# Inside of FFV-C Solver

**Frontflow / violet Cartesian**

**Ver. 0.7.0**

**Institute of Industrial Science**
**The University of Tokyo**
http://www.iis.u-tokyo.ac.jp/

September 2012

| | | | |
|---|---|---|---|
| First Edition | Version 0.6.0 | 10 Feb. | 2011 |
| | Version 0.7.0 | 3 Mar. | 2012 |
| | Version 0.7.2 | 15 Sep. | 2012 |

# 目次

# 第 1 章

# インストール

事前に必要な MPI 通信ライブラリと V-Sphere のインストール，および基本的な CBC ソルバークラスのインストールについては V-Sphere ユーザガイドを参照. この章では，CBC ソルバークラスの詳細なインストールとコンパイルについて説明する．

## 1.1　CBC SolverClass のインストール

### 1.1.1　インストール

#### 1.1.1.1　標準インストール

sphPrjTool を使ってインストール．あるプラットホームで開発したソースコードを異なるプラットホームでコンパイルする場合には，環境変数や V-Sphere のインストールディレクトリなどの違いにより多くパラメータの再設定が必要になる．煩雑な修正を避けるために，sphPrjTool には reset コマンドが用意されている．reset コマンドは，V-Sphere がインストールされているディレクトリ配下の~/config/sph-cfg.xml の内容[*1]を参照する．このため，reset コマンドを実行する前に V-Sphere のインストールディレクトリを環境変数 SPHEREDIR で指定する．

```
$ export SPHEREDIR=INSTALL_DIR
$ sphPrjTool PRJ_CBC.xml

sphPrjTool> reset localsettings
sphPrjTool> print
sphPrjTool> save
sphPrjTool> quit
```

この作業により，project_local_settings が上書きされるので，以下のコマンドによりソルバーをコンパイルする．

```
$ make allclean
$ make depend
$ make
```

必要に応じて，プロジェクトコンパイル環境設定ファイルを編集する．

- 使用環境に応じた適切なコンパイルオプションを指定する．
- LDFLAGS のの Library の PATH も異なっていたら、適切なパスに変更する．
- libxml2
  libxml2 ライブラリに関連する環境変数については，XML2LIBS と XML2FLAGS を，それぞれ次のコマンドにより得られる出力をそのまま記述する．

  ```
  $ xml2-config --libs
  $ xml2-config --cflags
  ```

## 1.2　各種プラットホームにおける CBC のコンパイル

### 1.2.1　RICC, FOCUS

RICC [1] と FOCUS [2] は，Intel 製 CPU のクラスタで，標準的なインストール手順を実行．

### 1.2.2　BlueGene/L

IBM BlueGene/L でのコンパイルについて説明する．コンパイル環境は，IBM XLFortran, XLC++ コンパイラ，クロスコンパイルである．クロスコンパイラのため，sphPrjTool は使用できないので，他の環境でプロジェクトを作成して

[*1] プラットホーム環境設定情報

持ってくる．

project_local_settings **ファイルの編集**　変更箇所は以下のとおり．[*2]

```
CC=blrts_xlc
CFLAGS=-qarch=440 -qtune=440 -O3
CXX=blrts_xlC
CXXFLAGS=-qarch=440 -qtune=440 -O3 -D_NON_P4_DEVICE_
FC=blrts_xlf
FCFLAGS=-qarch=440 -qtune=440 -O3 -qextname
F90=blrts_xlf90
F90FLAGS=-qarch=440 -qtune=440 -O3 -qextname
LDFLAGS=-L/bgl/BlueLight/ppcfloor/bglsys/lib -L/opt/ibmcmp/xlf/bg/10.1/blrts_lib
LIBS=-lmsglayer.rts -lrts.rts -ldevices.rts -lmass -lmassv -lxlf90
SPH_USR_DEF_LIBS=
SPHEREDIR=/gfs1/user/sphere/Vsphere_1_7_2_lib
SPH_DEVICE=Linux
MPICH_DIR=/bgl/BlueLight/ppcfloor/bglsys
MPICH_CFLAGS=-I/bgl/BlueLight/ppcfloor/bglsys/include
MPICH_LDFLAGS=-L/bgl/BlueLight/ppcfloor/bglsys/lib
MPICH_LIBS=-lmpich.rts
XML2FLAGS=-I/gfs1/user/XML2/include/libxml2
XML2LIBS=-L/gfs1/user/XML2/lib -lxml2
SPHERE_CFLAGS=-DSKL_TIME_MEASURED -D_CATCH_BAD_ALLOC
              -I/gfs1/user/sphere/Vsphere_1_7_2_lib/include
SPHERE_LDFLAGS=-L/gfs1/user/sphere/Vsphere_1_7_2_lib/lib
SPH_PARA_MODULE=MPI
```

### 1.2.3　AMD Opteron

本節では，AMD Opteron 上での sphere コンパイルについて述べる．

sphPrjTool を使用してプロジェクトを作成する．sphPrjTool では, 以下のコマンドで LIBS を指定する．PGI Compiler の場合,

```
sphPrjTool> env LIBS "-lpgf90 -lpgf90_rpm1 -lpgf902 -lpgftnrtl"
```

Intel Compiler の場合，project_local_settings でのコンパイルオプションは”-O3”のみを指定する．

### 1.2.4　QUEST

本節では，理化学研究所の Quest システム[*3]（PFU 製 RG1000×64 台, 1024node, 1CPU/node, 2cores/CPU, 2GB/node, GE）環境でのコンパイルについて示す．Quest システムでは幾種類かの MPI ライブラリが利用可能なので，各ライブラリに合わせてコンパイルを行う．下記に，インストールシェルのサンプルを示す．PRJ_CBC.xml を次のように設定する．

```
<?xml version="1.0"?>
<SphereProject>
```

[*2] SPHEREDIR, SPHERE_CFLAGS, SPHERE_LDFLAGS は,「/gfs1/user/sphere/Vsphere_1_7_2_lib」を指定した sphere ライブラリインストールディレクトリ（INSTALLDIR）に置き換えること．XML2FLAGS, XML2LIBS は,「/gfs1/user/XML2」を指定した libxml2 インストールディレクトリ（–prefix）に置き換えること．

[*3] Quest システムの詳細については，VPN 経由で，http://quest.q.riken.jp

```
  <Param name="prj_name" dtype="STRING" value="PRJ_CBC"/>
  <Elem name="Environment">
    <Param name="CC" dtype="STRING" value="/opt/intel/Compiler/11.0/081/bin/intel64/icc"/>
    <Param name="CFLAGS" dtype="STRING" value="-O3"/>
    <Param name="CXX" dtype="STRING" value="/opt/intel/Compiler/11.0/081/bin/intel64/icpc"/>
    <Param name="CXXFLAGS" dtype="STRING" value="-O3"/>
    <Param name="FC" dtype="STRING" value="/opt/intel/Compiler/11.0/081/bin/intel64/ifort"/>
    <Param name="FCFLAGS" dtype="STRING" value="-O3 -r8"/>
    <Param name="F90" dtype="STRING" value="/opt/intel/Compiler/11.0/081/bin/intel64/ifort"/>
    <Param name="F90FLAGS" dtype="STRING" value="-O3 -r8"/>
    <Param name="LDFLAGS" dtype="STRING" value="-L/opt/intel/Compiler/11.0/081/lib/intel64"/>
    <Param name="LIBS" dtype="STRING" value="-lifport -lifcore"/>
    <Param name="SPH_USR_DEF_LIBS" dtype="STRING" value=""/>
    <Param name="SPHEREDIR" dtype="STRING" value="/gfs1/keno/SPHERE/bin/vsph175_dbl"/>
    <Param name="SPH_DEVICE" dtype="STRING" value="IA64_Linux"/>
    <Param name="MPICH_DIR" dtype="STRING" value="/usr/local/openmpi/intel"/>
    <Param name="MPICH_CFLAGS" dtype="STRING" value="-I/usr/local/openmpi/intel/include"/>
    <Param name="MPICH_LDFLAGS" dtype="STRING" value="-L/usr/local/openmpi/intel/lib"/>
    <Param name="MPICH_LIBS" dtype="STRING" value="-lmpi"/>
    <Param name="XML2FLAGS" dtype="STRING" value="-I/usr/include/libxml2"/>
    <Param name="XML2LIBS" dtype="STRING" value="-lxml2 -lz -lpthread -lm"/>
    <Param name="SPHERE_CFLAGS" dtype="STRING" value="-DSKL_TIME_MEASURED
                -D_CATCH_BAD_ALLOC -I/gfs1/keno/SPHERE/bin/vsph175_dbl/include"/> \
    <Param name="SPHERE_LDFLAGS" dtype="STRING" \
                value="-L/gfs1/keno/SPHERE/bin/vsph175_dbl/lib"/>
    <Param name="SPHERE_LIBS" dtype="STRING" value="-lsphapp -lsphbase -lsphls -lsphfio \
                -lsphdc -lsphcrd -lsphcfg -lsphftt -lsphvcar"/>
    <Param name="REALOPT" dtype="STRING" value="-DREAL_IS_DOUBLE"/>
    <Param name="SPH_EXTERNAL_HEADER_PATH" dtype="STRING" value="../../FB"/>
    <Param name="SPH_EXTERNAL_HEADER_PATH" dtype="STRING" value="../../IP"/>
    <Param name="SPH_PARA_MODULE" dtype="STRING" value="MPI"/>
    <Param name="SPH_PLS_FNAME" dtype="STRING" value="../project_local_settings"/>
  </Elem>
  <Elem name="SolverClass">
    <Elem name="CBC">
    </Elem>
  </Elem>
  <Elem name="NonSolverClass">
    <Elem name="IP">
    </Elem>
    <Elem name="FB">
    </Elem>
  </Elem>
</SphereProject>
```

### 1.2.5　Windows

V-Sphere::CBC Version 1.4.4 と V-Sphere::FlowBase Version 1.8.4 の場合の WindowsXP におけるコンパイル手順を以下に示す.

**sphPrjTool でのリセット**　Windows 上でプロジェクトツールを実行して、”reset localsettings”を行う.

```
$ export SPHEREDIR=INSTALL_DIR
$ sphPrjTool PRJ_CBC.xml
sphPrjTool> reset localsettings
```

```
sphPrjTool> print
sphPrjTool> save
sphPrjTool> quit
```

この作業により，project_local_settings が上書きされ，Makefile.win が生成される．上記作業を行った後に，nmake -f Makefile.win を実行すると，Windows 上でコンパイルが可能になる．

**コンパイル**　カレントディレクトリを SolverClass として，以下のようにコンパイルを実施する．

```
$PRJ_CBC> nmake -f Makefile.win
```

実行モジュール sphere.exe が生成され，PRJ_CBC/bin ディレクトリにコピーされる．

Windows 環境では DOS プロンプトを用いてコンパイルを行う．ただし，DOS プロンプトの環境で以下のパスが設定されている必要がある．DOS プロンプト上で set コマンドを用いて以下が環境変数 PATH に設定されていることを確認のこと．

- libxml2 ライブラリのパス
- zlib ライブラリのパス
- iconv ライブラリのパス

詳細は V-Sphere の Windows マニュアル（18_windows.pdf）を参照．アクセサリメニューにある DOS プロンプトには，Visual Studio や Intel Compiler の環境情報が登録されていない場合がある．その際には，Visual Studio もしくは Intel Compiler に付属している DOS プロンプトを使用すること．

**Intel Compiler 11.x への対応**　表 1.1 の環境変数について修正を行う．

表 1.1　Intel Compiler 11.x に対する変更

| | | 修正内容 |
|---|---|---|
| 1 | 修正前 | `CXX="C:\Program Files\Intel\Compiler\C++\10.1.021\IA32\bin\icl.exe"` |
| | 修正後 | `CXX="C:\Program Files\Intel\Compiler\11.0\066\cpp\bin\ia32\icl.exe"` |
| 2 | 修正前 | `F90="C:\Program Files\Intel\Compiler\Fortran\10.1.021\IA32\bin\ifort.exe"` |
| | 修正後 | `F90="C:\Program Files\Intel\Compiler\11.0\066\fortran\bin\ia32\ifort.exe"` |
| 3 | 修正前 | `LDFLAGS=/LIBPATH:"C:\Program Files\Intel\Compiler\Fortran\10.1.021\IA32\lib"` |
| | 修正後 | `LDFLAGS=/LIBPATH:"C:\Program Files\Intel\Compiler\11.0\066\fortran\lib\ia32"` |
| 4 | 修正前 | `INTELF90_DIR=C:\Program Files\Intel\Compiler\Fortran\10.1.021\IA32` |
| | 修正後 | `INTELF90_DIR=C:\Program Files\Intel\Compiler\11.0\066\fortran` |
| 5 | 修正前 | `INTELCXX_DIR=C:\Program Files\Intel\Compiler\C++\10.1.021\IA32` |
| | 修正後 | `INTELCXX_DIR=C:\Program Files\Intel\Compiler\11.0\066\cpp` |

**Makefile.win の修正**　表 1.2 の環境変数について修正を行う．

表 1.2　Intel Compiler 11.x に対する変更

| | | 修正内容 |
|---|---|---|
| 1 | 修正前 | `AR = "$(INTELCXX_DIR)\bin\xilib.exe"` |
| | 修正後 | `AR = "$(INTELCXX_DIR)\bin\ia32\xilib.exe"` |
| 2 | 修正前 | `INTEL_LINK = "$(INTELCXX_DIR)\bin\xilink.exe"` |
| | 修正後 | `INTEL_LINK = "$(INTELCXX_DIR)\bin\ia32\xilink.exe"` |

なお，Makefile.win ファイルはプロジェクトフォルダ直下，および FB フォルダ，CBC フォルダに存在するので，すべての Makefile.win ファイルに対して上記の修正を行う．

### 1.2.6　Windows の実行モジュール

Windows 版のバイナリパッケージのインストールについて説明する．Windows 版のバイナリパッケージは CD-ROM1 のディスクイメージにより提供する．

```
CD-ROM1
+- CBC
    +- doc
    |   +- CBC_for_win_manual.pdf      CBC インストールマニュアル
    |   +- cbc_UG.pdf                  CBC ユーザガイド
    |   +- Sphere_UG.pdf               V-Sphere ユーザガイド
    +- bin
        +- bin.zip                     CBC 実行モジュール（sphere.exe）
```

#### 1.2.6.1　インストール手順

1. Visual C++ 2005 SP1 再頒布可能パッケージのインストール
   vcredist_x86.exe
2. WindowsInstaller 3.1 Redistributable のインストール
   WindowsInstaller-KB893803-v2-x86.exe
3. .NET Framework Version 3.5 再頒布可能パッケージのインストール
   dotNetFx35setup.exe
4. MPICH2 のインストール
   mpich2-1.2.1p1-win-ia32.msi
5. libxml2 のインストール
   libxml2-2.6.32+.win32
6. iconv のインストール
   iconv-1.9.2.win32
7. zlib のインストール
   zlib-1.2.3.win32
8. 環境変数の設定
   システム環境変数”Path”に libxml2, iconv, zlib の各 bin フォルダを登録する．また，MPICH2 の bin フォルダも登録しておくと便利.
9. ソルバのインストール
   CD-ROM1 の CBC\bin\bin.zip を展開してできるフォルダ”bin”を任意の位置 (例えば c:\Program Files) にフォルダ”CBC”を作成して，”CBC”フォルダは配下に展開した ”bin”フォルダごと移動する．この bin フォルダも環境変数 Path に登録しておくと便利.

# 第 2 章

# 基礎方程式と解析方法の概要

本章では，CBC ソルバークラスが扱う流体の基礎方程式と解法アルゴリズム，離散化について概略を述べる．

## 2.1　支配方程式

流体の密度変化は圧力や温度の変化により生じる．代表的な流速が音速に比べてかなり低い場合，流体の挙動は圧縮性，つまり密度変化による媒質の弾性の影響が小さいと近似でき，非圧縮性流体として扱うことができる．一方，代表流速が音速よりかなり低い場合でも，弾性の影響を考慮しなければならない場合もある．この場合，対象とする流れの現象を考慮して基礎方程式を適切にモデル化して解く．モデル化のレベルによって，下記のようなバリエーションが考えられる．

1) 圧縮性方程式を近似せずに扱う [3]
2) 弾性の影響を質量保存則のみに適用し，運動量に与える密度変化を省略する
3) 温度変化によって生じる密度変化を運動方程式の外力としてモデル化する
4) 完全に非圧縮性流体として扱う

ここでは，2)-4) の定式化を説明する．

### 2.1.1　非圧縮性流体

解析対象とする流れの特徴を以下のように仮定し，支配方程式を記述する．

- 流れの速度が音速に比べて十分に低く，流れの運動に対する圧縮性の影響は小さいと仮定し，流れを非圧縮性として取り扱う．
- 温度場の代表的な温度差スケールが 30 °C 以下で，密度変化が小さいと仮定すると，密度変化が質量保存則へ与える影響は小さい．密度変化が流れの運動に及ぼす影響を Boussinesq 近似によりモデル化する．

支配方程式として，非圧縮性流れに対する質量保存則 **(2.1)**，運動量保存則 **(2.2)**，エネルギー保存則 **(2.3)** を用いる．$\delta$ はクロネッカーのデルタで重力方向 (i=3) のときに浮力が作用する．ここで，プライム [′] は有次元量を表す．物性値など有次元であることが明らかなものにはプライムは付けていない．

$$\frac{\partial u_i'}{\partial x_i'} = 0 \tag{2.1}$$

$$\rho' \frac{\partial u_i'}{\partial t'} + \rho' \frac{\partial}{\partial x_j'} \left\{ \left( u_j' - u_j^{g\prime} \right) u_i' \right\} = - \frac{\partial P'}{\partial x_i'} + \frac{\partial}{\partial x_j'} \left[ \mu \left( \frac{\partial u_i'}{\partial x_j'} + \frac{\partial u_j'}{\partial x_i'} \right) \right] - \rho' g \delta_{i3} \tag{2.2}$$

$$\rho' C_p \left[ \frac{\partial \theta'}{\partial t'} + \frac{\partial}{\partial x_i'} \left\{ \left( u_i' - u_i^{g\prime} \right) \theta' \right\} \right] = \frac{DP'}{Dt'} + \frac{\partial}{\partial x_i'} \left( \lambda \frac{\partial \theta'}{\partial x_i'} \right) + \mu \Phi + Q' \tag{2.3}$$

| | | |
|---|---|---|
| $\rho'$ | $[kg\,/\,m^3]$ | density |
| $P'$ | $[Pa]$ | pressure |
| $C_p$ | $[J\,/\,(kg\,K)]$ | specific heat at constant pressure |
| $\theta'$ | $[K]$ | temperature |
| $\lambda$ | $[W\,/\,(m\,K)]$ | heat conductivity |
| $u_j{}'$ | $[m\,/\,s]$ | velocity components |
| $u_j^{g\prime}$ | $[m\,/\,s]$ | velocity components of a grid point |
| $x_j{}'$ | $[m]$ | coordinate axis |
| $t'$ | $[s]$ | time |
| $\mu$ | $[Pa\,s]$ | viscosity |
| $g$ | $[m\,/\,s^2]$ | gravitational acceleration |
| $\Phi$ | $[1/s^2]$ | dissipation function |
| $Q'$ | $[W\,/\,m^3]$ | heat source |

**式 (2.2)** は形式的に ALE(Arbitrary Lagrangian and Eulerian) で書かれているが，速度 $u_j^{g\prime}$ で移動する格子系での保存則を表現している．格子点を固定 ($u_j^{g\prime}=0$) すれば Euler 表現，流体粒子と一緒に移動 ($u_j^{g\prime}=u_j'$) させれば Lagrangian 表現となる．ここでは，並進や回転などの任意の格子移動速度を与えるために $u_j^{g\prime}$ を利用する．

一様で平衡な状態においては，密度は圧力と温度の関数となる．理想気体の場合には，$\mathcal{R}\,(8.314472\ [J\,/\,(mol\,K)]$ を気体定数 [4][*1]とすると**式 (2.4)** により表される．

$$P' = \rho'\,\mathcal{R}\,\theta' \tag{2.4}$$

平衡状態 0 からのずれが小さく変化が線形であると仮定すると，テーラー展開から

$$\rho' \;=\; \rho_0{}' + \left(\frac{\partial\rho'}{\partial\theta'}\right)_0 (\theta' - \theta_0{}') + \left(\frac{\partial\rho'}{\partial P'}\right)_0 (P' - P_0{}') \tag{2.5}$$

ここで，体積膨張率を次式で定義すると，

$$\beta_0 = -\frac{1}{\rho_0{}'}\left(\frac{\partial\rho'}{\partial\theta'}\right)_0 \tag{2.6}$$

理想気体の場合には，

$$\beta_0 = \frac{1}{\theta_0{}'} \tag{2.7}$$

となる．**(2.5)** の右辺第三項は音速の 2 乗に反比例するので省略でき，最終的に次のようになる．

$$\rho' = \rho_0{}' - \rho_0{}'\beta_0\,(\theta' - \theta_0{}') \tag{2.8}$$

**式 (2.8)** は，基準状態 0 を基にして密度変化を温度変化によって表す Boussinesq 近似モデルである．

[*1] 気体定数の値は CODATA（科学技術データ委員会）の 2002 年の勧告で，それ以前から変更になっている．

さて，重力下にある静止流体では下層にある流体ほど圧力が高い．三次元の z 方向 (i=3) を鉛直方向にとると，Navier-Stokes 方程式 **(式 2.2)** の右辺は，状態 0 のとき，

$$-\frac{\partial P'}{\partial x_i'} - \rho_0' g = -\frac{\partial}{\partial z'}\left(P' + \rho_0' g z'\right) = -\frac{\partial \tilde{p}'}{\partial z'} \tag{2.9}$$

である．重力ポテンシャルの影響を加えた新しい定義の圧力を導入し，支配方程式を表すと，**式 (2.2)** は以下のようになる．

$$\tilde{p}' = P' + \rho_0' g z' \tag{2.10}$$

$$\frac{\partial u_i'}{\partial t'} + \frac{\partial}{\partial x_j'}\left\{\left(u_j' - u_j^{g\prime}\right) u_i'\right\} = -\frac{\partial p'}{\partial x_i'} + \frac{\partial}{\partial x_j'}\left[\nu\left(\frac{\partial u_i'}{\partial x_j'} + \frac{\partial u_j'}{\partial x_i'}\right)\right] - (\theta' - \theta_0')\beta_0 g \delta_{i3} \tag{2.11}$$

