INCOMPACT3D USER GUIDE version 2.0

Sylvain Laizet (Imperial College London)
Tradução: Luísa Vieira Lucchese, Leonardo Romero Monteiro

Conteúdo

1 Vista Geral do Incompact3d

O **Incompact3d** é um poderoso modelo de escoamentos para a pesquisa acadêmica. Ele pode combinar a versatilidade de códigos industriais com a precisão de códigos espectrais. Graças a projetos que obtiveram sucesso com NAG e HECToR (antiga instalação de supercomputação do Reino Unido), o **Incompact3d** pode ser usado em mais de um milhão de núcleos computacionais para resolver as equações incompressíveis de Navier-Stokes. O alto nível de paralelização é alcançado graças a uma biblioteca de comandos de decomposição 2D altamente escalonáveis, e também graças a uma interface de Transformações Rápidas de Fourier (FFT) distribuídas. [2] Essa biblioteca de comandos está disponível em http://www.2decomp.org e pode ser utilizada gratuitamente.
O **Incompact3d** é baseado em malha cartesiana. O uso de uma malha tão simplificada oferece a oportunidade de implementar esquemas compactos de alta ordem para a discretização espacial, enquanto que o uso do Método de Fronteiras Imersas (IBM) permite a implementação de qualquer sólido com geometria rombuda e/ou com paredes sólidas dentro do domínio computacional. A originalidade principal do código é a resolução da equação de Poisson (que garante a incompressibilidade), que é completamente resolvida no espaço espectral, utilizando a formalidade do números de onda modificados, não importando quais são as condições de contorno (periódica, deslizamento livre, não deslizamento, entrada/saída, etc.). Por fim, nota-se que a malha da pressão é deslocada em meio elemento de malha de velocidades, para evitar oscilações espúrias de pressão que poderiam ser introduzidas pelo IBM.

Mais informação sobre os métodos numéricos pode ser encontrada em:

- Laizet S.& Lamballais E., **High-order compact schemes for incompressible flows: a simple and efficient method with the quasi-spectral accuracy**, J. Comp. Phys., vol 228-15, pp 5989-6015, 2009
- Lamballais E., Fortune V. & Laizet S., **Straightforward high-order numerical dissipation via the viscous term for Direct and Large Eddy Simulation**, J. Comp. Phys., Vol 230-9, pp 3270-3275, 2011

Mais informação sobre a estratégia de paralelização do código pode ser encontrada em:

- Laizet S.& Li N., **Incompact3d, a powerful tool to tackle turbulence problems with up to** $0(10^5)$ **computational cores**, Int. J. of Numerical Methods in Fluids, Vol 67-11, pp 1735-1757, 2011
- Li N. & Laizet S., **2DECOMP&FFT - a highly scalable 2D decomposition library and FFT interface**, Cray User Group meeting: Simulation comes of age Edinburgh, Scotland – 24/05-27/05, 2010
- Laizet S., Lamballais E. & Vassilicos J.C., **A numerical strategy to combine high-order schemes, complex geometry and parallel computing for high resolution DNS of fractal generated turbulence**, Computers & Fluids, vol 39-3, pp 471-484, 2010

IMPORTANTE:

- **É fortemente recomendado ler as referências anteriores antes de começar a usar o Incompact3d.**
- **Nós pedimos educadamente que você cite as referências anteriores (quando adequado) no seu trabalho baseado no Incompact3d.**

2 Instalação

Quando se inicia o uso do **Incompact3d** em seu computador, é recomendado instalar os programas **gfortran** e **openmpi** ((preferencialmente com a distribuição **Ubuntu**). É possível de fazer download dos arquivos mais atuais do **openmpi** no endereço http://www.open-mpi.org/software/ompi/v1.10/. Veja a baixo os comandos para instalar o **gfortran** e o **openmpi**:

```
sudo apt-get install build-essential
sudo apt-get install gfortran
tar -xvzf openmpi-***.tar.gz
cd openmpi***
sudo ./configure  --prefix=/usr/local F77=gfortran FC=gfortran
sudo make all install
```

Também é necessário adicionar uma linha no arquivo `.bashrc`:

```
export LD_LIBRARY_PATH=/usr/local/lib/
```

2.1 Compilando incompact3d

Há um arquivo Makefile disponível e a única coisa a fazer é escolher os comandos relevantes. Para utilizar a combinação **gfortran+openmpi**, deve-se utilizar:

```
FC = mpif90
OPTFC = -O3 -funroll-loops -ftree-vectorize -fcray-pointer -cpp
CC = mpicc
CFLAGS = -O3
```

Para compilar o código, você apenas precisa escrever no terminal com o caminho até a pasta onde o código está os comandos: `make clean` e `make`.

2.2 Problemas conhecidos

Alguns usuários tiverem dificuldades com a biblioteca MPI `mpich`. O **Incompact3d** já foi usado em uma ampla variedades de supercomputadores com vários compiladores e bibliotecas de MPI.

É altamente recomendado de ler cuidadosamente a documentação do supercomputador que você está utilizando e você deve encontrar todas as informações que você precisa sobre as melhores opções de compilação.

3 O código

Nesta seção, somente os arquivos relacionados com a resolução das equações de Navier-Stokes para fluidos incompressíveis são descritos. A decomposição do domínio 2D e seus arquivos vem da biblioteca do código-aberto 2DECOMP&FFT e não serão explicados neste documento.

3.1 incompact3d.f90

Este é o arquivo principal do código, que contém o laço temporal relativo a resolução das equações de Navier-Stokes para escoamentos incompressíveis.

