483.euler/4, 8483.euler/5, 8483.euler/6, 8483.euler/7,
       8483.euler/8, 8483.euler/9, 8483.euler/10, 8483.euler/11,
       8483.euler/12, 8483.euler/13, 8483.euler/14, 8483.euler/15,
       8483.euler/16, 8483.euler/17, 8483.euler/18, 8483.euler/19,
       8483.euler/20, 8483.euler/21, 8483.euler/22, 8483.euler/23,
       8483.euler/24, 8483.euler/25, 8483.euler/26, 8483.euler/27,
       8483.euler/28, 8483.euler/29, 8483.euler/30, 8483.euler/31,
       8483.euler/32, 8483.euler/33, 8483.euler/34, 8483.euler/35,
       8483.euler/36, 8483.euler/37, 8483.euler/38, 8483.euler/39,
       8483.euler/40, 8483.euler/41, 8483.euler/42, 8483.euler/43,
       8483.euler/44, 8483.euler/45, 8483.euler/46, 8483.euler/47,
       8483.euler/48, 8483.euler/49, 8483.euler/50, 8483.euler/51,
       8483.euler/52, 8483.euler/53, 8483.euler/54, 8483.euler/55,
       8483.euler/56, 8483.euler/57, 8483.euler/58, 8483.euler/59,
       8483.euler/60, 8483.euler/61, 8483.euler/62, 8483.euler/63
     resources_available.arch = linux
     resources_available.host = node01
     resources_available.mem = 263588384kb
     resources_available.ncpus = 64
     resources_available.vnode = node01
     resources_assigned.accelerator_memory = 0kb
     resources_assigned.hbmem = 0kb
     resources_assigned.mem = 83886080kb
     resources_assigned.naccelerators = 0
     resources_assigned.ncpus = 64
     resources_assigned.vmem = 0kb
     resv_enable = True
     sharing = default_shared
     last_state_change_time = Thu Mar 30 15:41:25 2023
     last_used_time = Thu Mar 30 15:40:50 2023

node17
     Mom = node17.cluster.mflab.mecanica.ufu.br
     ntype = PBS
     state = free
     pcpus = 20
     resources_available.arch = linux
     resources_available.host = node17
     resources_available.mem = 97399664kb
     resources_available.ncpus = 20
     resources_available.vnode = node17
     resources_assigned.accelerator_memory = 0kb
```

```
resources_assigned.hbmem = 0kb
resources_assigned.mem = 0kb
resources_assigned.naccelerators = 0
resources_assigned.ncpus = 0
resources_assigned.vmem = 0kb
resv_enable = True
sharing = default_shared
last_state_change_time = Sat Feb 25 13:11:27 2023
last_used_time = Fri Mar  3 14:14:48 2023
```

A execução do comando acima mostra dois **nós** computacionais do cluster: o `node01` e o `node17`. Observe que para ambos existem os mesmos atributos, porém os valores são diferentes. Vamos entender esses atributos, pois é a partir deles que você será capaz de verificar o que já de recursos disponíveis no cluster para a execução do seu **JOB**. Por conveniência, os atributos importantes para essa tarefa estão identificados de acordo.

1. `Mom`: nome do nó no PBS
2. `ntype`: software de batch
3. `state`: estado do nó. Pode ser free, down ou job-busy [**Importante!**]
4. `pcpus`: quantidade de cores reais [**Importante!**]
5. `jobs`: lista os jobs que estão executando no nó
6. `resources_available.arch`: sistema operacional
7. `resources_available.host`: nome do nó
8. `resources_available.mem`: quantidade de RAM [**Importante!**]
9. `resources_available.ncpus`: quantidade de cores (virtuais + reais)
10. `resources_available.vnode`: nome virtual do nó
11. `resources_assigned.accelerator_memory`: quantidade de RAM gasta por aceleradores
12. `resources_assigned.hbmem`: memória especial usada (sempre 0 pois depende de hardware específico)
13. `resources_assigned.mem`: RAM em uso [**Importante!**]
14. `resources_assigned.naccelerators`: quantidade de acelearados usados (também 0, pois depende de hardware específico)
15. `resources_assigned.ncpus`: quantidade de cores em uso [**Importante!**]
16. `resources_assigned.vmem`: quantidade de memória virtual em uso
17. `resv_enable`: indica se o nó pode ou não ser reservado
18. `sharing`: política de compartilhamento de dados do PBS
19. `last_state_change_time`: Última vez que o PBS verificou o estado do nó
20. `last_used_time`: Última vez que um JOB rodou no nó

Como indicado acima, as informações mais importantes para o objetivo de executar o MFSim em um cluster, ou seja, **submeter um JOB** no PBS, são: quantidade de cores e RAM disponíveis e utilizados e o estado do nó. O estado desejado para submeter um JOB para o nó é `free`, que indica que o mesmo está disponível. Observe que no exemplo postado anteriormente o nó `node17` está disponível para uso enquanto o nó `node01` não está.

Uma vez que conseguiu indentificar o que pode pedir de recursos nos cluster para o seu JOB, é hora de montar o arquivo PBS, informando os recursos que precisará para executar o MFSim.

De modo geral, os arquivos PBS possuem uma estrutura como a mostrada a seguir:

```
#!/bin/bash
#PBS -N NOME_DO_JOB
#PBS -e NOME_DO_JOB.err
#PBS -o NOME_DO_JOB.out
#PBS -l walltime=TEMPO_DO_JOB
#PBS -l RECURSOS_DO_JOB
#PBS -r n
#PBS -q FILA_DO_JOB

FABRIC=shm:dapl

CORES=$[ `cat $PBS_NODEFILE | wc -l` ]
NODES=$[ `uniq $PBS_NODEFILE | wc -l` ]

cd $PBS_O_WORKDIR

printf "################################################\n";
printf "Current time is: `date`\n";
printf "Current PBS work directory is: $PBS_O_WORKDIR\n";
printf "Current PBS queue is: $PBS_O_QUEUE\n";
printf "Current PBS job ID is: $PBS_JOBID\n";
printf "Current PBS job name is: $PBS_JOBNAME\n";
printf "PBS stdout log is: $PBS_O_WORKDIR/NOME_DO_JOB.err\n";
printf "PBS stderr log is: $PBS_O_WORKDIR/NOME_DO_JOB.out\n";
printf "Fabric interconnect selected is: $FABRIC\n";
printf "This jobs will run on the following ($CORES) processors:\n";
echo `cat $PBS_NODEFILE`

CARGA_DE_MÓDULOS
source ../etc/mfsim-env.sh
mpirun ALGUMAS_CONFIGURAÇÕES --oversubscribe -np NUMERO_DE_CORES $PBS_O_WORKDIR/amr3d ARQUIVO_CFG > out 2>err
```

Explicando ...

1. `#!/bin/bash`: indica que o arquivo PBS é um script bash
2. `#PBS -N NOME_DO_JOB`: informa o nome do job
3. `#PBS -e NOME_DO_JOB.err`: arquivo para o qual o PBS reportará os erros que encontrar
4. `#PBS -o NOME_DO_JOB.out`: arquivo para o qual o PBS reportará o que está acontecendo no job, exceto erros
5. `#PBS -l walltime=TEMPO_DO_JOB`: tempo máximo de execução do job, em horas:minutos:segundos
6. `#PBS -l RECURSOS_DO_JOB`: quantidade de recursos que job precisará, mensurado em nós, processadores e memória por nó
7. `#PBS -r n`: indica para o PBS não tentar reiniciar o job caso ele pare por algum motivo
8. `#PBS -q FILA_DO_JOB`: indica a fila do PBS que seu job pretende usar. A frente vamos detalhar essa informação.
9. O bloco

```
FABRIC=shm:dapl

   CORES=$[ `cat $PBS_NODEFILE | wc -l` ]
   NODES=$[ `uniq $PBS_NODEFILE | wc -l` ]

   cd $PBS_O_WORKDIR

   printf "################################################\n";
   printf "Current time is: `date`\n";
   printf "Current PBS work directory is: $PBS_O_WORKDIR\n";
```

```
printf "Current PBS queue is: $PBS_O_QUEUE\n";
printf "Current PBS job ID is: $PBS_JOBID\n";
printf "Current PBS job name is: $PBS_JOBNAME\n";
printf "PBS stdout log is: $PBS_O_WORKDIR/NOME_DO_JOB.err\n";
printf "PBS stderr log is: $PBS_O_WORKDIR/NOME_DO_JOB.out\n";
printf "Fabric interconnect selected is: $FABRIC\n";
printf "This jobs will run on the following ($CORES) processors:\n";
echo `cat $PBS_NODEFILE`
```

Trás informações sobre o job gerados pelo próprio PBS

10. `CARGA_DE_MÓDULOS`: é onde deve carregar os módulos com o `lmod`, como mostrado na seção 3.4
11. `source ../etc/mfsim-env.sh`: configura as variáveis de ambiente do MFSim no JOB
12. `mpirun ALGUMAS_CONFIGURAÇÕES --oversubscribe -np NUMERO_DE_CORES $PBS_O_WORKDIR/amr3d ARQUIVO_CFG > out` instrui o PBS a como executar o MFSim

Você deve ter percebido que temos uma informação nova agora, a **fila**. Filas são listas de configurações que o PBS usa para gerenciar recursos no cluster. O que cada fila faz varia um pouco entre os clusters e a versão do PBS em cada um, então o que precisa se atentar é que **você não pode pedir mais recursos do que as filas permitem**.

Nesse sentido os clusters possuem algumas filas, cada uma com limites de recursos

Fila	Limites	Observações
serial	1 core	apenas cluster 3
parallel_8	8 cores; 4 JOBs por vez	cluster 3 e 6
parallel_16	16 cores; 4 JOBs por vez	cluster 3 e 6
parallel_32	32 cores; 2 JOBs por vez	todos os clusters
parallel_64	64 cores; 2 JOBs por vez	todos os clusters
parallel_128	128 cores; 2 JOBs por vez	todos os clusters
parallel_256	256 cores; 2 JOBs por vez	todos os clusters
parallel_512	512 cores; 2 JOBs por vez	todos os clusters

Logo, se seu job precisará de até 8 cores, então a fila que você deve utilizar é a `parallel_8`.

Uma última observação são as configurações específicas de cada cluster que precisam ser fornecidas no `mpirun`.

Cluster	Configurações do mpirun

cluster 3	`-bind-to none:overload-allowed`
	`--mca btl tcp,vader,self`
	`--mca btl_tcp_if_exclude lo,eno33,eno1`

cluster 5	`-bind-to none:overload-allowed`
	`--mca btl openib,vader,self`
	`--mca btl_tcp_if_exclude lo,eth0`
	`--mca btl_openib_allow_ib 1`

cluster 6	`-bind-to none:overload-allowed`

Agora que já temos tudo que precisamos, vamos exemplificar o uso do PBS.

Considere que você quer rodar o MFSim no cluster 3. Você baixou o código fonte do MFSim para a pasta `/home/USER/MFSim-cmake` e o o instalou na pasta `/home/USER/st_cluster3/MFSim-install`, usando a opção `-Dprefix` do `cmake` explicada na seção "Escolhendo o local de instalação" no manual de instalação. O arquivo cfg está em `/home/USER/st_cluster3/MFSim-install/bin/arquivo.cfg`. Considere também que seu caso utilizará 16 cores, 50 GB de RAM e precisará rodar por até 100 horas. Vamos então montar o arquivo PBS e submeter esse **JOB**.

```
#!/bin/bash
#PBS -N teste
#PBS -e teste.err
#PBS -o teste.out
#PBS -l walltime=100:00:00
#PBS -l select=1:ncpus=16:mpiprocs=16:mem=50gb
#PBS -r n
```

```
#PBS -q parallel_16

FABRIC=shm:dapl

CORES=$[ `cat $PBS_NODEFILE | wc -l` ]
NODES=$[ `uniq $PBS_NODEFILE | wc -l` ]

cd $PBS_O_WORKDIR

printf "################################################\n";
printf "Current time is: `date`\n";
printf "Current PBS work directory is: $PBS_O_WORKDIR\n";
printf "Current PBS queue is: $PBS_O_QUEUE\n";
printf "Current PBS job ID is: $PBS_JOBID\n";
printf "Current PBS job name is: $PBS_JOBNAME\n";
printf "PBS stdout log is: $PBS_O_WORKDIR/teste.err\n";
printf "PBS stderr log is: $PBS_O_WORKDIR/teste.out\n";
printf "Fabric interconnect selected is: $FABRIC\n";
printf "This jobs will run on the following ($CORES) processors:\n";
echo `cat $PBS_NODEFILE`

CARGA_DE_MÓDULOS
source ../etc/mfsim-env.sh
mpirun --mca btl tcp,vader,self --mca btl_tcp_if_exclude lo,eno33,eno1 --oversubscribe -np 16
↪  $PBS_O_WORKDIR/amr3d /home/USER/st_cluster3/MFSim-install/bin/arquivo.cfg > out 2>err
```

Vamos salvar o conteúdo do script acima no arquivo `/home/USER/st_cluster3/MFSim-install/bin/run-job.pbs`. Com ele salvo, submeteremos o JOB ao PBS com