また，低マッハ数を仮定すると，散逸関数 $\Phi$ は $M^2$ に比例するので，その寄与は小さいとしてよい．圧力の全微分の項の影響も小さいとすると，**式 (2.3)** は，

$$\frac{\partial \theta'}{\partial t'} + \frac{\partial}{\partial x_i'}\left\{\left(u_i' - u_i^{g\prime}\right)\theta'\right\} = \frac{\partial}{\partial x_i'}\left(\alpha \frac{\partial \theta'}{\partial x_i'}\right) + \frac{Q'}{\rho' C_p} \tag{2.12}$$

ここで $\alpha$ は温度拡散係数で $[m^2/s]$ の単位をもつ．

$$\left.\begin{aligned} \alpha &= \frac{\lambda}{\rho' C_p} \quad [m^2/s] \\ \nu &= \frac{\mu}{\rho'} \quad [m^2 s] \\ p &= \frac{\tilde{p}'}{\rho_0'} \quad [m^2/s^2] \end{aligned}\right\} \tag{2.13}$$

とおいた．温度拡散係数 $\alpha$ が一定，さらに定常の場合には下記のようになる．

$$\frac{\partial \theta'}{\partial t'} + \frac{\partial}{\partial x_i'}\left\{\left(u_i' - u_i^{g\prime}\right)\theta'\right\} = \alpha \frac{\partial}{\partial x_i'}\left(\frac{\partial \theta'}{\partial x_i'}\right) + \frac{Q'}{\rho' C_p} \tag{2.14}$$

$$\frac{\partial}{\partial x_i'}\left\{\left(u_i' - u_i^{g\prime}\right)\theta'\right\} = \alpha \frac{\partial}{\partial x_i'}\left(\frac{\partial \theta'}{\partial x_i'}\right) + \frac{Q'}{\rho' C_p} \tag{2.15}$$

## 2.2　無次元化

代表速度 $u_0'$, 代表長さ $L'$, 代表温度スケール $\Delta\theta'$ と基準温度 $\theta_0'$ で**式 (2.1)**, **(2.11)**, **(2.12)** を無次元化する．

$$\left.\begin{aligned} u &= \frac{u'}{u_0'} \\ x &= \frac{x'}{L'} \\ p &= \frac{p' - p_0'}{\rho' u_0'^2} \\ \theta &= \frac{\theta' - \theta_0'}{\Delta\theta'} \end{aligned}\right\} \tag{2.16}$$

### 2.2.1 強制対流と自然対流

以下の**式 (2.17)–(2.19)** は，単一成分の熱流動を表す．

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{2.17}$$

$$\frac{\partial u_i}{\partial t} + \frac{\partial}{\partial x_j}\left\{\left(u_j - u_j^g\right) u_i\right\} = -\frac{\partial p}{\partial x_i} + \frac{1}{Re}\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + \frac{Gr}{Re^2}\delta_{i3}\theta \tag{2.18}$$

$$\frac{\partial \theta}{\partial t} + \frac{\partial}{\partial x_i}\left\{\left(u_i - u_i^g\right) \theta\right\} = \frac{1}{Pe}\frac{\partial}{\partial x_i}\frac{\partial \theta}{\partial x_i} + \Theta \tag{2.19}$$

$$\left.\begin{aligned} Re &= \frac{u'_0 L'}{\nu} \\ Pr &= \frac{\mu C_p}{\lambda} = \frac{\nu}{\alpha} \\ Gr &= \frac{g\beta\Delta\theta' L'^3}{\nu^2} \\ Ra &= Pr \cdot Gr \\ Pe &= Pr \cdot Re \\ \Theta &= \frac{Q'}{\rho' C_p}\frac{L'}{u'_0 \Delta\theta'} \end{aligned}\right\} \tag{2.20}$$

ここで，

| | | |
|---|---|---|
| $Pr$ | Prandtl 数 | 粘性と熱の拡散率の比 |
| $Re$ | Reynolds 数 | 慣性力と粘性力の比 |
| $Gr$ | Grashof 数 | 浮力と粘性力の比 |
| $Ra$ | Rayleigh 数 | 不安定性のパラメータ |
| $Pe$ | Peclet 数 | 対流と熱伝達のエネルギー輸送の比 |
| $\Theta$ | - | 無次元の温度変化率 |

**式 (2.18)** は強制対流と自然対流を表現している．右辺第三項は自然対流と強制対流の比を表している．つまり，$Gr/Re^2 \gg 1$ の場合には自然対流が支配的で，$Gr/Re^2 \ll 1$ の場合には強制対流が支配的となる．$Gr = 0$ つまり温度差が無い場合には純強制対流である．一方，$Gr/Re^2 \to \infty$ の場合には純自然対流で，流れは浮力によって駆動されるため代表速度が自明ではない．また，$Gr > 10^9$ となるような流れは非定常性が強くなる．

### 2.2.2 自然対流のスケールアナリシス

純自然対流の場合の代表流速をスケールアナリシスから推測する [5]．自然対流の場合には流れを駆動する支配要因が熱拡散であり，対流の影響は小さいと考えられるので，温度境界層と粘性境界層の厚さが同程度と見積もられる．ところで，$Pr$ 数が大きな流体の場合には粘性が支配的なので，**式 (2.18)** において，粘性項と浮力項のオーダーが等しい．

一方，低 $Pr$ 数流体の場合には慣性力が支配的となるので，慣性項と浮力項のオーダーが等しくなる．これらをまとめると，

$$\left.\begin{aligned} \frac{Gr}{Re^2}\delta_{i3}\theta &\sim \frac{1}{Re}\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right) \quad (Pr \gg 1) \\ \frac{Gr}{Re^2}\delta_{i3}\theta &\sim \frac{\partial}{\partial x_j}\left(u_i u_j\right) \qquad\qquad\quad (Pr \ll 1) \end{aligned}\right\} \tag{2.21}$$

$Pr$ 数が小さい場合は**式 (2.21)** から，

$$u_0' = \sqrt{g\beta\Delta\theta' L'} \tag{2.22}$$

と見積もることができる．したがって，自然対流の場合の代表速度は**式 (2.22)** の関係を用いて見積もり，代表速度パラメータとして与える．自然対流と強制対流が共存する共存対流の場合には，各々の代表スケールの平均値や大きい方の値を代表速度とする．

### 2.2.3　熱流動計算の支配方程式とパラメータ

各流動現象の支配方程式に対する入力パラメータ (Kind_Of_Solver, Buoyancy) と支配方程式の関係を**表 2.1** にまとめる[*2]．パラメータは全て有次元で入力[*3] するが，標準出力には対応する無次元数も出力する．

表 2.1　支配方程式と熱対流計算のパラメータ指定の関係

| 支配方程式 | Kind_Of_Solver | Buoyancy |
|---|---|---|
| 純強制対流 | Flow_Only | - |
| 熱対流（浮力なし） | Thermal_Flow | No_Buoyancy |
| 熱対流（浮力あり） | Thermal_Flow | Boussinesq |
| 自然対流 | Thermal_Flow_Natural | Boussinesq |
| 固体熱伝導 | Solid_Conduction | - |

#### 2.2.3.1　純強制対流

**式 (2.18)** においては $Gr = 0$ なので $Re$ が支配パラメータとなる．無次元化のスケーリングは，$u_0'$, $L'$, $\nu$, $\alpha\,(=\lambda/\rho' C_p)$ を与える．

#### 2.2.3.2　熱対流

**■浮力の効果を考慮しない場合　式 (2.18)** において，純強制対流と同じく $Gr = 0$ である．**式 (2.19)** では $Pe$ が支配パラメータとなる．無次元化のスケーリングは，$u_0'$, $L'$, $\nu$, $\alpha\,(=\lambda/\rho' C_p)$ を与える．

**■浮力の効果を考慮する場合　式 (2.18)** では $Gr$, $Re$ が，**式 (2.19)** では $Pe$ が支配パラメータとなる．無次元化のスケーリングは，$u_0'$, $L'$, $\Delta\theta'$, $\beta$, $g$, $\nu$, $\alpha$, $Pr$ を与える．

[*2] 共役熱移動と固体熱伝導についての詳細は，6 章で説明する．
[*3] 純強制対流の場合のみ無次元パラメータにも対応している．

#### 2.2.3.3 純自然対流

浮力の効果を考慮した熱対流と同じである．ただし，$u_0'$ は自明でないので，**式 (2.22)** により適切に見積る点に注意する．

#### 2.2.3.4 固体熱伝導

**式 (2.19)** の形式で $Pe$ が支配パラメータとなる．ただし，対流項の寄与はゼロである．無次元化のスケーリングは，$L'$, $\Delta\theta'$, $\alpha$, を与え，$u_0'$ には**式 (6.7)** を用いる．

#### 2.2.3.5 共役熱移動

共役熱移動は，固体中の熱移動と流体中の熱流動を同時に扱うので，必然的に多媒質の熱移動問題となる．熱流動は浮力効果を考慮している．

## 2.3 解法アルゴリズム

この節では前節の支配方程式に対して，非圧縮性流体の解法に使われる分離解法を適用し，有限体積法で離散化する．

### 2.3.1 Fractional Step 法

非圧縮性の Navier-Stokes 方程式 **(2.18)** の解法として，Fractional step 法を用いる．これは，任意のベクトル場が非回転場と湧き出し無しの直交するベクトル場に分解できる性質を利用して，二つのベクトルの和をとることにより解を求める分離解法である．

離散式のコーディングポリシーとして，各セル単位で計算を進めていく．保存的な支配方程式を解くのでセル界面の流束ベースの評価が素直で演算量も少なくなるが，コロケートでは固体面や境界面の処理を考える上でセル単位毎の方が計算処理がしやすい．

#### 2.3.1.1 Euler Explicit

一次精度の時間進行法である．

■Navier-Stokes equations

**式 (2.18)** の対流項と粘性項をそれぞれ $C_i$, $D_i$，浮力項を外力 $f_i$ で表すと，

$$\left.\begin{aligned} \frac{\partial u_i}{\partial t} + C_i &= -\frac{\partial p}{\partial x_i} + D_i + f_i \\ C_i &= \frac{\partial}{\partial x_j}\left\{\left(u_j - u_j^g\right) u_i\right\} \\ D_i &= \frac{1}{Re}\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \\ f_i &= \frac{Gr}{Re^2}\delta_{i3}\theta \end{aligned}\right\} \tag{2.23}$$

疑似ベクトルの予測式は，

$$u_i^* = u_i^n + \Delta t\left(D_i^n - C_i^n + f_i^n\right) \tag{2.24}$$

連続の式による拘束条件から，圧力の Poisson 方程式は,

$$\frac{\partial}{\partial x_i}\frac{\partial p}{\partial x_i}^{n+1} = \frac{1}{\Delta t}\frac{\partial \bar{u}_i}{\partial x_i}^{*} \tag{2.25}$$

圧力ポテンシャルによるセルセンターとスタガード位置の速度ベクトルの修正式は,

$$u_i^{n+1} = u_i^{*} - \Delta t \frac{\partial p}{\partial x_i}^{n+1} \tag{2.26}$$

$$u_{i,\,face}^{n+1} = \bar{u}_{i,\,face}^{*} - \Delta t \frac{\partial p}{\partial x_i}^{n+1} \tag{2.27}$$

■Thermal transport equation

**式 (2.19)** の移流項と拡散項をそれぞれ $Cs_i$, $Ds_i$ で表すと,

$$\left.\begin{aligned} \frac{\partial \theta}{\partial t} + Cs_i &= Ds_i + \Theta \\ Cs_i &= \frac{\partial}{\partial x_i}\left\{\left(u_i - u_i^g\right)\theta\right\} \\ Ds_i &= \frac{1}{Pe}\frac{\partial}{\partial x_i}\frac{\partial \theta}{\partial x_i} \end{aligned}\right\} \tag{2.28}$$

$$\theta^{n+1} = \theta^n + \Delta t\left(Ds_i^n - Cs_i^n + \Theta^n\right) \tag{2.29}$$

#### 2.3.1.2 Adams-Bashforth

二次精度ではあるが，安定条件が厳しい.

■Navier-Stokes equations

$$u_i^{*} = u_i^n + \Delta t\left[\frac{1}{2}\left\{3\left(D_i^n - C_i^n\right) - \left(D_i^{n-1} - C_i^{n-1}\right)\right\} + \frac{1}{2}\left(3f_i^n - f_i^{n-1}\right)\right] \tag{2.30}$$

■Thermal transport equation

$$\theta^{n+1} = \theta^n + \Delta t\left[\frac{1}{2}\left\{3\left(Ds_i^n - Cs_i^n\right) - \left(Ds_i^{n-1} - Cs_i^{n-1}\right)\right\} + \frac{1}{2}\left(3\Theta^n - \Theta^{n-1}\right)\right] \tag{2.31}$$

#### 2.3.1.3 Adams-Bashforth + Crank-Nicolson

拡散項に由来する安定条件による時間積分幅の制限を緩和するため，陰解法を導入する.

■Navier-Stokes equations

$$u_i^{*} = u_i^n + \Delta t\left[-\frac{1}{2}\left(3C_i^n - C_i^{n-1}\right) + \frac{1}{2}\left(D_i^n + D_i^{*}\right) + \frac{1}{2}\left(3f_i^n - f_i^{n-1}\right)\right] \tag{2.32}$$

実装は,

$$\left.\begin{aligned}
\bar{u}_i &= u_i^n + \Delta t\left[-\frac{1}{2}\left(3C_i^n - C_i^{n-1}\right) + \frac{1}{2}D_i^n + \frac{1}{2}\left(3f_i^n - f_i^{n-1}\right)\right] \\
u_i^* &= \bar{u}_i + \frac{\Delta t}{2}\bar{D} \\
u_{i,j,k}^* &= \bar{u}_{i,j,k} + \frac{\Delta t}{2\,Re\,h^2}\left(\sum_l u_l^* - 6\,u_{i,j,k}^*\right) \\
\left(1 + \frac{3\,\Delta t}{Re\,h^2}\right)u_{i,j,k}^* &= \bar{u}_{i,j,k} + \frac{\Delta t}{2\,Re\,h^2}\sum_l u_l^*
\end{aligned}\right\} \tag{2.33}$$

■Thermal transport equation

$$\theta^{n+1} = \theta^n + \Delta t\left[-\frac{1}{2}\left(3Cs_i^n - Cs_i^{n-1}\right) + \frac{1}{2}\left(Ds_i^n + Ds_i^{n+1}\right) + \frac{1}{2}\left(3\Theta^n - \Theta^{n-1}\right)\right] \tag{2.34}$$

実装は,

$$\left.\begin{aligned}
\bar{\theta} &= \theta^n + \Delta t\left[-\frac{1}{2}\left(3Cs_i^n - Cs_i^{n-1}\right) + \frac{1}{2}Ds_i^n + \frac{1}{2}\left(3\Theta^n - \Theta^{n-1}\right)\right] \\
\theta^{n+1} &= \bar{\theta} + \frac{\Delta t}{2}Ds_i^{n+1} \\
\theta_{i,j,k}^{n+1} &= \bar{\theta}_{i,j,k} + \frac{\Delta t}{2\,Pe\,h^2}\left(\sum_l \theta_l^{n+1} - 6\,\theta_{i,j,k}^{n+1}\right) \\
\left(1 + \frac{3\,\Delta t}{Pe\,h^2}\right)\theta_{i,j,k}^{n+1} &= \bar{\theta}_{i,j,k} + \frac{\Delta t}{2\,Pe\,h^2}\sum_l \theta_l^{n+1}
\end{aligned}\right\} \tag{2.35}$$

#### 2.3.1.4　Runge-Kutta + Crank-Nicolson

二次精度の時間進行法，対流項には 2 段階 Runge-Kutta 法，拡散項には Crank-Nicolson 法を用いる．

■1st step : Predictor

積分幅を $\Delta t/2$ にとり Euler 陽解法で時間積分し，n+1/2 タイムレベルでの予測値を得る．

Navier-Stokes equations

$$u_i^{*,n+1/2} = u_i^n + \frac{\Delta t}{2}\left(D_i^n - C_i^n + f_i^n\right) \tag{2.36}$$

$n + 1/2$ タイムレベルの圧力 Poisson 式

$$\frac{\partial}{\partial x_i}\frac{\partial p}{\partial x_i}^{n+1/2} = \frac{2}{\Delta t}\frac{\partial u_i}{\partial x_i}^{*,n+1/2} \tag{2.37}$$

$n + 1/2$ タイムレベルの圧力ポテンシャルによる速度の修正式

$$u_i^{n+1/2} = u_i^{*,n+1/2} - \frac{\Delta t}{2}\frac{\partial p}{\partial x_i}^{n+1/2} \tag{2.38}$$

Thermal transport equation

$$\theta^{n+1/2} = \theta^n + \frac{\Delta t}{2}\left(Ds_i^n - Cs_i^n + \Theta^n\right) \tag{2.39}$$

■2nd step : Corrector

Navier-Stokes equations

$u^{*,n+1}$ について反復的に解く.

$$u_i^{*,n+1} = u_i^n + \Delta t \left\{ \frac{1}{2} \left( D_i^n + D_i^{*,n+1} \right) - C_i^{n+1/2} + f_i^{n+1/2} \right\} \tag{2.40}$$

$$\frac{\partial}{\partial x_i} \frac{\partial p}{\partial x_i}^{n+1} = \frac{1}{\Delta t} \frac{\partial u_i}{\partial x_i}^{*,n+1} \tag{2.41}$$

$$u_i^{n+1} = u_i^{*,n+1} - \Delta t \frac{\partial p}{\partial x_i}^{n+1} \tag{2.42}$$

Thermal transport equation

拡散項に Crank-Nicolson 法を用いると,

$$\theta^{n+1} = \theta^n + \Delta t \left\{ \frac{1}{2} \left( Ds_i^n + Ds_i^{n+1} \right) - Cs_i^{n+1/2} + \Theta^{n+1/2} \right\} \tag{2.43}$$

一方，拡散項にも Runge-Kutta スキームを用いる場合には，次のようになる.

$$\theta^{n+1} = \theta^n + \Delta t \left( Ds_i^{n+1/2} - Cs_i^{n+1/2} + \Theta^{n+1/2} \right) \tag{2.44}$$

## 2.4　乱流解析

現在の計算機リソースでは，Reynolds 数が $10^5 \sim 10^6$ オーダの高 Reynolds 数の乱流を直接計算することはできないため，乱流現象を表現する何らかの数学モデルの導入が必要になる．

### 2.4.1　LES と RANS

主な乱流モデルは LES(Large Eddy Simulation) と Reynolds 平均モデル (RANS; Reynolds Averaged Navier-Stokes simulation) に大別される．

LES は計算格子よりも大きなスケールの渦は直接計算し，格子幅よりも小さいスケール (SGS; Sub-Grid Scale) の渦をモデル化する手法である．一般に低周波の大きな渦は 流れ場によって異なるが，高周波の小さな渦は流れ場の形態によらず普遍性をもち，高周波の小さな渦は等方的でエネルギーを散逸する役割を担っているとされている．このような考えに基づいて，LES は普遍性のある小さな渦の影響だけをモデル化し，流れ場の形態の影響を強く受ける大きな渦の影響はモデル化せず直接解く．しかし，この大きな渦も大小さまざまなスケールの渦が相互作用し合って形成されているため，大きな渦の計算といえども十分に細かい計算格子が必要である [6].

一方，RANS は時間平均化された Navier-Stokes 方程式を解く手法であり，時間平均的な流れ場や乱流成分の定常的な統計量を得るのに適した手法である．LES が大きな渦の影響をモデル化せずに直接解いているのに対して，RANS モデルは大きい渦から小さい渦まで全てのスケールの渦の影響をモデル化している．このため LES ほど細かい格子は要求されないので，LES に比べれば計算負荷は小さいため，計算負荷の面から言えば産業利用に向いた解析手法であり，市販の CFD コードには必ずと言って良いほど RANS 系の乱流モデルが実装されている．しかし，この渦のモデル化には経験的あるいは直観的な関係式や基礎的な実験データに基づいて整理された定数群が使用されることが多く，普遍的に使用できる乱流モデルは今のところ開発されていない．また，数多くの乱流モデルが存在するため乱流モデルを適切に選択し適用することは難しく，あるモデルが合わない場合，他のモデルを試すといった試行錯誤が行われる場合も多い．

### 2.4.2　LES 乱流モデル

LES では格子サイズ以下の小さな渦の影響を表す SGS(Sub-Grid Scale) モデルを導入する．乱流中ではエネルギーカスケードといわれる現象が起きており，流れ場と同程度な大きさの渦から徐々に小さい渦へと運動エネルギーが受け渡され，最終的には最小の渦のスケールで運動エネルギーが熱に変換されている (局所的にはこの逆もあり得る)．SGS 渦粘性モデルは，実効的な粘性を大きくすることによって，最終的に熱に変換されるべき量の運動エネルギーを格子スケールの渦の運動で散逸させる．

標準 Smagorinsky モデルは GS(Grid Scale) と SGS 間のエネルギーの輸送は常に散逸的であるため，乱流場に局所的に存在する SGS から GS へのエネルギーの逆輸送である Backward cascade を再現する仕組みを持たない．このため局所的なエネルギー散逸の再現性には欠けるものの，唯一のモデル定数である Smagorinsky 定数 ($Cs$) が適切であれば，エネルギー散逸の総量に関しては妥当な値を予測することが知られている．また，モデル式がシンプルで計算安定性もよいことから近年，工学的な問題にも広く利用されている．$Cs$ には理論値 (Cs=0.173) が存在するが，いくつかの計算例をみるとより低い値に修正することで実験結果とよく合うことが報告されており（例えば [7]），これが普遍的な定数でないことがわかる．

そこで，Germano ら [8] が提案した DSM(Dynamic Smagorinsky Model) および，その改良モデル (例えば [9]) は，$Cs$ を流れ場の状態から自動的に決定し，エネルギーの Backward cascade を再現することも可能である．このため，DSM は各種の流れ場への LES の適用を促進できると期待され，今なお精力的に研究が行われている．しかし，標準 Smagorinsky モデルには無いフィルタリングや多くの計算が必要となるので，標準 Smagorinsky モデルに比べて計算負荷は大きい．また，$Cs$ が負の値をとることは数値計算には不安定に作用するので，安定化のために何らかの工夫が必要になる．

以上，LES における代表的な 2 種類のモデルについて述べたが，標準 Smagorinsky モデルの安定性と計算負荷が少ない点は魅力的である．また唯一の定数値 $Cs$ についても，過去の研究結果を参考に決めることは難しくない．

### 2.4.3　Smagorinsky モデル

標準 Smagorinsky モデルの離散化手法については，既に多くの研究・文献 (例えば [7, 8]) が存在する．このため，ここでは解法の概略をのみを述べる．LES の基礎方程式には，**式 (2.46)**，**式 (2.45)** に示すフィルタリング操作を施した GS での Navier-Stokes 方程式と流体の連続の式を用いる [6, 10].