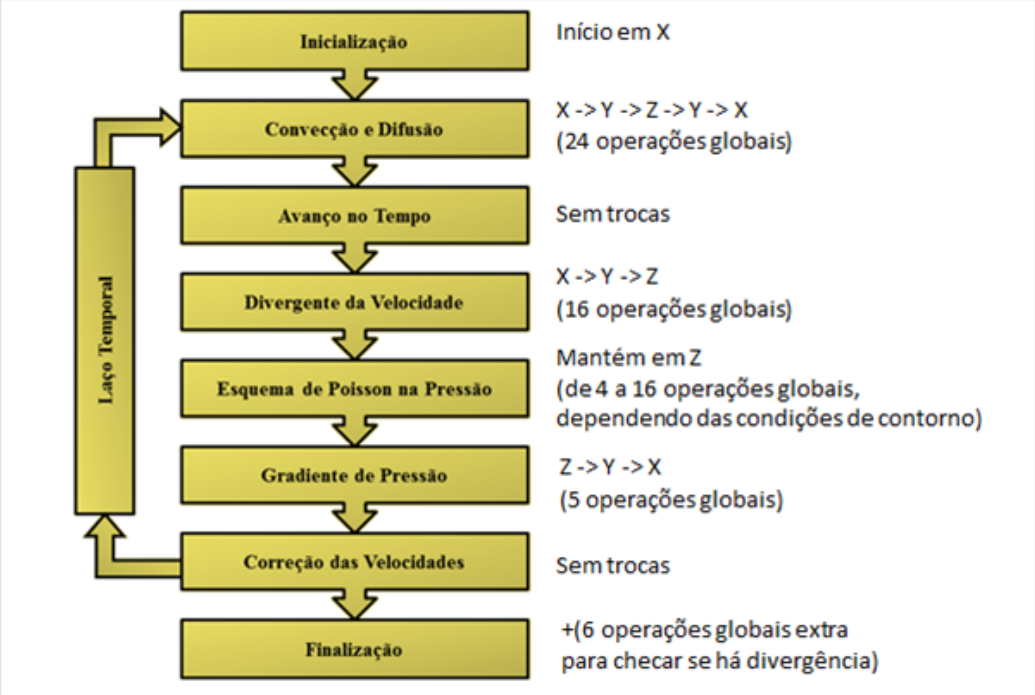

Figura 1: *Estrutura do* **Incompact3d** *com a decomposição de domínio 2D.*

A estrutura do modelo é apresentada na figura 1. Esta figura também mostra o gerenciamento da troca de lápis. Para resolver a equação de Poisson no campo espectral, uma única divisão e necessária e os números de onda modificados, combinado com a função de transferência, são totalmente independentes uns dos outros. Percebe-se que, quando o FTT 3D forward está sendo utilizado, estamos no espaço físico do lápis Z, e então se passa ao lápis X no espaço espectral e finalmente, ao lápis Z novamente após o FFT 3D backward, com o intuito de reduzir as operações globais de transposição. Para o código que usar condições de contorno tri-periódicas, a cada passo de tempo, 55 operações globais de transposição são necessárias a cada passo de tempo. Esse número pode subir para 69 por causa das condições de contorno.

Como é possível de ver, o código utiliza vários módulos (para as constantes, arranjos 1D e 2D). Todos os módulos podem ser encontrados em `module_param.f90`.

As variáveis $t1$ e $t2$ são utilizadas para calcular a média do custo por passo de tempo da simulação.

É possível de salvar as estatísticas e as visualizações em uma malha grosseira (a cada `nstat` e `nvisu` pontos de malha, respectivamente).

```
call init_coarser_mesh_statS(nstat,nstat,nstat,.true.)
call init_coarser_mesh_statV(nvisu,nvisu,nvisu,.true.)
```

As estatísticas e a visualização são calculadas no `visu.f90`.

Os parâmetros da simulação são inicializados com

```
call parameter()
call schemes ()
```

Nota-se que pelo campo de pressões ser deslocado por meia malha em respeito ao campo de velocidade, é necessário definir diversas tamanhos de malha (para as interpolações) dependendo das condições de contorno da simulação (`phG` variável).

As rotinas a seguir são chamadas para inicializar o campo do escoamento. É possível utilizar um procedimento de checkpoint para reiniciar a simulação a partir de uma anterior (parâmetro `ilit`). O procedimento de reinicialização pode ser encontrado em `tools.f90` e o arquivo de reinicialização é chamado de `sauve.dat`. Nota-se que para alguns supercomputadores (normalmente os IBM) e para simulações muito grandes, talvez seja necessário dividir o `sauve.dat` em arquivos menores, pois alguns supercomputadores não lidam com arquivos muito grandes.

```
if (ilit==0) call init(ux1,uy1,uz1,ep1,phi1,gx1,gy1,gz1,phis1,hx1,hy1,hz1,phiss1)
if (ilit==1) call restart(ux1,uy1,uz1,ep1,pp3,phi1,gx1,gy1,gz1,&
px1,py1,pz1,phis1,hx1,hy1,hz1,phiss1,phG,0)
call test_speed_min_max(ux1,uy1,uz1)
if (iscalar==1) call test_scalar_min_max(phi1)
```

3.2 incompact3d.prm

O nome de cada variável é claramente explicado no arquivo. Todas as quantidades são adimensionais com um comprimento e uma velocidade.

O tamanho do domínio computacional é $xlx \times yly \times zlz$.

Não existe verificação do intervalo de tempo, então é responsabilidade do usuário verificar cuidadosamente sobre as condições de estabilidade para a simulação. O intervalo de tempo ótimo tem que ser definido pelo procedimento de tentativa e erro. É possível utilizar o esquema explicito de Adam-Bashforth (recomendado quando se utiliza o Método de Fronteiras Imersas - IBM) ou esquemas explícitos de Runge-Kutta (parâmetro `ncheme`). Em algumas versões do código, uma estratégia semi-implícita foi implementada para melhorar a condição de estabilidade para o intervalo de tempo.

O intervalo de tempo fixo para uma dada simulação, não sendo possível (ainda?) de usar diferentes intervalos de tempos ao longo da simulação.