```
$ cd /home/USER/st_cluster3/MFSim-install/bin
$ qsub run-job.pbs
```

Agora é só aguardar o PBS executar seu JOB.

4 Casos de Teste

O MFSim disponibiliza vários casos de teste e verificação na pasta *tests*. A exceção do primeiro caso, que é um teste automatizado, todos os demais podem ser setados e executados manualmente pelo usuário, servindo assim de tutorial.

É possível também executar cada um dos testes utilizando a ferramenta de testes disponibilizada pelo MFSim. Sendo assim, um caminho recomendado é primeiro executar cada teste via ferramenta, para ver como funciona, e depois setar e executar cada caso manualmente.

4.1 Uso de solução manufaturadas

O primeiro caso de teste utiliza soluções manufaturadas para validar o MFSim como um todo, sendo portanto um caso extremamente interessante, visto que se esse teste falhar, a instalação do MFSim estará incorreta. Por este mesmo motivo, esse é o único caso que é setado automaticamente pelo próprio MFSim, não requerendo do usuário intervenção, exceto, a própria execução do teste.

O código desse teste está disponível em *tests/manuf*.

O procedimento básico deste caso envolve a definição de uma solução manufaturada **para cada variável do problema, de forma que cada variável está presente nas equações diferenciais do escoamento**.

Trata-se de um problema com as variáveis u, v, w, p, ρ e μ. As variáveis são definidas pelas seguintes soluções manufaturadas:

$$p_e = \cos\left(\omega_1 \pi x + \omega_2 \pi y + \omega_3 \pi z + \omega_4 t\right) \tag{1}$$

$$u_e = \sin^2\left(\omega_1 \pi x + \omega_2 \pi y + \omega_3 \pi z + \omega_4 t\right) \tag{2}$$

$$v_e = \cos^2\left(\omega_1 \pi x + \omega_2 \pi y + \omega_3 \pi z + \omega_4 t\right) \tag{3}$$

$$w_e = \omega_1 \left(\cos\left(\omega_1 \pi x + \omega_2 \pi y + \omega_3 \pi z + \omega_4 t\right)^2\right)/\omega_3 + \omega_2 \left(\cos\left(\omega_1 \pi x + \omega_2 \pi y + \omega_3 \pi z + \omega_4 t\right)^2\right)/\omega_3 \tag{4}$$

$$\rho_e = 1.0 + 0.1\left(\sin\left(\omega_1 \pi x + \omega_2 \pi y + \omega_3 \pi z + \omega_4 t\right)\right)^2 \tag{5}$$

$$\mu_e = 1.0 + 0.2\left(\cos\left(\omega_1 \pi x + \omega_2 \pi y + \omega_3 \pi z + \omega_4 t\right)\right)^2 \tag{6}$$

onde $\omega_1 = \omega_2 = \omega_3 = 2$ e $\omega_4 = 1$

O termo forçante é o responsável por fazer a solução numérica se aproximar da solução exata, sendo este termo acrescentado na equação do momentum, tal que:

$$\rho \frac{\partial u}{\partial t} + \rho u \nabla . u = -\nabla p + \nabla\left(\mu\left(\nabla u + \nabla u^T\right)\right) + f \tag{7}$$

onde f é o termo forçante, e sua solução é obtida ao substituir na Eq.: $u = u_e, v = v_e, w = w_e, p = p_e, \mu = \mu_e$ e $\rho = \rho_e$.

O uso de soluções manufaturadas periódicas faz com que a visualização dessas variáveis seja periódica conforme o comportamento das funções seno e cosseno.

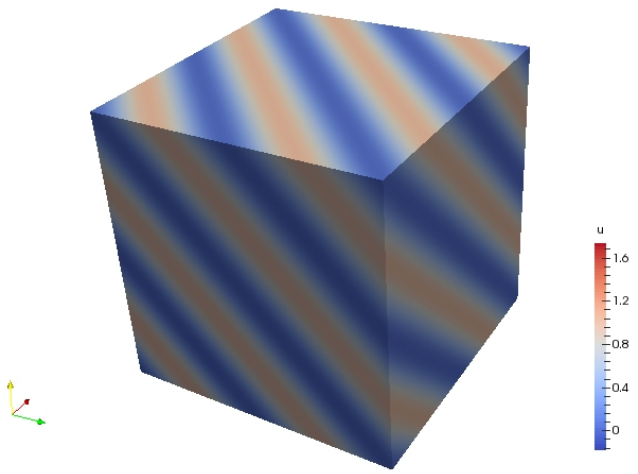


Figure 4: Periodiocidade da solução manufatura por campo de velocidade U

A solução numérica da equação 7 é então obtida em uma sequência de malhas, onde a norma L2 e a norma infinito permitem avaliar a ordem de convergência do método, através do termo forçante. Pelo fato de se utilizar um método de segunda ordem para a discretização espacial e pequenos passos no tempo, pode-se afirmar que a ordem de convergência a ser obtida deve ser de segunda ordem, pois os erros da discretização temporal não prevalecem devido ao pequeno passo no tempo.

A taxa de convergência para as velocidades é então avaliada pela razão entre os erros da solução exata e numérica.

São testadas as taxas de convergência de oito diferentes casos, sendo cada caso avaliado por malhas de diferentes refinamentos.

4.1.1 Caso 1

Este caso é executado em quatro processos em paralelo, utilizando a seguinte malha:

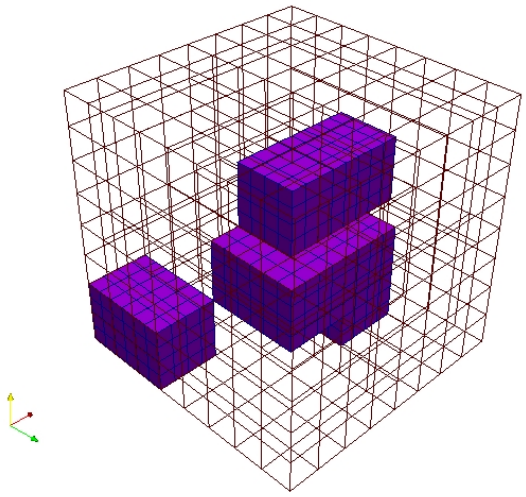


Figure 5: Bloco utilizado no patch 1

A seguir são apresentados os resultados para a norma L2.

Table 1: Norma L2 e razão de convergência

Malha	$\| u \|_2$	r_u	$\| v \|_2$	r_v	$\| w \|_2$	r_w
8X8X8 L2	7.930E-2	-	7.937E-2	-	1.467E-1	-
16X16X16 L2	1.818E-2	4.36	1.822E-2	4.36	3.390E-2	4.35
32X32X32 L2	4.485E-3	4.05	4.494E-3	4.05	8.377E-3	4.05
64X64X64 L2	1.119E-3	4.01	1.121E-3	4.01	2.084E-3	4.02
128X128X128 L2	2.800E-4	4.00	2.802E-4	4.00	5.171E-4	4.03

A seguir são apresentados os resultados para a norma infinito.

Table 2: Norma ∞ e razão de convergência

Malha	$\| u \|_\infty$	r_u	$\| v \|_\infty$	r_v	$\| w \|_\infty$	r_w
8X8X8 L2	2.003E-1	-	1.983E-1	-	2.959E-1	-
16X16X16 L2	4.744E-2	4.22	4.647E-2	4.27	7.060E-2	4.19
32X32X32 L2	1.216E-2	3.90	1.179E-2	3.94	1.889E-2	3.74
64X64X64 L2	3.132E-3	3.88	3.083E-3	3.82	4.785E-3	3.95
128X128X128 L2	8.143E-4	3.85	8.154E-4	3.78	1.201E-3	3.98

4.1.2 Caso 2

Nesse caso, o código é executado com um processo. A seguir ilustra-se a malha utlizada.

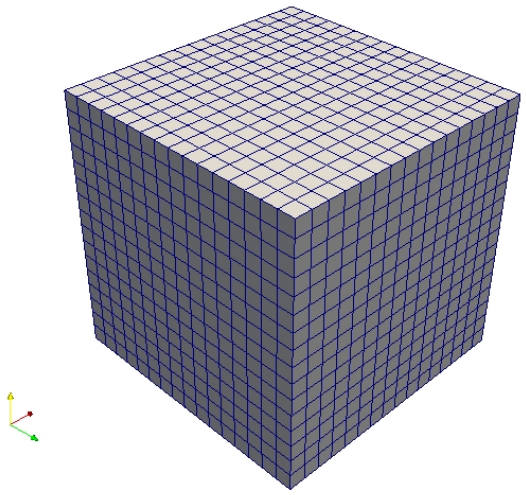


Figure 6: Bloco utilizado no patch 3

A seguir apresenta-se os resultados para a norma L2.

Table 3: Norma L2 e razão de convergência

Malha	$\| u \|_2$	r_u	$\| v \|_2$	r_v	$\| w \|_2$	r_w
8X8X8 L2	8.327E-2	-	8.326E-2	-	1.608E-1	-
16X16X16 L2	1.925E-2	4.33	1.925E-2	4.33	3.717E-2	4.33
32X32X32 L2	4.763E-3	4.04	4.763E-3	4.04	9.189E-3	4.04
64X64X64 L2	1.192E-3	4.00	1.192E-3	4.00	2.296E-3	4.00
128X128X128 L2	2.982E-4	4.00	2.982E-4	4.00	5.742E-4	4.00

A seguir apresenta-se os resultados para a norma ∞.

Table 4: Norma ∞ e razão de convergência

Malha	$\| u \|_\infty$	r_u	$\| v \|_\infty$	r_v	$\| w \|_\infty$	r_w
8X8X8 L2	1.864E-1	-	1.876E-1	-	2.798E-1	-
16X16X16 L2	4.235E-2	4.40	4.237E-2	4.43	7.125E-2	3.93
32X32X32 L2	1.127E-2	3.76	1.127E-2	3.76	1.906E-2	3.74
64X64X64 L2	3.036E-3	3.71	3.026E-3	3.71	4.812E-3	3.96
128X128X128 L2	7.805E-4	3.89	7.772E-4	3.89	1.206E-3	3.99

4.1.3 Caso 3

Nesse caso, o código é executado com um processo. A seguir ilustra-se a malha utilizada nesse caso.

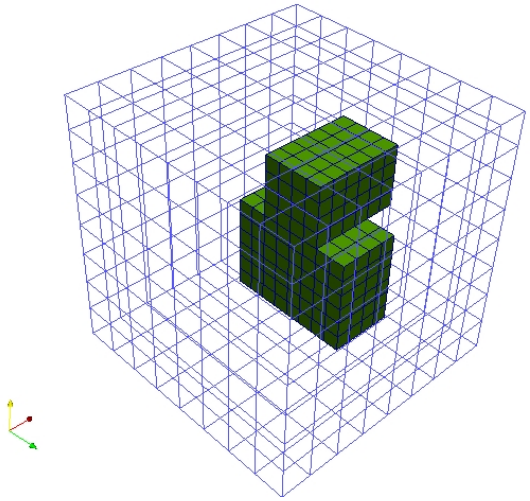


Figure 7: Bloco utilizado no patch 4

A seguir apresenta-se os resultados para a norma L2.

Table 5: Norma L2 e razão de convergência

Malha	$\| u \|_2$	r_u	$\| v \|_2$	r_v	$\| w \|_2$	r_w
8X8X8 L2	8.102E-2	-	8.245E-2	-	1.528E-1	-
16X16X16 L2	1.865E-2	4.34	1.891E-2	4.36	3.531E-2	4.33
32X32X32 L2	4.609E-3	4.05	4.666E-3	4.05	8.730E-3	4.04
64X64X64 L2	1.150E-3	4.01	1.166E-3	4.00	2.175E-3	4.01
128X128X128 L2	2.872E-4	4.00	2.924E-4	3.99	5.404E-4	4.02

A seguir apresenta-se os resultados para a norma infinito.

Table 6: Norma ∞ e razão de convergência

Malha	$\| u \|_\infty$	r_u	$\| v \|_\infty$	r_v	$\| w \|_\infty$	r_w
8X8X8 L2	2.039E-1	-	2.000E-1	-	2.818E-1	-
16X16X16 L2	4.400E-2	4.63	4.426E-2	4.52	7.113E-2	3.96
32X32X32 L2	1.127E-2	3.90	1.195E-2	3.70	1.905E-2	3.73
64X64X64 L2	3.035E-3	3.71	3.101E-3	3.85	4.812E-3	3.96
128X128X128 L2	8.056E-4	3.77	8.183E-4	3.79	1.205E-3	3.99

4.1.4 Caso 4

Nesse caso, o código é executado com um processo. A seguir ilustra-se a malha utlizada.

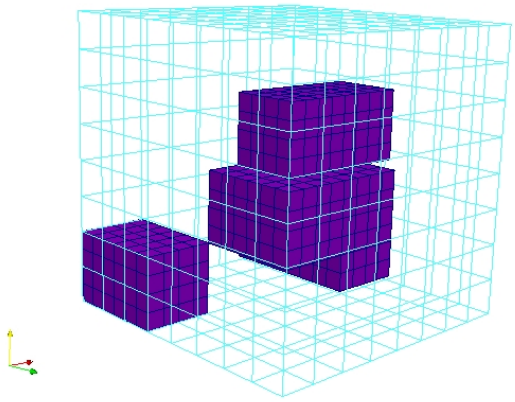


Figure 8: Bloco utilizado no patch 5

A seguir apresenta-se os resultados para a norma L2.

Table 7: Norma L2 e razão de convergência

Malha	$\| u \|_2$	r_u	$\| v \|_2$	r_v	$\| w \|_2$	r_w
8X8X8 L2	7.930E-2	-	7.942E-2	-	1.466E-1	-
16X16X16 L2	1.818E-2	4.36	1.822E-2	4.36	3.389E-2	4.33
32X32X32 L2	4.484E-3	4.05	4.494E-3	4.05	8.376E-3	4.05
64X64X64 L2	1.119E-3	4.01	1.121E-3	4.01	2.084E-3	4.02
128X128X128 L2	2.800E-4	4.00	2.802E-4	4.00	5.171E-4	4.03

A seguir apresenta-se os resultados para a norma infinito.

Table 8: Norma ∞ e razão de convergência

Malha	$\| u \|_\infty$	r_u	$\| v \|_\infty$	r_v	$\| w \|_\infty$	r_w
8X8X8 L2	2.006E-1	-	1.981E-1	-	2.961E-1	-
16X16X16 L2	4.739E-2	4.23	4.651E-2	4.26	7.061E-2	4.19
32X32X32 L2	1.215E-2	3.90	1.180E-2	3.94	1.889E-2	3.74
64X64X64 L2	3.130E-3	3.88	3.079E-3	3.83	4.785E-3	3.95
128X128X128 L2	8.144E-4	3.84	8.149E-4	3.78	1.202E-3	3.98

4.1.5 Caso 5

Nesse caso, o código é executado com quatro processos em paralelo. A seguir ilustra-se a malha utlizada.

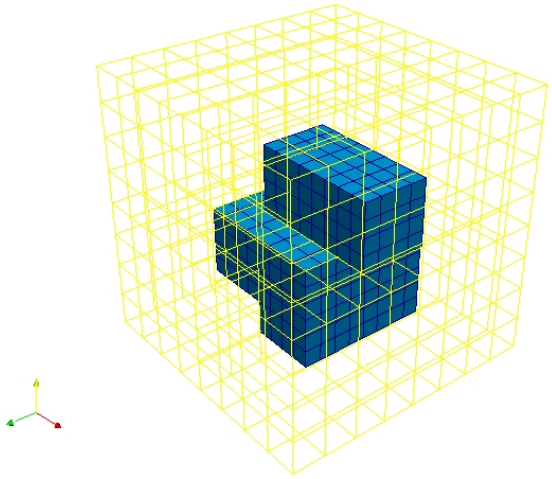


Figure 9: Bloco utilizado no patch 6

A seguir apresenta-se os resultados para a norma L2.

Table 9: Norma L2 e razão de convergência

Malha	$\| u \|_2$	r_u	$\| v \|_2$	r_v	$\| w \|_2$	r_w
8X8X8 L2	8.057E-2	-	8.027E-2	-	1.491E-1	-
16X16X16 L2	1.851E-2	4.35	1.846E-2	4.35	3.449E-2	4.32
32X32X32 L2	4.570E-3	4.05	4.561E-3	4.05	8.528E-3	4.04
64X64X64 L2	1.142E-3	4.00	1.138E-3	4.01	2.123E-3	4.02
128X128X128 L2	2.871E-4	3.98	2.849E-4	3.99	5.283E-4	4.02

A seguir apresenta-se os resultados para a norma infinito.

Table 10: Norma ∞ e razão de convergência

Malha	$\| u \|_\infty$	r_u	$\| v \|_\infty$	r_v	$\| w \|_\infty$	r_w
8X8X8 L2	2.015E-1	-	1.987E-1	-	2.962E-1	-
16X16X16 L2	4.472E-2	4.51	4.389E-2	4.53	7.087E-2	4.18
32X32X32 L2	1.155E-2	3.87	1.169E-2	3.75	1.901E-2	3.73
64X64X64 L2	3.049E-3	3.79	3.083E-3	3.79	4.808E-3	3.95
128X128X128 L2	8.177E-4	3.73	8.161E-4	3.78	1.205E-3	3.99

4.1.6 Caso 6

Nesse caso, o código é executado com um processo. A seguir ilustra-se a malha utlizada.

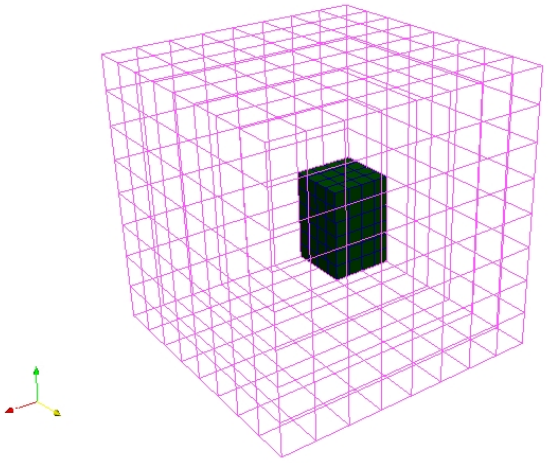


Figure 10: Bloco utilizado no patch 7

A seguir apresenta-se os resultados para a norma L2.

Table 11: Norma L2 e razão de convergência

Malha	$\| u \|_2$	r_u	$\| v \|_2$	r_v	$\| w \|_2$	r_w
8X8X8 L2	8.238E-2	-	8.237E-2	-	1.588E-1	-
16X16X16 L2	1.901E-2	4.33	1.901E-2	4.33	3.668E-2	4.33
32X32X32 L2	4.699E-3	4.05	4.699E-3	4.05	9.064E-3	4.05
64X64X64 L2	1.175E-3	4.00	1.175E-3	4.00	2.264E-3	4.00
128X128X128 L2	2.941E-4	4.00	2.940E-4	4.00	5.660E-4	4.00

A seguir apresenta-se os resultados para a norma infinito.

Table 12: Norma ∞ e razão de convergência

Malha	$\| u \|_\infty$	r_u	$\| v \|_\infty$	r_v	$\| w \|_\infty$	r_w
8X8X8 L2	1.860E-1	-	1.872E-1	-	2.808E-1	-
16X16X16 L2	4.439E-2	4.19	4.474E-2	4.18	7.045E-2	3.99
32X32X32 L2	1.130E-2	3.93	1.138E-2	3.93	1.884E-2	3.74
64X64X64 L2	3.041E-3	3.71	3.031E-3	3.75	4.782E-3	3.94
128X128X128 L2	7.814E-4	3.89	7.781E-4	3.89	1.201E-3	3.98

4.1.7 Caso 7

Nesse caso, o código é executado em quatro processos em paralelo. A seguir ilustra-se os blocos utlizados nesse caso.

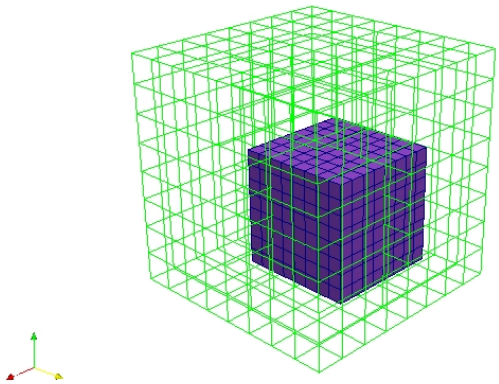


Figure 11: Bloco utilizado no patch 8

A seguir apresenta-se os resultados para a norma L2.

Table 13: Norma L2 e razão de convergência

Malha	$\| u \|_2$	r_u	$\| v \|_2$	r_v	$\| w \|_2$	r_w
8X8X8 L2	7.847E-2	-	8.044E-2	-	1.479E-1	-
16X16X16 L2	1.811E-2	4.33	1.849E-2	4.35	3.438E-2	4.30
32X32X32 L2	4.477E-3	4.05	4.573E-3	4.04	8.510E-3	4.04
64X64X64 L2	1.118E-3	4.00	1.143E-3	4.00	2.119E-3	4.02
128X128X128 L2	2.798E-4	4.00	2.866E-4	3.99	5.266E-4	4.02

A seguir apresenta-se os resultados para a norma infinito.

Table 14: Norma ∞ e razão de convergência

Malha	$\| u \|_\infty$	r_u	$\| v \|_\infty$	r_v	$\| w \|_\infty$	r_w
8X8X8 L2	1.972E-1	-	2.029E-1	-	2.977E-1	-
16X16X16 L2	4.393E-2	4.49	4.507E-2	4.50	7.082E-2	4.20
32X32X32 L2	1.128E-2	3.89	1.135E-2	3.97	1.903E-2	3.72
64X64X64 L2	3.040E-3	3.71	3.033E-3	3.74	4.809E-3	3.96
128X128X128 L2	7.814E-4	3.89	7.790E-4	3.89	1.205E-3	3.99

4.1.8 Caso 8

Nesse caso, o código é executado em apenas um processo. A seguir ilustra-se os blocos utlizados nesse caso.

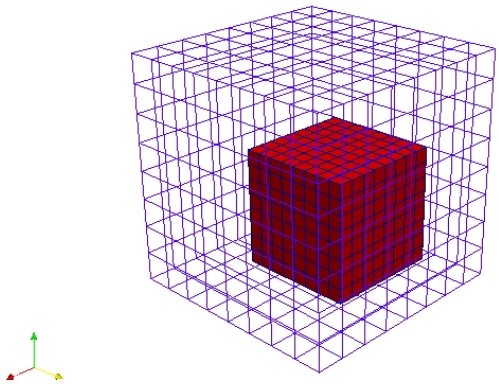


Figure 12: Bloco utilizado no patch 9

A seguir apresenta-se os resultados para a norma L2.

Table 15: Norma L2 e razão de convergência

Malha	$\| u \|_2$	r_u	$\| v \|_2$	r_v	$\| w \|_2$	r_w
8X8X8 L2	7.845E-2	-	8.044E-2	-	1.479E-1	-
16X16X16 L2	1.811E-2	4.33	1.849E-2	4.35	3.438E-2	4.30
32X32X32 L2	4.477E-3	4.05	4.573E-3	4.04	8.509E-3	4.04
64X64X64 L2	1.117E-3	4.01	1.143E-3	4.01	2.119E-3	4.02
128X128X128 L2	2.798E-4	3.99	2.866E-4	3.99	5.266E-4	4.02

A seguir apresenta-se os resultados para a norma infinito.

Table 16: Norma ∞ e razão de convergência

Malha	$\| u \|_\infty$	r_u	$\| v \|_\infty$	r_v	$\| w \|_\infty$	r_w
8X8X8 L2	1.972E-1	-	2.029E-1	-	2.977E-1	-
16X16X16 L2	4.393E-2	4.49	4.507E-2	4.50	7.082E-2	4.20
32X32X32 L2	1.128E-2	3.89	1.136E-2	3.97	1.903E-2	3.72
64X64X64 L2	3.040E-3	3.71	3.033E-3	3.75	4.809E-3	3.96
128X128X128 L2	7.814E-4	3.89	7.790E-4	3.89	1.205E-3	3.99

4.1.9 Executando o teste

O caso está setado na pasta *tests/manuf*, então para executar esse teste basta criar o arquivo cfg apontando para as pastas `input`, `geo` e `probes` contidas na pasta *tests/manuf* e executar o MFSim.

Usando um exemplo, considere que o código fonte do MFSim está em `~/MFLab/MFSim-cmake` e a instalação em `~/MFLab/MFSim-cmake/install`

4.1.9.1 Windows

Primeiro monte ou ative o *container* docker apontando para `/mflab` onde está o código fonte do MFSim e `/mfsim` onde o MFSim foi instalado (caso não se lembre o que é *container* e os apontamentos do docker, dá uma olhada na seção 3.2).

Depois gere o arquivo cfg em `/mfsim/bin/manuf.cfg` com o seguinte conteúdo:

```
input_path: "/mflab/tests/manuf/input"
geo_path: "/mflab/tests/manuf/geo"
output_path: "/mfsim/output"
restart_path: "/mfsim/restart"
probes_path: "/mflab/tests/manuf/probes"
```

Depois dispare:

```
Docker# cd /mfsim/bin
Docker# mpirun --allow-run-as-root -n 2 ./amr3d /mfsim/bin/manuf.cfg 2>err
```

4.1.9.2 Linux: Singularity

Primeiro deve instanciar o *container* singularity como mostrado na seção 3.3. Depois, deve montar o arquivo CFG informando onde está o caso setado e onde serão gravados os resultados. Considerando que o fonte do MFSim está em `~/MFLab/MFSim-cmake`, o instalado em `~/MFLab/MFSim-cmake/install` e o cfg em `~/MFLab/MFSim-cmake/install/bin/manuf.cfg` teremos algo como:

```
input_path: "~/MFLab/MFSim-cmake/tests/manuf/input"
geo_path: "~/MFLab/MFSim-cmake/tests/manuf/geo"
output_path: "~/MFLab/MFSim-cmake/install/output"
restart_path: "~/MFLab/MFSim-cmake/install/restart"
probes_path: "~/MFLab/MFSim-cmake/tests/manuf/probes"
```

Agora é só disparar

```
Singularity> cd ~/MFLab/MFSim-cmake/install/bin
Singularity> source ../etc/mfsim-env.sh
Singularity> mpirun -n 2 ./amr3d ~/MFLab/MFSim-cmake/install/bin/manuf.cfg 2>err
```

4.1.9.3 Linux: Lmod

Primeiro, carregue os módulos do lmod como mostrado na seção 3.4. Depois monte o cfg indicando a pasta `tests/manuf` do código fonte do MFSim para a `input`, `geo` e `probes` e dispare.

Considerando que o fonte do MFSim está em `~/MFLab/MFSim-cmake`, o instalado em `~/MFLab/MFSim-cmake/install` e o cfg em `~/MFLab/MFSim-cmake/install/bin/manuf.cfg`, teremos algo como:

```
input_path: "~/MFLab/MFSim-cmake/tests/manuf/input"
geo_path: "~/MFLab/MFSim-cmake/tests/manuf/geo"
output_path: "~/MFLab/MFSim-cmake/install/output"
restart_path: "~/MFLab/MFSim-cmake/install/restart"
probes_path: "~/MFLab/MFSim-cmake/tests/manuf/probes"
```

E a execução:

```
$ cd ../install/bin
$ source ../etc/mfsim-env.sh
$ mpirun -n 2 ./amr3d ~/MFLab/MFSim-cmake/install/bin/manuf.cfg 2>err
```

4.2 Escoamento sobre uma esfera estacionária

Esse caso é um escoamento monofásico sobre uma esfera estacionária. A seguir, ilustra-se as principais características desse tipo de escoamento para diferentes regimes (Reynolds de 100 a 1000).

O código desse teste está disponível em *tests/sphere*.

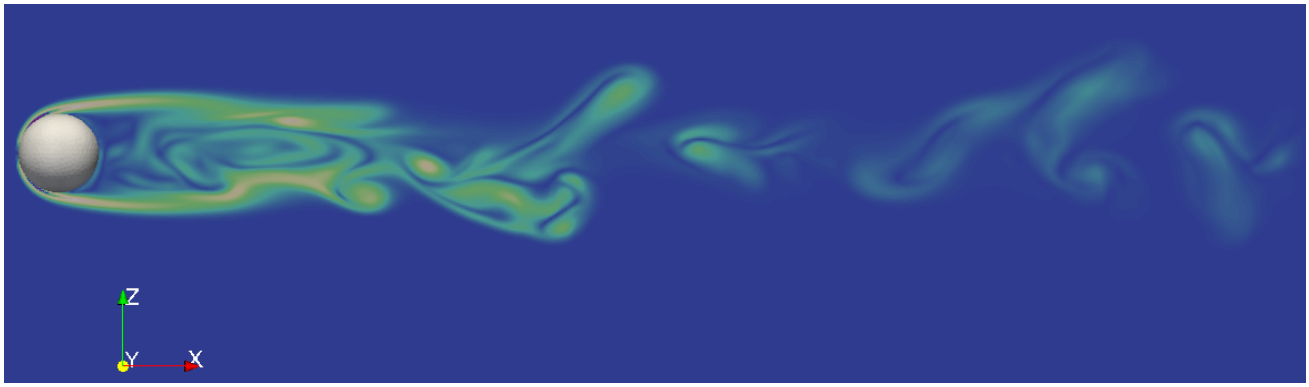


Figure 13: Escoamento sobre uma esfera estacionária

O primeiro passo para setar o caso é acessar a pasta *input* e abrir o arquivo *input.amr3d*. A seguir veremos como modificar esse arquivo.

4.2.1 Parâmetros de controle da malha

O domínio do problema é um paralelepípedo retangular definido por $[0, 1.024] \times [0, 0.512] \times [0, 0.512]$. Vamos Considerar uma esfera dentro deste domínio com centro em (0.36, 0.256, 0.256) e diâmetro de 0.04m. Definiremos 5 níveis físicos, de forma que o nível LBOT é $32 \times 16 \times 16$. Além disso, adotaremos uma malha adaptativa e usaremos dois critérios de refinamento: fronteira imersa(ib-points) e vorticidade.

```
1  !!!****mesh control parameters****!!!
2  0.0d0 1.024d0 0.0d0 0.512d0 0.0d0 0.512d0  ![a1,b1]x[a2,b2]x[a3,b3]: domain (the largest parallelepiped)
3  2 2 2                                        !refinement ratio                (r)
4  1                                           !number of ghost cells           (ngc)
5  5                                            !number of physical levels       (npl)
6  32 16 16                                   !number of cells in lbot level
7  0.99d0                                       !(cuttof) - goes from 0 to 1, and define how slim is the patch
8  2                                            !buffer zone between levels      (bzbl)
9  2                                            !buffer zone to flagged points   (bzfp)
10 .true.                                      !remesh in the initial configuration: .true. or .false.
11 .true.                                      !.true./.false. allow the user to choose if he wants an adaptive mesh (dynamic_mesh)
12 100                                          !number of time step of regriding (rfn)
13 1                                            !Remesh type: 1- Standard Interp+Project, 2- Popinet Div-free Interp
14 .true.                                       !interface is covered by the finest level (interface_finest_level)
15 25   0.01d0   0.1d0   0.1d0   0     !options of refinement: 1- density, 2- ib points, 3- curvature, 4- vorticidade_1 e 5-
       vorticity_2 and Control parameters of the refinement criteria number 4 (lower_lim_input, upper_lim_input), 5 (eps_input) and
       vorticity level limiter (vort_lim)
16 0          !0: set the initial patch by coordinates, 1: read the initial patch from corner.amr3d file
17 1                                            !number of initial patches (number_of_initial_patchs)
18 0.2d0 0.5d0 0.2d0 0.6d0 0.2d0 0.8d0
```

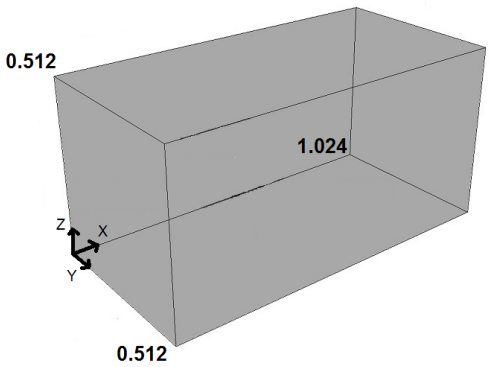


Figure 14: Domínio do problema

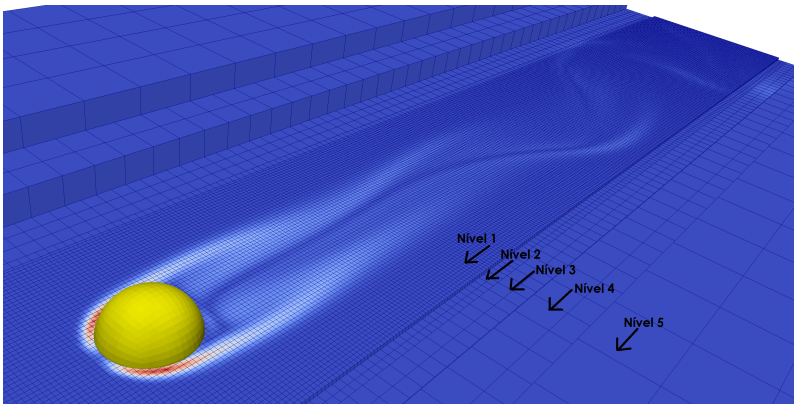
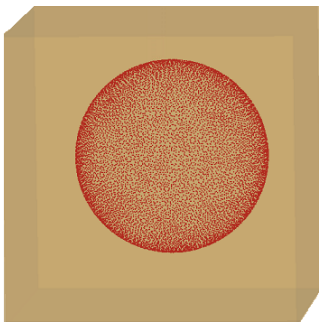


Figure 15: Domínio com os diferentes níveis de refinamento

Note também que o bloco de malha inicial (initial patch) deve ser determinado de modo a conter a esfera.



4.2.2 Parâmetros de controle do modelo de discretização e acoplamento de pressão-velocidade