$$\frac{\partial \overline{u}_i}{\partial x_i} = 0 \tag{2.45}$$

$$\frac{\partial \overline{u}_i}{\partial t} + \overline{u}_j \frac{\partial \overline{u}_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \frac{\partial}{\partial x_i}\left(-\tau_{ij} + 2\nu \overline{D}_{ij}\right) \tag{2.46}$$

$\overline{D}_{ij}$ はひずみ速度テンソルの GS 成分で，

$$\overline{D}_{ij} = \left(\frac{\partial \overline{u}_i}{\partial x_j} + \frac{\partial \overline{u}_j}{\partial x_i}\right) \tag{2.47}$$

ここで $\tau_{ij}$ は GS で粗視化した場合の残余の応力で，$\overline{p}$ および $\overline{u}_i$ はフィルタ化を施した GS の圧力と速度成分を表している．SGS 応力 $\tau_{ij}$ は渦粘性近似を用いてモデル化される．

$$\tau_{ij} = -2\nu_e \overline{D}_{ij} \tag{2.48}$$

Smagorinsky モデルは SGS の乱流エネルギーの収支において生成と散逸が局所平衡の状態にあると仮定しており，$\nu_e$[*4]を以下のようにモデル化する．

$$\nu_e = \left(Cs\overline{\Delta}\right)^2 \left|\overline{D}\right| \tag{2.49}$$

ここで $Cs$, $\Delta_i$ は，それぞれ Smagorinsky 定数および $i$ 方向の格子幅を表している．乱流統計理論からコルモゴロフのスペクトルを仮定し，$Cs = 0.173$ の無次元定数を得た．

フィルタ代表長さは，各軸方向の格子幅の平均値とする．

$$\overline{\Delta} = (\Delta_1\Delta_2\Delta_3)^{\frac{1}{3}} \tag{2.50}$$

ひずみ速度テンソルの大きさは，

$$\left|\overline{D}\right| = \sqrt{2\overline{D}_{ij}\,\overline{D}_{ij}} \tag{2.51}$$

$$\left|\overline{D}\right|^2 = 2\left(D_{xx}^2 + D_{yy}^2 + D_{zz}^2\right) + 4\left(\left[\overline{D}_{xy}^{xy}\right]^2 + \left[\overline{D}_{yz}^{yz}\right]^2 + \left[\overline{D}_{zx}^{zx}\right]^2\right) \tag{2.52}$$

Smagorinsky モデルでは，GS 成分に速度勾配があれば必ず $\left|\overline{D}\right| > 0$ となるので，**式 (2.49)** より SGS 渦粘性係数は正の値となり，そこでは散逸性を示す．例えば，層流域で速度勾配があるような部分では $\tau_{ij} = 0$ となるはずであるが，そ

[*4] 渦粘性モデルでは，係数は常に正（散逸）であり，エネルギーの逆カスケードは表現できない．

うならない．したがって，遷移問題や再層流化などの現象を含む問題に対しては，Smagorinsky モデルの使用を層流場と乱流場で切り替える必要がある．

一方，固体壁面近傍でもエネルギー収支は生成と散逸が釣り合う局所平衡にはなっていない．したがって，固体境界付近ですべりなし条件を与えて壁近傍も解析する場合には，壁面で $\tau_{ij} = 0$ となるように壁近傍で減衰をかける必要がある．

### 2.4.4　Smagorinsky モデルにおける壁面境界条件

壁面付近の取り扱いについて，(1) 壁法則を用いて格子点を節約する方法と，(2) 十分な格子点を用いてすべりなし条件を適用する方法を説明する．

#### 2.4.4.1　壁関数

壁面近くの流れは，$Re$ 数によって表される平均量とは無関係と推定され，壁近傍で重要な役割を果たす密度 $\rho'$, 動粘性係数 $\nu$, 壁面剪断応力 $\tau'_w$, 壁からの距離 $y'$ によって支配されると考えられている．プライム $'$ は有次元変数を示す．

ここで，固体壁面から第一番目の格子点を $y'_p$ を壁から離れた位置 ($y^+_p = 30 \sim 200$) にとり，その点の速度 $u'_p$ を考える．この領域での速度の代表スケールは，次式の壁面に沿う方向の摩擦速度 (frinction velocity) $u'_\tau$ で表される．

$$u'_\tau = \sqrt{\tau'_w/\rho'} \qquad (m/s) \tag{2.53}$$

ここで無次元長さ $y^+$ を壁座標 (wall coordinate) として定義する．

$$y^+ = \frac{u'_\tau y'}{\nu} \tag{2.54}$$

同様に無次元の速度は,

$$u^+ = \frac{u'}{u'_\tau} \tag{2.55}$$

壁座標と摩擦速度を用いると，壁面近傍の速度プロファイルは次式のように表せ，$Re$ 数に無関係な $y^+$ のみの関数となる (Prandtl の壁法則)．$y^+_p$ は壁から第一点目の格子点である[*5]．

$$u^+_p = \frac{1}{\kappa}\ln(y^+_p) + C \tag{2.56}$$

プロファイルは対数分布則となり，$\kappa$, $C$ は平滑面では，それぞれカルマン定数 (0.4)，普遍定数 (5.5) である．常用対数表示では,

$$u^+_p = 5.75\log_{10}(y^+_p) + 5.5 \tag{2.57}$$

壁関数を使えば，壁面から第一格子点までの間は積分する必要がなく，対数則 **式 (2.56)** を使えば良い．

#### 2.4.4.2　壁法則の適用

**■発達した流れの場合**　チャネル流や円管内の発達流の場合，平均圧力勾配 $\partial p'/\partial x'_i$ と壁面摩擦力 $\tau'_w$ が釣り合うので，**式 (2.53)** から摩擦速度がわかる．第一格子点が対数則の範囲内であることが確認できれば，**式 (2.56)** を利用できる．

[*5] 粘性底層は $0 < y^+ < 4$，バッファー域は $4 < y^+ < 30 \sim 70$，乱流域は $30 \sim 100 < y^+$

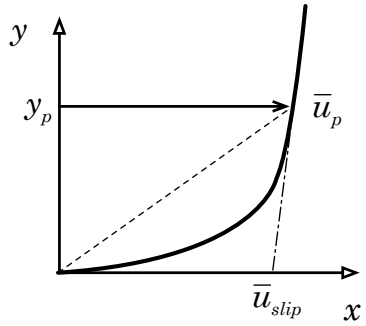



図 2.1　壁面近傍の無次元速度プロファイル（ [6] から転写）

**■流れの状況に応じて壁面摩擦が決まる場合**　一般的には，流れの様子により局所的な摩擦速度の大きさは異なるため，反復的に求める．ある時点での $y'_p$ における $u'_p$ を与えて，関数として次の形を満たす $u'_\tau$ をニュートン反復により求める．

$$F(u'_\tau) \equiv \frac{u'_p}{u'_\tau} - \frac{1}{\kappa}\ln\left(\frac{u'_\tau y'_p}{\nu}\right) - C = 0 \tag{2.58}$$

m を反復回数とすると，

$$u'^{m+1}_\tau = u'^{m}_\tau - \frac{F\left(u'^{m}_\tau\right)}{F^{(1)}\left(u'^{m}_\tau\right)} \tag{2.59}$$

対数則の場合には $F$ の一階導関数は，

$$F^{(1)}(u'_\tau) = -\left(\frac{u'_p}{u'_\tau} + \frac{1}{\kappa}\right)\frac{1}{u'_\tau} \tag{2.60}$$

**式 (2.58)~(2.60)** から反復式を表すと，

$$u'^{m+1}_\tau = u'^{m}_\tau + \frac{\left\{\dfrac{u'_p}{u'^{m}_\tau} - \dfrac{1}{\kappa}\ln\left(\dfrac{y'_p u'^{m}_\tau}{\nu}\right) - C\right\} u'^{m}_\tau}{\dfrac{u'_p}{u'^{m}_\tau} + \dfrac{1}{\kappa}} \tag{2.61}$$

摩擦速度を無次元で表し $u_\tau = u'_\tau/u'_0$ とすると，

$$u^{m+1}_\tau = u^{m}_\tau + \frac{\left\{u^{+m}_p - \dfrac{1}{\kappa}\ln\left(y^{+m}_p\right) - C\right\} u^{m}_\tau}{u^{+m}_p + \dfrac{1}{\kappa}} \tag{2.62}$$

上記の手順により求めた $u'_\tau$ に対して，$y'_p$ が対数則の範囲にあるかどうかを判定し，壁法則を適用する．

**■摩擦速度求解のアルゴリズム**　与えられたセル近傍の速度から摩擦速度を求める手順は以下のようになる．

1. 反復の初期値を計算する．

$$
\begin{aligned}
\tau_w^0 &= \frac{\tau_w'}{\rho' {u_0'}^2} = \frac{1}{Re}\frac{\partial u}{\partial y} \\
u_\tau^0 &= \frac{u_\tau'^0}{u_0'} = \frac{1}{u_0'}\sqrt{\frac{\tau_w'}{\rho'}} = \sqrt{\frac{1}{Re}\frac{\partial u}{\partial y}} \\
y_p^{+\,0} &= \frac{u_\tau^0 u_0' y_p'}{\nu} = u_\tau^0 y_p' \frac{Re}{L'}
\end{aligned}
\tag{2.63}
$$

ここで，$L'$ は代表長さ，$y_p' = h'/2$（バイナリボクセルの場合は，壁面から半セルの距離）である．速度勾配は次の 2 つの候補がある．

（a）壁面から半セルの距離にある速度定義点の速度の大きさで評価

$$
\frac{\partial u}{\partial y} = \frac{\Delta u}{\Delta y} = \frac{|u|}{h/2} \tag{2.64}
$$

（b）**図 2.1** の $y_p$ の位置での速度勾配を壁関数 **(2.57)** から求める

$$
\frac{\partial u}{\partial y} = \frac{5.75}{y_p^+ \ln 10} \tag{2.65}
$$

2. Newton 反復（常用対数を用いて）

$$
u_\tau^{m+1} = u_\tau^m + \frac{\left\{u_p^{+\,m} - 5.75\log_{10} y_p^{+m} - 5.5\right\} u_\tau^m}{u_p^{+\,m} + \dfrac{1}{\kappa}} \tag{2.66}
$$

3. 収束判定次式により収束判定を行う．

$$
\frac{|u_\tau^{m+1} - u_\tau^m|}{|u_\tau^m|} < \epsilon_2 \quad (= 10^{-3}) \tag{2.67}
$$

ここで，$u_\tau^{m+1} < \epsilon_2$ の場合には，安定性のために $u_\tau^{m+1} = \epsilon_2$ とする．

4. 更新

$$
\begin{aligned}
u_\tau^m &= u_\tau^{m+1} \\
y_p^{+\,m} &= u_\tau^{m+1} y_p' \frac{Re}{L'}
\end{aligned}
\tag{2.68}
$$

5. 上記を収束するまで繰り返す．反復上限を 10 回程度で抑えておく．

#### 2.4.4.3　減衰関数を用いたすべりなし条件

壁近傍では GS で速度勾配があるため，SGS 応力が生じる．しかし，壁面上では $\tau_{ij} = 0$ なので，これを打ち消す必要がある．Smagorinsky モデルでは減衰関数 $f_s$ を導入し，渦粘性係数を補正する．

$$
\nu_e = \left(Cs\, f_s\, \overline{\Delta}\right)^2 \left|\overline{D}\right| \tag{2.69}
$$

減衰関数としては van Driest 関数を用いる．

$$
f_s = 1 - \exp\left(-\frac{y^+}{A^+}\right) \tag{2.70}
$$

$A^+$ は無次元の定数で約 25 が用いられる．

すべりなし条件を用いる場合には，粘性底層からバッファ層にかけて，格子点が数点必要である．この場合，粘性底層内では乱流応力は無視できるので，壁面摩擦応力 $\tau'_w$ は次のように決められる．

$$\tau'_w = \mu \left| \frac{\partial \overline{u'}}{\partial y'} \right|_{wall} \tag{2.71}$$

### 2.4.5　壁法則から求めた壁面摩擦応力による粘性応力の置換

壁面に接する流体セルの固体セルへの接平面上の壁面摩擦応力 $\tau'_w = \mu(\partial u'/\partial y')$ は壁法則 **(2.53)** から求められる摩擦速度から $\tau'_w = \rho' u_\tau'^2$ として計算される．この壁面摩擦応力を粘性項の離散式に代入し境界条件とする．たとえば，z 軸のマイナス方向に壁面がある場合，無次元形式で，

$$\frac{1}{Re}\frac{\partial}{\partial z}\left(\frac{\partial u}{\partial z}\right) = \frac{1}{h}\left(\frac{1}{Re}\frac{\partial u}{\partial z}_t - \tau_{w,b}\right) \tag{2.72}$$

計算された摩擦速度から壁面摩擦応力を計算し，**式 (2.72)** により境界条件を与える．固体セルに隣接する流体セルにおいて摩擦速度が求まると，固体に接する面上の応力は次式により計算される．

$$\tau_w = u_\tau^2 \tag{2.73}$$

しかしながら，複数の固体面に接するセルでは各方向同じ大きさの壁面摩擦応力となるし，軸方向に沿わない速度ベクトルの場合には過大評価になる可能性がある．そこで，速度ベクトルの方向成分を計算し，各方向へ射影する．つまり，**式 (2.73)** に用いる $u_\tau$ を再定義する．

$$\tau_{w,i} = \left(u_\tau \frac{u_i}{|u|}\right)^2 \tag{2.74}$$

# 第 3 章

# EBC スキームと境界条件処理

この章では，直交 Euler 格子のコロケート変数配置の場合に，任意位置の境界条件を効率よく処理する方法として，スキーム中に境界条件を実装方法について述べる．

## 3.1　Embedded Boundary Condition Scheme

Embedded Boundary Condition Scheme は，直交格子で任意形状を扱うために境界条件をスキーム中に埋め込む方法の総称とする．物体の形状近似の程度によってその実装は異なる．最も簡単な物体表現としては，物体を 0 と 1（マスク情報）で表現する Binary 近似，つまりレゴブロックのような物体形状がある．この場合には，マスク情報を使って必要な評価点での流束などを計算できるようにスキームを実装する．形状近似度を改善する場合，格子に沿わない方向の物体形状の影響を取り込めるように，スキームを構築する方法がある．セル内を平面で近似する方法や曲面近似など様々なバリエーションが考えられる．古典的なカットセルはこの範囲になる．

本章では，バイナリ近似と面近似の実装について述べる．

## 3.2　Binary 近似の解法

計算領域における任意の位置における境界条件をバイナリ近似の基に統一的に処理するために，**式 (3.1)** に示すフラグ $\phi$ を用いた線形結合により**式 (3.2)** のように離散式を表現する．

$$\phi = \begin{cases} 0 & \text{Non} - \text{fluid cell} \\ 1 & \text{Fluid cell} \end{cases} \tag{3.1}$$

$$p = p\phi + (1-\phi)\, p^{BC} \tag{3.2}$$

上式は，$\phi = 1$ のとき本来の値 $p$ になり，$\phi = 0$ のときに指定する境界条件の値 $p^{BC}$ になる．基本的には if 文でも記述できるが，パイプラインの効率的な動作を期待した実装を行う．

### 3.2.1　圧力の Poisson 方程式

非圧縮コードのホットスポットの一つは圧力の Poisson 方程式の反復部分であるため，この反復部分を効率よく計算したい．そのためには，収束性の高い反復解法の選択，効率的な反復法の実装が必要になる．ここでは後者の実装について，メモリと実行性能の点を検討する．

近年の計算機アーキテクチャは，階層的なメモリ構造，パイプライン演算器，キャッシュ構造などのハードウェア構成により，SIMD 命令の有効利用が高性能計算に必要になる．高性能計算に必要なことは，プロセッサへデータを連続的に供給し，かつ取り出すことである．そのためにはメモリからプロセッサへのデータ転送能力が重要な鍵となる．反復解法ではメモリからのデータ供給と演算数の比 (Byte/Flop 値) が実効性能に大きく影響する．特に，構造格子上の差分法では B/F=2 ~ 5 程度となるだろう．B/F 値をできるだけ小さく抑えるためには，メモリからのロードストア数を減らすことが必要である．

一方，実用コードの特徴的な点として，多様な境界条件に対応することが要求される．このため，境界条件を含んだ反復解法部分の高速処理を考える必要がある．これらの点を考慮して，次のポリシーにより反復解法部分を設計する．

1. EBCS による境界条件を含んだ統一的な反復式の表現.
2. ロードの抑制のため，必要な係数をビットにエンコードする．
3. ライブラリを含む反復解法を選択可能なように反復解法クラスを設計する．

#### 3.2.1.1　圧力境界条件の種類

圧力の境界条件の分類として，次の 2 通りがある．

1. 数学的な分類 Neumann 型条件と Dirichlet 型条件

2. 物理的な分類計算内部領域の境界条件と外部境界条件

数学的な分類の観点からは，次のような状況が考えられる．

- Neumann 条件
  - 勾配値が不変の場合
    $\nabla p = const.$ を反復ループの外側で与える．
  - 勾配値がタイムステップ間では一定であるが，時間的には変化する場合
    勾配値を反復ループの外側で与える．
  - 勾配値が反復毎に変化する場合
    勾配値が Navier-Stokes 方程式から計算される場合で，壁面境界条件，流入条件などに用いる．
- Dirichlet 条件
  - 値が不変の場合
    値を反復ループの外側で与える．
  - 値がタイムステップ間では一定であるが，時間的には変化する場合
    値を反復ループの外側で与える．

#### 3.2.1.2　EBCS による Poisson 反復離散式の表現

連続の式と Fractional Step 法のポテンシャルの修正式から導かれる圧力の Poisson 方程式は**式 (3.3)** のようになる．

$$\nabla\left(\nabla p^{n+1}\right) = \frac{1}{\Delta t} div\left(u^{*}\right) \equiv \psi \tag{3.3}$$

反復式に組み込む境界条件とパターンして，以下の 3 種類を考慮する．

1. Dirichlet 条件

$$p = p^{BC} \tag{3.4}$$

外部境界条件での利用を想定している．

2. Neumann 条件

$$\nabla p = \begin{cases} \nabla p^{BC} & \text{1 time step 中で一定値をとり，圧力 Poisson の反復過程で固定値となる．} \\ f(u_i) & \text{NS 式から圧力勾配項を計算するので，反復毎に値が変化する．} \end{cases} \tag{3.5}$$

3. 固体中の圧力
   固体中の圧力はマスクなどによりゼロとして扱うと，圧力がドリフトした場合にゼロ固定で残るため，結果を見る場合に誤解を招いたり，描画処理で不都合がある．そこで，周囲の値の範囲内に自然に収まるように，ラプラス方程式 **(3.6)** を解く．

$$\nabla\left(\nabla p^{n+1}\right) = 0 \tag{3.6}$$

■Neumann **条件の導入**

EBCS を用いて，**式 (3.3)** に Neumann 条件を導入すると，

$$\begin{aligned}\nabla(\nabla p) = \psi \quad \rightarrow \quad \frac{1}{h}\{ & \left(\nabla p\,\phi^N\right)_e + \left(1-\phi_e^N\right)\nabla p_e^{BC} - \left(\nabla p\,\phi^N\right)_w - \left(1-\phi_w^N\right)\nabla p_w^{BC} \\ + & \left(\nabla p\,\phi^N\right)_n + \left(1-\phi_n^N\right)\nabla p_n^{BC} - \left(\nabla p\,\phi^N\right)_s - \left(1-\phi_s^N\right)\nabla p_s^{BC} \\ + & \left(\nabla p\,\phi^N\right)_t + \left(1-\phi_t^N\right)\nabla p_t^{BC} - \left(\nabla p\,\phi^N\right)_b - \left(1-\phi_b^N\right)\nabla p_b^{BC} \quad \} = \psi\end{aligned} \tag{3.7}$$

ここで $\phi_l^N$ は，Neumann 条件を表すセルフェイスの境界条件マスクで，三次元の場合，1 つのセルに対して 6 方向 $l = \{e, w, n, s, t, b\}$ 存在し，**式 (3.8)** の値をとる．

$$\phi_l^N = \begin{cases} 1 & \text{Normal face in a fluid cell} \\ 0 & \text{Neumann boundary face} \end{cases} \tag{3.8}$$

**式 (3.7)** を整理して，境界条件部分を右辺項に移動すると，

$$\frac{1}{h}\left\{\left(\nabla p\,\phi^N\right)_e - \left(\nabla p\,\phi^N\right)_w + \left(\nabla p\,\phi^N\right)_n - \left(\nabla p\,\phi^N\right)_s + \left(\nabla p\,\phi^N\right)_t - \left(\nabla p\,\phi^N\right)_b\right\} = \psi \tag{3.9}$$

$$\sum_l \left(\nabla p\,\phi^N\right)_l n_l = h\psi - \sum_l \left(1-\phi_l^N\right)\nabla p_l^{BC}\, n_l \tag{3.10}$$

Volterra の原理を用いて半離散的に表し[*1]，$l$ は隣接セルフェイスの意味で $l = \{e, w, n, s, t, b\}$，$n_l$ はセルの外側方向の単位法線とする．$\phi$ はセルフェイスで定義された値であるが，$p$ についてはその方向の隣のセルの値を意味する．

■Dirichlet **条件の導入**

次に，Dirichlet 条件を組み込む．**式 (3.8)** と同様に，Dirichlet 条件を表すセルフェイスの境界条件マスクを次式で表す．

$$\phi_l^D = \begin{cases} 1 & \text{Normal face in a fluid cell} \\ 0 & \text{Dirichlet boundary face} \end{cases} \tag{3.11}$$

セルを構成する面に対しては，一つの面について Neumann 条件か Dirichlet 条件のどちらか一方しか指定できないので，両者は排他的である点に注意する．EBCS を用いて**式 (3.10)** 左辺の各勾配項に Dirichlet 条件のフラグ **(3.11)** を導入する．

$$\left.\begin{aligned}\left(\nabla p\,\phi^N\right)_e &= \frac{1}{h}\left\{p_{i+1}\,\phi_e^D + \left(1-\phi_e^D\right)p_e^{BC} - p_i\right\}\phi_e^N \\ \left(\nabla p\,\phi^N\right)_n &= \frac{1}{h}\left\{p_{j+1}\,\phi_n^D + \left(1-\phi_n^D\right)p_n^{BC} - p_j\right\}\phi_n^N \\ \left(\nabla p\,\phi^N\right)_t &= \frac{1}{h}\left\{p_{k+1}\,\phi_t^D + \left(1-\phi_t^D\right)p_t^{BC} - p_k\right\}\phi_t^N \\ \left(\nabla p\,\phi^N\right)_w &= \frac{1}{h}\left\{p_i - p_{i-1}\,\phi_w^D - \left(1-\phi_w^D\right)p_w^{BC}\right\}\phi_w^N \\ \left(\nabla p\,\phi^N\right)_s &= \frac{1}{h}\left\{p_j - p_{j-1}\,\phi_s^D - \left(1-\phi_s^D\right)p_s^{BC}\right\}\phi_s^N \\ \left(\nabla p\,\phi^N\right)_b &= \frac{1}{h}\left\{p_k - p_{k-1}\,\phi_b^D - \left(1-\phi_b^D\right)p_b^{BC}\right\}\phi_b^N\end{aligned}\right\} \tag{3.12}$$

[*1] 表記はベクトルで，添え字はセル界面の番号であることに注意する [11; p. 39].