Diferentes condições de contorno pode ser usadas nas três direções espaciais (`nclx,ncly e nclz`) como é possível de ver na figura 2:

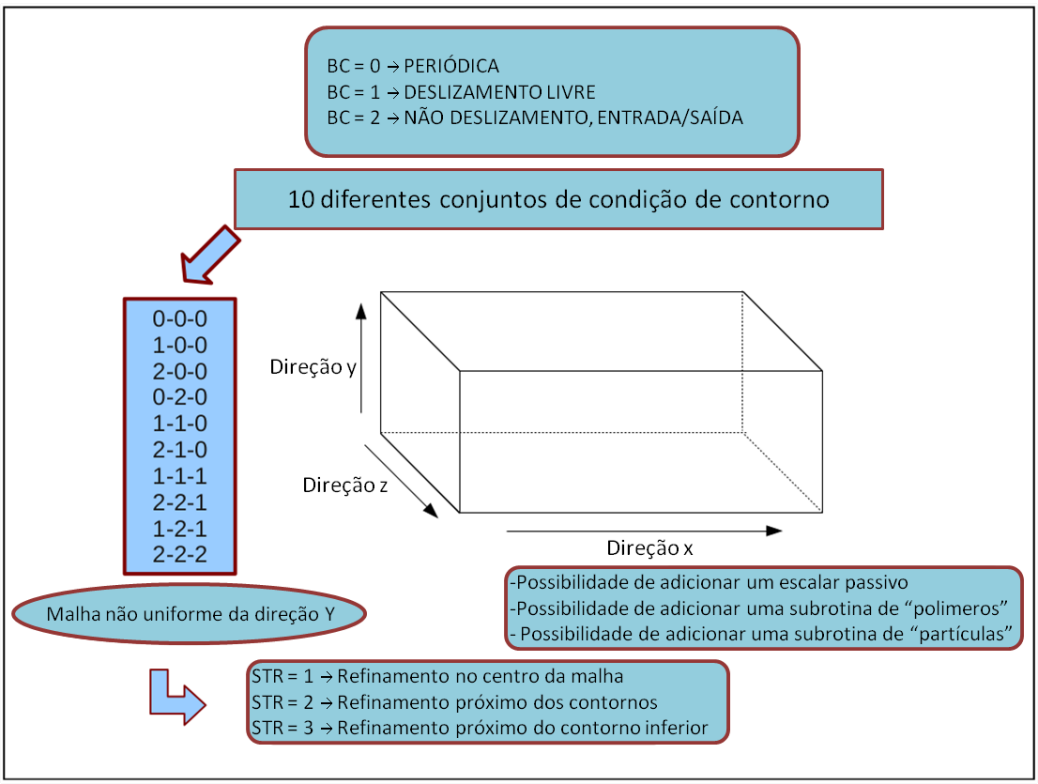

Figura 2: *Descrição das diferentes condições de contorno (falta traduzir a figura).*

- Condições periódicas correspondentes a $ncl = 0$
- Condições de livre-deslizamento correspondentes a $ncl = 1$
- Condições de contorno abertas (condição de Dirichlet para a velocidade para não-deslizamento ou condições de entrada e saída), correspondentes a $ncl = 2$

Até agora e como pode ser visto na figura acima, 12 combinações diferentes podem ser usadas no código, cobrindo uma larga gama de configurações de fluxo: $(0-0-0)$, $(1-0-0)$, $(2-0-0)$, $(0-2-0)$, $(1-1-0)$, $(2-1-0)$, $(1-1-1)$, $(2-2-1)$, $(2-1-2)$, $(1-2-1)$, $(1-1-2)$ and $(2-2-2)$.

Para exemplificar, a combinação para um fluxo em um canal é $(0-2-0)$. Para um fluxo que evolui espacialmente, com condições de contorno periódicas na direção do fluxo e fronteiras laterais com livre-deslizamento, a combinação é $(2-1-0)$. Nem todas as combinações foram testadas então podem ser encontrados alguns problemas.

No momento, as mesmas condições de contorno tem que ser impostas em uma mesma direção.

Alguma codificação é necessária se quiser impor, vamos dizer, uma condição de não-deslizamento e uma condição de deslizamento livre em uma direção espacial (por exemplo em uma camada limite turbulenta).

Se quiser utilizar um escalar passivo com diferentes condições de contorno e o campo de velocidades como 1, alguma codificação é necessária: terá de serão de ser criadas novas subrotinas de derivativos com o correto conjunto de coeficientes, dependendo da condição de contorno.

É possível usar uma malha com stretching na direção y com o parâmetro *istret*. Se $istret = 0$, não é utilizado stretching. Caso $istret = 1$ então ele é utilizado e com um refinamento da malha no centro do domínio computacional. Caso $istret = 2$ então o refinamento se dá na fronteira do domínio computacional. Finalmente, caso $istret = 3$ então o refinamento é somente em baixo no domínio computacional. $istret = 2$ pode ser utilizado para um fluxo turbulento

em canal, $istret = 3$ para uma camada limite turbulenta. O parâmetro `beta` é o parâmetro de refinamento (pequenos valores para um grande refinamento e grandes valores para uma malha quase que regular).

É possivel escolher uma configuração diferente de escoamento com o parâmetro `itype` , mas ele está inteiramente implementado ainda. A definição da configuração do escoamento (e as condições iniciais e/ou de entrada) pode ser feita pelas subrotinas `init` e `ecoule`.

Os parâmetros `ifirst` e `ilast`, correspondem, respectivamente, a primeira e a última iteração. Se o procedimento de checkpoint for utilizado, é possível de dividir a simulação e menores.

Duas diferentes formulações podem ser usadas para os termos convectivos da equação de Navier-Stokes:

- a formulação rotacional $H_i = u_j \left(\frac{\partial u_i}{\partial x_j} - \frac{\partial u_j}{\partial x_i} \right)$ correspondendo a $iskew = 0$,
- a formulação antissimétrica $H_i = \frac{1}{2} \left(\frac{\partial u_i u_j}{\partial x_j} + u_j \frac{\partial u_i}{\partial x_j} \right)$ correspondendo a $iskew = 1$.

Recomenda-se a formulação antissimétrica para melhor modelagem das pequenas escalas e menos efeitos de aliasing.

É possivel de combinar **Incompact3d** com Métodos de Fronteiras Virtuais (IBM). Para as últimas implementações no modelo, por favor verificar:

Gautier R., Laizet S. & Lamballais E., 2014, **A DNS study of jet control with microjets using an alternating direction forcing strategy**, *Int. J. of Computational Fluid Dynamics*, **28**, 393–410

3.3 parameters.f90

Este arquivo contém a subrotina parameter que é usada para inicialização da simulação (por exemplo, os coeficientes de tempo e o tamanho das malhas Δx, Δy e Δz).

A subrotina `parameter` lê o arquivo `incompact3d.prm`. Se um stretching é utilizado na direção y então a subrotina `stretching` é chamada para definir o arranjo 1D `yp`, o qual contém as coordenadas da malha na direção y.