```
!!!!*****Pressure velocity coupling and discretization models*****!!!!
1 39 1.0d-5                    !kind_pressure_velocity_coupling(1-F.S., 2-SIMPLE, 3-SIMPLEC); maximum number of iterations(SIMPLE, SIMPLEC);   maximum divergent(SIMPLE, SIMPLEC)
0.7d0     0.3d0     0.1d0    0.7d0  !Relaxation coefficient of velocity(SIMPLE, SIMPLEC); Relaxation coefficient of pressure(SIMPLE); !Relaxation coefficient of temperature(SIMPLE, SIMPLEC); !Relaxation coefficient of concentrations(SIMPLE, SIMPLEC)
1                      !temporal discretization model: 1 semi-implicit , 2 implicit
"sbdf"                              !time discretization -"sbdf","mcnab","abcn","cnlf" (implicit_disc)
1                           !Conservative form: 1, not conservative form: 2
"cubista"                   !advection model:  IMPLICIT: upwind1O, upwind2O, CDS, QUICK, QUICKhayase;   SEMI-IMPLICIT: upwind1O, upwind2O, godunov, Barton, MSOU, CDS, skew, cubista
1                                   !flux discretization at momentum equation: 1- malalasekera, 2- traditional
0.2d0                               !Courant-Friedrichs-Lewy condition, CFL (cflGlobal)
```

4.2.3 Parâmetros de controle do método multigrid

```
!!!****multigrid control parameters****!!!
50                                  !maximum number of v-cycles or w-cycles (nvcmax)
3                                !smoothers options: 1- gsrb, 2- msip, 3- sip, 4- cg, 5- bicgstab (smoothers_pressure)
.false.                             !red black smooth at pressure multigrid (redblack)
1.0d0                               !over relaxation parameter at pressure multigrid (omeg)
1.3d0                               !over relaxation parameter at velocity multigrid (omegv)
1.0d0                               !over relaxation parameter at scalar multigrid (omegs)
1                                   !solver pressure :1- v_cycle, 2 - PETScVI
1                                   !solver velocity :1- v_cycle, 2 - PETSc, 3 - SIP
.true.                              !PETSc update matrix ?
.false. 0                           !PETSc at lbase level??, wich level?? (petsc_level=lbot-petsc_level)
25 6 8                              !number of smoothness in the 2x2x2 level (relax_base), in down process (relax_down) and in up process (relax_up)
```

4.2.4 Parâmetros de controle da fase contínua

O primeiro caso a ser simulado é para Re = 400. Sabemos que $Re = \frac{uL\rho}{\mu}$. Portanto, considerando $\rho = 1, u = 1, L = 0.04$ e como $Re = 400$ tem-se $\mu = 0.0001$.

```
!!!****two-phase control parameters****!!!
1.0d0      1.0d0                    !density at continuous phase and disperse phase (dens_phase_c,dens_phase_d)
0.0001d0     1.0d-1               !viscosity at continuous phase and disperse phase (visc_phase_c,visc_phase_d)
0.0d0                               !surface tension (surface_tension)
0.0d0    0.0d0   0.0d0              !gravity acceleration (gravity_x, gravity_y, gravity_z)
```

4.2.5 Parâmetros de controle da simulação

Nesse caso resolve-se a equação completa de Navier-Stokes e adota-se que o tempo final de simulação é 1s. O método escolhido para a discretização do termo temporal é o método semi-implícito sem truque. Uma definição importante é o esquema de discretização do termo advectivo das equações de Navier-Stokes, pois deve-se utilizar um esquema robusto como cubista ou Barton.

```
!!!****simulations control parameters****!!!
.false.                             !.true./.false. allow the user to choose if he wants to restart the code (re_start)
500                                 !save the data each xxx time step to restart the code (ctrst)
2099                                !code restart point (rct). Please check the folder "restart/". (USE THE SAME EXPRESSION AFTER "ct_")
5                                !number of time steps to print the results (prt)
100000000                           !maximum number of time steps (ctmax)
1.0d0                               !final time of simulation (final_t)
"hdf5"                              !printing format: "tecplot"; "paraview"; "hdf5"(printing)
```

4.2.6 Parâmetros de controle das condições de contorno

Como o escoamento é na direção x temos que a entrada é a face west e nesse problema foi definido que essa entrada tem campo de velocidade imposta com valor unitário. E na saída (face east) foi definida condição advectiva. Todas as demais faces tem condição de Neumann para a velocidade. Para a pressão, foi definida condição de Neumann para a entrada (face

west) e em todas demais faces foi definida condição de Dirichlet com valor nulo.
Para as propriedades físicas, foi adotada condição de contorno de derivada nula sendo que o valor das propriedades varia em função do número de Reynolds.

```
!!!****boundary conditions control parameter ****!!!
0   0   0                          !periodic boundary conditions: (0 -> no periodic, 1 -> periodic)
!!!**** alfa u + beta du/dn + gamma du/dt = g****!!!!!
!########### U velocity boundary conditions ############
 1.0d0   0.0d0   0.0d0                          !u boundary condition at WEST face (alfa, beta, gamma)
 0.0d0   0.0d0   1.0d0                          !u boundary condition at EAST face (alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !u boundary condition at SOUTH face (alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !u boundary condition at NORTH face (alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !u boundary condition at BOTTOM face (alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !u boundary condition at TOP face (alfa, beta, gamma)
!########## V velocity boundary conditions ############
 1.0d0   0.0d0   0.0d0                          !v boundary condition at WEST face (alfa, beta, gamma)
 0.0d0   0.0d0   1.0d0                          !v boundary condition at EAST face (alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !v boundary condition at SOUTH face (alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !v boundary condition at NORTH face (alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !v boundary condition at BOTTOM face (alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !v boundary condition at TOP face (alfa, beta, gamma)
!########## W velocity boundary conditions ############
 1.0d0   0.0d0   0.0d0                          !w boundary condition at WEST face (alfa, beta, gamma)
 0.0d0   0.0d0   1.0d0                          !w boundary condition at EAST face (alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !w boundary condition at SOUTH face (alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !w boundary condition at NORTH face (alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !w boundary condition at BOTTOM face (alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !w boundary condition at TOP face (alfa, beta, gamma)
!############ pressure boundary conditions ##############
 0.0d0   1.0d0   0.0d0                          !p boundary condition at WEST face(alfa,beta,gamma)
 1.0d0   0.0d0   0.0d0                          !p boundary condition at EAST face(alfa,beta,gamma)
 1.0d0   0.0d0   0.0d0                          !p boundary condition at SOUTH face(alfa, beta, gamma)
 1.0d0   0.0d0   0.0d0                          !p boundary condition at NORTH face(alfa, beta, gamma)
 1.0d0   0.0d0   0.0d0                          !p boundary condition at BOTTOM face(alfa, beta, gamma)
 1.0d0   0.0d0   0.0d0                          !p boundary condition at TOP face(alfa, beta, gamma)
!############ density boundary conditions ###############
 0.0d0   1.0d0   0.0d0                          !p boundary condition at WEST face(alfa,beta,gamma)
 0.0d0   1.0d0   0.0d0                          !p boundary condition at EAST face(alfa,beta,gamma)
 0.0d0   1.0d0   0.0d0                          !p boundary condition at SOUTH face(alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !p boundary condition at NORTH face(alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !p boundary condition at BOTTOM face(alfa, beta, gamma)
 0.0d0   1.0d0   0.0d0                          !p boundary condition at TOP face(alfa, beta, gamma)

!!!****initial partition topology****!!!
4 1 1 !number of processors in x,y,z axis

!!!****Perform load balance
0 !0: dont perform load balance; 1: perform load balance

!!!****Start by HDF5
false                                           !use start by hdf5
"start.hdf5"                                    !hdf5 file

!!!**** Method
2               ! method: 1 - Noone; 2 - IB; 3 - VoF; 4 - FT; 5 - VoF+IB; 6 - FT+IB

!!!**** EWF
.false.                     ! use_ewf [true|false]. Default: false

!!!**** Libs
.false.             ! use_petsc
.false.             ! use_slepc
.false.             ! use_zoltan

!!!**** Manuf
.false.             | use_manuf [true|false]. Default: false
```

A definição das condições de contorno podem ser ajustadas no arquivo *bc_functions.f90* na pasta *input*:

```
function exact_u(t,x,y,z)
    implicit none
    double precision :: t, x, y, z
    double precision :: exact_u

    exact_u = 1.0d0

end function exact_u
```

4.2.6.1 Construção da geometria

Para a construção da geometria vamos gerar o arquivo *esfera.geo*. A geometria é determinada a seguir.

Como temos 5 níveis físicos e LBOT é $32 \times 16 \times 16$ temos que LTOP é $512 \times 256 \times 256$. Sabendo que $dx = dy = dz$ temos que o volume do cubo do nível mais fino da malha euleriana é dado por $dx^3 = \left(\frac{1.024}{512}\right)^3 = 8 \times 10^{-9}$. Por outro lado, o volume do prisma de base triangular da malha que cobre a esfera é dado por $\frac{l^3\sqrt{3}}{4}$. Igualando os dois volumes citados anteriormente conclui-se que $l = 0.0026$. Defini-se $cl = l = 0.0026$.

Desse modo, o texto contido no arquivo é ilustrado a seguir:

```
R = 0.02;
Cx = 0.36;
```

```
Cy = 0.256;
Cz = 0.256;

cl = 0.0026;

Point(1)={Cx, Cy, Cz,cl};
Point(2)={Cx+R,Cy,Cz,cl};
Point(3)={Cx, Cy+R,Cz,cl};
Point(4)={Cx, Cy - R,Cz,cl};

Circle(1) = {2,1,4};
Circle(2) = {3,1,2};

Extrude {{0, 1, 0}, {Cx, Cy, Cz}, Pi/2} {
  Line{1, 2};
  }
Extrude {{0, 1, 0}, {Cx, Cy, Cz}, Pi/2} {
  Line{3, 6};
  }
Extrude {{0, 1, 0}, {Cx, Cy, Cz}, Pi/2} {
  Line{9, 12};
  }
Extrude {{0, 1, 0}, {Cx, Cy, Cz}, Pi/2} {
  Line{18, 15};
 }
```

Ao abrir esse arquivo no software Gmsh, tem-se a visualização a seguir.

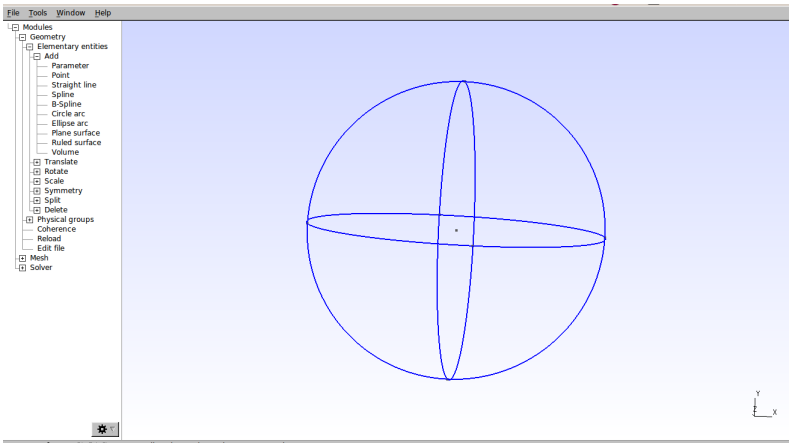


Figure 16: Geometria da esfera

4.2.7 Construção da malha lagrangiana

Para construir a malha, basta abrir no Gmsh o arquivo de geometria *esfera.geo* criado anteriormente e definir inicialmente a dimensão dos elementos da malha 2D a ser criada. Para isso, clique em Tools → Options → Mesh → General e preencha os campos *Min/Man element size* com o valor de $cl = 0.0026$. Em seguida, clique em Modules → Mesh → 2d.

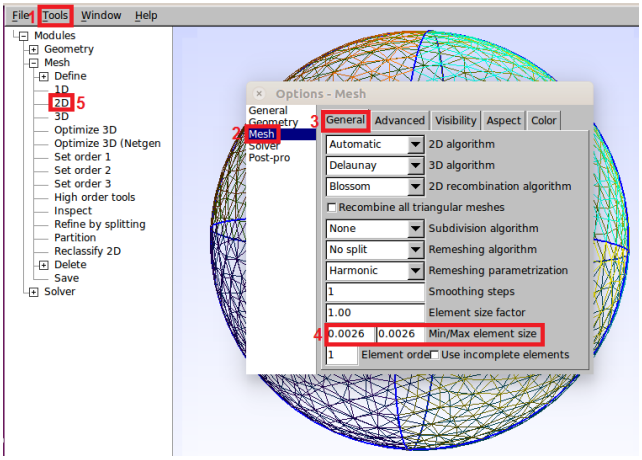


Figure 17: Geometria da esfera

Agora, gere o arquivo *.stl* clicando em *File* > *Export* > salve o arquivo *esfera.stl*, na mesma pasta onde está o *esfera.geo*.

Após salvar o arquivo da malha lagrangiana na pasta *geo* do código, devemos informar o caminho desse arquivo no arquivo de configuração de fronteira imersa, o *ib.amr3d*, dentro da pasta *input*.