**式 (3.10)** と**式 (3.12)** をまとめると，

$$\sum_l \left(p\,\phi^D\,\phi^N\right)_l - p\sum_l \phi_l^N = \underbrace{h^2\psi}_{\gamma^0} \underbrace{- h\sum_l \left(1-\phi_l^N\right)\nabla p_l^{BC} n_l}_{\gamma^N} \underbrace{- \sum_l \left(1-\phi_l^D\right) p_l^{BC}\phi_l^N n_l}_{\gamma^D} \tag{3.13}$$

右辺第一項目 $\gamma^0$ は連続条件と速度境界条件に由来する項である．第二項目は Neumann 条件 $\gamma^N$，第三項目は Dirichlet 条件 $\gamma^D$ の寄与である．更に，Dirichlet, Neumann 条件を反復間で固定値をとる $\gamma^{D1}, \gamma^{N1}$ と，反復毎に値が変わる $\gamma^{D2}, \gamma^{N2}$ に分ける．**式 (3.13)** 右辺のソース項の種類を**表 3.1** に分類して示す．$p_l^{BC}$ は仮想的に与える圧力値であることに注意する．つまり，計算空間のセルには直接値を代入しない．

反復回数を $k$ として SOR 法系の反復式を構成すると，

$$\left.\begin{aligned}
\tilde{p} &= \frac{1}{\sum_l \left(\phi_l^N\right)} \left\{ \sum_l \left(p\,\phi^D\,\phi^N\right)_l \underbrace{-\gamma^0 + \gamma^{D1} + \gamma^{N1}}_{\text{反復中固定値}} \underbrace{+\gamma^{D2} + \gamma^{N2}}_{\text{反復毎に変化}} \right\} \\
\Delta p &= \tilde{p} - p^k \\
p^{k+1} &= p^k + \omega\,\Delta p \\
\gamma^0 &= h^2\psi \\
\gamma^{D^1} &= \sum_l \left(1-\phi_l^D\right) p_l^{BC}\phi_l^N n_l \\
\gamma^{D^2} &= \sum_l \left(1-\phi_l^D\right) p_l^{BC}\phi_l^N n_l \\
\gamma^{N^1} &= h\sum_l \left(1-\phi_l^N\right)\nabla p_l^{BC} n_l \\
\gamma^{N^2} &= h\sum_l \left(1-\phi_l^N\right)\nabla p_l^{BC} n_l
\end{aligned}\right\} \tag{3.14}$$

上式は，Neumann 条件と Dirichlet 条件を含む圧力反復式で，流体セルと固体セルに適用する．ここで問題点は，周囲を壁で囲まれた孤立した流体セルの場合，対角項の係数がゼロとなる場合である．そのようなセルでは流れは生じないので強制的に固体壁にして，少なくとも 1 面は流体セルと接することを保証する[*2]．

**表 3.1** に示す壁面条件については，圧力勾配の評価方法に 2 種類ある．$\nabla p = 0$ は高レイノルズ数の場合に用いられる近似的な圧力境界条件であり，もう一方は低レイノルズ数の場合などに用いる厳密な境界条件である．後者の方がよい近似であるが，計算コストは高い．壁面条件の評価方法は，Steer セクション内の `Pressure_BC_for_wall_surface` タグで指定する[*3]．

Neumann 条件と Dirichlet 条件で値がゼロの場合には，ソース項の計算は簡単になる．つまり，内部境界，外部境界ともに $\gamma^{N1}, \gamma^D$ については計算不要となり，$\gamma^{N2}$ のみを計算すればよい[*4]．したがって，**表 3.1** の境界条件に対して，固定ソース項は速度境界条件による $\gamma^0$ のみを考慮すればよい．

**■Dirichlet 境界条件の利用**　Dirichlet 型の境界条件は，計算外部領域であるガイドセルに与える場合を想定している．ガイドセル上に指定される場合には，そのセルは計算せず，単に参照値として利用されるのみである．具体的には，遠方境界と流入出境界条件の場合の流入時（トラクションフリー）には $p = 0$ を利用する．この場合は，外部境界を含む計算内部セルの該当面に Dirichlet BC フラグをゼロにセットして計算する．

---

[*2] 処理としては，壁面と流入出条件に由来する全ての NeumannBC フラグをセットした後，全周 NeumannBC フラグのセルをみつけ，状態を変更する．

[*3] Pressure_BC_for_wall_surface = grad_zero or grad_NS

[*4] 境界条件のビットフラグ $\phi^D, \phi^N = 1$ で有効な場合でもその値がゼロであるから．

表 3.1　内部と外部境界に指定できる圧力境界条件の種類

| 境界条件の種類 | 内部 | 外部 | ソース項 | 実装 |
|---|---|---|---|---|
| 壁面境界 $\nabla p = 0$ | ○ | ○ | $\gamma^{N1}$ | $\partial p/\partial x_i = 0$ |
| $\nabla p = f(u_i)$ | ○ | ○ | $\gamma^{N2}$ | $\partial p/\partial x_i = 1/Re(\partial^2 u_i/\partial x_i^2)$ |
| 対称境界 $\nabla p = 0$ | - | ○ | $\gamma^{N1}$ | $\partial p/\partial x_i = 0$ |
| 流入境界 $\nabla p = f(u_i)$ | ○ | ○ | $\gamma^{N2}$ | $\partial p/\partial x_i = -\partial u_i/\partial t - (u_j - u_j^g)(\partial u_i/\partial x_j) + 1/Re(\partial^2 u_i/\partial x_i^2)$ |
| 流出境界 $\nabla p = 0$ | ○ | ○ | $\gamma^{N1}$ | $\partial p/\partial x_i = 0$ |
| $p = 0$ | ○ | ○ | $\gamma^{D}$ | |
| 遠方境界 $p = 0$ | - | ○ | $\gamma^{D}$ | トラクションフリー |
| 流入出境界 | - | ○ | - | 流出境界と，（流入として）遠方境界の切り替え |
| 周期境界 | - | ○ | - | データコピー |
| 周期境界 | - | ○ | $\gamma^{D^2}$ | ドライバ |

**■内部境界条件と外部境界条件の取り扱い　表 3.1** に示すように，内部境界条件と外部境界条件は $\partial p/\partial x_i = 0$ や $p = 0$ など比較的簡単な形式である．計算時には内部と外部を区別せずに，ビットフラグの状況によりソース項の計算の後，**式 (3.14)** を計算する実装としている．ただし，周期境界条件のみは，反復後に処理する．

**■固体セルの場合の Poisson 反復**　固体セルの場合には，前述のようにラプラス方程式を解き，値が平滑化され周囲の値の平均値となることを期待する．この場合，求まった固体中の圧力には物理的な意味はない．Laplace 方程式を解く場合，**図 3.1** において，S セルは全ての面で $\phi^N = 1$ を与え，**式 (3.14)** において $\psi = 0$ として反復する．

固体中の圧力も解くことになるので，収束ノルムを L2 とする場合には，固体セルの残差もノルム評価に加えられる点に注意する．流体を 1 セルの固体セルで分離しているような状況で，固体セルの両側の流体セルで圧力値が大きく異なる場合には，固体セルの圧力勾配が大きくなり，収束性が悪くなることもあり得る．一方，ノルムに $div\,(u^{n+1})$ をとる場合には，収束性は流体のみを解いている場合と変わりない（はず..）．

**■圧力境界条件に利用するビットフラグ**　セルの面の状態を表す Neumann 条件 **(3.8)** や Dirichlet 条件 **(3.11)** のフラグを効率的に実装するため，フラグを 1bit で表現し，整数型の変数にまとめて保持する．ビット操作による演算コストは，フラグを通常の変数で保持しロードするコストに比べれば少ない．4 バイト整数へのビットの割り当てを**表 11.1**，**表 11.2** に示す．

**図 3.1** に示すように，通常の流体セル (F) と固体セル (S) を考える．Neumann 条件と Dirichlet 条件をフラグにより各フェイスに与え，**式 (3.13)** を計算する．

**■連立一次方程式の係数行列と定数ベクトルの計算　式 (3.13)** を離散化して得られる連立一次方程式 $\mathrm{A}\boldsymbol{x} = \boldsymbol{b}$ の係数行列 A と定数ベクトル $\boldsymbol{b}$ については，前述したポリシーから以下のようにまとめられる．

- ビットフラグの計算
  エンコードして保持しておくフラグは，次の 4 種類である．

$$\left.\begin{array}{lllll} \phi_l^D\,\phi_l^N & 6\text{ 方向} \times 1\,\text{bit} & (6\,\text{bits}) & \text{BC\_NDAG\_\{E, W, N, S, T, B\}} \\ \phi_l^N & 6\text{ 方向} \times 1\,\text{bit} & (6\,\text{bits}) & \text{BC\_N\_\{E, W, N, S, T, B\}} \\ \phi_l^D & 6\text{ 方向} \times 1\,\text{bit} & (6\,\text{bits}) & \text{BC\_D\_\{E, W, N, S, T, B\}} \\ \sum_l \left(\phi_l^N\right) & 1 \sim 6 & (3\,\text{bits}) & \text{BC\_DIAG} \end{array}\right\} \tag{3.15}$$

- 定数ベクトルの計算
  計算内部領域における固体セルを計算する場合，定数項を $\psi = 0$ と修正する．

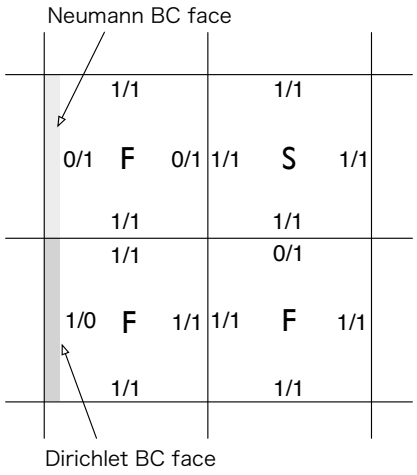



図 3.1　圧力反復式に用いるビットフラグ．各セルについて，6 つのフラグ（三次元の場合）を設定する．S セルは固体セル，F セルは通常の流体セルであり，この例では，左下のセルに Dirichlet 境界，左上のセルに Neumann 境界が与えられている．Neumann BC フラグは，Neumann 条件が与えられたセル面と固体セルに接する流体セルの接触面をゼロに設定する．

**■圧力 Poisson 部の計算手順　式 (3.13) を 3.14 の手順で解く場合，ソース項の処理は以下のように考える．**

begin: Time step loop
- pseudo vector の計算
- Poisson のソース項 $\gamma^0$ の計算
- Poisson のソース項 $\gamma^{N1}$ の計算（反復中に変化しない条件：壁面，対称，流出）
- Poisson のソース項 $\gamma^D$ の計算（流出，遠方，流入出）
- begin: Iteration loop
  - Poisson のソース項 $\gamma^{N2}$ の計算（反復中に変化する条件：壁面，流入）
  - Poisson の求解 (**式 3.14**)
- end: Iteration loop

end: Time step loop

反復中に変化する Neumann 条件は，壁面境界で圧力勾配を Navier-Stokes 方程式から求める場合と流入境界の場合である．ソース項の計算タイミングとして，圧力 Poisson の反復前と反復内の 2 通りがある．$\gamma^{N1}$ は反復前に決まる条件である．一方，壁面境界や流入境界で NS 式から圧力勾配を計算する $\gamma^{N2}$ は反復内で処理する．したがって，ソース項は 2 つの配列を使って実装している．

#### 3.2.1.3　Fractional Step 法における射影ステップの処理

コロケート変数配置の Fractional Step 法の計算処理では，セルセンタとセルフェイスに配置された速度ベクトルに対して，**式 (2.26, 2.27)** に示すそれぞれ圧力勾配による射影ステップの計算が必要になる．再掲すると，

$$\left.\begin{aligned} u_i^{n+1} &= u_i^* - \Delta t \frac{\partial p}{\partial x_i}^{n+1} \quad \text{(Cell center)} \\ u_{i,\,face}^{n+1} &= \bar{u}_{i,\,face}^* - \Delta t \frac{\partial p}{\partial x_i}^{n+1} \quad \text{(Cell face)} \end{aligned}\right\} \tag{3.16}$$

**式 (3.16)** の圧力勾配項に対して，**式 (3.12)** の圧力勾配の離散式を適用する．

**■セルセンタの圧力勾配**　x 方向を例にして，セルセンタの場合の圧力勾配の離散化を示す．

$$
\begin{aligned}
\nabla p_{c,x} &= \frac{1}{2}\left(\nabla p_e + \nabla p_w\right) \\
&= \frac{1}{2}\left[\left(\nabla p\,\phi^N\right)_e + \left(1-\phi_e^N\right)\nabla p_e^{BC} + \left(\nabla p\,\phi^N\right)_w + \left(1-\phi_w^N\right)\nabla p_w^{BC}\right] \\
&= \frac{1}{2}\left[\frac{1}{h}\left\{\phi_e^D p_{i+1} + \left(1-\phi_e^D\right)p_{i+1}^{BC} - p_i\right\}\phi_e^N + \left(1-\phi_e^N\right)\nabla p_e^{BC}\right. \\
&\quad \left. + \frac{1}{h}\left\{p_i - \phi_w^D p_{i-1} - \left(1-\phi_w^D\right)p_{i-1}^{BC}\right\}\phi_w^N + \left(1-\phi_w^N\right)\nabla p_w^{BC}\right] \\
&= \underbrace{\frac{1}{2h}\left\{\phi_e^D\phi_e^N p_{i+1} + \left(\phi_w^N - \phi_e^N\right)p_i - \phi_w^D\phi_w^N p_{i-1}\right\}}_{(i)} \\
&\quad + \underbrace{\frac{1}{2}\left[\frac{1}{h}\left\{\left(1-\phi_e^D\right)\phi_e^N p_{i+1}^{BC} - \left(1-\phi_w^D\right)\phi_w^N p_{i-1}^{BC}\right\} + \left(1-\phi_e^N\right)\nabla p_e^{BC} + \left(1-\phi_w^N\right)\nabla p_w^{BC}\right]}_{(ii)}
\end{aligned}
\tag{3.17}
$$

**式 (3.17)** において，(i) の項は非境界条件部分の圧力勾配を示す．一方，(ii) の項は Dirichlet 条件と Neumann 条件による圧力勾配を示し，反復ループ外で設定する．ガイドセルに接しない内部領域は $\nabla p = 0$ の条件のみなので，(ii) 項はゼロとしてよい．ガイドセルに接する内部領域では，外部境界条件 OUTFLOW, TRACTION_FREE により $p = 0$ の Dirichlet 条件が課せられる場合がある．その場合も参照値がゼロなので，(ii) の評価は不要になる．$\nabla p = 0$ の条件を考慮して，**式 (3.17)** を簡潔に表記すると，

$$
\nabla p_{c,x} = \frac{1}{2h}\left\{\phi_e^D\phi_e^N p_{i+1} + \left(\phi_w^N - \phi_e^N\right)p_i - \phi_w^D\phi_w^N p_{i-1}\right\} \tag{3.18}
$$

**■セルフェイスの圧力勾配**　次に，$i+1$ セルがディリクレ境界の場合のセルフェイスの圧力勾配を示す．

$$
\begin{aligned}
\nabla p_{i+1/2,x} &= \left(\nabla p\,\phi^N\right)_{i+1/2} + \left(1-\phi_{i+1/2}^N\right)\nabla p_{i+1/2}^{BC} \\
&= \frac{1}{h}\left\{\phi_{i+1/2}^D\, p_{i+1} + \left(1-\phi_{i+1/2}^D\right)p_{i+1}^{BC} - p_i\right\}\phi_{i+1/2}^N + \left(1-\phi_{i+1/2}^N\right)\nabla p_{i+1/2}^{BC} \\
&= \underbrace{\frac{1}{h}\left(\phi_{i+1/2}^D\,\phi_{i+1/2}^N\, p_{i+1} - \phi_{i+1/2}^N\, p_i\right)}_{(i)} + \underbrace{\frac{1}{h}\left(1-\phi_{i+1/2}^D\right)\phi_{i+1/2}^N\, p_{i+1}^{BC} + \left(1-\phi_{i+1/2}^N\right)\nabla p_{i+1/2}^{BC}}_{(ii)}
\end{aligned}
\tag{3.19}
$$

**式 (3.19)** において，(i) の項は非境界条件部分の圧力勾配，(ii) は境界条件により指定される圧力勾配を示す．圧力勾配の境界条件によるセルフェイスにおける修正も，セルセンターの場合と同様に，領域境界上の圧力勾配が ririchlet 型で与えられる場合のみを考慮すればよい．また，スタガード配置の境界上における速度ベクトルのインデクスは $i = 0, imax$ であり，$i = 0$ の場合には対応するビットフラグを設定する．$\nabla p = 0$, $p = 0$ を考慮すると，

$$
\begin{aligned}
X-\text{面)}\quad \nabla p_{1/2} &= \frac{1}{h}\left(\phi_{1/2}^N\, p_1 - \phi_{1/2}^D\,\phi_{1/2}^N\, p_0\right) \\
X+\text{面)}\quad \nabla p_{ix+1/2} &= \frac{1}{h}\left(\phi_{i+1/2}^D\,\phi_{i+1/2}^N\, p_{ix+1} - \phi_{i+1/2}^N\, p_{ix}\right)
\end{aligned}
\tag{3.20}
$$

結局，ディリクレ型として $p = 0$ を考える限りにおいては，修正項は不要となる．

### 3.2.2　対流項の扱い

対流項の空間スキームは保存的な差分法により離散化する．つまり，有限体積法と同様にセル界面での数値流束を求め，発散型と親和性の高い流束ベースの評価を行う．

#### 3.2.2.1　MUSCL 形式

対流項スキームについて，一次元を例に示す．インデクス $_i$ はセルセンター位置を示し，ハーフインデクス $_{i\pm1/2}$ はセル界面を意味する．

対流項スキームには，セル内部の分布形を仮定し，分布を再構築して精度を上げる MUSCL スキームを形式的に利用する [12]．分布形として線形 (piecewise linear) と放物形 (piecewise parabolic) の仮定はそれぞれ二次と三次の近似精度に相当する．Taylor 展開から非移流物理量 $\varphi$ の分布を再構成すると，

$$\varphi(x) = \varphi_i + (x - x_i)\frac{\varphi_{i+1} - \varphi_{i-1}}{2\Delta x} + \frac{3\kappa}{2}\left[(x - x_i)^2 - \frac{(\Delta x)^2}{12}\right]\frac{\varphi_{i+1} - 2\varphi_i + \varphi_{i-1}}{(\Delta x)^2} \quad (x_{i-1/2} \le x \le x_{i+1/2}) \tag{3.21}$$

セル境界における左右 (L, R) の物理量は，

$$\left.\begin{aligned} (\varphi_R)_{i+1/2} &= \varphi_{i+1} - \frac{\varepsilon}{4}\left[(1-\kappa)\Delta_{i+1}\varphi + (1+\kappa)\Delta_i\varphi\right] \\ (\varphi_L)_{i+1/2} &= \varphi_i + \frac{\varepsilon}{4}\left[(1-\kappa)\Delta_{i-1}\varphi + (1+\kappa)\Delta_i\varphi\right] \\ \Delta_i\varphi &= \varphi_{i+1} - \varphi_i \end{aligned}\right\} \tag{3.22}$$

となる．$\varepsilon$ は精度を切り替えるパラメータで $\kappa$ は分布形を制御する．

$$\varepsilon = \begin{cases} 0 & \text{1st order upwind} \\ 1 & \text{Higher order} \end{cases} \tag{3.23}$$

$$\kappa = \begin{cases} 1 & \text{2nd order central} \\ 1/3 & \text{3rd order upwind} \end{cases} \tag{3.24}$$

上記により求められたセル界面の両側の物理量の値を用いて，数値流束を計算する．ALE 形式を考慮して移流速度を $u - u^g$，非移流物理量を $\varphi$ とすると，物理流束は次式となる．

$$f = (u - u^g)\varphi \tag{3.25}$$

セル界面 $_{i\pm1/2}$ における数値流束 $\tilde{f}_{i\pm1/2}$ は次式で表せる．

$$\tilde{f}_{i\pm1/2} = \frac{1}{2}\left[(f_R + f_L) - |(u - u^g)|(\varphi_R - \varphi_L)\right]_{i\pm1/2} \tag{3.26}$$

#### 3.2.2.2　minmod リミター

MUSCL スキームのリミターは，主に圧縮性で衝撃波が生じるような不連続を扱う場合に，単調性を維持し，TVD 安定性を確保するために導入される．ここでは minmod limiter について示す．minmod limiter は次式で表される勾配制限関数である．

$$\mathrm{minmod}(x, y) = \mathrm{sgn}(x)\max[0, \min\{|x|, \mathrm{sgn}(x)y\}] \tag{3.27}$$

いま，$i$ セルの両側のセル界面における数値流束を考える．**図 3.2** を参照して，**式 (3.22)** に minmod limiter を適用すると，

$$\left.\begin{aligned}
(\varphi_R)_{i+1/2} &= \varphi_{i+1} - \frac{\varepsilon}{4}\left[(1-\kappa)\overline{\Delta^+} + (1+\kappa)\overline{\Delta^-}\right]_{i+1} \\
(\varphi_L)_{i+1/2} &= \varphi_i + \frac{\varepsilon}{4}\left[(1-\kappa)\overline{\Delta^-} + (1+\kappa)\overline{\Delta^+}\right]_{i} \\
(\varphi_R)_{i-1/2} &= \varphi_i - \frac{\varepsilon}{4}\left[(1-\kappa)\overline{\Delta^+} + (1+\kappa)\overline{\Delta^-}\right]_{i} \\
(\varphi_L)_{i-1/2} &= \varphi_{i-1} + \frac{\varepsilon}{4}\left[(1-\kappa)\overline{\Delta^-} + (1+\kappa)\overline{\Delta^+}\right]_{i-1}
\end{aligned}\right\} \tag{3.28}$$