3.4 variable.f90

Os arranjos 3D são dinamicamente definidos neste arquivo, dependendo do tamanho da simulação e do número de núcleos computacionais. O arranjo 3D com 1 no final (ou com 2 ou 3) são definidos nos lápis X (ou nos lápis Y ou Z). Para os arranjos 3D no lápis X, duas opções são possíveis. O $*size$ é usado se não há necessidade de usar os coeficientes i, j, ou k em uma subrotina (por exemplo na subrotina do avanço do tempo `intt`). Para guardar a velocidade uy do passo de tempo anterior, p arranjo gy é definida com $allocate(gy(xsize(1), xsize(2), xsize(3)$, bem como uyl é definida com $allocate(uy1(xstart(1) : xend(1), xstart(2) : xend(2), xstart(3) : xend(3)))$. Na prática, as duas matrizes tem o mesmo tamanho, mas os coeficientes i, j, k são diferentes.

EXEMPLO:
Uma simulação é baseada em$n_x \times n_y \times n_z = 129 \times 64 \times 32$ nós de malha, usando uma mapeamento 2D $p_{row} \times p_{col} = 4 \times 8 = 32$ núcleos de processamento, numerados de 0 a 31.

Para esta configuração, $xsize(1) = n_x$, $xsize(2) = n_y/p_{row} = 16$, $xsize(3) = n_z/p_{col} = 4$.

O tamanho de The size of gy é $gy(129, 16, 4)$ com $i = 1, .., 129$, $j = 1, .., 16$ e $k = 1, .., 4$ para todos os números.

O tamanho de uy é $uy(129, 16, 4)$ com $i = 1, .., 129$ para todas as numerações, $j = 1, .., 16$ para numerações de 0 a 7, $j = 17, .., 32$ para numerações de 8 a 15, $j = 33, .., 48$ para numerações de 16 a 23 e $j = 49, .., 64$ para numerações de 24 a 31; $k = 1, .., 4$ para numerações 0/8/16 e 24; $k = 5, .., 8$ para numerações 1/9/17 e 25, $k = 9, .., 12$ para numerações 2/10/18 e 26; $k = 13, .., 16$ para numerações 3/11/19 e 27, $k = 17, .., 20$ para numerações 4/12/20 e 28, $k = 21, .., 24$ para numerações 5/13/21 e 29, $k = 25, .., 28$ para numerações 6/14/22 e 30 e finalmente $k = 29, .., 32$ para numerações 7/15/23 e 31.

Quando estiver escrevendo sua própria sub-rotina no código, ou antes de escreve-la, é recomendado verificar o $*size, *start$ e $*end$ para entender melhor o mapeamento de decomposição 2D.

3.5 schemes.f90 e derive.f90

Os coeficientes para o esquema de sexta ordem são definidos em `schemes.f90`. Os arranjos 1D são utilizados para definir diferentes matrizes relacionadas ao cálculo dos derivativos de primeira e segunda ordem. As subrotinas de derivativos e interpolações são definidas em `derive.f90`

Mais detalhas são fornecidos a seguir. A referência principal para os esquemas compactos (derivativos, interpolação do ponto intermediário, para diferentes condições de contorno) é:

LELE, K. S., 1992, **Compact finite difference schemes with spectral-like resolution**, *Journal of computational physics*, **103(1)**, 16-42.

3.5.1 Discretização Espacial

Primeiro consideraremos uma distribuição uniforme de n_x pontos x_i no domínio $[0, L_x]$ com $x_i = (i-1)\Delta x$ para $1 \leq i \leq n_x$. A aproximação dos valores $f'_i = f'(x_i)$ do primeiro derivativo $f'(x)$ da função $f(x)$ pode estar relacionado a valores $f_i = f(x_i)$ por um esquema de diferenças finitas na forma

$$\alpha f'_{i-1} + f'_i + \alpha f'_{i+1} = a\ \frac{f_{i+1} - f_{i-1}}{2\Delta x} + b\ \frac{f_{i+2} - f_{i-2}}{4\Delta x} \tag{1}$$

Escolhendo $\alpha = 1/3$, $a = 14/9$ e $b = 1/9$, esta aproximação possui a precisão de sexta ordem que possui um então chamado "comportamento quasi-espectral" que fornece a capacidade de representar precisamente uma ampla variedade de escalas.

Analogamente relações podem ser estabelecidas para as aproximações dos valores dos derivativos de segunda ordem $f''_i = f''(x_i)$ com

$$\alpha f''_{i-1} + f''_i + \alpha f''_{i+1} = a\ \frac{f_{i+1} - 2f_i + f_{i-1}}{\Delta x^2} + b\ \frac{f_{i+2} - 2f_i + f_{i-2}}{4\Delta x^2} + c\ \frac{f_{i+3} - 2f_i + f_{i-3}}{9\Delta x^2} \tag{2}$$

Escolhendo $\alpha = 2/11$, $a = 12/11$, $b = 3/11$ e $c = 0$, está aproximação possui a precisão de sexta ordem. Para controlar do erro de aliasing pelo termo viscoso, este tipo de esquema também pode ser definido para ser sobre dissipativo nos maiores números de onda (no alcance espectral, onde mesmo um esquema de diferenças finitas de alta ordem se torna impreciso).Mais detalhes podem ser encontrados em:

LAMBALLAIS E., FORTUNE V. & LAIZET S., 2011 **Straightforward high-order numerical dissipation via the viscous term for Direct and Large Eddy Simulation**, *J. Comp. Phys.*, **230(9)**, 3270-3275.