```
!!!!*****Immersed Boundary parameters****!!!
.true. 1 ! Immersed Boundary TRUE/FALSE (used in models)
```

```
3   5 0.001d0 !maximum numbem of cycles in the direct forcing scheme (maxc), and eslon of directing force (rigid_epslon) - for rigid
        body
4   1 !kind of file: '1-stl ascii file', '2-stl2amr file', '3-manual file' (kindIBFile)
5   1
6   "esfera.stl"
7   .false.
8   0d0 0d0 0d0
9   0d0 0d0 0d0
10  0d0 0d0 0d0
11
12  1 !kind of the interp./distrib. function: 1-hat, 2-Gauss, 3-cubic (kindIntIB)
13  1 !strategy for the construction of the lagrangian volume [1 -> (L^2*sqrt(3)/4)*L = dx^3]; [2 -> (L^2*sqrt(3)/4)*dx = dx^3] (
        stratLagVol)
14  .false. ! print pvd, pvtp, and vtp first time step.
15  .false. ! print pvd, pvtp, and vtp EVERY time step.
16
17  !!!******IB indicator funcyion - normal must be pointed at inner region of  immersed boundary
18  .false. ! active the IB indicator function that is 1 in the inner region and zero outter
19  .false. ! Is the flow external to the IBs?
20
21  !!!!***** Compressible parameters****!!!
22  .false.                 !use free slip boundary condition (freeSlipIB)
23
24  !!!!*****Scalar Immersed Boundary parameters****!!!
25  .false. !Use scalar immersed boundary (termical ib)
26  .false. ! Use scalar immersed boundary (scalar ib)
27  1 0d0 ! Type of Temperature BC (1-dirichlet,2-neumann,3-robin) and impose volue
28  1 0d0 ! Type of Scalar BC (1-dirichlet,2-neumann,3-robin) and impose volue
```

A malha lagrangiana construída pode ser visualizada a seguir, onde o tamanho das arestas dos triângulos é 0,0026.

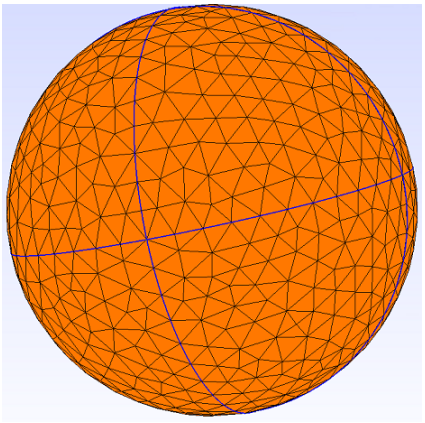


Figure 18: Malha da esfera

4.2.8 Executando o teste

Agora que temos o caso todo setado, só nos falta montar o arquivo cfg e rodar.

4.2.8.1 Windows

Primeiro instancie o *container* docker e monte o arquivo cfg conforme mostrado na seção 3.2. Depois dispare:

```
Docker# cd /mfsim/bin
Docker# mpirun --allow-run-as-root -n 4 ./amr3d arquivo_cfg 2>err
```

4.2.8.2 Linux: Singularity

Primeiro instancie o *container* Singularity e monte o arquivo cfg conforme mostrado na seção 3.3. Depois dispare:

```
Singularity> cd PATH_INSTALACAO_MFSIM/bin
Singularity> source ../etc/mfsim-env.sh
Singularity> mpirun -n 4 ./amr3d arquivo_cfg 2>err
```

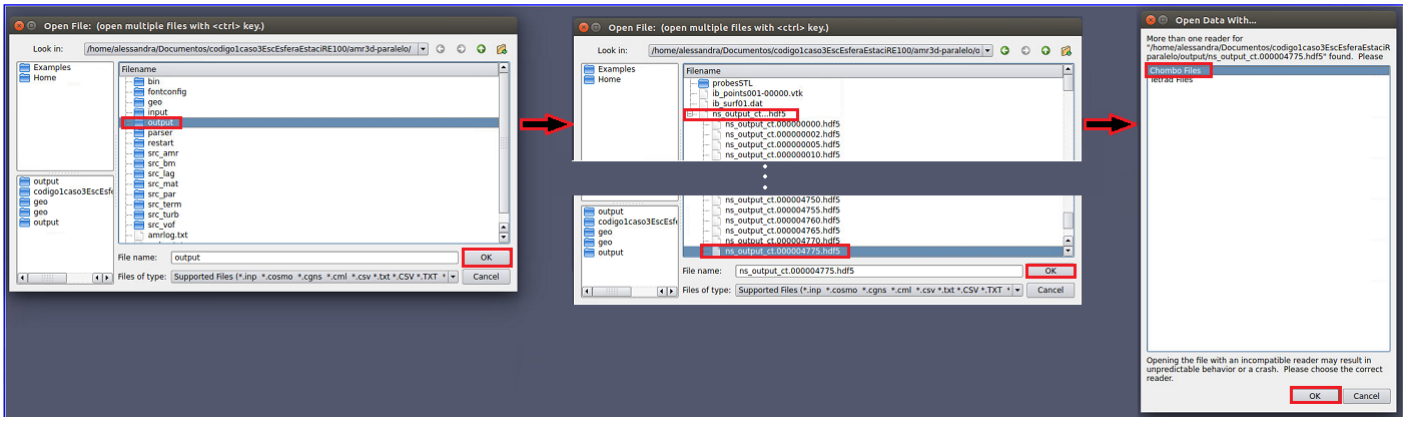
4.2.8.3 Linux: Lmod

Primeiro carregue os módulos e monte o arquivo cfg como mostrado na seção 3.4. Depois dispare:

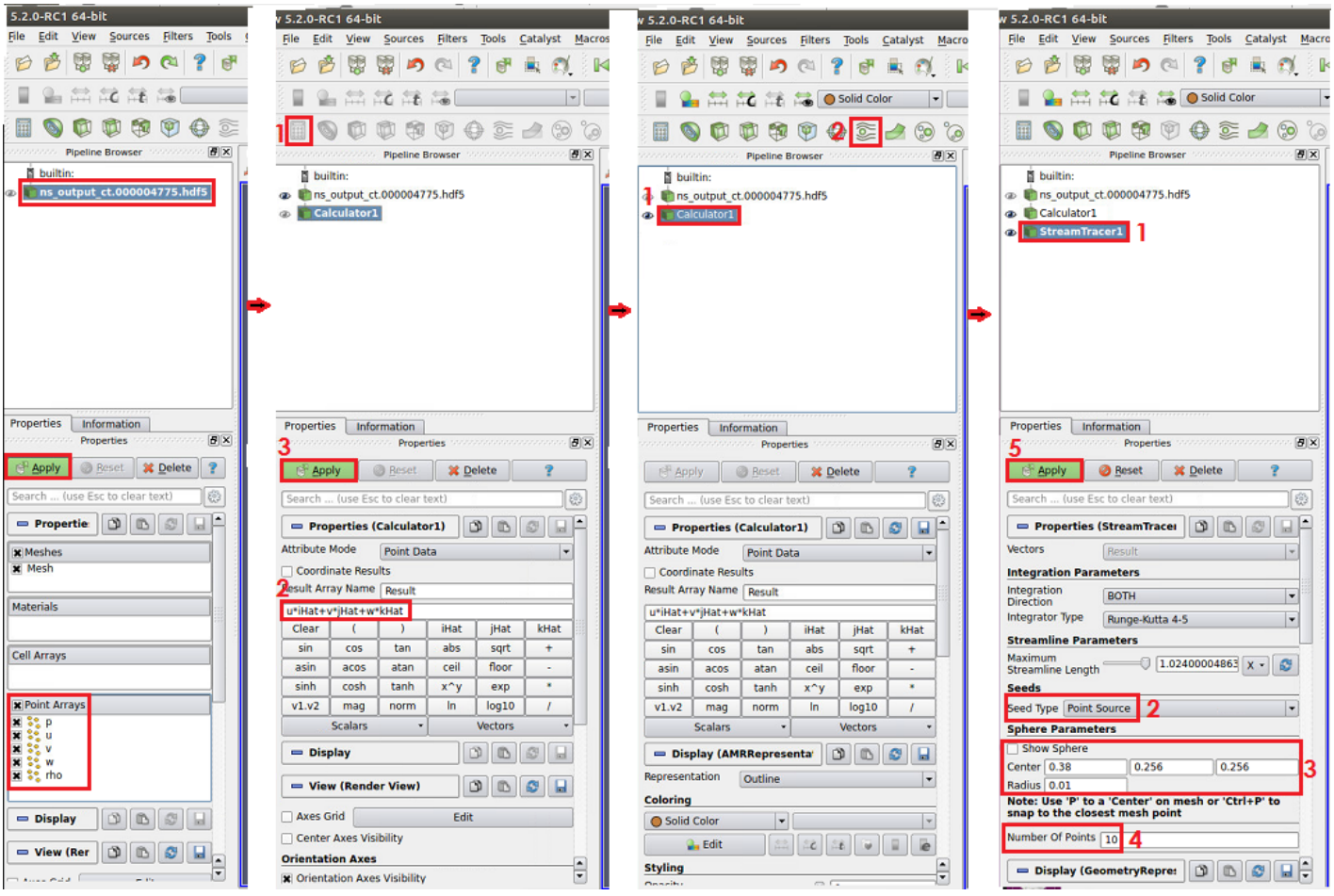
```
$ cd PATH_INSTALACAO_MFSIM/bin
$ source ../etc/mfsim-env.sh
$ mpirun -n 4 ./amr3d arquivo_cfg 2>err
```

4.2.9 Resultados

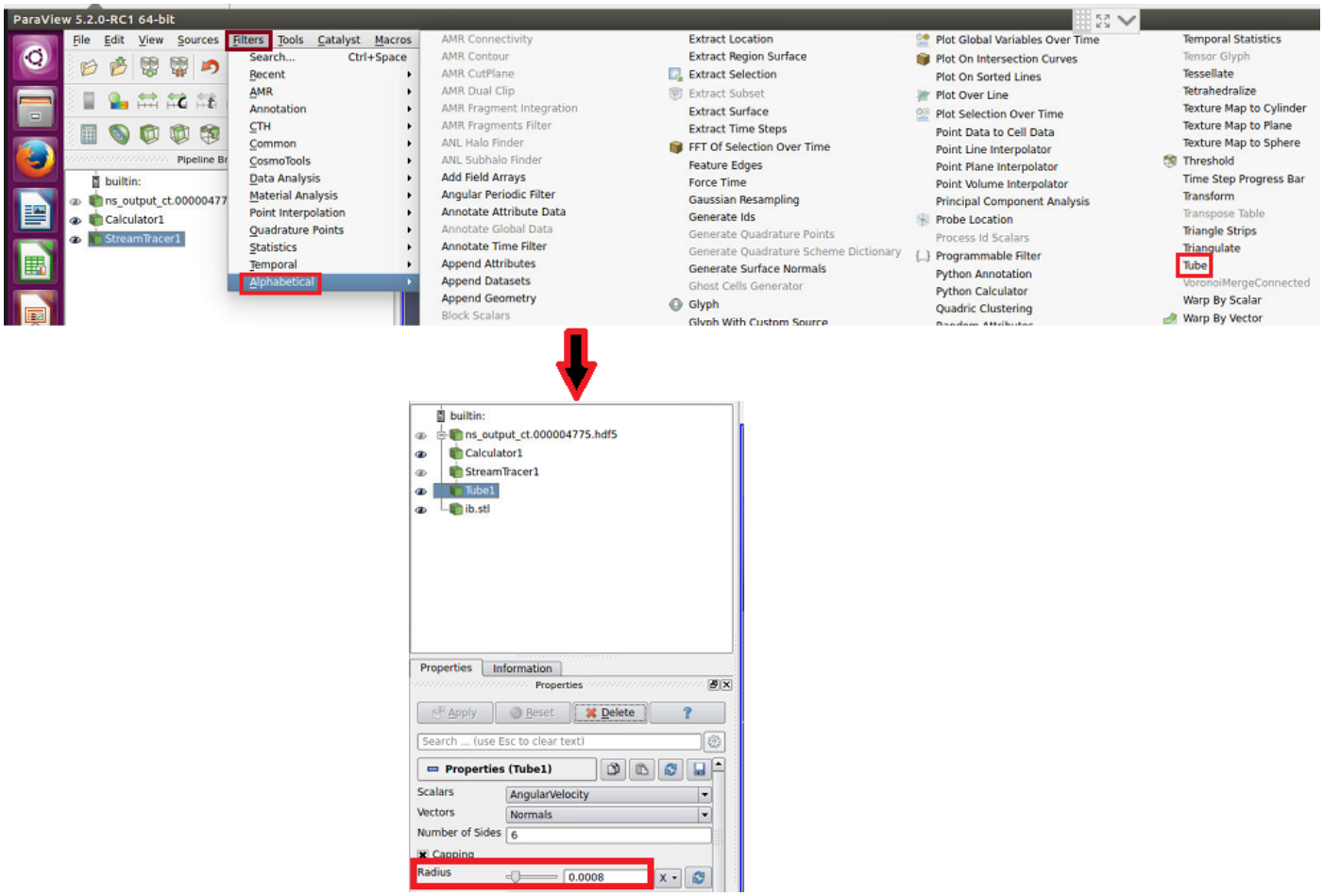
Para a construção dos gráficos pode-se usar o Paraview. Abra a pasta output no Paraview e selecione o último arquivo gravado:



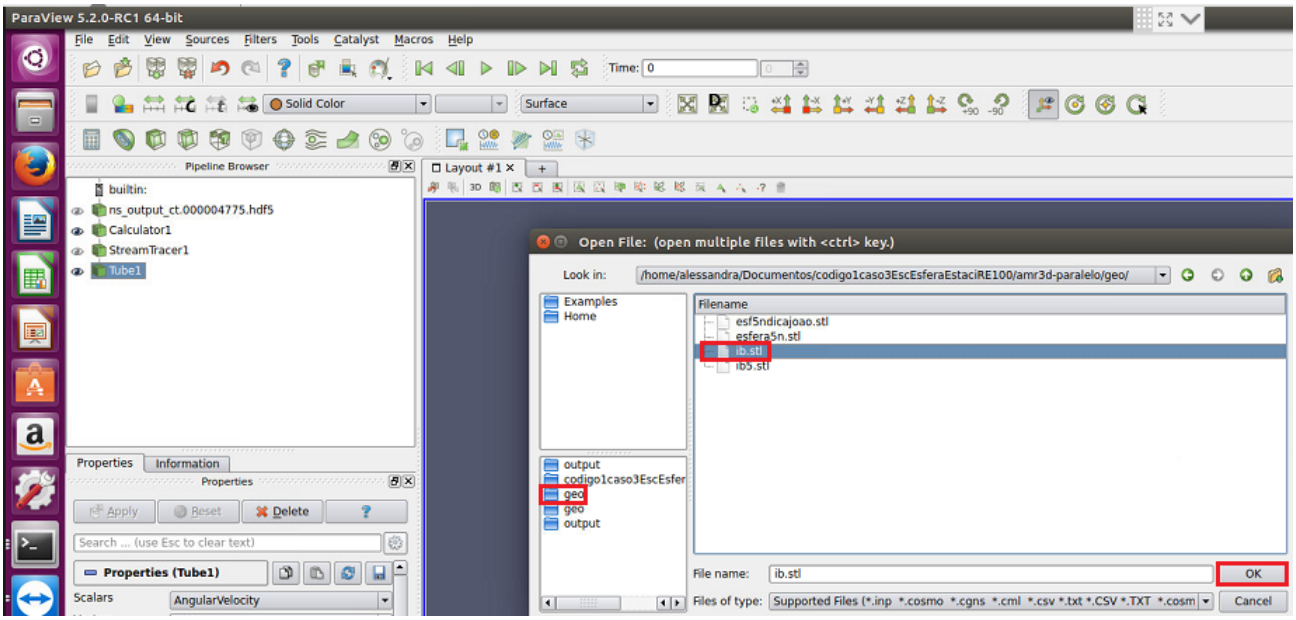
Em seguida, selecione as variáveis de interesse e clique em aplicar. Depois clique no símbolo da calculadora no canto superior esquerdo, digite `u*iHat+v*jHat+w*kHat` e aperte em aplicar. Selecione Calculator1 e clique no comando *Stream Tracer*. Faça os ajustes nos campos *Seeds* e *Sphere Parameters*:



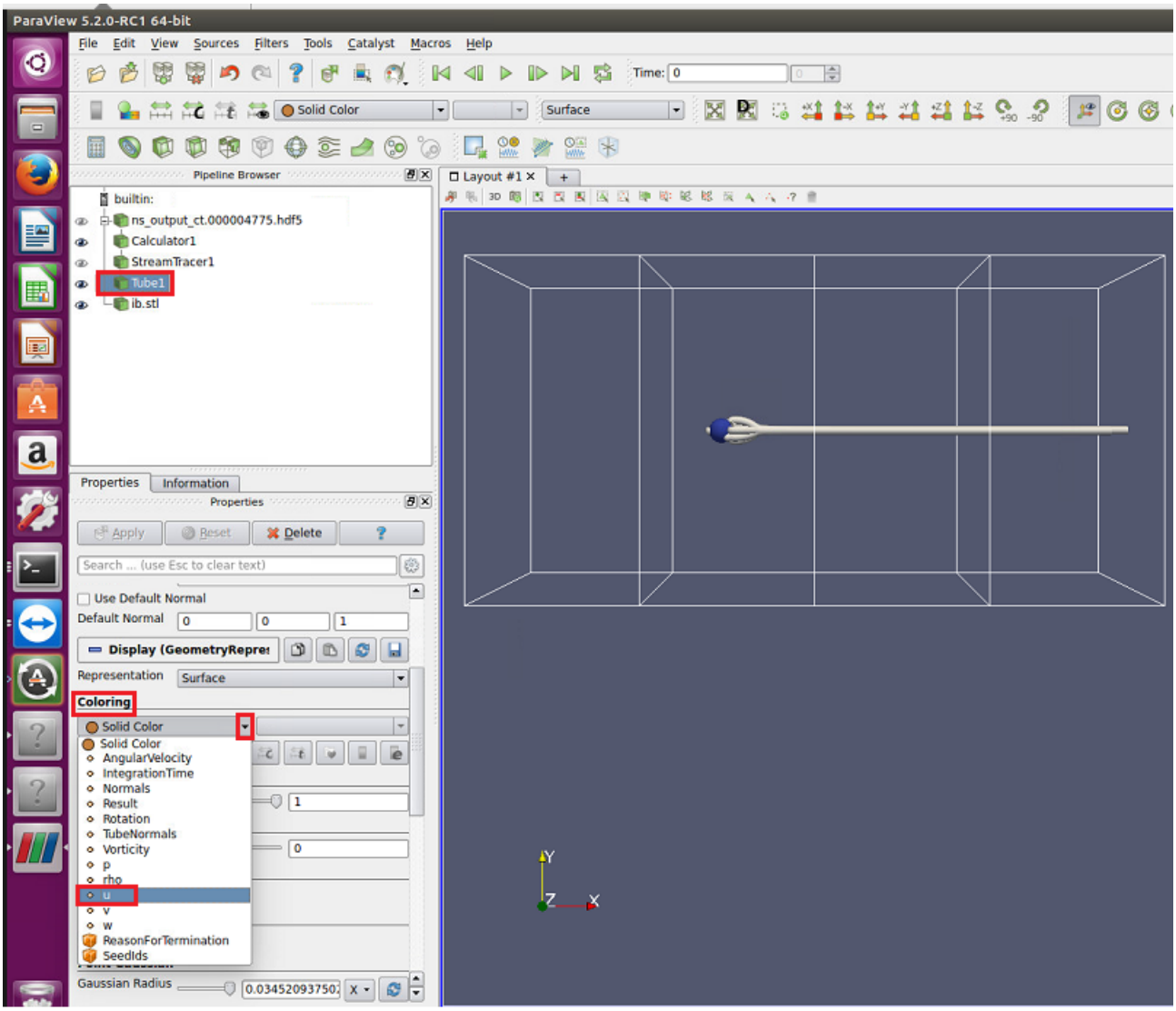
Siga o caminho Filters→Alphabetical→Tube e lembre-se de clicar em aplicar. Em *radius* coloque o valor 0.0008.



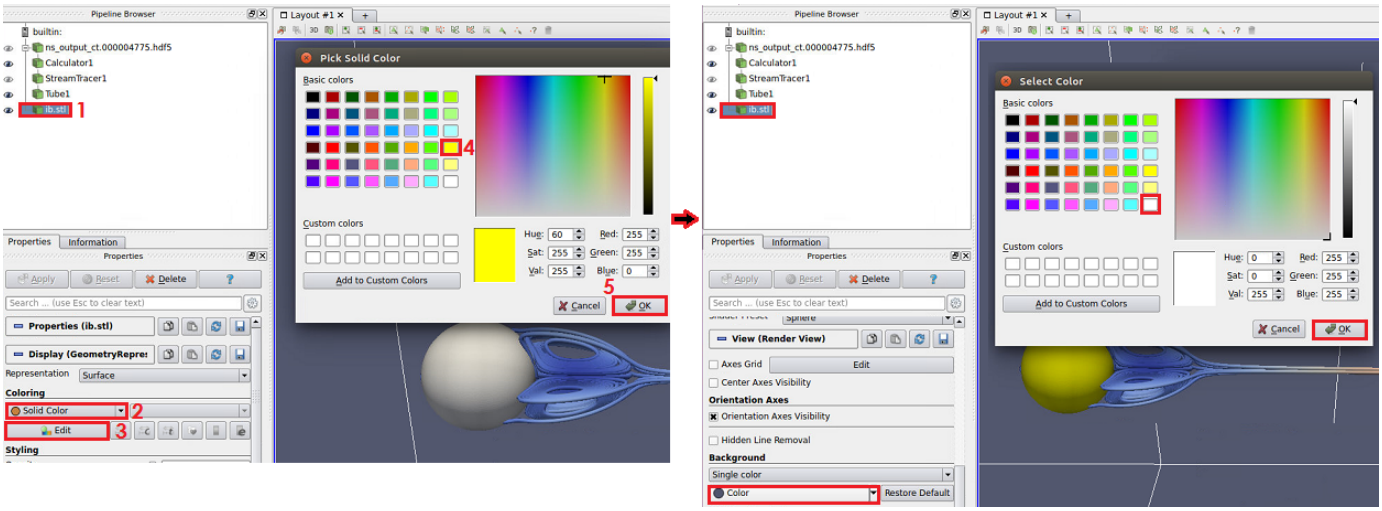
Para plotar a esfera abra o arquivo *.stl* que está na pasta geo e clique em aplicar.



Selecione Tube1 e na opção Coloring escolha u:



Por último pode-se trocar a cor da esfera e a cor do plano de fundo:



Após aplicar um zoom nas figuras obtém-se os gráficos a seguir:

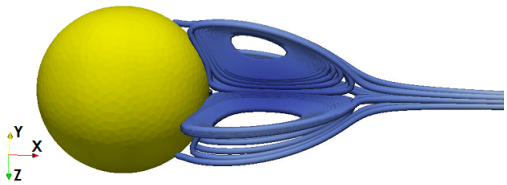


Figure 19: Linhas de corrente a Reynolds 100

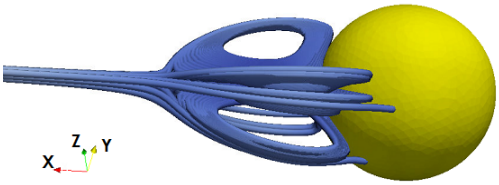


Figure 20: Linhas de corrente a Reynolds 100

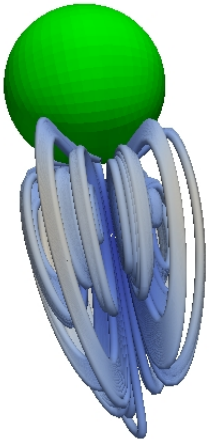


Figure 21: Linhas de corrente a Reynolds 200

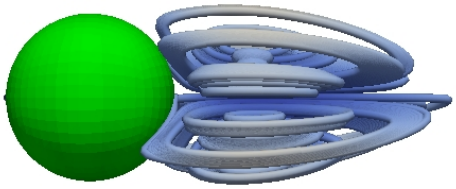


Figure 22: Linhas de corrente a Reynolds 200

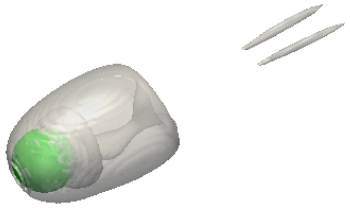


Figure 23: Esteira dupla para Reynolds 300 por meio de critério iso-Q (0.1)

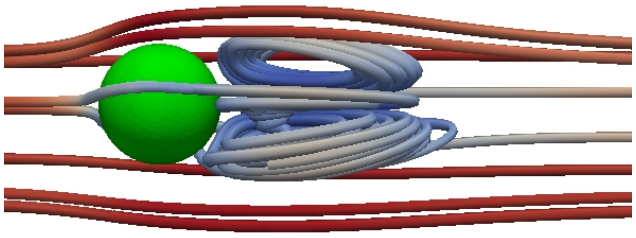


Figure 24: Linhas de correntes para Reynolds 300

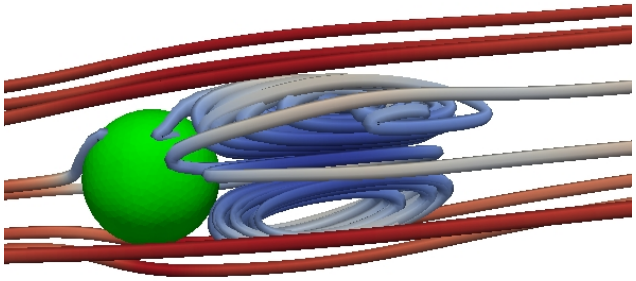
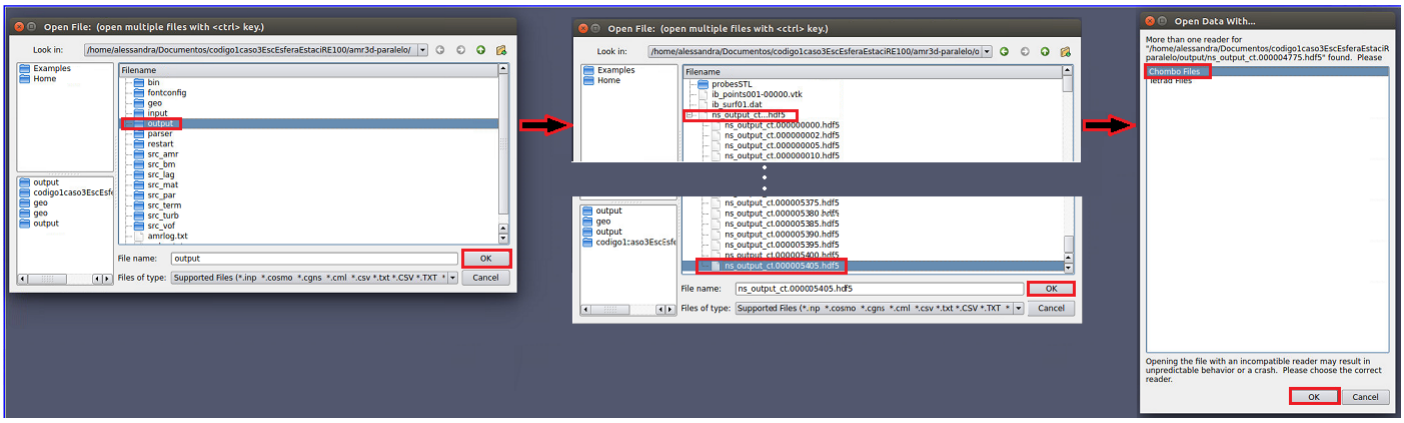


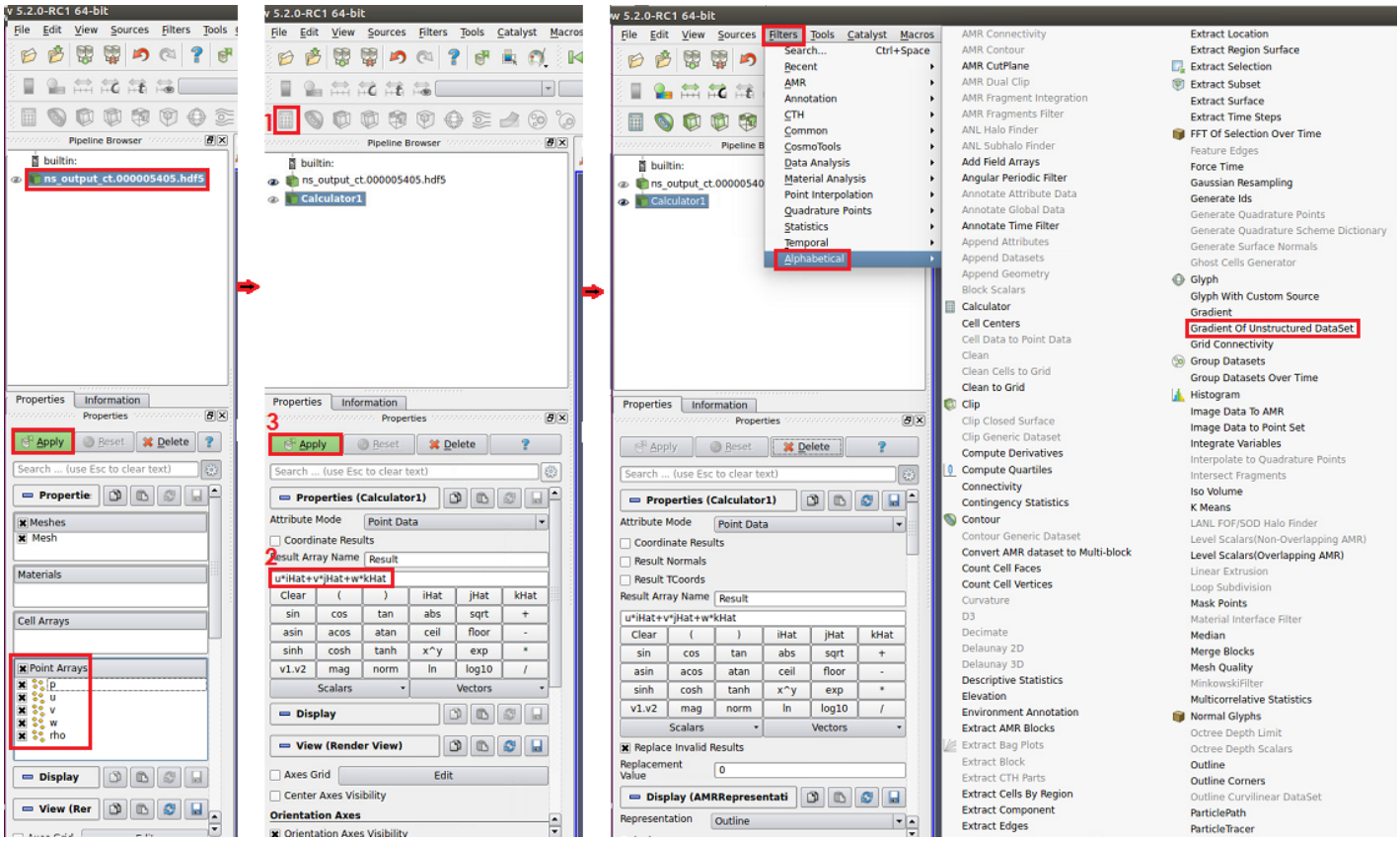
Figure 25: Linhas de correntes para Reynolds 300

Para a construção dos gráficos(Reynolds 400) usando o critério q pode-se também usar o Paraview. Abra a pasta output no Paraview e selecione o último arquivo gravado:

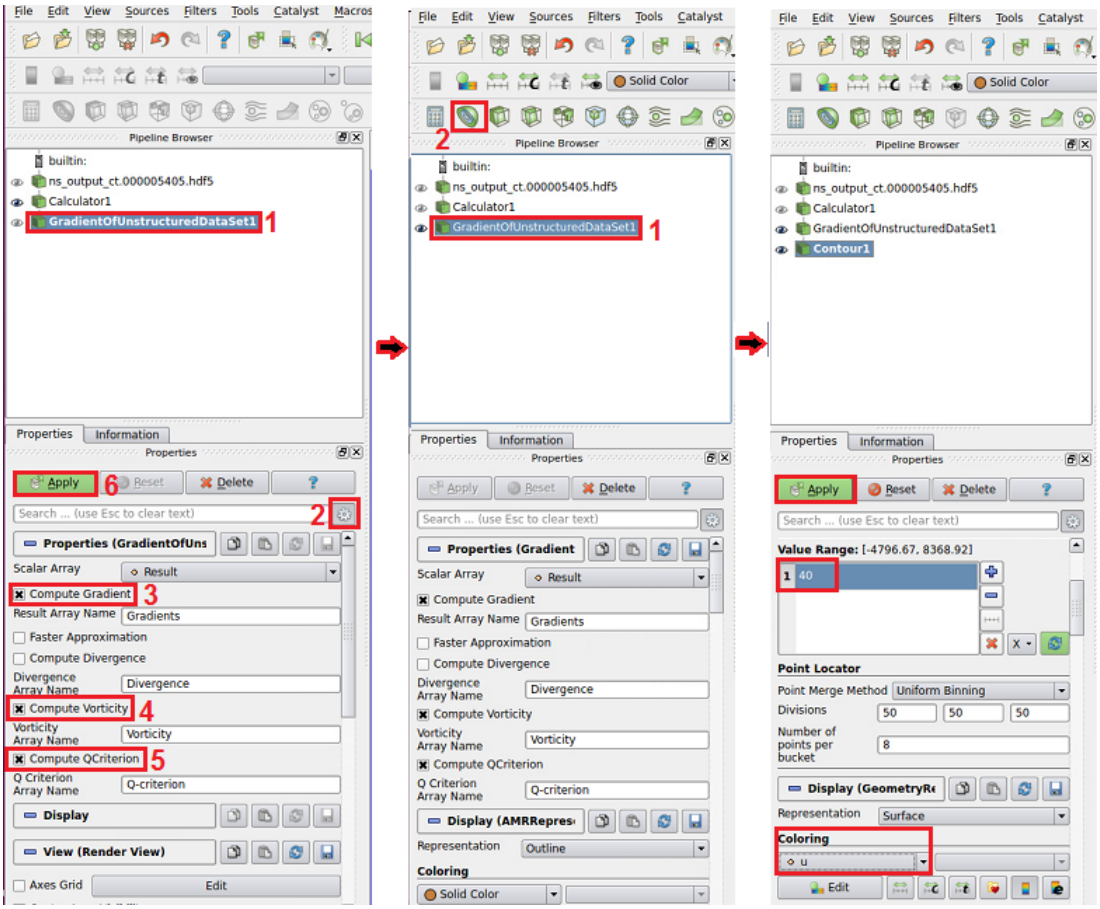


Em seguida, selecione as variáveis de interesse e clique em aplicar. Depois clique no símbolo da calculadora no canto superior esquerdo, digite $u^*iHat + v^*jHat + w^*kHat$ e aperte em aplicar. Selecione Calculator1 e siga o caminho

Filters→Alphabetical→Gradient Of Unstructured DataSet.



Em *Properties(GradientOfUnstructureDataSet1)* selecione *Compute Gradient, Compute Vorticity e Compute QCriterion.* Selecione GradientOfUnstructuredDataSet1 e clique em Contour. No campo Value Range coloque o valor 40 e em *Coloring* escolha u.