ここで，

$$\left.\begin{aligned}
\Delta_i^+ &= \varphi_{i+1} - \varphi_i \\
\Delta_i^- &= \varphi_i - \varphi_{i-1}
\end{aligned}\right\} \quad \equiv d_{(1,2,3,4)}$$

$$\left.\begin{aligned}
\overline{\Delta_i^+} &= \mathrm{minmod}(\Delta_i^+,\, b\Delta_i^-) \\
\overline{\Delta_i^-} &= \mathrm{minmod}(\Delta_i^-,\, b\Delta_i^+)
\end{aligned}\right\}$$

$$b = \frac{3-\kappa}{1-\kappa}$$

である．

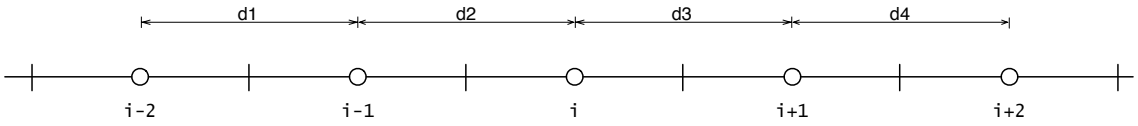



図 3.2　MUSCL スキームのステンシル．d1,d2,d3,d4 はセルセンタで定義される物理量の区間で，セル界面位置の速度の符号を s1,s2,s3,s4 とする．

コード中の実装は次の通り．

$$\begin{aligned}
&\text{g6;}\quad \overline{\Delta_{i+1}^+} = \mathrm{minmod}(\Delta_{i+1}^+,\, b\Delta_{i+1}^-) = \mathrm{s4}\cdot\max[0,\ \min\{|\mathrm{d4}|,\ \mathrm{s4\,b\,d3}\}] \\
&\text{g5;}\quad \overline{\Delta_{i+1}^-} = \mathrm{minmod}(\Delta_{i+1}^-,\, b\Delta_{i+1}^+) = \mathrm{s3}\cdot\max[0,\ \min\{|\mathrm{d3}|,\ \mathrm{s3\,b\,d4}\}] \\
&\text{g4;}\quad \overline{\Delta_{i}^+} = \mathrm{minmod}(\Delta_{i}^+,\, b\Delta_{i}^-) = \mathrm{s3}\cdot\max[0,\ \min\{|\mathrm{d3}|,\ \mathrm{s3\,b\,d2}\}] \\
&\text{g3;}\quad \overline{\Delta_{i}^-} = \mathrm{minmod}(\Delta_{i}^-,\, b\Delta_{i}^+) = \mathrm{s2}\cdot\max[0,\ \min\{|\mathrm{d2}|,\ \mathrm{s2\,b\,d3}\}] \\
&\text{g2;}\quad \overline{\Delta_{i-1}^+} = \mathrm{minmod}(\Delta_{i-1}^+,\, b\Delta_{i-1}^-) = \mathrm{s2}\cdot\max[0,\ \min\{|\mathrm{d2}|,\ \mathrm{s2\,b\,d1}\}] \\
&\text{g1;}\quad \overline{\Delta_{i-1}^-} = \mathrm{minmod}(\Delta_{i-1}^-,\, b\Delta_{i-1}^+) = \mathrm{s1}\cdot\max[0,\ \min\{|\mathrm{d1}|,\ \mathrm{s1\,b\,d2}\}]
\end{aligned}$$

#### 3.2.2.3　バイナリボクセルの場合の固体壁面境界処理

バイナリボクセルによる物体形状表現では，計算空間内の任意のセルが物体（非流体）セルになる．対流項スキームの近似精度が高くなるとステンシルが広がり，参照セルが非流体セルのときの処理を適切に行わなくてはならない．

物理流束は**式 (3.25)** で表せるので，セルの 6 面について和をとると，対流項は次のように離散化される．

$$\frac{\partial f}{\partial x_j} \equiv \frac{1}{h}\sum_l \tilde{f}_l\, n_l = \frac{1}{h}\sum_l \frac{1}{2}\left[\,(f_R + f_L) - |(u-u^g)|\,(\varphi_R - \varphi_L)\,\right]_l n_l \tag{3.29}$$

ここで $n_l$ はセルの各面における外側法線であり，移流速度は各面に垂直な方向成分と考える．

バイナリボクセルでは，セルの状態を表すマスク関数 $\phi$ により形状を階段状に近似する．$\phi$ は**式 (3.1)** として定義され，このマスク関数を用いて界面位置を調べ[*5]，壁面境界条件をスキーム中に取り込む [13]．**図 3.3** のようなコロケート配置におけるセルの状態パターンに対して，セル i での計算を考える．

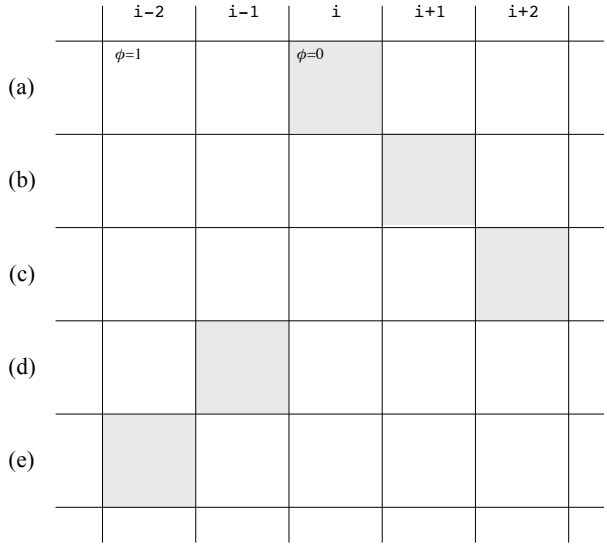



図 3.3　対流項の計算で考慮するマスクパターン．網掛けの部分は非流体セル $\phi = 0$，それ以外は流体セル $\phi = 1$ を示す．(a)~(e) のパターンは横方向に一次元的な分布を表している．

1. (a) の場合は，セル i の計算結果自体がマスクにより無効となるので考慮する必要はない．
2. ステンシルが $i \pm 2$ である (b)~(e) の場合は，参照値を修正する．
3. (b), (d) のパターン，つまり参照する $i \pm 1$ セルが固体の場合にも参照値を修正する．更に，$i \pm 1/2$ 位置の流束は，固定壁面があるので $\tilde{f} = 0$ となる．次式のように界面のフラグによりマスクし，流束をゼロにするように実装している．

$$\phi_{i\pm1/2} \;=\; \phi_i \times \phi_{i\pm1} \;=\; \begin{cases} 0 & \text{Wall boundary face} \\ 1 & \text{Fluid face} \end{cases} \tag{3.30}$$

$$\tilde{f}_{i\pm1/2} \;=\; \phi_{i\pm1/2}\, \tilde{f}_{i\pm1/2} \tag{3.31}$$

ここで右辺の $\tilde{f}_{i\pm1/2}$ は**式 (3.26)** の評価式である．

#### 3.2.2.4　参照値の修正

**図 3.3** の各パターンに対して，対象セルの対流項を計算する場合の参照値の修正について考える．バイナリボクセルの形状表現では，物体境界は必ずセル界面にある．どのセル界面が固体境界面であるかは，セルの状態 $\phi$ を見て判断し，固体境界面における参照値は流体セル側からの線形分布を仮定する．パターン (b), (d) の場合，参照値 $\varphi_{i\pm1}$ を次のように近似する．$\varphi_{i\pm1/2\,face}$ は境界値として与えられるものとする．

$$\varphi_{i\pm1} \;=\; 2\varphi_{i\pm1/2\,face} - \varphi_i \tag{3.32}$$

[*5] 事前に各セルの状態について，ビットフラグ (STATE_BIT) にエンコードしておく．計算時には，デコードしたフラグから**式 (3.30)** と式 (3.31) を評価する．対流項部分の計算は演算量が多く，if 文の分岐でも速度低下は小さいことを確認している (Intel Compiler 11.1).

同様に，$\varphi_{i\pm2}$ が参照値となる (c), (e) の場合についても $\varphi_{i\pm3/2\,face}$ を境界値として与え，次式で修正する．

$$\varphi_{i\pm2} \;=\; 2\varphi_{i\pm3/2\,face} - \varphi_{i\pm1} \tag{3.33}$$

### 3.2.3　粘性項の扱い

粘性項は，非圧縮条件を用いて簡略化して扱う．

$$\frac{1}{Re}\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right) = \frac{1}{Re}\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j}\right) \tag{3.34}$$

対流項と同様に参照値の修正を行い，評価する．ベクトル成分 $\varphi$ について，Euler 陽解法の場合について**式 (3.35)** に示す．

$$\frac{1}{Re}\frac{\partial}{\partial x_j}\left(\frac{\partial \varphi}{\partial x_j}\right) \approx \frac{1}{Re\,h^2}\left\{\sum_l(\varphi_l) - 6\varphi_{i,j,k}\right\} \tag{3.35}$$

$\sum\limits_l(\varphi_l)$ は，対象セルの隣接 6 セルの $\varphi$ の和を意味する．

### 3.2.4　境界条件処理

**2.3.1** の Fractional Step 法の手順では，疑似速度ベクトルを求めるプロセスで対流項と粘性項の計算を行い，その後境界条件処理を行う．境界条件には，外部境界条件と計算領域内部で用いる境界条件の 2 種類が設定できる．CBC ソルバークラスで指定できる速度の境界条件を**表 3.2** に示す．

表 3.2　計算領域内部と外部境界に指定できる速度境界条件の種類

| 境界条件の種類 | 内部 | 外部 | 外部境界の実装方式 | コメント |
|---|---|---|---|---|
| 壁面境界 | ○ | ○ | 流束形式 | フラグによりスキーム中に組み込まれている |
| 対称境界 | - | ○ | 境界値指定 | |
| 流入境界 | ○ | ○ | 流束形式 | Dirichlet 型による速度指定（一定値と単振動条件） |
| 流出境界 | ○ | ○ | 流束形式 | 対流流出，内部境界は面直のみ |
| 遠方境界 | - | ○ | 境界値指定 | トラクションフリー |
| 流入出境界 | - | ○ | | 流出境界と，（流入として）遠方境界の切り替え |
| 周期境界 | - | ○ | 境界値指定 | データコピー |

#### 3.2.4.1　速度境界条件のビットフラグ

速度境界条件を認識するために，圧力の場合と同様にビットフラグを導入する．**図 3.4** に示すように，各セルの 6 面についてフラグがセットされる．このフラグは各面に割り当てられた境界条件の識別子を示す．具体的には，媒質と境界条件の属性リストであるコンポーネントの登録番号である．CBC の実装では，コンポーネントは媒質と境界条件の個数を合わせて 30 個まで保持できる．各面に 5 ビット幅を割り当て，値の 0 は不使用（つまり，境界条件なし），31 は外部境界の認識に利用している（$2^5-2$ 個）．[*6]．**表 11.3** に具体的なビット割り当てを示す．

[*6] コンポーネント個数は ParseBC::setControlVars() でチェック．

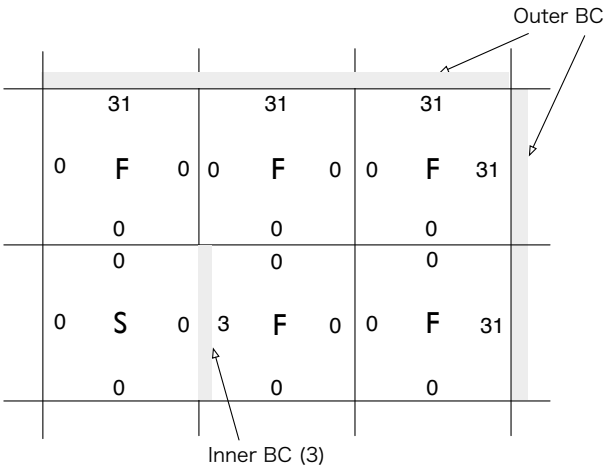



図 3.4　速度境界条件に用いるビットフラグ．各セルについて，6 つのフラグ（三次元の場合）を設定する．S セルは固体セル，F セルは通常の流体セルである．この例では内部境界条件として，コンポーネントの登録番号 3 の境界条件がひとつ指定されている．外部境界には，認識番号 31 が登録される．

#### 3.2.4.2　内部境界条件

固体壁条件については，計算空間内に任意の箇所に現れる可能性があるため，前述したようにフラグを用いて壁面の影響を流束形式によりスキームに導入する．一方，流入と流出条件は，その処理は煩雑で効率的な実装が難しい場合が多いので，演算速度はあまり期待できない．しかしながら，指定する速度境界条件は一般的に局所的になる傾向があるため，境界条件処理かどうかを探索する範囲をバウンディングボックスで絞り込んでおくことにより，演算処理自体を少なくできる．このため，コンポーネント単位で処理を行う．境界条件の導入は，圧力の場合の**式 (3.2)** と同様に，境界マスクと流束を次式により評価する．

$$\phi^{BC} = \begin{cases} 0 & \text{Face that boundary condition is given} \\ 1 & \text{Normal face} \end{cases} \tag{3.36}$$

$$\tilde{f}_{i\pm1/2} = \phi^{BC}\tilde{f}_{i\pm1/2} + \left(1-\phi^{BC}\right)\tilde{f}^{BC}_{i\pm1/2} \tag{3.37}$$

実装としては，**式 (3.37)** を 2 パスで構成する．つまり，最初のループで固体壁の影響を考慮した流束 $\tilde{f}_{i\pm1/2}$ を計算し，境界条件が指定された面の流束は計算しない．次のループで，境界条件面のみ流束 $\tilde{f}^{BC}_{i\pm1/2}$ を計算し，寄与分を加算する．内部境界条件は，ひとかたまりで格子線に沿って面直な流入・流出面をもつものを想定している．これらの境界条件は，流束形式でスキーム中に取り込まれる．また，補助的にセルフェイス位置の速度にも境界条件を与える．

#### 3.2.4.3　外部境界条件

外部境界条件の実装には，2 種類の方法を用いている．一つは，内部境界と同じく空間項の評価時に境界部分の流束を評価する方法である．もう一つは，一般的に行われている値を代入する方法である．後者の方法は，代入すべき位置（セル）が存在し，一意に決まる値で参照のみが行われる場合に用いる．

壁面境界は，空間項の評価時に壁面位置を感知し，その影響を境界条件として取り込む離散化を行っているので，特別な処理を必要としない．ただし，前処理として，指定された境界面に対して固体の ID をガイドセル部分に付与しておく．

流入と流出境界は，内部境界と同様な実装をで，処理単位は各面毎としている．

対称条件，遠方条件，周期条件については，境界値を指定する．

## 3.3　距離情報を用いた形状近似の解法

形状の近似として，物理量の定義点から物体までの距離情報のみを用いた形状近似を考える．前処理時の入力情報として STL などの形状データを考え，含まれる三角形要素と格子の交点情報を計算する．これより，距離情報が得られる．この場合，入力の STL が完全に形状を表現できていなくても，STL として三角形要素が定義できていれば交点情報が必ず得られる．また，定義点間に複数の交点があってもよい．したがって，データフォーマットとして有効であれば，形状を完全に再現できていなくても，薄い物体でも扱うことが可能な利点がある．

### 3.3.1　実装の基本的な考え方

コロケート格子における有限体積法ベースの流束計算に基づいて計算する．物体近傍付近では，流束を構築する際に必要な速度の参照点の値を流体側から外挿することにより，物体の影響を考慮した流束を計算する．したがって，物体の影響は全てセル界面の数値流束の近似に集約され，レギュラーのステンシルでの計算となる．

**図 3.5** を参照して，セル $i$ を積分する場合の壁面の影響をみる．セル界面 $i+1/2$ 側を考えると，次の 2 つの場合に分けられる．

(*a*)　$1/2 < d_i^+ < 1$　$\varphi_{i+1/2}$が定義でき, $\tilde{f}_{i+1/2}$が値をもつ.
(*b*)　$0 \le d_i^+ \le 1/2$　セル界面位置が固体内であるので$\varphi_{i+1/2}$が定義できない.

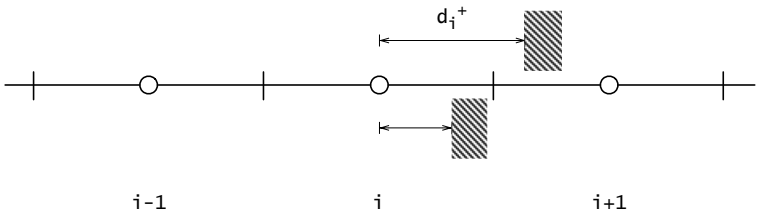



図 3.5　セル $i$ を積分する場合の壁位置による影響．$d$ は格子幅で正規化された無次元距離を表す．

$d_i^{\pm}$ は格子幅で正規化された距離で，$i$ 点における正負方向の距離を表し，[0,1] の値をとる．$\varphi$ は物理量で，例えば速度である．

(b) の場合には，セル界面位置が固体壁内部に位置するので，取り扱いを検討する．方法として，$\tilde{f}_{i+1/2} = 0$ と考えるやり方と，以下のように参照点の値を修正してそのまま計算するやり方が考えられる. とりあえず，後者を選択．したがって，(a) の場合のみについて流束計算を考える．

参照点の外挿は，**図 3.6** のように $d_i^+ \to 0$ となっても破綻しないように次式を用いる [14].

$$\varphi_{i+1} = \frac{1}{1/2 + d_i^+}\left[\frac{3}{2}\varphi_I - (1 - d_i^+)\,\varphi_{i-1/2}\right] \tag{3.38}$$

同様に，界面 $i - 1/2$ の場合には，

$$\varphi_{i-1} = \frac{1}{1/2 + d_i^-}\left[\frac{3}{2}\varphi_I - (1 - d_i^-)\,\varphi_{i+1/2}\right] \tag{3.39}$$

ここで $\varphi_I$ は壁面の値で指定値である．また，$\varphi_{i\pm1/2}$ の値は，定義されていなければ内挿する．

あるいは，少々不安定な実装として，次も考慮.

$$\varphi_{i\pm1} = \frac{1}{d_i^{\pm}}\left[\varphi_I - (1 - d_i^{\pm})\,\varphi_i\right] \tag{3.40}$$

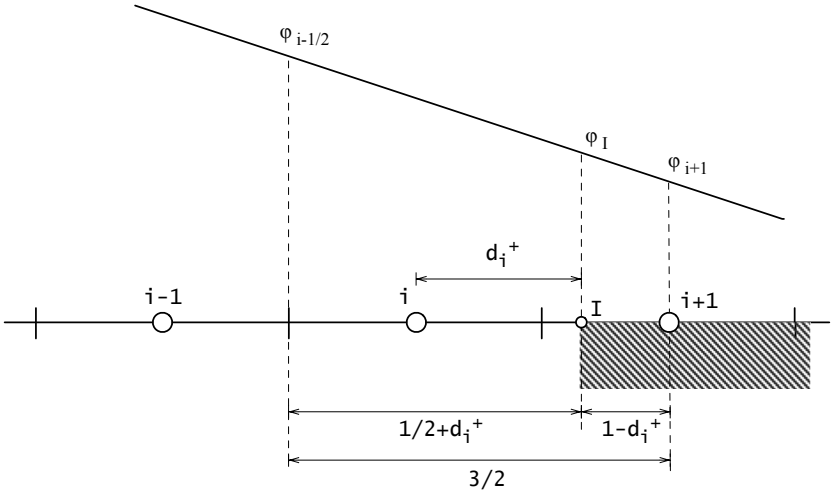



図 3.6　$(i+1)$ セルが固体内にある場合の参照値 $\varphi_{i+1}$ の外挿．距離は格子幅 $\Delta x$ により正規化している．

### 3.3.2　距離情報を用いた MUSCL スキームの実装

セル $i$ に関する流束計算を高次精度のスキームを用いるとステンシルが広がる．参照点間にカットが入るパターンに分類して，スキームの実装を述べる．

#### 3.3.2.1　対流項スキーム実装の詳細

MUSCL スキームは片側 3 点の参照点をもつので，参照する物理量が壁面位置による修正が必要かどうかを見ていく．

■$\tilde{f}_{i-1/2}$ **の評価**　$\tilde{f}_{i-1/2}$ を評価する場合，$u_{i-1/2}$ の符号により参照するセルが異なる．

$$\tilde{f}_{i-1/2} = \begin{cases} f(\varphi_{i-2}, \varphi_{i-1}, \varphi_i) & u_{i-1/2} \geq 0 \\ f(\varphi_{i-1}, \varphi_i, \varphi_{i+1}) & u_{i-1/2} < 0 \end{cases}$$

壁面の影響による修正が必要な場合とは，参照する定義点間に交点が存在する場合である．各セルで計算した距離情報の値を判定すれば，交点があるかどうかわかる．つまり，$d \neq 1$ であれば交点が存在し，参照するセルの値を修正する必要がある．

$u_{i-1/2} \geq 0$ **の場合**　参照するセルは $(i-2, i-1, i)$ であるが，$i$ セルについての計算なので，修正対象となるセルは $(i-2, i-1)$ セルである．

1. **図 3.7** に示すように，もし $d_i^- \neq 1$ であれば，$(i-1) \sim (i)$ 間に交点が存在し，**式 (3.39)** により $\varphi_{i-1}$ の参照値を修正する．
2. 同じように，$d_{i-1}^- \neq 1$ であれば，$\varphi_{i-2}$ の修正が必要になる．

$$\varphi_{i-2} = \frac{1}{1/2 + d_{i-1}^-} \left[ \frac{3}{2} \varphi_I - (1 - d_{i-1}^-) \varphi_{i-1/2} \right] \tag{3.41}$$

$u_{i-1/2} < 0$ **の場合**　参照するセルは $(i-1, i, i+1)$ であるが，$i$ セルについての計算なので，壁面が存在する場合の修正対象は $(i-1, i+1)$ セルである．

1. **図 3.8** に示すように，もし $d_i^- \neq 1$ であれば，$(i-1) \sim (i)$ 間に交点が存在し，**式 (3.39)** により $\varphi_{i-1}$ の参照値を修正する．

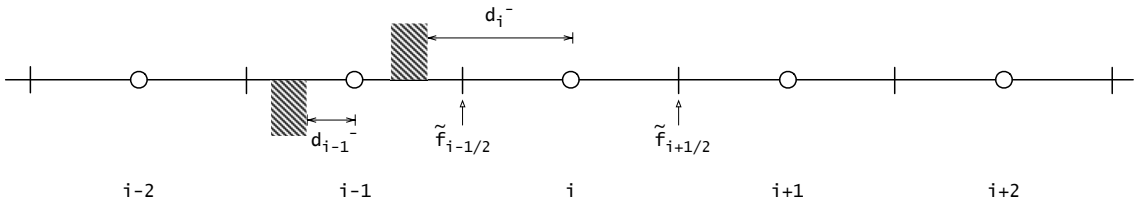



図 3.7　$\tilde{f}_{i-1/2}$ 評価時の $u_{i-1/2} \geq 0$ の場合の修正の判定．このときステンシルは $(i-2,\, i-1,\, i)$ で，壁面の影響による修正を考慮する対象は $(i-2,\, i-1)$.