Como já dito, quatro diferentes condições de contorno podem ser consideradas em cada direção espacial. As condições de contorno periódica e de deslizamento livre (equivalentes as condições simétricas e antissimétricas) permitem o uso de esquemas (1, 2) para todos os pontos considerados. Mais precisamente, os esquemas (1, 2) apenas precisam ser relevantemente modificados próximo dos contornos $i = 1$ e $i = n_x$ pela substituição dos valores dos pontos extras (algumas vezes chamados de "fantasmas") f_0, f_{-1}, f_{n_x+1}, f_{n_x+2} pelo seus homólogos f_1, f_2, f_{n_x-1}, f_{n_x-2}. Para uma condição de contorno periódica, este tipo de substituição para f, f' e f'' pode ser escrita

$$f_0 \to f_{n_x},\ f_{-1} \to f_{n_x-1},\ f'_0 \to f'_{n_x},\ f''_0 \to f''_{n_x}, \tag{3}$$

enquanto as condições simétricas e antissimétricas levam a

$$f_0 \to f_2,\ f_{-1} \to f_3,\ f'_0 \to -f'_2,\ f''_0 \to f''_2, \tag{4}$$

e

$$f_0 \to -f_2,\ f_{-1} \to -f_3,\ f'_0 \to f'_2,\ f''_0 \to -f''_2, \tag{5}$$

respectivamente. Por simplicidade, apenas as relações no contorno esquerdo (próximo de $i = 1$) são apresentadas aqui, suas partes homólogas a direita (próximo de $i = n_x$) sendo facilmente deduzido.

Quando condições de contorno de não deslizamento ou abertas (i.e. Dirichlet para a velocidade) são usadas, quase nada é assumido fora o escoamento de saída do domínio computacional. Formulações de um único lado são utilizadas para as aproximações de primeira e de segunda ordem para estes tipos de contorno, usando relações da forma

$$\begin{aligned} f'_1 + 2f'_2 &= \frac{1}{2\Delta x}(-5f_1 + 4f_2 + f_3) \\ f''_1 + 11f''_2 &= \frac{1}{\Delta x^2}(13f_1 - 27f_2 + 15f_3 - f_4) \end{aligned} \tag{6}$$

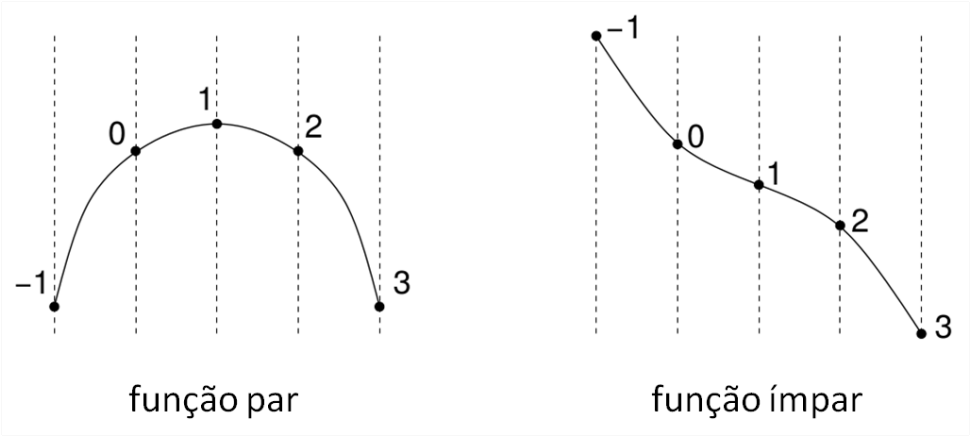

Figura 3: *condições de contorno simétrica e antissimétrica.*

que possuem precisão de terceira ordem. Nos pontos adjacentes, por que uma formulação de três pontos deve ser utilizada, o esquema de Padé deve ser aplicado com

$$\begin{aligned} \frac{1}{4}f_1' + f_2' + \frac{1}{4}f_3' &= \frac{3}{2}\,\frac{f_3 - f_1}{2\Delta x} \\ \frac{1}{10}f_1'' + f_2'' + \frac{1}{10}f_3'' &= \frac{6}{5}\,\frac{f_3 - 2f_2 + f_1}{\Delta x^2} \end{aligned} \tag{7}$$

sendo estes esquemas de quarta ordem de precisão.

3.5.2 Formulação de Matriz

As relações antecedentes podem ser escritas como matrizes:

$$\mathbf{A}_x\mathbf{f}' = \frac{1}{\Delta x}\mathbf{B}_x\mathbf{f} \tag{8}$$

$$\mathbf{A}'_x\mathbf{f}'' = \frac{1}{\Delta x^2}\mathbf{B}'_x\mathbf{f} \tag{9}$$

onde $\mathbf{A}_x, \mathbf{A}'_x, \mathbf{B}_x, \mathbf{B}'_x$ são matrizes com coeficientes $n_x \times n_x$. $\mathbf{f}$, $\mathbf{f}'$ e $\mathbf{f}''$ são vetores de tamanho n_x.

1. *caso periódico* : Nós temos $f_{n_x+1} = f_1$ and $f_0 = f_{n_x}$. $\mathbf{A}_x$ e $\mathbf{B}_x$ podem ser escritos como:

$$\mathbf{A}_x = \begin{pmatrix} 1 & \alpha & & & & & & \alpha \\ \alpha & 1 & \alpha & & & & & \\ & \cdot & \cdot & \cdot & & & & \\ & & \cdot & \cdot & \cdot & & & \\ & & & \cdot & \cdot & \cdot & & \\ & & & & \cdot & \cdot & \cdot & \\ & & & & & \alpha & 1 & \alpha \\ \alpha & & & & & & \alpha & 1 \end{pmatrix} \quad \text{e} \quad \mathbf{B}_x = \begin{pmatrix} 0 & a & b & & & & -b & -a \\ -a & 0 & a & b & & & & -b \\ -b & -a & 0 & a & b & & & \\ & \cdot & \cdot & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \cdot & \cdot & \\ & & & -b & -a & 0 & a & b \\ b & & & & -b & -a & 0 & a \\ a & b & & & & -b & -a & 0 \end{pmatrix}$$

2. *caso de deslizamento livre* : Nós podemos ter duas situações, sendo elas as condições de contorno simétrica e antissimétrica. f pode ser ímpar ou par como demonstrado na figura 3. Quando f é par nós temos:

$$\mathbf{A}_x = \begin{pmatrix} 1 & 0 & & & & & \\ \alpha & 1 & \alpha & & & & \\ & \cdot & \cdot & \cdot & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & \alpha & 1 & \alpha \\ & & & & & 0 & 1 \end{pmatrix} \quad \text{e} \quad \mathbf{B}_x = \begin{pmatrix} 0 & & & & & & \\ -a & -b & a & b & & & \\ -b & -a & 0 & a & b & & \\ & \cdot & \cdot & \cdot & \cdot & \cdot & \\ & & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & -b & -a & 0 & a & b \\ & & & & -b & -a & b & a \\ & & & & & & & 0 \end{pmatrix}$$