O resultado final é o gráfico a seguir:

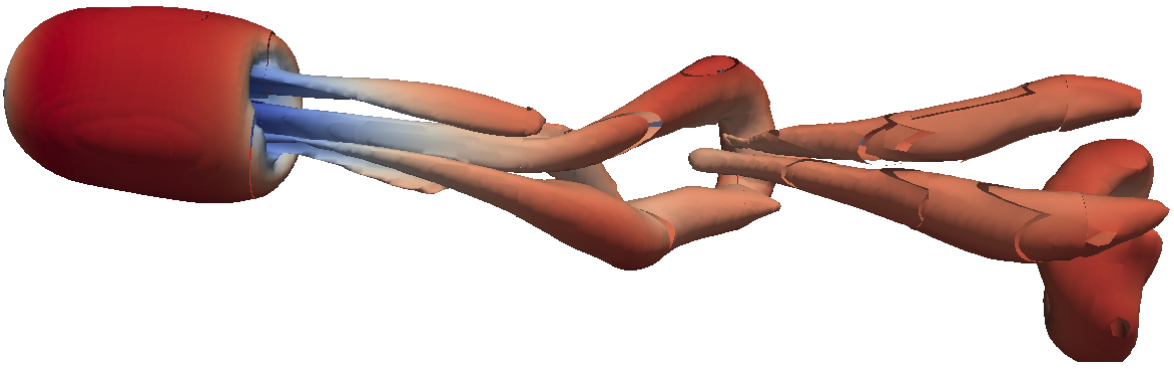


Figure 26: Grampos de cabelo a Reynolds 400

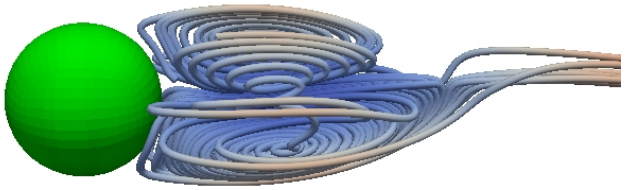


Figure 27: Linhas de corrente para Reynolds 400

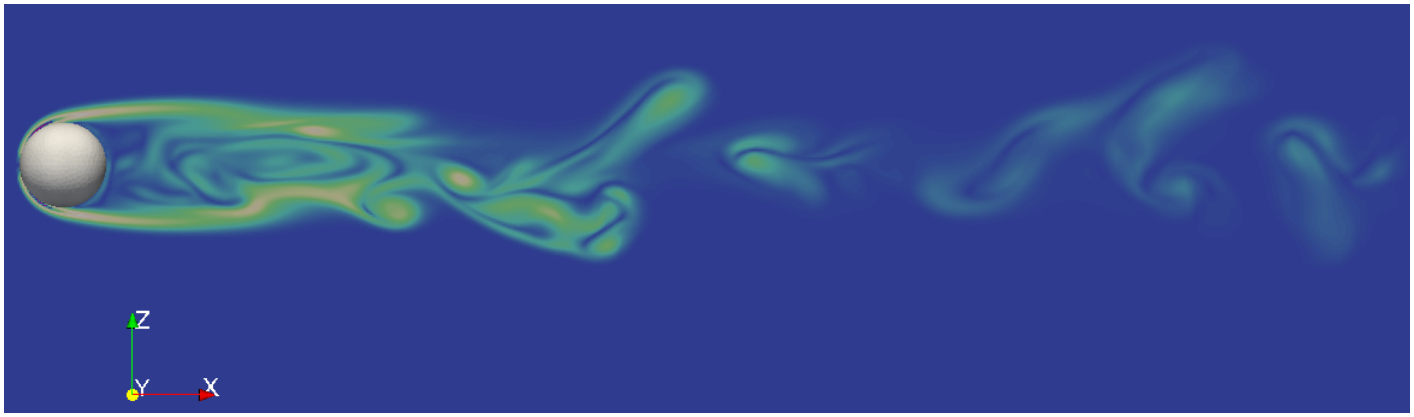


Figure 28: Vorticidade para Reynolds 1000

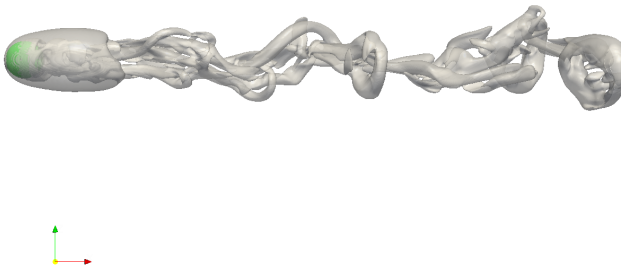


Figure 29: Critério q (0.1) para Reynolds 1000

4.2.10 Validação dos resultados

O coeficiente de arrasto é definido como:

$$C_D = \frac{F_x}{\left(\frac{1}{2}\right)\rho U_\infty^2 \left(\frac{\pi D^2}{4}\right)}$$

sendo F_x a força de arrasto, agindo sobre o corpo imerso (no caso, a esfera), U_∞ é a velocidade da corrente livre, ρ é a massa específica e D é o diâmetro da esfera.

Algumas correlações matemáticas apresentam resultados para o coeficiente de arrasto em função do número de Reynolds, para escoamentos sobre esferas, como as apresentadas abaixo (Subramanian, 2003):

$Re \leq 0.01$	$C_D = \frac{9}{2} + \frac{24}{Re}$
$0.01 < Re \leq 20$	$C_D = \frac{24}{Re}\left[1 + 0.1315Re^{0.82-0.05log_{10}Re}\right]$
$20 \leq Re \leq 260$	$C_D = \frac{24}{Re}\left[1 + 0.1935Re^{0.6305}\right]$
$260 \leq Re \leq 1.5 \times 10^3$	$C_D = 10^{1.6435-1.1242log_{10}Re+0.1558(log_{10}Re)^2}$
$1.5 \times 10^3 \leq Re \leq 1.2 \times 10^4$	$C_D = 10^{-2.4571+2.5558log_{10}Re-0.9295(log_{10}Re)^2+0.1049(log_{10}Re)^3}$
$1.2 \times 10^4 \leq Re \leq 4.4 \times 10^4$	$C_D = 10^{-1.9181+0.6370log_{10}Re-0.0636(log_{10}Re)^2}$
$4.4 \times 10^4 \leq Re \leq 3.38 \times 10^5$	$C_D = 10^{-4.3390+1.5809log_{10}Re-0.1546(log_{10}Re)^2}$
$3.38 \times 10^5 \leq Re \leq 4 \times 10^5$	$C_D = 29.78 - 5.3log_{10}Re$
$4 \times 10^5 \leq Re \leq 10^6$	$C_D = 0.1log_{10}Re - 0.49$
$Re > 10^6$	$C_D = 0.19 - \frac{8 \times 10^4}{Re}$

	Reynolds 100	Reynolds 200	Reynolds 400	Reynolds 1000
Correlação	1,087	0,776	0,593	0,471
Presente trabalho	1.07	0,71	0,6	0,48

Um parâmetro utilizado na literatura em casos de escoamento ao redor de esferas é o comprimento de bolha de recirculação. Em geral, calcula-se um parâmetro adimensional dado por:

$$Adimensional = \frac{comprimento \quad da \quad bolha}{diametro} = \frac{X_S}{0.04} \tag{8}$$

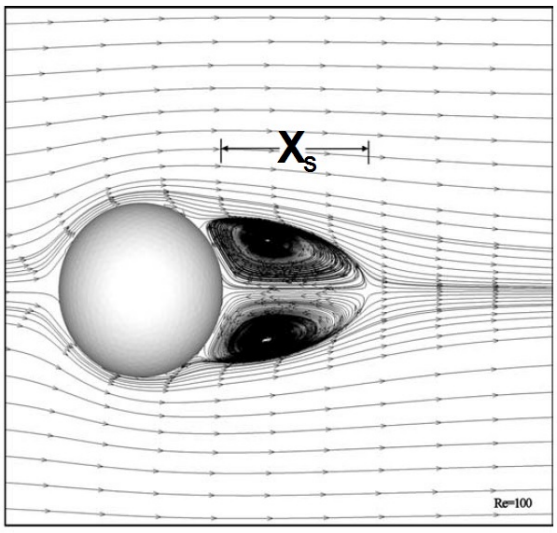


Figure 30: Representação do comprimento da bolha de recirculação.

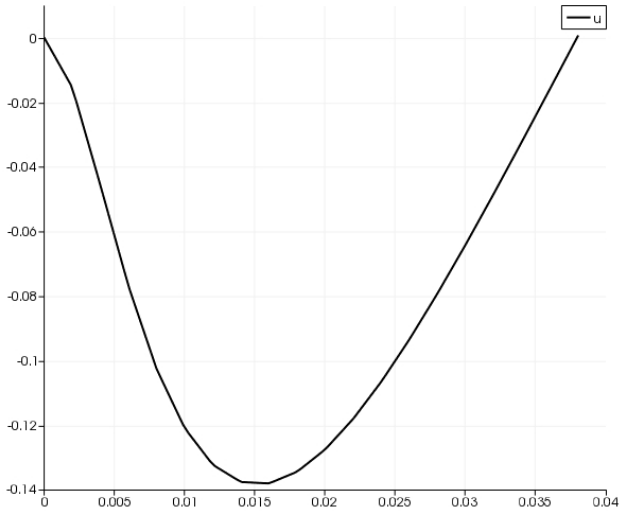
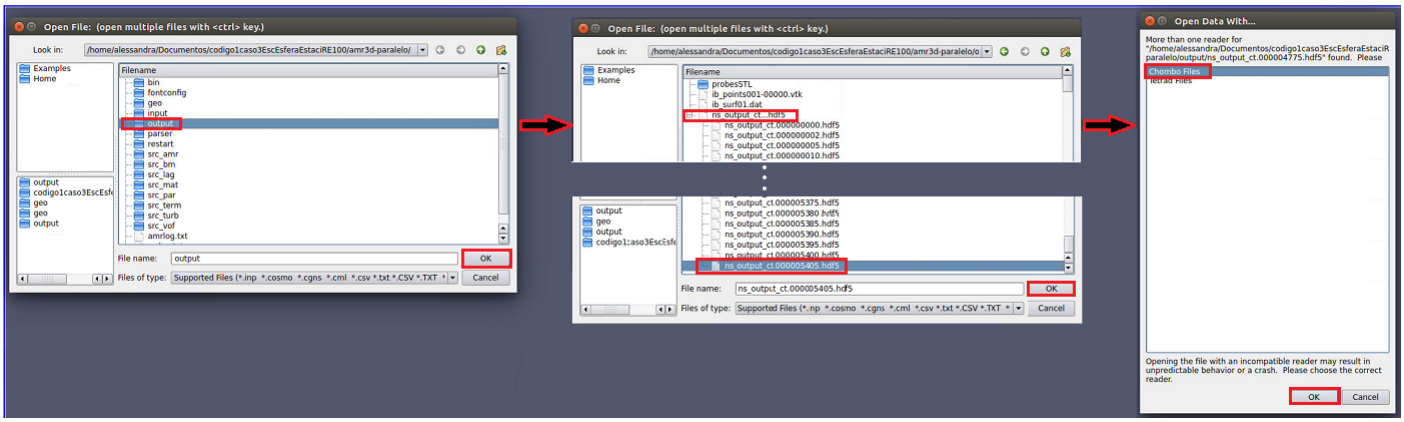
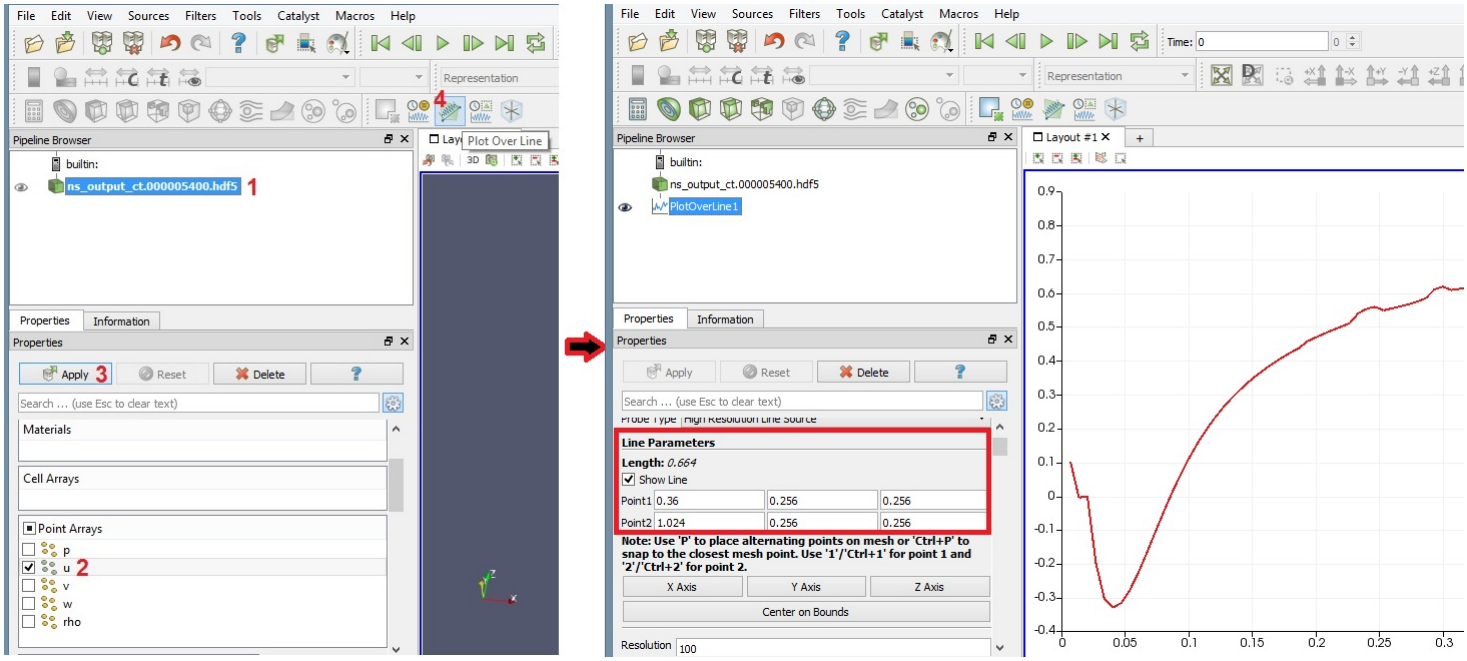


Figure 31: Bolha de recirculação para Reynolds 100

Para a construção do gráfico acima basta abrir a pasta output no Paraview e selecionar o último arquivo gravado:



Selecione a variável de interesse e clique em aplicar. Depois clique em Plot Over Line e preencha o campo *Line Parameters*.



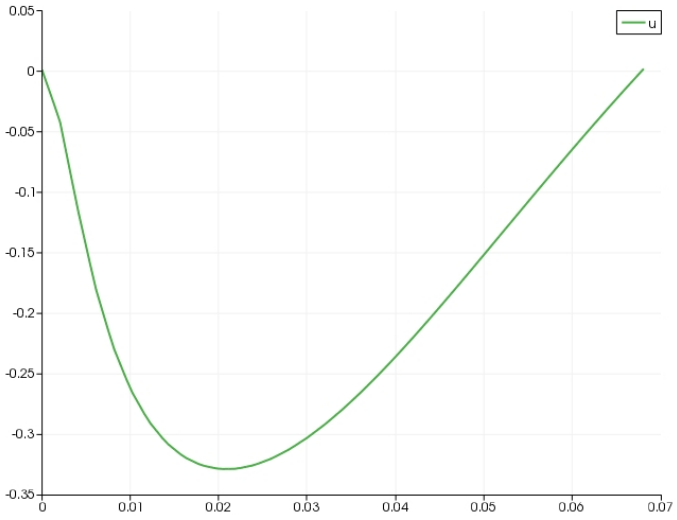


Figure 32: Bolha de recirculação para Reynolds 200

5 Erros Comuns

5.1 Erro de mpirun não encontrado

Ao tentar utilizar o comando `mpirun` é possível aparecer o seguinte erro:

```
johnatas@njam:~/MFLab/Code/MFSim-cmake/install/bin$ mpirun -n 4 ./amr3d sphere.cfg

Comando 'mpirun' não encontrado, mas poder ser instalado com:

sudo apt install lam-runtime       # version 7.1.4-6build2, or
sudo apt install mpich             # version 3.3.2-2build1
sudo apt install openmpi-bin       # version 4.0.3-0ubuntu1
sudo apt install slurm-wlm-torque  # version 19.05.5-1
```

Se esse erro aparecer é porque seu sistema não possui alguma implementação de MPI instalado. As *stacks* Docker e Singularity **dificilmente** apresentarão esse problema, já que os *containers* vem o MPI já configurado por padrão. É mais provável que o problema ocorra com quem usa a *stack* lmod.

A solução nesse caso é bem simples: carregar os módulos como explicado na seção 3.4.

5.2 Erro de *_path

Outro erro possível é de alguma falta de entrada *_path:

```
johnatas@njam:~/MFLab/Code/MFSim-cmake/install/bin$ mpirun -n 4 ./amr3d sphere.cfg
        probes_path not given!
        probes_path not given!
        probes_path not given!
        probes_path not given!
--------------------------------------------------------------------------
Primary job  terminated normally, but 1 process returned
a non-zero exit code. Per user-direction, the job has been aborted.
--------------------------------------------------------------------------
--------------------------------------------------------------------------
mpirun detected that one or more processes exited with non-zero status, thus causing
the job to be terminated. The first process to do so was:

  Process name: [[29436,1],0]
  Exit code:    1
--------------------------------------------------------------------------
```

Esse problema decorre de má configuração do arquivo cfg. Todas as entradas nesse arquivo são **obrigatórias**. A seção 2.4 descreve como setar o arquivo cfg.

5.3 Erro de boundary conditions functions

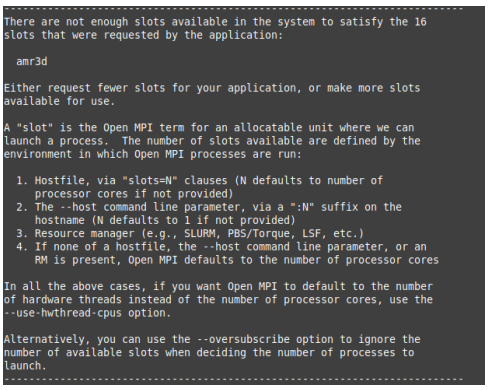
É possível ao disparar o MFSim ocorrer um erro de condições de contorno dinâmicas (bc dinâmico):

```
johnatas@njam:~/MFLab/Code/MFSim-cmake/install/bin$ mpirun -n 4 ./amr3d sphere.cfg
f951: Warning: Nonexistent include directory '-L' [-Wmissing-include-dirs]
/opt/ohpc/pub/compiler/gcc/10.2.0/bin/ld: não foi possivel localizar -lmfsim
collect2: erro: ld returned 1 exit status
Compilation error on file "/home/johnatas/MFLab/Code/MFSim-cmake/tests/sphere/input/./bc_functions.f90"!Error loadi
ng bc functions: "/tmp/Nf2ThCpk3lDZbR0kim5WIuYGoESesz.so: cannot open shared object file: No such file or directory
"!Error loading bc functions: "/tmp/Nf2ThCpk3lDZbR0kim5WIuYGoESesz.so: cannot open shared object file: No such file
 or directory"!Error loading bc functions: "/tmp/Nf2ThCpk3lDZbR0kim5WIuYGoESesz.so: cannot open shared object file:
 No such file or directory"!Error loading bc functions: "/tmp/Nf2ThCpk3lDZbR0kim5WIuYGoESesz.so: cannot open shared
 object file: No such file or directory"! Error! Could not load boundary conditions functions.
 Error! Could not load boundary conditions functions.
 Error! Could not load boundary conditions functions.
```

Esse é um erro comum nas *stacks* lmod e singularity e decorre da falta do comando `source ../etc/mfsim-env.sh` que seta as variáveis de ambiente requeridas pelo bc dinâmico, conforme apresentado na seção 3.3 (singularity) e 3.4 (lmod)

5.4 Erro de slots

Ao tentar executar o MFSim é possível que ocorra o seguinte erro:



Esse erro acontece quando você tenta utilizar no MPI *cores* virtuais ao invés de reais. Elaborando melhor, os processadores modernos, incluindo os presentes em *smpartphones*, são construídos empacotando em um só chip vários processadores menores. A esses processadores menores chamamos de *cores*. Dependendo da versão do seu processador é possível que além dos *cores* presentes fisicamente no chip também existam *cores* virtuais que expandem as funcionalidades dos *cores* reais. Como isso de fato funciona é uma explicação extremamente longa e totalmente fora do escopo desse manual, que é ensinar você a como usar o MFSim. Por agora, você precisa apenas entender que **um programa paralelizado utilizando MPI, como MFSim, sempre executará melhor em cores reais do que virtuais**. O MPI presente nos ambientes suportados pelo MFSim "sabe" disso e por isso gera esse erro, que acaba não sendo bem um erro.

Existe uma forma de contornar essa questão especificando para o MPI que é pra ele considerar o uso de *cores* virtuais. Isso é feito utilizando a flag `--oversubscribe`. Ex:

Suponha que você precisa rodar uma simulação em 8 processos, mas o seu processador possui 4 *cores* reais e 4 *cores* virtuais. Se você executar o comando:

```
$ mpirun -n 8 ./amr3d arquivo_cfg
```

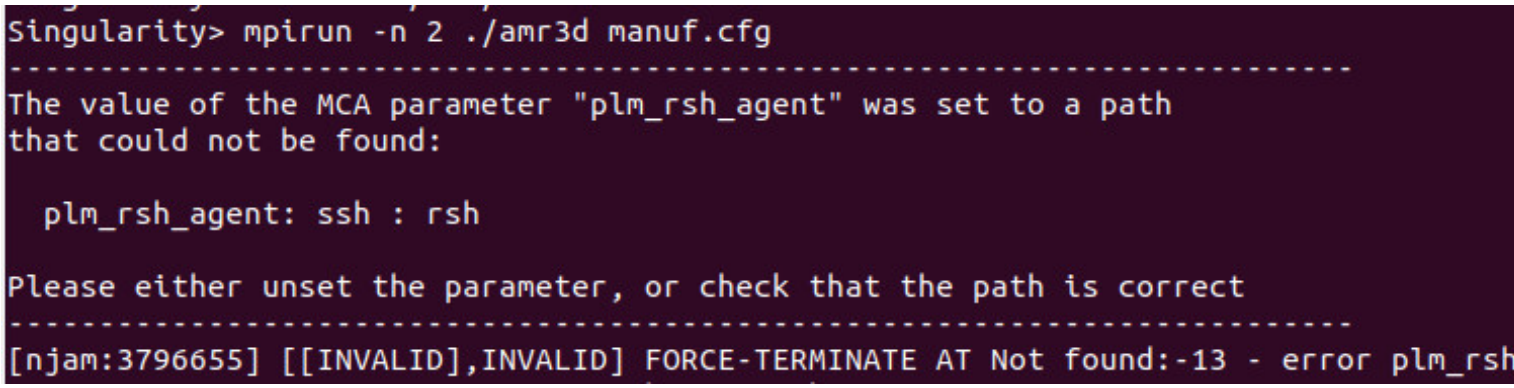
Você terá o erro de *slots*. Porém, se utilizar a flag:

```
$ mpirun -n 8 --oversubscribe ./amr3d arquivo_cfg
```

O código executará normalmente.

5.5 Erro de RSH

Ao tentar executar o MFSim via Singularity na stack 12 é possível que o seguinte erro aconteça:



Esse erro decorre por um problema interno do Singularity ao instanciar o *container*. A solução é bem simples e consiste em remapear os `rsh` para o `sh` **antes** de executar o MFSim:

```
Singularity> export OMPI_MCA_plm_rsh_agent=sh
Singularity> mpirun -n CORES ./amr3d arquivo_cfg
```

References

[1] Franco Barbi et al. Experimentação numérica de bolhas em ascensão. 2016.

[2] Marcelo Maia Ribeiro Damasceno et al. Desenvolvimento de uma modelagem para escoamentos reativos em malhas adaptativas do tipo bloco-estruturada. 2018.

[3] MMR Damasceno, JM Vedovoto, and A da Silveira-Neto. Turbulent inlet conditions modeling using large-eddy simulations. *Computer Modeling in Engineering & Sciences*, 104(2):105–132, 2015.

[4] Bernardo Alan de Freitas Duarte, Rafael Romão da Silva Melo, Millena Martins Villar, Ricardo Serfaty, and Aristeu da Silveira Neto. An extension of oberbeck–boussinesq approximation for thermal convection problems. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 40(6):317, 2018.

[5] Bernardo Alan de Freitas Duarte, Millena Martins Villar, Ricardo Serfaty, and Aristeu da Silveira Neto. Evaluation of a diffuse interface treatment for pressure in phase change simulations using adaptive mesh refinement. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 41(2):82, 2019.

[6] Alex José Elias et al. Modelagem híbrida urans-les para escoamentos turbulentos. 2018.

[7] M. Ferziger, J. H. e Peric. *Computational Methods for Fluid Dynamic*. Springer, Berlin, Germany, 2001.

[8] Massimo Germano, Ugo Piomelli, Parviz Moin, and William H Cabot. A dynamic subgrid-scale eddy viscosity model. *Physics of Fluids A: Fluid Dynamics*, 3(7):1760–1765, 1991.

[9] Brian Edward Launder and Dudley Brian Spalding. *Mathematical models of turbulence*. Number BOOK. Academic press, 1972.

[10] Douglas K Lilly. A proposed modification of the germano subgrid-scale closure method. *Physics of Fluids A: Fluid Dynamics*, 4(3):633–635, 1992.

[11] Clovis Raimundo Maliska. *Transferência de calor e mecânica dos fluidos computacional* . Grupo Gen-LTC, 2017.

[12] Rafael Romão da Silva Melo et al. Modelagem e simulação de escoamentos turbulentos com efeitos térmicos, utilizando a metodologia da fronteira imersa e malha adaptativa. 2017.

[13] F. R. Menter. Two-equation eddy-viscosity turbulence models for engineering applications. *AIAA Journal*, 32(8):1598–1605, 1994. URL: `https://arc.aiaa.org/doi/abs/10.2514/3.12149?journalCode=aiaaj`, `doi:doi.org/10.2514/3.12149`.

[14] Stéphane Popinet. An accurate adaptive solver for surface-tension-driven interfacial flows. *Journal of Computational Physics*, 228(16):5838–5866, 2009.

[15] Rodrigo Lisita Ribera et al. Understanding the dynamics of gas-liquid-solid contact points. 2015.

[16] Tsan-Hsing Shih, William W. Liou, Aamir Shabbir, Zhigang Yang, and Jiang Zhu. A new k-ϵ eddy viscosity model for high reynolds number turbulent flows. *Computers & Fluids*, 24(3):227 – 238, 1995. URL: `http://www.sciencedirect.com/science/article/pii/004579309400032T`, `doi:https://doi.org/10.1016/0045-7930(94)00032-T`.

[17] Maurits H Silvis, Ronald A Remmerswaal, and Roel Verstappen. Physical consistency of subgrid-scale models for large-eddy simulation of incompressible turbulent flows. *Physics of Fluids*, 29(1):015105, 2017.

[18] Joseph Smagorinsky. General circulation experiments with the primitive equations: I. the basic experiment. *Monthly weather review*, 91(3):99–164, 1963.

[19] P. R. Spalart and S. R. Allmaras. A one-equation turbulence model for aerodynamic flows. *AIAA Paper*, 1992. URL: `https://arc.aiaa.org/doi/10.2514/6.1992-439`, `doi:doi.org/10.2514/6.1992-439`.

[20] DAVID C. WILCOX. Reassessment of the scale-determining equation for advanced turbulence models. *AIAA Journal*, 26(11):1299–1310, 1988. URL: `https://arc.aiaa.org/doi/abs/10.2514/3.10041?journalCode`, `doi:10.2514/3.10041`.