2. $d_i^+ \neq 1$ であれば，**式 (3.38)** により $\varphi_{i+1}$ を修正する．

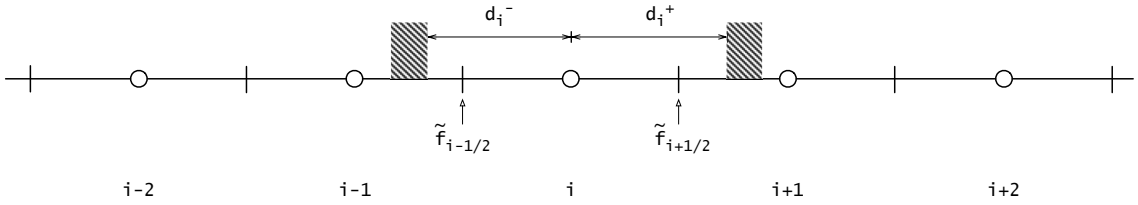



図 3.8　$\tilde{f}_{i-1/2}$ 評価時の $u_{i-1/2} < 0$ の場合，ステンシルは $(i-1,\, i,\, i+1)$ で修正対象は $(i-1,\, i+1)$．$\tilde{f}_{i+1/2}$ 評価時の $u_{i+1/2} \geq 0$ の場合のステンシルも同様.

■$\tilde{f}_{i+1/2}$ **の評価**　$\tilde{f}_{i+1/2}$ を評価する場合，$u_{i+1/2}$ の符号により参照するセルが異なる．

$$\tilde{f}_{i+1/2} = \begin{cases} f(\varphi_{i-1},\, \varphi_i,\, \varphi_{i+1}) & u_{i+1/2} \geq 0 \\ f(\varphi_i,\, \varphi_{i+1},\, \varphi_{i+2}) & u_{i+1/2} < 0 \end{cases}$$

$u_{i+1/2} \geq 0$ **の場合**　修正対象となるセルは $(i-1,\, i+1)$ セルである．

1. **図 3.8** に示すように，もし $d_i^- \neq 1$ であれば，$(i-1) \sim (i)$ 間に交点が存在し，**式 (3.39)** により $\varphi_{i-1}$ の参照値を修正する．
2. $d_i^+ \neq 1$ であれば，**式 (3.38)** により $\varphi_{i+1}$ を修正する．

$u_{i+1/2} < 0$ **の場合**　修正対象となるセルは $(i+1,\, i+2)$ セルである．

1. **図 3.9** に示すように，もし $d_i^+ \neq 1$ であれば，**式 (3.38)** により $\varphi_{i+1}$ の参照値を修正する．
2. $d_{i+1}^+ \neq 1$ であれば，次式により $\varphi_{i+2}$ を修正する．

$$\varphi_{i+2} = \frac{1}{1/2 + d_{i+1}^+} \left[ \frac{3}{2} \varphi_I - (1 - d_{i+1}^+)\, \varphi_{i+1/2} \right] \tag{3.42}$$

#### 3.3.2.2　粘性項の計算

粘性項を二次精度の中心差分で近似する場合は，**図 3.7** などでセル $(i-1,\, i+1)$ について，壁面の影響を考慮して参照値の修正をすればよい．あとは，参照値を用いて通常どおりのスキームにより計算をする[*7]．

[*7] 実装としては，壁面の影響を修正する処理を二重にするのを避けるため，対流項スキームとは分けず同一関数内で処理する．

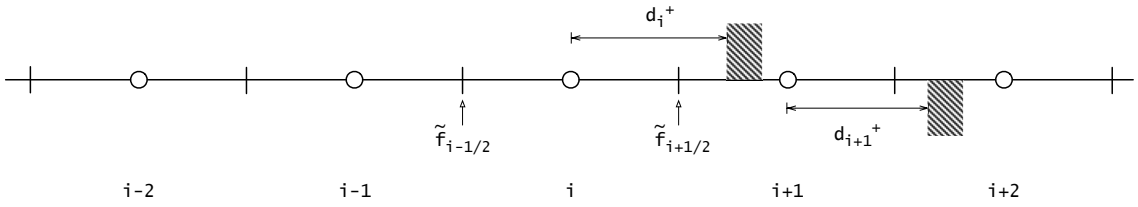



図 3.9　$\tilde{f}_{i+1/2}$ 評価時の $u_{i+1/2} < 0$ の場合の修正の判定.

#### 3.3.2.3　速度ベクトル発散の計算

Poisson 方程式のソース項や収束判定に用いる速度ベクトルの発散の計算について説明する．**図 3.10** に示すセル a は通常のステンシルで計算できる．一方，セル b とセル c は壁面の影響があるので，参照値を修正する．修正の方法は前述した方法を適用する.

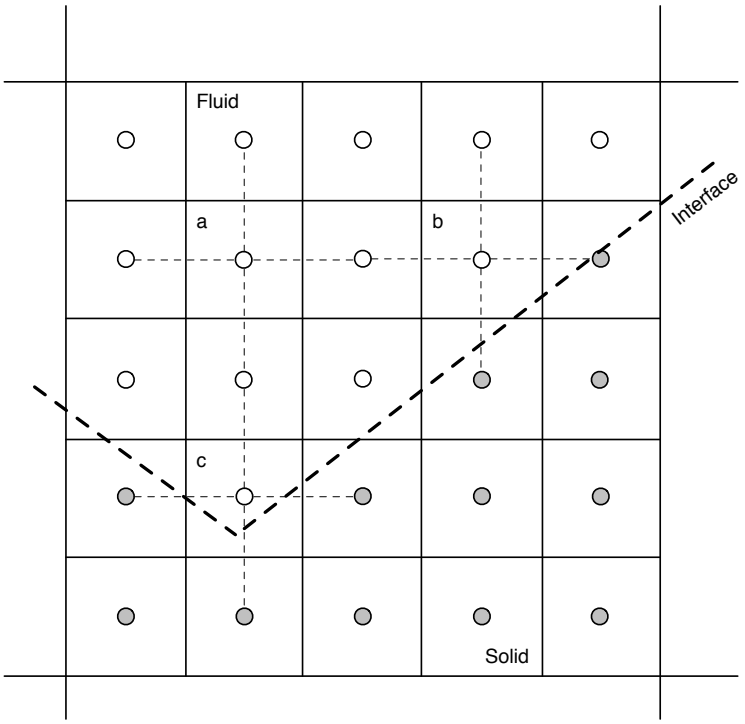



図 3.10　$\partial u_i/\partial x_i$ の計算と圧力計算．○はセルセンター位置が流体で計算するセル，●は計算しないセル．a は通常のステンシル，セル b,c にはそれぞれ 2 方向，3 方向の修正が必要.

### 3.3.3　圧力計算

圧力計算の方法として，Binary 近似の方法と法線情報を用いて形状近似度を改善する方法が適用可能.

#### 3.3.3.1　バイナリ近似の前処理

距離情報を用いた計算で圧力をバイナリで扱う場合，前処理時に距離情報を用いてノイマン条件をセットしておく．各セル定義点で計算された 6 方向の距離情報をみて，それらが 1.0[*8]でなければ壁面が存在することになる.

[*8] 格子幅 $\Delta x$ で正規化.

#### 3.3.3.2　法線を利用した参照点への内挿

法線情報を用いて精度を改善する方法について述べる．

### 3.3.4　物体との交点を含むセル間の速度ベクトルの内挿

セルセンター速度からセルフェイス速度への内挿処理が必要になるが，物体との交点を含むセル間では注意する必要がある．**図 3.11** には，セル i の両側のセルフェイス位置の速度 $\varphi_{i+1/2}, \varphi_{i-1/2}$ に内挿する処理を示している．左右のセルフェイスへは，それぞれ次式で内挿する．

$$\varphi_{i\pm1/2} = \frac{1}{d_i^{\pm}}\left[\frac{1}{2}\varphi_w + \left(d_i^{\pm} - \frac{1}{2}\right)\varphi_i\right] \tag{3.43}$$

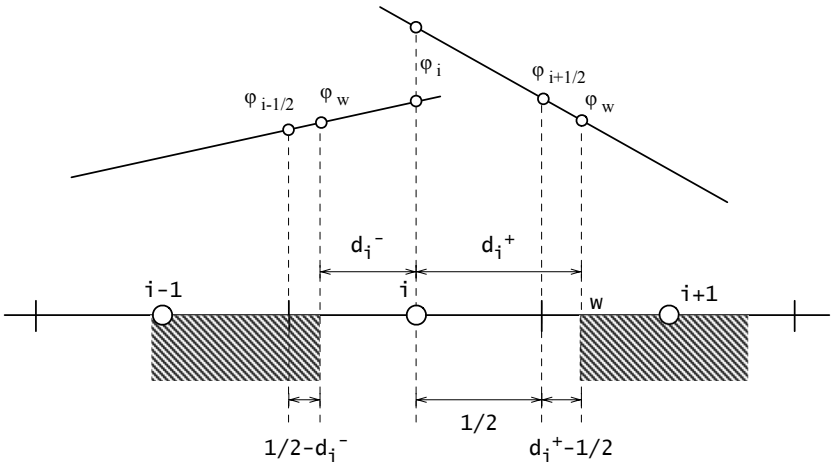



図 3.11　セルセンターの速度ベクトルからセルフェイスの速度ベクトルへの内挿.

セル間に薄い物体がある場合には，共有する同一のセル界面でも異なる参照値をもつことになる．この場合には，セルフェイスの値を保持して，それを積分するやり方ではなく，セルセンターの値に直接積分して集約するようにする．また，ここで述べる方法は前述した方法とは異なるので，暫定的な実装.

## 3.4　体積力を用いた境界条件

複雑形状周りの流れを解析する場合，Cartesian Method は格子作成の労力を低減できる点で魅力的な方法である．格子生成の手間は減少するが，それと引き換えにソルバー側で任意の形状の境界条件を取り扱う工夫が必要になる．また，形状近似のレベルに応じて必要な形状パラメータを得る前処理プログラムの開発が必要となる．Cartesian Method では格子密度が重要なパラメータとなるが，計算機資源や計算時間の制限から十分な格子が得られない場合がある．現象の中に空間スケールの異なる流れがあり相互に影響するような問題，例えば，多孔質層を通過する大空間の流れを解析する場合，興味の対象は大空間内の流動挙動であり，多孔質層内はマクロに見て適切な流れ場になっていればよい．この場合，多孔質部分はダルシー則などによりモデル化される．メッシュ解像度以下の微細な構造が流動特性に与える影響は，ダルシー則などのように理論的，あるいは実験式などで与えられる場合も多い．自動車のエンジンルーム内の流れを解析する場合にも，エンジンルーム内に配置される熱交換器は同様の特徴をもち，その特性は通過風速と圧力損失量の関係として実験式で与えられる．本節では Cartesian Method で有用な体積力を用いた種々の境界条件の導入について述べる．

### 3.4.1　外力項による境界条件の導入

**式 (3.44)** に示す Navier-Stokes 方程式中の外力項 $F_i$ を通して，体積力の形で様々な効果を導入する．各論の前に，まず Fractional Step 法に従って，大まかな手順を示す．

$$\frac{\partial u_i}{\partial t}^{n+1} + \frac{\partial}{\partial x_j}\left(u_i u_j\right) = -\frac{\partial p}{\partial x_i} + \frac{1}{Re}\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + F_i \tag{3.44}$$

体積力として与える境界条件の位置は，ボクセルモデルのセル毎に付与される ID により指定される．この ID により，境界条件の種類と計算空間内の場所が特定できる．この境界条件は，コンポーネントとして扱う[*9]．体積境界条件コンポーネントの占有率 $\beta$ をセル中心で次のように定義する．

$$\begin{cases} \beta = 0.0 & (Fluid) \\ 0.0 \le \beta \le 1.0 & (Component) \end{cases} \tag{3.45}$$

$\beta$ を用いると，**式 (3.44)** は次のように書ける．外力項を時間に関して陽的に扱う場合，非定常性が強い場合には不安定になることが懸念されるため，外力項を陰的に扱い Poisson の反復過程に組み込む．

$$\frac{\partial u_i}{\partial t}^{n+1} + \frac{\partial}{\partial x_j}\left(u_i u_j\right)^n = -\frac{\partial p}{\partial x_i}^{n+1} + \frac{(1-\beta)}{Re}\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j}^n + \frac{\partial u_j}{\partial x_i}^n\right) + \beta {F_i}^{n+1} \tag{3.46}$$

Euler 陽解法を用いると疑似ベクトルは，

$$u_i^* = u_i^n - \Delta t\left\{\frac{\partial}{\partial x_j}\left(u_i u_j\right)^n - \frac{(1-\beta)}{Re}\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j}^n + \frac{\partial u_j}{\partial x_i}^n\right)\right\} \tag{3.47}$$

射影ステップは，

$$u_i^{n+1} = u_i^* - \Delta t\left(\frac{\partial p}{\partial x_i}^{n+1} - \beta {F_i}^{n+1}\right) \tag{3.48}$$

圧力反復は連続の式から導かれ，次のようになる．

$$\frac{\partial}{\partial x_i}\left(\frac{\partial p}{\partial x_i}\right)^{n+1} = \frac{1}{\Delta t}\frac{\partial u_i^*}{\partial x_i} + \frac{\partial}{\partial x}\left(\beta {F_i}^{n+1}\right) \tag{3.49}$$

[*9] コンポーネントは，計算空間内に配置される境界条件や媒質を扱うための仕組みである．

多孔質内における流動は，対流項の影響よりも粘性力の影響が大きく，ダルシー抵抗と圧力のバランスが重要になる．**式 (3.49)** による外力の扱いはこの事実を直接的に反映しており，多孔質内流動などではより早い収束が期待できる．

### 3.4.2　熱交換器（圧力損失部）

熱交換器の特性を想定した圧力損失モデルを外力項の形式で表し，**3.4.1** に示した計算アルゴリズム中にコンシステントに組み込む．熱交換器は**図 3.12** のようにチューブとフィンから構成され，熱交換器面に対して面直の法線方向に流れが限定される．熱交換器の流動特性として，平均通過風速と熱交換器部で発生する静圧損失のマクロな関係が JIS で定められる測定により得られる．これは**図 3.13** に示すようにほぼ二次関数としてモデル化できる．熱交換器モデルとしては，下記の特性をもつ数値モデルを考える．

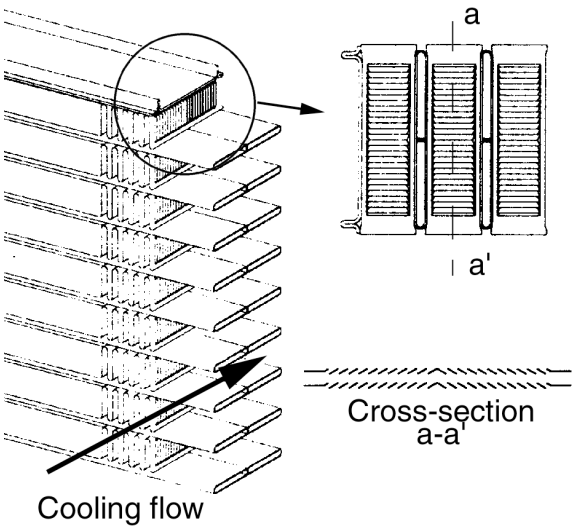



図 3.12　自動車用熱交換器の構造

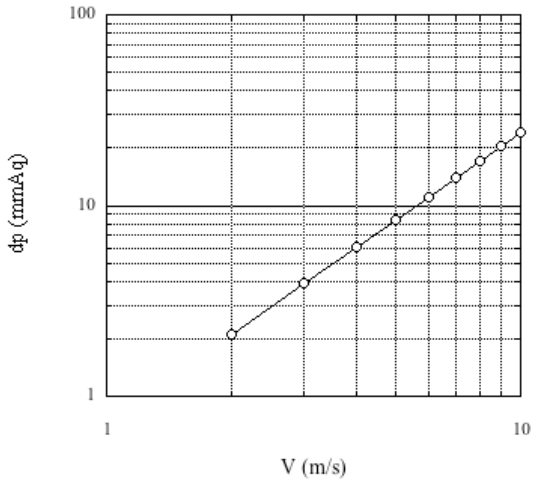



図 3.13　熱交換器の通過風量（風速）と圧力損失量の測定値

- 流れの方向は法線 $n_i'$ 方向（**図 3.12** 中の矢印の方向）に限定される．マクロな視点から $n_i'$ と垂直方向には，チューブにより流れが発生しない．
- $n_i'$ 方向に $\Delta p' = f(u_i')$ の圧力損失が発生する．
- 関数形として二次多項式の形式と離散データ形式を考慮する．
- 熱交換器部では，傾斜して配置された場合の入口損失なども実験式に含まれることとする．

以下の基礎式を考える．

$$\frac{\partial {u_i'}^{n+1}}{\partial x_i'} = 0 \tag{3.50}$$

$$\frac{\partial {u_i'}^{n+1}}{\partial t'} + \frac{\partial}{\partial x_j'}\left(u_i' u_j'\right) = -\frac{\partial p'}{\partial x_i'} + \frac{\partial \tau_{ij}'}{\partial x_j'} \tag{3.51}$$

$$\tau_{ij}' = \nu\left(\frac{\partial u_i'}{\partial x_j'} + \frac{\partial u_j'}{\partial x_i'}\right) \tag{3.52}$$

熱交換器部分では粘性項の影響を無視し，熱交換器による圧力損失の影響を $F_{R_i}'$ とすると，

$$\frac{\partial {u_i}'^{n+1}}{\partial t'} + \frac{\partial}{\partial x_j'}\left({u_i}'{u_j}'\right) = -\frac{\partial {p'}^R}{\partial x_i'} + F_{R_i}' \tag{3.53}$$

ここで，圧力勾配項は熱交換器による静圧変化分を表す．熱交換器の占有率として**式 (3.45)** の $\beta$ を用い，**式 (3.51)** と **式 (3.53)** をまとめると，

$$\frac{\partial {u_i}'^{n+1}}{\partial t'} + \frac{\partial}{\partial x_j'}\left({u_i}'{u_j}'\right) = -\left\{\beta\frac{\partial {p'}^R}{\partial x_i'} + (1-\beta)\frac{\partial p'}{\partial x_i'}\right\}^{n+1} + (1-\beta)\frac{\partial {\tau_{ij}}'}{\partial x_j'} \tag{3.54}$$

$\beta = 0.0$ のとき，つまり流体部分では，**式 (3.54)** は**式 (3.51)** と等価である．

ここで，熱交換器の圧力損失について考える．コア厚さ $\Delta r'$ の熱交換器で $\Delta {p'}^R = f({u_i}'^R)$ の圧力損失が得られる．圧力損失は速度に比例して大きくなり，通常の圧力勾配と同じ符号となる．つまり，正の向きの速度ベクトルに対して逆圧力勾配となる．

$$F_{R_i}' = -sgn\left({u_i}'\right)\frac{\Delta {p'}^R}{\Delta r'}\,{n_i}' \tag{3.55}$$

**式 (3.55)** の無次元化にあたり，$sgn({u_i}')$, ${n_i}'$ は単に方向を表すものであるので，無次元化には影響しない点に注意する．熱交換器部の圧力勾配は，流れ場の圧力勾配に対して線形な関係を仮定すると，次のようにモデル化できる．

$$\frac{\partial {p'}^R}{\partial x_i'} = \frac{\partial p'}{\partial x_i'} + F_{R_i}' \tag{3.56}$$

$$\frac{\partial {u_i}'^{n+1}}{\partial t'} + \frac{\partial}{\partial x_j'}\left({u_i}'{u_j}'\right) = -\frac{\partial {p'}^{n+1}}{\partial x_i'} + (1-\beta)\frac{\partial {\tau_{ij}}'}{\partial x_j'} + \beta {F_{R_i}'}^{n+1} \tag{3.57}$$

**式 (3.57)** を無次元化すると，**式 (3.44)** の形式になる．

#### 3.4.2.1　熱交換器の境界条件

熱交換器に特有の境界条件として，**式 (3.58)** に従い，速度ベクトルを熱交換器の流出方向（面法線ベクトル ${n_i}'$）へ射影する．

$$u_i^* = (1-\beta)u_i^* + \beta\left|u_i^*\right| n_i \tag{3.58}$$

もし，単に**式 (3.59)** により方向を強制すると，ベクトルの成す角度が大きくなった場合，内積の値が小さくなり，速度ベクトルの大きさが小さくなる．そうすると，保存則を満たすように逆に圧力勾配が大きくなり熱交換器部分で加速するようになる．したがって，動圧が変化しないように**式 (3.58)** を採用する．

$$u_i^* = (1-\beta)u_i^* + \beta\left(u_i^* n_i\right) n_i \tag{3.59}$$

$\beta$ と $F_i$ の各値は，速度ベクトル成分のセルセンタの点で定義されることに注意する．

熱交換器内部では，**式 (3.49)** の右辺第 3 項もゼロになる．したがって，Laplace 方程式を解いていることになり，単に前後の圧力差から平滑化を行うことになる．これは，熱交換器の前後で圧力勾配の跳びが発生し，内部は流速一定の流れ場になっているとする仮定とコンシステントである．**式 (3.49)** を解いて得られた圧力場は，熱交換器部で通過風速に応じた圧力勾配となっている．

本手法は流体の圧力場の一部分の勾配を指定しながら Poisson 方程式を解いている．これは，圧力 Poisson 方程式中において勾配の境界条件を満たすように解いていることと同じで，Liu ら [15] の Laplace/Poisson 方程式への境界条件の導入方法と基本的に同じアイデアである．また，圧力と速度の同時緩和を行いながら圧力損失項をアップデートするため，n+1 ステップでより良い近似になっている．

#### 3.4.2.2　圧力損失パラメータ

圧力損失パラメータは，熱交換器の性能試験結果により得られる．$\Delta p - V$ の性能線図を $[mmAq - m/s]$ 単位とした場合のパラメータの取得について示す．**図 3.13** のように横軸に熱交換器の平均通過流速 $V'[m/s]$，縦軸に圧力損失ヘッド $h'[mmAq]$ とする．多くの場合，流速の二次曲線で近似できる．プライム [′] は有次元量を示すものとして，

$$h' = \begin{cases} c_1 u'^2 + c_2 u' + c_3 & (u' \geq u_{th}') \\ c_4 u'^2 & (u' < u_{th}') \end{cases} \quad [mm] \tag{3.60}$$

係数 $c_4$ の式は速度が小さいときに圧力損失量をゼロに漸近させるためのもので，閾値 $u_{th}'$ で切り替える．

圧力損失と $\Delta r'$ $[mm]$ の厚さの熱交換器部での圧力勾配は次のようになる．

$$\Delta p' = \rho_w' g h' \times 10^{-3} \quad [Pa] \tag{3.61}$$

$$\frac{\Delta p'}{\Delta r'} = \frac{\rho_w' g h'}{\Delta r'} \quad [Pa/m] \tag{3.62}$$

$\rho_w'$ は水の密度 $[kg/m^3]$, $g$ は重力加速度 $[m/s^2]$ である．これらを次式で無次元化する．

$$\left.\begin{aligned} \Delta p &= \frac{\Delta p'}{\rho_0' u_0'^2} \\ \Delta r &= \frac{\Delta r'}{L'} \\ u &= \frac{u'}{u_0'} \end{aligned}\right\} \tag{3.63}$$

無次元の熱交換器部の圧力勾配は，基準密度を $\rho_0'$ として，

$$\frac{\Delta p}{\Delta r} = \frac{\rho_w' g h'}{\Delta r'} \frac{L'}{\rho_0' u_0'^2} \tag{3.64}$$

これより，

$$h' \equiv \left(\frac{\Delta p}{\Delta r}\right) \frac{\rho_0'}{\rho_w'} \frac{\Delta r' u_0'^2}{L' g} = c_1 (u\, u_0')^2 + c_2 (u\, u_0') + c_3 \tag{3.65}$$

$$\frac{\Delta p}{\Delta r} = \frac{\rho_w' L' g}{\rho_0' \Delta r' u_0'^2} \left\{ c_1 (u\, u_0')^2 + c_2 (u\, u_0') + c_3 \right\} = \frac{\rho_w' L' g}{\rho_0' \Delta r'} \left( c_1 u^2 + \frac{c_2}{u_0'} u + \frac{c_3}{u_0'^2} \right) \tag{3.66}$$

係数を**式 (3.67)** でまとめると,

$$\left.\begin{aligned} \xi_1 &= \frac{\rho_w{}' L' g}{\rho_0{}' \Delta r'} c_1 \\ \xi_2 &= \frac{\rho_w{}' L' g}{\rho_0{}' \Delta r'} \frac{c_2}{u_0{}'} \\ \xi_3 &= \frac{\rho_w{}' L' g}{\rho_0{}' \Delta r'} \frac{c_3}{u_0{}'^2} \\ \xi_4 &= \frac{\rho_w{}' L' g}{\rho_0{}' \Delta r'} c_4 \end{aligned}\right\} \tag{3.67}$$

無次元の圧力勾配は**式 (3.68)** で表せる.

$$\frac{\Delta p}{\Delta r} = \begin{cases} \xi_1 u^2 + \xi_2 u + \xi_3 & (u \geq u_{th}) \\ \xi_4 u^2 & (u < u_{th}) \end{cases} \tag{3.68}$$

無次元の圧力勾配から有次元の圧力損失量へは**式 (3.64)** を用いて算出する.