Quando **f** é ímpar nós temos:

$$\mathbf{A}_x = \begin{pmatrix} 1 & 2\alpha & & & & & & \\ \alpha & 1 & \alpha & & & & & \\ & \cdot & \cdot & \cdot & & & & \\ & & \cdot & \cdot & \cdot & & & \\ & & & \cdot & \cdot & \cdot & & \\ & & & & \cdot & \cdot & \cdot & \\ & & & & & \alpha & 1 & \alpha \\ & & & & & & 2\alpha & 1 \end{pmatrix} \quad \text{e} \quad \mathbf{B}_x = \begin{pmatrix} 0 & 2a & 2b & & & & & \\ -a & b & a & b & & & & \\ -b & -a & 0 & a & b & & & \\ & \cdot & \cdot & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \cdot & \cdot & \\ & & & -b & -a & 0 & a & b \\ & & & & -b & -a & b & a \\ & & & & & 2b & 2a & 0 \end{pmatrix}$$

3. *Caso de Dirichlet* : Neste caso nós utilizamos o esquema não centrado e temos:

$$\mathbf{A}_x = \begin{pmatrix} 1 & \alpha_1 & & & & & & \\ \alpha_2 & 1 & \alpha_2 & & & & & \\ & \alpha & 1 & \alpha & & & & \\ & & \cdot & \cdot & \cdot & & & \\ & & & \cdot & \cdot & \cdot & & \\ & & & & \cdot & \cdot & \cdot & \\ & & & & & \alpha_{n_x-1} & 1 & \alpha_{n_x-1} \\ & & & & & & \alpha_{n_x} & 1 \end{pmatrix}$$

$$\mathbf{B}_x = \begin{pmatrix} a_1 & b_1 & c_1 & & & & & \\ -a_2 & 0 & a_2 & & & & & \\ -b & -a & 0 & a & b & & & \\ & \cdot & \cdot & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \cdot & \cdot & \\ & & & -b & -a & 0 & a & b \\ & & & & & -a_{n_x-1} & 0 & a_{n_x-1} \\ & & & & & c_{n_x} & b_{n_x} & a_{n_x} \end{pmatrix}$$

COMENTÁRIO IMPORTANTE: Note que o parâmetro `npaire` é apenas ativado quando `ncl=1`. Por favor, veja abaixo os valores que dependem das componentes de velocidade e da direção espacial:

	X	Y	Z
ux	0	1	1
uy	1	0	1
uz	1	1	0
uxux	1	1	1
uyuy	1	1	1
uzuz	1	1	1
uxuy	0	0	1
uxuz	0	1	0
uyuz	1	0	0
dux/dx	1	1	1
duy/dx	0	0	1
duz/dx	0	1	0
dux/dy	0	0	1
duy/dy	1	1	1
duz/dy	1	0	0
dux/dz	0	1	0
duy/dz	1	0	0
duz/dz	1	1	1

3.6 tools.f90

3.6.1 Subroutina test_speed_min_max

Esta subrotina é utilizada para calcular o máximo e o mínimo do campo de velocidade. Note que o valor 1609 não possui um significado físico, e sim, corresponde a data de aniversário de Sylvain Laizet (16 de Setembro) e é usado para inicialização dos valores de min/max.

3.6.2 Subroutina test_scalar_min_max

Esta subrotina é utilizada para calcular o máximo e mínimo do campo escalar.

3.6.3 Subroutina restart

Esta subrotina é utilizada ou gerar um arquivo de restart ou para ler um arquivo de restart (parâmetro `ilit` no `incompact3d.prm`). Como já mencionado, você talvez precisa dividir o arquivo de restart em supercomputadores IBM quando a simulação é muito grande.

3.6.4 Subroutina stretching

Esta subrotina é ativada apenas quando uma malha com *stretch* é utilizada na direção y. Ela define uma malha com *stretch* e algum arranjo 1D que é utilizado para os derivativos e interpolações. Mais detalhes podem ser encontrados em

LAIZET S.& LAMBALLAIS E., 2009, **High-order compact schemes for incompressible flows: a simple and efficient method with the quasi-spectral accuracy**, *J. Comp. Phys.*, **228(15)**, 5989-6015.

3.6.5 Subroutinas de inversão

Diversas subrotinas estão disponíveis para a inversão de matrizes pentadiagonais. Isto é necessário quando uma malha com *stretch* é utilizada para direção y.

3.7 decomp_2d.f90

Este arquivo faz parte da biblioteca `2D Decomp%FFT` que está gratuitamente disponível em http://www.2decomp.org/. Ele implementa uma decomposição 2D com um propósito geral no `Incompact3d`. Os arquivos que estão relacionados a esta biblioteca não serão explicados neste documento. Como uma biblioteca de comunicação, ela implementa a assim chamada decomposição em lápis 2D para particionar do conjunto de dados tridimensionais em sistemas distribuídos e fazer comunicação baseada em transposição. Também provém uma interface eficiente e altamente útil para fazer Transformações Rápidas de Fourier (FFT) tridimensionais. A biblioteca foi escrita em Fortran e montada com base em MPI. Não é recomendado modificar este arquivo.

3.8 convdiff.f90

Os termos convectivos e difusivos das equações de Navier Stokes são calculados neste arquivo. Como já mencionado, duas formulações diferentes podem ser utilizadas para os termos convectivos:

- a formulação rotacional $H_i = u_j \left(\frac{\partial u_i}{\partial x_j} - \frac{\partial u_j}{\partial x_i} \right)$ correspondente a $iskew = 0$,
- a formulação antissimétrica $H_i = \frac{1}{2} \left(\frac{\partial u_i u_j}{\partial x_j} + u_j \frac{\partial u_i}{\partial x_j} \right)$ correspondente a $iskew = 1$.

Nós recomendamos utilizar a formulação antissimétrica para uma melhor modelação das pequenas escalas e para menos efeito de *aliasing*. Note que no começo e no final da subrotina dados do lápis X são armazenados.

Note que, quando uma malha com *strech* é utilizada na direção y, a estimativa do derivativo de segundo ordem é

$$\frac{\partial^2 f}{\partial y^2} = \frac{\partial^2 f}{\partial s^2} \left(\frac{ds}{dy} \right)^2 + \frac{\partial f}{\partial s} \frac{d^2 s}{dy^2} = \frac{1}{h'^2} \frac{\partial^2 f}{\partial s^2} - \frac{h''}{h'^3} \frac{\partial f}{\partial s}$$

com

$$y = h(s), \qquad 0 \leq s \leq 1, \qquad 0 \leq y \leq L_y$$

onde $h(s)$ é o mapeamento de coordenadas igualmente espaçadas e s para o as coordenadas y com espaçamento com *stretch*.

Modelos de Simulação de Grandes Escalas (Large-Eddy Simulations/LES) devem ser implementados no final da subrotina `convdiff`.

Qualquer termo de força também deve ser implementado nesta subrotina (por exemplo, a força de Coriolis).

3.9 navier.f90

3.9.1 Subrotina intt

Subrotina para o avanço explícito do tempo da equação de Navier Stokes. Note que os ciclos 3D são completamente verotizados, com a notação `(ijk,1,1)`. Os dados são armazenados no lápis X.

3.9.2 Subrotina corgp

Subrotina para o passo de correção no método do passo fracionado. O campo de velocidade é corrigido pelo gradiente de pressão para se tornar incompressível. Os dados são armazenados no lápis X.

No caso do escoamento em canal temporal, é necessário modificar a taxa de escoamento para ser levado em consideração na perda de pressão na parede (subroutina `channel`).

3.9.3 Subrotinas inflow and outflow

Estas subrotinas são utilizadas somente se as condições de contorno de entrada/saída são usadas na direção fluxo do escoamento (`nclx=2`). A condição de saída é uma simples equação de convecção 1D. A condição de entrada pode ser adaptada para qualquer tipo de condição de entrada pela subrotina `ecoule`.

Caso necessário, um ruído aleatório pode ser adicionado na entrada.

3.9.4 Subrotina ecoule

Nesta subrotina, é definida a condição de entrada. No momento, apenas algumas configurações de escoamentos estão implementadas (`itype==1` para um campo constante, `itype==2` para um perfil Poisseuille, `itype==6` para os vórtices de Taylor-Green).

Note que a subrotina `ecoule` é chamada na subrotina `inflow` a cada passo de tempo.

3.9.5 Subrotina init

Subrotina dedicada para a inicialização da simulação pela subrotina `ecoule`. Os dados são armazenados no lápis X.

Um ruído aleatório pode ser adicionado na condição inicial.

3.9.6 Subrotina divergence

Subrotina dedicada para o cálculo de $\boldsymbol{\nabla}.\mathbf{u}^{**}$ quando `nlock=1` e para $\boldsymbol{\nabla}.\mathbf{u}^{n+1}$ quando `nlock=2`.

Note que as entradas são as três componentes da velocidade armazenadas no lápis X e a saída é definida no lápis Z (o primeiro argumento da subrotina). A saída é definida na malha deslocada.

3.9.7 Subrotina gradp

Calculado dos três gradientes de pressão no lápis X (na malha da velocidade) para o campo de pressão (definido na malha deslocada) o qual é calculado no lápis Z.

3.9.8 Subrotina corgp_IBM

Apenas usada quando Métods de Fronteira Imersa (Immersed Boundary Methods) são utilizados. Ela tem que fazer uma pre- e pós-correção no gradiente de pressão no lado de dentro da região do sólido. Mais detalhes podem ser encontrados em:

Laizet S.& Lamballais E., 2009, **High-order compact schemes for incompressible flows: a simple and efficient method with the quasi-spectral accuracy**, *J. Comp. Phys.*, **228(15)**, 5989-6015.

3.9.9 Subrotina body

Somente utilizado quando os Métodos de Fronteira Imersa (Immersed Boundary Methods -IBM) são utilizados. Nesta subrotina, o campo de velocidades é forçado a zero dentro das regiões sólidas.

Note que novos IBM estão em desenvolvimento. O último método implementado para no *solver* esta descrito em:

Gautier R., Laizet S. & Lamballais E. 2014 **A DNS study of jet control with microjets using an alternating direction forcing strategy**, *Int. J. of Computational Fluid Dynamics*, **28**, 393–410

3.9.10 Subrotina pre_correc

É possível impor condições de contorno em $\mathbf{u}^{k+1}$ por causa da incompressibilidade (o campo de velocidades não pode ser modificado após o passo de correção com os gradientes de pressão). Assim, a imposição das condições de contorno são realizadas em $\mathbf{u}^{**}$ ao invés de $\mathbf{u}^{k+1}$.

$\Rightarrow$ Veja o exemplo para o escoamento em canal:

$$\mathbf{u}^{*}_{wall} = 0$$

$$\mathbf{u}^{k+1}_{wall} = \mathbf{u}^{*}_{wall} - \boldsymbol{\nabla}\tilde{p}^{k+1}$$

$$\mathbf{u}^{k+1}_{wall} = -\boldsymbol{\nabla}\tilde{p}^{k+1} \neq 0$$

Isto não é satisfatório. É melhor fazer o seguinte:

$$\mathbf{u}^{*}_{wall} = \boldsymbol{\nabla}\tilde{p}^{k}$$

$$\mathbf{u}^{k+1}_{wall} = \mathbf{u}^{*}_{wall} - \boldsymbol{\nabla}\tilde{p}^{k+1}$$

$$\mathbf{u}^{k+1}_{wall} = \boldsymbol{\nabla}\tilde{p}^{k} - \boldsymbol{\nabla}\tilde{p}^{k+1} \approx 0$$

Como pode ser visto, correções com os gradientes de pressão (no passo de tempo anterior) são necessárias para que as condições de contorno sejam mais precisas.

Note que a condição de contorno somente precisa ser imposta na subrotina `pre_correc` quando `ncl=2`.

Para configurações de escoamento particulares, é necessário verificar que a taxa de escoamento na saída é a mesma que na entrada. Este correção pode ser realizada na subrotina `pre_correc`.

3.10 poisson.f90

O arquivo tem que resolver a equação de Poisson, que é realizada no espaço espectral, mesmo quando uma condição de contorno de Dirichlet são utilizadas para a velocidade.

Note que o campo de pressões é obtido na malha deslocada, precisando assim do uso de um procedimento de pré e pós correção das Transformadas Rápidas de Fourier.