圧力損失ヘッドの測定単位が $p'$ $[mmHg]$ の場合には，水の密度を水銀密度 $\rho'_{mercury}$ に変更する[*10]．圧力損失が $\Delta p'$ $[Pa]$ で与えられる場合には,

$$\Delta p' = \begin{cases} c_1 u'^2 + c_2 u' + c_3 & (u' \geq u_{th}{}') \\ c_4 u'^2 & (u' < u_{th}{}') \end{cases} \quad [Pa] \tag{3.69}$$

$$\frac{\Delta p}{\Delta r} = \frac{\Delta p'}{\Delta r'} \frac{L'}{\rho_0{}' u_0{}'^2} \tag{3.70}$$

$$\Delta p' \equiv \left(\frac{\Delta p}{\Delta r}\right) \frac{\Delta r'}{L'} \rho'_0 u_0{}'^2 = c_1 (u\, u_0{}')^2 + c_2 (u\, u_0{}') + c_3 \tag{3.71}$$

$$\frac{\Delta p}{\Delta r} = \frac{L'}{\rho_0{}' \Delta r' u_0{}'^2} \left\{ c_1 (u\, u_0{}')^2 + c_2 (u\, u_0{}') + c_3 \right\} = \frac{L'}{\rho_0{}' \Delta r'} \left( c_1 u^2 + \frac{c_2}{u_0{}'} u + \frac{c_3}{u_0{}'^2} \right) \tag{3.72}$$

$$\left.\begin{aligned} \xi_1 &= \frac{L'}{\rho_0{}' \Delta r'} c_1 \\ \xi_2 &= \frac{L'}{\rho_0{}' \Delta r'} \frac{c_2}{u_0{}'} \\ \xi_3 &= \frac{L'}{\rho_0{}' \Delta r'} \frac{c_3}{u_0{}'^2} \\ \xi_4 &= \frac{L'}{\rho_0{}' \Delta r'} c_4 \end{aligned}\right\} \tag{3.73}$$

[*10] mmAq の場合には，$\theta = 300[K]$, $p = 101.325[kPa]$ で，$\rho'_{water} = 996.62[kg/m^3]$，mmHg の場合には，$\theta = 300[K]$ で，$\rho'_{mercury} = 13538[kg/m^3]$ をハードコード.

#### 3.4.2.3　実装の方針

体積力コンポーネントには，下記のような特徴がある．

- 一般に，コンポーネントは計算空間中に局在し，そのセル数も少ない．したがって，そのコンポーネントの計算に必要な体積率などを全空間にわたって保持する配列を用いるのはメモリコストがかかる．
- 局在性のため，並列処理時には負荷バランスが崩れることは避けられない．
- 必要な計算のために，体積率，重複計算を避けるためのワーク配列が必要．その配列の大きさは Bbox のサイズで与えられるが，前述のように全領域の中の一部分である．また，コンポーネントは一つでも全領域の配列が必要となる．

以上の点から，次の実装が考えられる．

- ワーク配列を必要とするコンポーネントがある場合には，そのの配列 (dc_wvex, dc_cfr) を全空間にわたって確保する．各コンポーネントの Bbox の大きさはサーチ用に利用する．メモリの無駄が生じるが，並列処理は標準的な方法が利用可能．コンポーネント（熱交換器を含む一般的な要素）の占有率 $\beta$ は，バイナリボクセルの場合，BinaryVoxel::setCmpFractionBV() により， bcx[] の情報から生成し，cfr[] に保持する．
- 別の方法としては，各コンポーネントの Bbox の大きさの配列をアロケートし，体積率とワークに用いて計算する．これにより，メモリ効率と計算負荷を小さく抑えられる．また，一方で並列処理時のインデクスの計算などが煩雑になるが，V-Sphere のコンポーネント機能を用いて実装する[*11]．

ここでは，前者の方法で実装した．

BVH を使って絞り込みを行っているが，Hybrid 並列では OpenMP のループ分割でロードインバランスが生じる可能性が高いので，一次元配列をコア数で分割する方がよい．

#### 3.4.2.4　解法の手順

1. 圧力損失部の疑似速度ベクトルの修正
   疑似速度ベクトルの計算式 (3.47) を式 (3.74) のように表し，流束の計算を通常の計算処理 (i) と圧力損失部分による修正 (ii) の 2 ステップに分解して計算する．(ii) の計算は，コンポーネントリストクラスと bcd[] の情報を用いて対象部分のみ計算する．実装では有限体積的に流束を評価しているので，修正も流束に対して行う．

$$
\begin{aligned}
u_i^* &= u_i^n - \Delta t \left\{ \frac{\partial}{\partial x_j}\left(u_i u_j\right)^n - \frac{(1-\beta)}{Re}\frac{\partial}{\partial x_j}\left(\frac{\partial {u_i}^n}{\partial x_j} + \frac{\partial {u_j}^n}{\partial x_i}\right)\right\} \\
&= u_i^n - \Delta t \left[ \underbrace{\frac{\partial}{\partial x_j}\left(u_i u_j\right)^n - \frac{1}{Re}\frac{\partial}{\partial x_j}\left(\frac{\partial {u_i}^n}{\partial x_j} + \frac{\partial {u_j}^n}{\partial x_i}\right)}_{(i)} + \underbrace{\frac{\beta}{Re}\frac{\partial}{\partial x_j}\left(\frac{\partial {u_i}^n}{\partial x_j} + \frac{\partial {u_j}^n}{\partial x_i}\right)}_{(ii)} \right]
\end{aligned}
\tag{3.74}
$$

   **式 (3.58)** により，圧力損失部の疑似速度ベクトルを流出方向に修正する (`SetBC3D::mod_Pvec_Forcing()`)．オプションとして，方向修正をしない場合も考慮する．

2. Poisson の圧力反復式とソース項の追加
   圧力損失部は外力項として Navier-Stokes 方程式に組み込まれ，その発散量が Poisson 方程式のソース項として作用する．つまり，圧力の反復式には圧力損失が寄与するソース項として，**式 (3.49)** の右辺第 2 項が追加され

[*11] テスト実装を試みるが，きわめて複雑で断念...

る．反復の半離散式は，**式 (3.14)** を参照して以下のようになる．

$$\left.\begin{aligned} \tilde{p} &= \frac{1}{\sum_l \left(\phi_l^N\right)} \left\{ \sum_l \left(p\,\phi^D\,\phi^N\right)_l \underbrace{-\gamma^0 + \gamma^{D1} + \gamma^{N1}}_{\text{反復中固定値}} \underbrace{+\gamma^{D2} + \gamma^{N2} - \gamma^F}_{\text{反復毎に変化}} \right\} \\ \Delta p &= \tilde{p} - p^k \\ p^{k+1} &= p^k + \omega\,\Delta p \\ \gamma^F &= h \sum_l \left(\beta F^{n+1}\right)_l n_l \end{aligned}\right\} \tag{3.75}$$

ここで，$\gamma^F$ が圧力損失部によるソース項を表し，`SetBC3D::mod_Psrc_Forcing()` で計算している．反復プロセスは圧力と速度の同時緩和を採用しているので，$\gamma^F$ の項は反復毎に計算する．これは外力成分の発散を表しているので，各成分はセルフェイス位置で評価する．このため，セルセンターで定義された $F_i$ の算術平均をとり，セルフェイス位置に内挿する．つまり $i+1/2$ の場合には，次式を用いる[*12]．

$$\left(\beta F_1^{n+1}\right)_{i+1/2} = \frac{\beta_{i+1}+\beta_i}{2} \frac{F_{1,i+1}^{n+1} + F_{1,i}^{n+1}}{2} \tag{3.76}$$

圧力損失による外力は**式 (3.55)** で表される．無次元で表記すると，

$$F_i{}^{n+1} = -sgn\left(u_i^{n+1}\right) \frac{\Delta p}{\Delta r}^{n+1} n_i \tag{3.77}$$

ここで，$F_i$ の各成分はセルセンターで定義されることに注意する．無次元圧力勾配の評価には，**式 (3.68)** を用いる．

3. 圧力損失部の速度ベクトルの射影
   射影ステップは，式 (3.48) を式 (3.78) の (i), (ii) の 2 ステップに分解し，(ii) の修正項をコンポーネントリストを参照しながら計算する．

$$u_i{}^{n+1} = \underbrace{u_i{}^* - \Delta t \left(\frac{\partial p}{\partial x_i^{n+1}}\right)}_{(i)} \underbrace{-\Delta t \beta F_i{}^{n+1}}_{(ii)} \tag{3.78}$$

セルセンターとセルフェイスの速度成分に対して，**式 (3.48)** を 2 段階に分けて各々 `SetBC3D::mod_Vcf_Forcing()`, `SetBC3D::mod_Vcc_Forcing()` により処理する．

$$\begin{aligned} u_i{}^{n+1} &= u_i{}^* - \Delta t \frac{\partial p}{\partial x_i}^{n+1} \\ u_i{}^{n+1} &+= \Delta t \beta F_i{}^{n+1} \end{aligned} \tag{3.79}$$

4. 実装上のコメント

   (a) 外力項の計算
   **式 (3.77)** で評価する外力項と**式 (3.79)** の外力項の値は同じである必要があるが，反復の過程で前反復値とほぼ同じになると考え，以下のようにコード化している．

[*12] 赤坂 啓，小野 謙二:直交格子を用いた非圧縮流れ解析における圧力損失モデルの実装, Transaction of JSCES, 20070032(2007)

(1)　Poisson のソース項は前反復値　$F_i = f(u_i^k)$ を使う．

(2)　セルフェイスの速度の射影 $F_i = f(u_i^k)$

(3)　セルセンターの速度の射影 $F_i = f(u_i^{k+1})$

この処理は，上記の全計算で $F_i = f(u_i^k)$ を保証するためには，$F_i$ の値を計算してベクトル配列に保持しておく必要があること，並列時に通信が発生することの 2 点を避けるためである．

（b）外力項の Poisson のソース項の扱い

セルフェイスの速度の射影ステップで，熱交換器部分の影響が**式 (3.79)** のように入り，熱交換器を構成するセルの各面で外力項が作用する．したがって，熱交換器の隣接セルについても，熱交換器に接するセルフェイスは影響を受ける．このことは，ソース項を計算する場合，熱交換器の隣接セルについてもソース項を計算しておく必要があることを意味する．

5. モニタ値の計算

モニタ量は，熱交換器部の通過風速と圧力損失量の無次元平均値を次式で計算する．$m$ は熱交換器を構成するセル数である．

$$\begin{aligned} \bar{u}_{hex} &= \frac{1}{m}\sum_m sgn(n_i, u_i)\,|u|\beta \\ \frac{\bar{\Delta p}}{\Delta r}_{hex} &= -\frac{1}{m}\sum_m sgn(n_i, u_i)\,\frac{\Delta p}{\Delta r}\beta \end{aligned} \tag{3.80}$$

圧力損失量は，熱交換器の厚さ方向の値を次式で推定し，有次元化して履歴に出力する．

$$\bar{\Delta p}_{hex} = \frac{\bar{\Delta p}}{\Delta r}\,\Delta r\,\rho {u'_0}^2 L' \quad [Pa] \tag{3.81}$$

### 3.4.3　ファンモデル

前述の圧力損失モデルと同様な方法でファンの P-Q 特性を反映したマクロモデルを構築する．ファンモデルは，圧力損失モデルと異なり，ファン回転の仕事により圧力増加（Pressure Gain）が生じ，また回転方向の流速が発生する．最初に，軸流と斜流ファンに適用可能なファンモデル，その後，遠心ファンのモデルを説明する．

#### 3.4.3.1　ファン特性とモデル

ファンの圧力増加と流量の関係，いわゆる P-Q 特性は JIS で定められた**図 3.14** のチャンバー方式の測定方法により計測され，P-Q 特性線図が得られる．排気側にファンを設置し，流量を変化させた場合の圧力ゲインを測定する．圧力の測定はチャンバー内 a 点の静圧と大気圧の差を測定する．ただし，チャンバー断面積は十分に大きく，チャンバー中央部 a 点での流速はほぼゼロに近いため，動圧はゼロと見なして良い．したがって，a 点付近で測定した静圧は全圧と等しいと考えて良い．

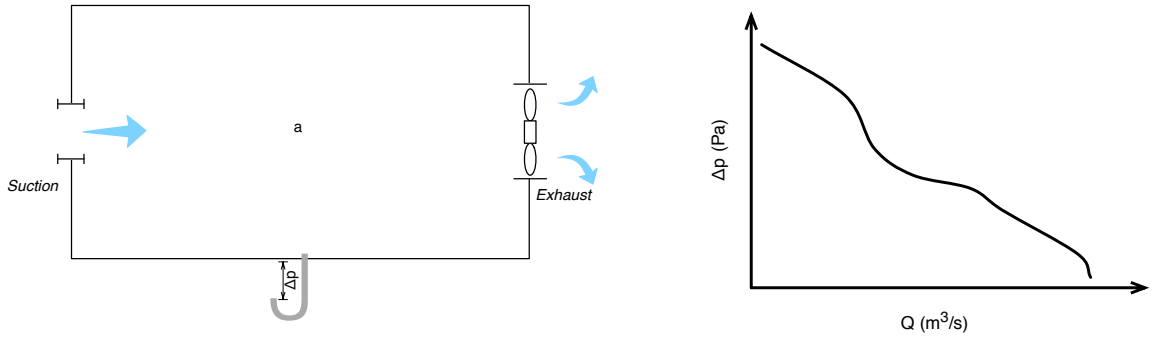



図 3.14　チャンバーによるファンの特性測定と P-Q 線図.

こうして得られたファンの P-Q 特性を計算に用いる場合には，次の点に注意する必要がある．合わせて，CBC ソルバーの実装方針も記す．

1. ファンの直近にある物体の影響

   ファン近傍に物体がある場合には，測定時の状況とは P-Q 特性が異なるために適用には注意が必要となる．基本的にはファン前後の近距離には物体を配置しないことであるが，シュラウドのようにほぼ一体となっているものは，シュラウド込みで測定し，P-Q 特性を取得するという利用方法も考えられる．

2. ラム圧の影響

   自動車の車載ファンのように走行風がある場合や，流路の狭い部分に設置されたファンの場合には，動圧の影響が無視できない．この場合，動圧分の補正をしないとファンの動作点がずれていく．計算モデルにおけるファン部分の上流側，下流側の位置をそれぞれ 1,2 で表し，静圧，動圧，全圧をそれぞれ $p_S$, $p_D$, $p_T$ ($p_T = p_S + p_D$) とする．圧力ゲインの計算方法として，測定データに基づく実装として，次の 2 通りが考えられる[*13]．

   (a) ファン前後の静圧差で計算する．つまり，$\Delta p = p_{S2} - p_{S1}$．この実装では，動圧の影響を考慮できずに動サテンがずれる．

   (b) 上流側の全圧と下流側の静圧の差で計算する．つまり，$\Delta p = p_{S2} - p_{T1}$．この実装により，上流側の動圧の影響を取り込むことができる．

   CBC では (b) の実装を行う．

3. 測定値の反映

   計算では測定データを多項式近似によりモデル化する方法が採られる．一般的には，P-Q 特性を一次式で近似することが多いが，一次式に乗らない点の影響が大きな場合がある．これは，多項式近似をしても根本的にはなく

[*13] ファン下流の速度が得られている場合には，$p_{T2}$ を用いた方法も考えられるが，一般的な測定データとしては得られない．

ならない．そこで，CBC では測定データ点をそのままテーブルとして保持し，データテーブルをピックアップして P-Q 特性を計算する．実装としては，DataHolder クラスを用いる．

4. 回転数や風量による P-Q 特性の補正
   P-Q 特性は定格回転数のみで有効である．回転数が異なる場合には補正したデータを用いる必要がある．

$$\text{風量比}\quad \frac{Q_2}{Q_1} = \left(\frac{L_2}{L_1}\right)^3 \frac{N_2}{N_1}$$

$$\text{静圧比}\quad \frac{P_2}{P_1} = \left(\frac{L_2}{L_1}\right)^2 \left(\frac{N_2}{N_1}\right)^2$$

ここで，$Q$, $L$, $N$, $P$ はそれぞれ流量，ファンブレード径，回転数，静圧を表し，1,2 は状態を表す．

#### 3.4.3.2　旋回流

ファンの軸流方向については P-Q 特性でその流量（速度成分）が与えられるが，回転方向成分は何らかのモデルが必要となる．よく使われる方法として，スリップ率を与える方法がある．スリップ率を一様に与える方法や半径方向の分布を与える方法などバリエーションは考えられるが，まず一様に与える方法を採用する．

ファンの通過風速については，流路面積を考慮する必要がある．つまり，ファンのボス部分は有効な通過面積ではないため，この部分の面積を除いた断面積を流路面積として扱う．これにより，同じ流量でも通過風速が異なり，動圧に反映され，P-Q 特性のルックアップ値が異なるため，注意する．

熱解析の場合には，ファンボス部分の背後に設置されているモータが発熱体として指定されることもある．

さらに，ファン停止時には逆流も許すように流路部分がフリーとなるように考える．これは，外力項をゼロにすることで対応する．

#### 3.4.3.3　実装と解法の手順

#### 3.4.3.4　遠心ファン

遠心ファンでは，吸入部分と排気部分で流速の方向が異なり，上記の軸流タイプの実装ができない．そこで，吸入部分と排気部分の速度ベクトルを規定し，圧力差を P-Q 線図により与える実装とする．

### 3.4.4　多孔質体モデル

本節では，多孔質モデルの取り扱いについて説明する．

#### 3.4.4.1　Darcy モデル

均質な空間に $A'\,[m^2] \times L'\,[m]$ の直方体の検査体積を考える．両端の圧力差が $\Delta P'\,[Pa]$ で流量が $Q'\,[m^3/s]$ の場合，透過率 (Permeability) を $k\,[m^2]$ として，

$$Q' = k\frac{A'}{\mu}\frac{\Delta P'}{L'} \tag{3.82}$$

$$u' = \frac{Q'}{A'} = \frac{k}{\mu}\frac{\Delta P'}{L'} \tag{3.83}$$

ここで，$\mu\,[Pa\cdot s]$ は粘性係数，$u'\,[m/s]$ は見かけの通過流速（多孔質内の平均速度）である．$\Delta P'/L'$ は圧力勾配なので，$u'\,(\mu/k)$ が力を表す．したがって，多孔質体部分の存在により流体に作用する外力は，

$$F_{D}{}'_{i} = -u_i{}'\frac{\mu}{k} \tag{3.84}$$

**式 (??)** の形式に合わせて，$\rho' u_0'^{\,2}/L'$ で無次元化すると，

$$F_{D\,i} = -\frac{1}{K}u_i \tag{3.85}$$

ここで，$K$ は無次元透過率を表す．

$$K = \frac{k}{\mu}\frac{\rho' u_0{}'}{L'} \tag{3.86}$$

#### 3.4.4.2　軸方向の異方性を考慮した Darcy モデル

非等方性を考慮した Darcy モデルは**式 (3.87)** のように表せる．

$$F_{D}{}'_{i} = -\frac{\mu}{k_{ji}}u_j{}' \tag{3.87}$$

$k_{ji}$ は異方性の係数を表すテンソルであるが，ここでは主軸方向の非等方性のみを考えるので，

$$k_i = \begin{pmatrix} k_1 & 0 & 0 \\ 0 & k_2 & 0 \\ 0 & 0 & k_3 \end{pmatrix} \tag{3.88}$$

**式 (3.86)** と同様に無次元化すると，

$$F_{D\,i} = -\frac{1}{K_i}u_i \tag{3.89}$$

Darcy モデルを用いる場合に必要なパラメータは**式 (3.88)** の主軸方向の成分となる．等方性の場合には，係数 $k_1, k_2, k_3$ に同じ値を与える．パラメータの記述については，**??**を参照．

### 3.4.5　共通の前処理

圧力損失，ファン，多孔質体の各境界条件は，基礎方程式の外力項としてモデル化している．この外力項は各コンポーネントの体積占有率の情報を必要とする．しばしば設計で用いられる不完全な入力形状データからコンポーネントの占有率を計算することは，内外判定に曖昧さが生じるため少々厄介である．また，マクロモデルは実際の詳細な形状があったとしても，計算のために少々形状変更を行う必要もある．そこで，モデルの特性と前処理のバランスを考慮して，以下の方針でモデル化し前処理を行う．

1. コンポーネントを矩形や円筒形に近似したボリュームとして扱い，体積占有率を自動計算する，
2. ボリュームを再現する幾何情報は数値データとして入力する．幾何情報データを CAD データから読み取る作業は手間がかかるが，V-Xgen の機能として実装することも可能．SV の場合にはボクセライズ時の内部データ構造に関係なく，直交等間隔となるので問題ないが，Octree の場合にはダミー STL を作成し，格子を細かくしておく必要がある．

#### 3.4.5.1　入力幾何形状情報

コンポーネントの幾何形状情報は**図 3.15** に示すように，圧力損失部とファンのボリュームを構成する情報となる．

- 圧力損失部

$$\left.\begin{array}{lll} W & \text{幅} & [m] \\ H & \text{高さ} & [m] \\ D & \text{厚さ} & [m] \\ \vec{n} & \text{流出方向法線ベクトル成分} & [-] \\ \vec{d} & \text{幅方向ベクトル成分} & [-] \\ O_R & \text{前面の中心座標} & [m] \end{array}\right\} \tag{3.90}$$

- ファン

$$\left.\begin{array}{lll} R_F & \text{ブレード半径} & [m] \\ R_B & \text{ボス半径} & [m] \\ D & \text{厚さ} & [m] \\ \vec{n} & \text{流出方向法線ベクトル成分} & [-] \\ O_F & \text{前面の中心座標} & [m] \end{array}\right\} \tag{3.91}$$

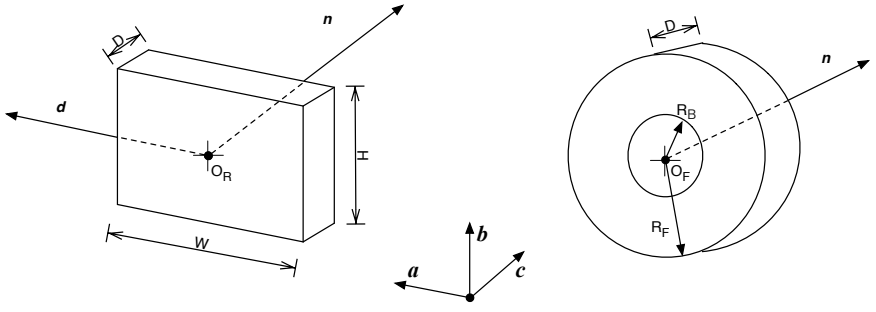



図 3.15　圧力損失部とファン部の幾何情報.