Os números de onda modificados são definidos na subrotina `waves`.

Note que a implementação do *solver* de Poisson that para a malha com *stretch* é bastante complicado e involve uma inversão de uma ou duas matrizes pentadiagonais. A teoria pode ser encontrada em

Laizet S.& Lamballais E., 2009, **High-order compact schemes for incompressible flows: a simple and efficient method with the quasi-spectral accuracy**, *J. Comp. Phys.*, **228(15)**, 5989-6015.

Por favor, preste atenção particular no Apêndice B.

É recomendado não modificar este arquivo.

12 diferentes combinações pode ser usadas para a equação de Poisson, cobrindo uma ampla variedade de configurações de escoamento: $(0-0-0)$, $(1-0-0)$, $(2-0-0)$, $(0-2-0)$, $(1-1-0)$, $(2-1-0)$, $(1-1-1)$, $(2-2-1)$, $(2-1-2)$, $(1-2-1)$, $(1-1-2)$ e $(2-2-2)$. Para o *solver* do Poisson, o caso `ncl=1` e o caso `ncl=2` são lidados da mesma maneira.

3.11 visu.f90

Arquivo que contem todas as subrotinas com procedimentos de pós-processamento. Várias ferramentas foram criadas para administrar eficientemente a entrada e saída de dados no código. Como foi explicado anteriormente, há um procedimento de reinício no código (arquivo `tools.f90`, subrotina `restart`). O arquivo restart `sauve.dat` é gerado a cada passo de tempo definido em `isave`. Nota-se que é possível recomeçar a simulação com um número diferente de núcleos computacionais. Em termos de capturas de tela em 2D e 3D, várias subrotinas podem ser usadas:

- **2D snapshots (resolução total):**
  subrotina `decomp_2d_write_plane(pencil,var,iplane,n,filename)`:
  `ipencil` é igual 1,2,3 para lápis em X, Y, e Z, respectivmente. `var` é o nome do arranjo 3D. `iplane` é igual a 1,2,3 para salvar um plano em X, Y, Z, respectivamente. `n` corresponde a localização deste plano. `filename` é o nome do arquivo de saída.

  EXEMPLO:
  `call decomp\_2d\_write\_plane(1,ux1,1,112,'ux2d')` vai escrever em `ux2d` um plano 2D (y-z) de um arranjo 3D chamada ux1 (definida no lápis X), para i=112.

- **3D snapshots (resolução total):**
  subrotina `decomp_2d_write_one(nx,ny,nz,ipencil,var,filename)`:
  `ipencil` é igual 1,2,3 para os lápis X, Y e Z respectivamente. `var` é o nome do arranjo 3D. `filename` é o nome do arquivo de saída.

  EXEMPLO:
  `call decomp_2d_write_one(nx,ny,nz,2,uy2,'uy2.dat')`vai escrever em `uy2.dat` o arranjo 3D `uy2` (definida no lápis Y).

- **snapshots em 3D (resolução diminuída):**
  subrotina `decomp_2d_write_one(ipencil,var,filename,icoarse)`:
  `ipencil` é igual a 1,2,3 para os lapis X, Y e Z, respectivamente. `var` é o nome do arranjo 3D. `filename` é o nome do arquivo de saída. `icoarse` é igual a `nstat` ou `nvisu` (eja o arquivo `module_param.f90`). O arranjo `var` é definido em uma malha mais aberta (a cada `nstat` ou `nvisu` nós). Antes de chamar essa subrotina é necessário chamar a subrotina
  `fine_to_coarseV(ipencil,var_full,var_coarseV)` ou
  `fine_to_coarseS(ipencil,var_full,var_coarseS)`.
  O tamanho de `var_coarseV` é `(xszV(1),xszV(2),xszV(3))`.
  O tamanho de `var_coarseS` é `(xszS(1),xszS(2),xszS(3))`.

  EXEMPLO:
  `call fine_to_coarseV(1,uz1,uvisu)` escreverá no arranjo com malha mais aberta `uvisu` os arranjos 3D (resolução totel) `uz1` (definidpono lápis Z).
  E então `call decomp_2d_write_one(1,uvisu,'uz_coarse.dat',2)` vai escrever em `uz_coarse.dat` no arranjo 3D `uvisu` (definida a cada `nvisu` nós de malha, no lápis X).

- **Lápis em X (várias resoluções):**
  É obviamente possível salvar os dados para apenas um pequeno número de lápis em X.

  EXEMPLO:
  Neste exemplo, uma simulação com $n_x \times n_y \times n_z = 2881 \times 360 \times 360$ está rodando em 3600 núcleos computacional um mapeamento 2D $p_{raw} \times p_{col} = 60 \times 60$. Eu quero salvar as três componentes da velocidade a cada passo de tempo para $j = k = 181$ e a cada 8 nós de malha na direção x. Vou simplesmente usar as seguintes linhas:

  ```
  if (nrank==4095) then
  ```

```
if (itime==ifirst) then
write (filename,923) nrank
open (nrank,file=filename,form='unformatted')
endif
write (nrank)(((ux1(i,j,k),i=1,xsize(1),8),j=1,xsize(2),2),k=1,xsize(3),2),&
(((uy1(i,j,k),i=1,xsize(1),8),j=1,xsize(2),2),k=1,xsize(3),2),&
(((uz1(i,j,k),i=1,xsize(1),8),j=1,xsize(2),2),k=1,xsize(3),2)
if (itime==ilast) close (nrank)
endif
```

Para identificar qual núcleo computacional corresponde a $j = k = 181$, é recomendado usar os arranjos $xstart$ e a matriz $xend$. Um procedimento similar pode ser utilizado para salvar dados em lápis em Y e em Z.

- **domínios sub-3D:**
subrotina `decomp_2d_write_subdomain(ipencil,var,imin,imax,jmin,jmax,kmin,kmax,filename)`:
Esta subrotina está disponível somente das últimas versões do código.

  EXEMPLO:
  `call decomp_2d_write_subdomain(1,ux1,851,1600,311,411,311,511,'ux_sub.dat')` irá escrever em `ux_sub.dat` parte do arranjo 3D `ux1` para $i = 851, 1600$, $j = 311, 411$ e $k = 311, 511$