上記のジオメトリ情報から完全に閉じたボリューム形状を再構築できる．また，**図 3.15** の二つの形状は比較的簡単であり，プログラムで内外判定が可能である．

#### 3.4.5.2　占有率の計算

コンポーネントの占有率の厳密な計算は難しいため，ある精度でサンプリングすることにより近似的に体積占有率を求める．また，コンポーネントは空間内に任意の位置・方向で配置されていると処理が煩雑になる．そこでまず，コンポーネントの法線と座標の $z$ 軸が一致するようにアフィン変換した上で，内外判定を行う．矩形形状の場合には，$z$ 軸に加えて方向ベクトルも合わせる．

**■占有率計算のアルゴリズム**

1. まず，幾何情報からジオメトリ BV のサイズを決定する．ジオメトリ BV からインデクススペースの BV を計算する．矩形形状については，8 頂点の最大値最小値を求める．円筒形状の場合には，円周上にサンプリング点を発生させ，回転変換した点群に対して最大値最小値を求める．
2. 座標変換を行う．
3. インデクス BV を探索範囲として，セルを構成する 8 つのノードに対して，内外判定を行う．内側であれば 1.0，外側の場合には 0.0 として，8 頂点分を加算する．コンポーネント内部は 1.0，外部は 0.0，部分的なセルは中間の値をとる．
4. 中間値をとるセルに対しては，サブサンプリングを行う．サブサンプリングは分割の基数を n とすると，O(表面積)/O (体積)=$\frac{1}{n}$ 程度の近似精度と考えることができるので，$n = 50$ 程度にすると，8 ビット幅で表現可能[*14]．

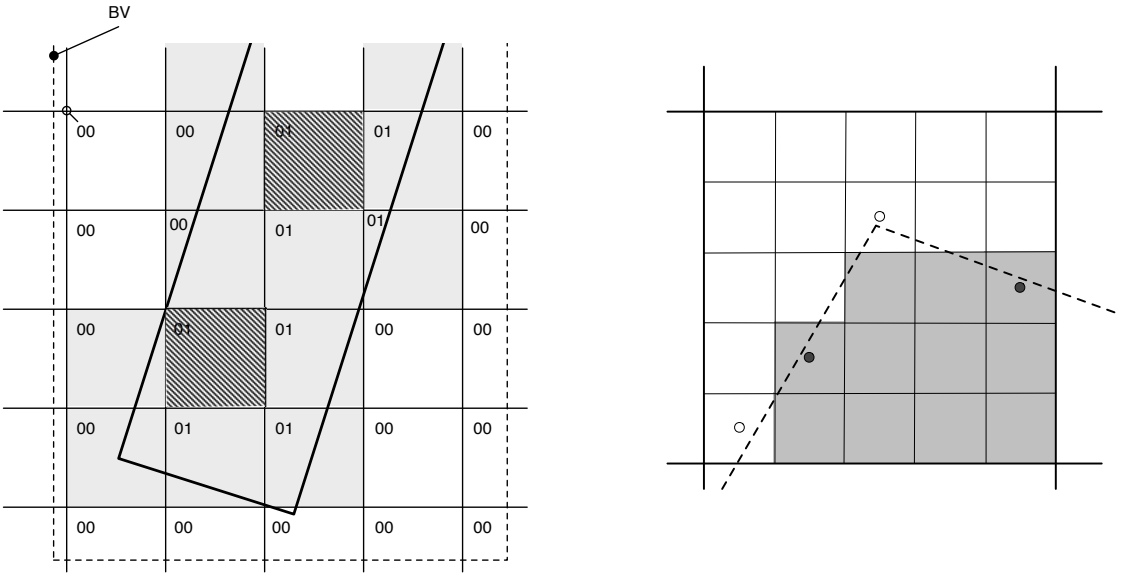



図 3.16　占有率の計算は，ノードの暫定的な体積率から，サブサンプリングする要素を特定する．グレー部分がサブサンプリング対象で，ハッチング部分は完全に内部にあるセル．サブサンプリングでは，セルセンタ位置でサンプリングを行い，占有率を計算する．右図は二次元の場合で n=5，11/25 の占有率となる．

**■アフィン変換**　コンポーネントの形状は，空間内の任意の位置に存在する．また，その配置は格子に沿うとは限らない．そこで内外判定を簡単にするため，コンポーネントの法線 $\vec{n}$ と座標の $z$ 軸方向の単位ベクトルが一致するように，コンポーネント BV の矩形領域をアフィン変換し，マッピングする．さらに，点 $O_R$ と原点を一致させるように平行移動する．この座標変換により，コンポーネントの BV は**図 3.17** のようになる．

1. Appendix に示すアフィン変換の準備のために，**図 3.15** の局所座標の基点を原点に平行移動し，**図 3.17** の右図のように配置する．コンポーネントの局所座標の取り方は，**図 3.15** の ($\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$) のように取る．まず，法線ベク

[*14] 計算時にも量子化した状態から float に変換して計算．ロードを減らせる

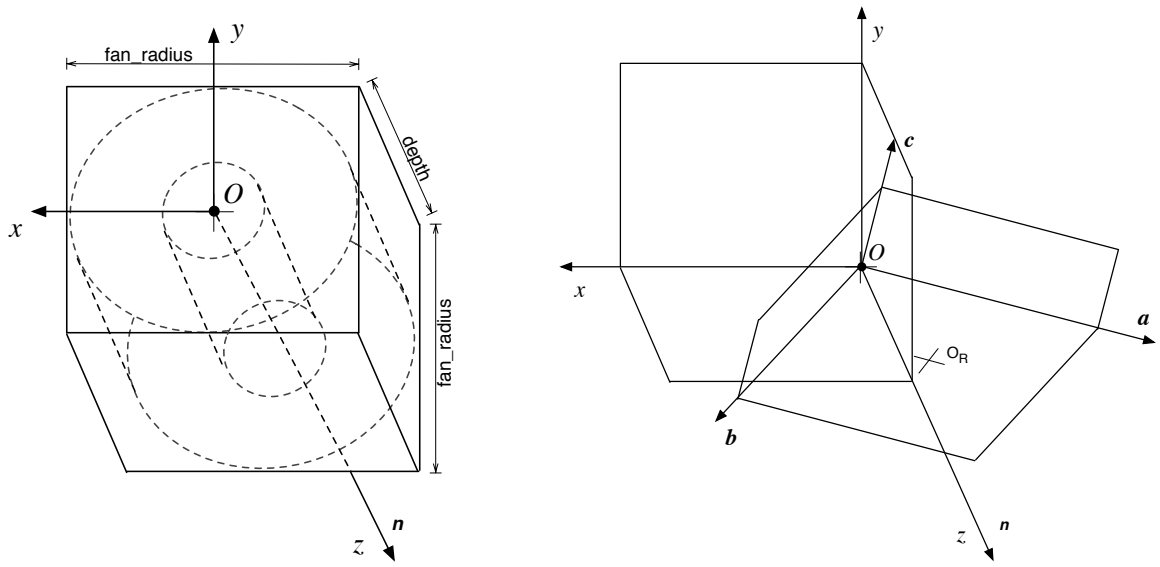



図 3.17　コンポーネントのアフィン変換．任意の配置から，単位立方体へマッピングする．局所基底の取り方は，流出方向が $\boldsymbol{c}$ となるようにする．

トル $\boldsymbol{c}$ を流出方向にとり，それと直交する補助ベクトル $\boldsymbol{a}$, $\boldsymbol{b}$ を右手系になるように取る．

2. このとき，変換行列の各成分は**式 (??)** となる．
3. $O_R$ を原点に重ねるように平行移動し，**図 3.17** の左図になるようにする．

**■内外判定**　内外判定は，矩形領域の場合には単位立方体内にあるかどうかの判定，ファンの場合には $xy$ 平面でブレード部にあるかどうかを見て，奥行き方向の範囲に収まっているかで判定する．

**■サブサンプリング**　対象となるセルに対して，$n^3$ のサブセルを想定し，サブセルのセルセンター位置でサンプリングを実行する．オリジナルの空間座標でサンプリング点を求め，それを座標変換して内外判定を行う．

## 3.5　Immersed Boundary

Immersed Boundary Method の中で Direct Forcing として知られる方法 [16] を実装する．これは，指定した速度成分を指定セルに与える境界条件である．Navier-Stokes 方程式に強制力を表す外力項 $F_{DFi}$ を導入する．

$$\begin{aligned} \frac{\partial u_i}{\partial t}^{n+1} &= H_i - \frac{\partial p}{\partial x_i}^{n+1} + F_{DFi} \\ H_i &= -\frac{\partial}{\partial x_j}\left(u_i u_j\right) + \frac{1}{Re}\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \end{aligned} \tag{3.92}$$

$F_{DFi}$ は ${u_i}^{n+1} = u_i^T$ を満足するように与えられる，つまり，

$$F_{DFi} = -H_i + \frac{\partial p}{\partial x_i}^{n+1} + \frac{u_i^T - u_i^n}{\Delta t} \tag{3.93}$$

これは，速度ベクトルに，次ステップで指定したい速度 ${u_i}^T$ を代入することで実装できる．この外力の導入により，強制した部分の連続の式にソース項 $S_{DF}$ が同時に導入される．したがって，Poisson 反復の収束判断には速度の発散値の指標を用いない方が良い．ただし，$S_{DF}$ の値は経験上，収束閾値よりも大きいが小さな値にとどまることが多い．

$$\frac{\partial u_i}{\partial x_i}^{n+1} = S_{DF} \tag{3.94}$$

Fractional Step 法による解法の手順を示すと，

1. Pseudo vector

$$\begin{aligned} \frac{{u_i}^* - {u_i}^n}{\Delta t} &= {H_i}^n && (1st\ Order) \\ \frac{{u_i}^* - {u_i}^n}{\Delta t} &= \frac{3}{2}{H_i}^n - {H_i}^{n-1} && (2nd\ Order) \end{aligned} \tag{3.95}$$

2. Poisson

$$\frac{\partial}{\partial x_i}\frac{\partial P}{\partial x_i}^{n+1} = \frac{1}{\Delta t}\frac{\partial {u_i}^*}{\partial x_i} \tag{3.96}$$

3. Projection

$$\frac{{u_i}^{**} - {u_i}^*}{\Delta t} = -\frac{\partial}{\partial x_i}\frac{\partial P}{\partial x_i}^{n+1} \tag{3.97}$$

4. Forcing

$$\begin{aligned} \frac{{u_i}^{n+1} - {u_i}^{**}}{\Delta t} &= F_{DFi} \\ F_{DFi} &= \frac{{u_i}^T - {u_i}^{**}}{\Delta t} \end{aligned} \tag{3.98}$$

このステップでは，指定領域内部の速度定義点に所望の速度 ${u_i}^T$ を強制する．

# 第 4 章

# 線形ソルバー

本章では，圧力の Poisson 方程式および運動方程式の時間積分に現れる連立一次方程式の解法について述べる．

## 4.1　圧力の Poisson 方程式

### 4.1.1　反復アルゴリズム

FFVC で扱う圧力の Poisson 方程式は，次のような形式となる．

$$\nabla\left(\nabla p^{n+1}\right) = \frac{1}{\Delta t} div\left(\boldsymbol{u}^{*}\right) + div(\beta \boldsymbol{F}) \tag{4.1}$$

ここで，$\beta$ は体積率，$\boldsymbol{F}$ は外力ベクトルである．

ノイマン条件とディリクレ条件のフラグを導入し，対角項が 1 となるようにスケーリングして，連立一次方程式 $Ax = b$ を得る．ここで $Face$ はセルを構成する面，$\boldsymbol{n}$ は各面の外側法線ベクトルとする．

$$\underbrace{p - \frac{1}{\sum_{l}^{Face} \phi_l^N} \sum_{l}^{Face} \left(p\, \phi^D\, \phi^N\right)_l}_{Ax} = \underbrace{-\frac{1}{\sum_{l}^{Face} \phi_l^N} \left\{ \frac{\Delta x}{\Delta t} \sum_{l}^{Face} \left(\boldsymbol{u}_l^{*} \cdot \boldsymbol{n}_l\right) + \Delta x \sum_{l}^{Face} (\beta \boldsymbol{F}_l \cdot \boldsymbol{n}_l) \right\}}_{b} \tag{4.2}$$

右辺ベクトルの第一項めは境界条件も含む．

Fractional step 法を基本に，$\boldsymbol{u}^{n+1}$ に基づく体積力を圧力反復過程に組み込む場合，収束判定は圧力 Poisson 方程式の収束と速度-圧力反復過程での非圧縮条件の 2 つを考える．アルゴリズムを 1 に示す．

### 4.1.2　ノルム

ノルムの定義は，下記とする．

$$\begin{array}{ll} L_2 & \|x\|_p = \left( \sum_{p=0}^{n} |x_i|^p \right)^{\frac{1}{p}} \\ Maximum & \|x\|_\infty = max|x_i| \end{array} \tag{4.3}$$

利用可能なノルムは，次の 3 種類である．

| $Type$ | $Norm$ |
|---|---|
| $\Delta x/b$ | $\frac{\|x^{m+1} - x^m\|_2}{\|b\|_2}$ |
| $r/b$ | $\frac{\|r\|_2}{\|b\|_2}$ |
| $r/r_0$ | $\frac{\|r\|_2}{\|r_0\|_2}$ |

(4.4)

## 4.2　GMRES

GMRES(Generalized Minimum Residual) は，大型疎行列を係数行列にもつ連立一次方程式 $Ax = b$ の解法アルゴリズムで，Krylov 部分空間法に分類される．

m 次の Krylov 部分空間は

$$K_m = span\{v^{(1)}, Av^{(1)}, A^2v^{(1)}, ..., A^{m-1}v^{(1)}\} \tag{4.5}$$

**Algorithm 1** Iterative procedure in Fractional step method.

Increment state : $\boldsymbol{u}^n \leftarrow \boldsymbol{u}^{n-1}, \quad p^n \leftarrow p^{n-1}$

Evaluate space term : $\boldsymbol{w} \leftarrow \dfrac{1}{Re}\dfrac{\partial^2 \boldsymbol{u}^n}{\partial x^2} - \dfrac{\partial}{\partial x}(\boldsymbol{u}\boldsymbol{u})^n$

Time integration : $\boldsymbol{u}^* = \boldsymbol{u}^n + \Delta t\, \boldsymbol{w}$

External force : $\boldsymbol{u}^* += \Delta t \boldsymbol{f}$

Boundary condition for $\boldsymbol{u}^*$

Set initial value for iteration : $\boldsymbol{u}^{(0)} \leftarrow \boldsymbol{u}^*$

Calculate a part of non-iterative source : $\boldsymbol{s}_0 \leftarrow \sum_l^{Face} \left(\boldsymbol{u}_l^* \cdot \boldsymbol{n}_l\right)$

Calculate a part of iterative source : $\boldsymbol{s}_1 \leftarrow \sum_l^{Face} \left(\beta \boldsymbol{F}_l^n \cdot \boldsymbol{n}_l\right)$

Initial RHS vector of Poisson : $\boldsymbol{b}^{(0)} \leftarrow -\dfrac{1}{\sum_l \phi_l^N}\left(\dfrac{\Delta x}{\Delta t}\, \boldsymbol{s}_0 + \Delta x\, \boldsymbol{s}_1\right)$

Calculate $L_2$ norm of $\boldsymbol{b}^{(0)}$ : $\|\boldsymbol{b}^{(0)}\|_2$

Calculate initial residual : $\boldsymbol{r}^{(0)} \leftarrow \boldsymbol{b}^{(0)} - A\boldsymbol{x}^{(0)}, \quad \boldsymbol{x}^{(0)} = \{p_0^n,\ p_1^n,\ ...,\ p_{imax\times jmax\times kmax}^n\}^T$

Take $L_2$ norm of $\boldsymbol{r}^{(0)}$ : $\|\boldsymbol{r}^{(0)}\|_2$

**for** $m$ = 0 to VP-IterMax **do**

  $\boldsymbol{x}^{(k)} \leftarrow \boldsymbol{x}^{(m)}$

  $\boldsymbol{b}^{(k)} \leftarrow \boldsymbol{b}^{(m)}$

  **for** $k$ = 0 to P-IterMax **do**

    Solve $A\boldsymbol{x}^{(k+1)} = \boldsymbol{b}^{(k)}$

    **if** convergence **then**

      Break k-Loop

    **end if**

  **end for**

  Update velocity : $\boldsymbol{u}^{(m+1)} \leftarrow \boldsymbol{u}^* - \Delta t\, \nabla p^{(m+1)}$

  Calculate a part of divergence : $\boldsymbol{dv} \leftarrow \sum_l^{Face} \left(\boldsymbol{u}_l^{(m+1)} \cdot \boldsymbol{n}_l\right)$

  Calculate a part of iterative source term : $\boldsymbol{s}_1^{(m+1)} \leftarrow \sum_l^{Face} \left(\beta \boldsymbol{F}_l^{(m+1)} \cdot \boldsymbol{n}_l\right)$

  Calculate RHS vector : $\boldsymbol{b}^{(m+1)} \leftarrow -\dfrac{1}{\sum_l \phi_l^N}\left(\dfrac{\Delta x}{\Delta t}\, \boldsymbol{s}_0 + \Delta x\, \boldsymbol{s}_1^{(m+1)}\right)$

  Calculate $L_2$ norm of $\boldsymbol{b}^{(m+1)}$ : $\|\boldsymbol{b}^{(m+1)}\|_2$

  **if** $(\boldsymbol{dv}/\Delta x < \varepsilon)$ **then**

    Break m-Loop

  **end if**

**end for**

で，$v^{(1)} = r^{(0)}/\|r^{(0)}\|$ である[*1]．GMRES 法は，Krylov 部分空間の直交基底ベクトル列を Arnoldi-modified Gram-Schmidt 法により構築し，$x^{(0)} + K_m$ の形式の全ベクトルの残差のユークリッドノルムを最小化する．各反復毎に残差ノルムが最小になるので，残差ノルムは単調に減少する特徴をもつ．

GMRES アルゴリズムを適用するためには，係数行列 $A$ は正則である必要があるが，非対称 $(A \neq A^T)$ でもよい．Arnoldi 過程に基づく長い漸化式を用いるので，GMRES 法は反復回数の増加に伴い，1 反復あたりの演算量と記憶容量が増大する問題点がある．このため，実用上「リスタート」が併用される．リスタートがない場合，GMRES 法は n

[*1] 上付きの $(i)$ を反復，下付きの $i$ を要素として表記する．

ステップ（$n \times n$ の行列サイズの場合）で収束が保証されるが，多大な記憶領域と演算量となるので，リスタートは必須である．

GMRES のアルゴリズムは modified Gram-Schmidt 直交化を用いるので，直交基底は次のようになる [17].

**Algorithm 2** Orthogonal basis function. Count i means a number of iteration.

$w^{(i)} \leftarrow Av^{(i)}$
**for** $k = 1$ to $i$ **do**
　$w^{(i)} \leftarrow w^{(i)} - (w^{(i)}, v^{(k)})\, v^{(k)}$
**end for**
$v^{(i+1)} \leftarrow w^{(i)} / \|w^{(i)}\|_2$

GMRES アルゴリズムは，残差ノルムを反復なしに計算することができる．十分に正確になるまで一連の残差ノルムを計算し，それから時間のかかる反復を行う効率的な計算ができる．$y_k$ は残差ノルムを最小化するベクトルに選ぶと，反復過程は次式になる．

$$x^{(i)} = x^{(0)} + y_1 v^{(1)} + y_2 v^{(2)} + \ldots + y_i v^{(i)} \tag{4.6}$$

アルゴリズム 2 に示すように，GMRES 法は反復回数の増加に従い，ベクトル $w^{(i)}$ の記憶容量が増加し，演算量も線形に増える．この問題点を解消するためにリスタートが用いられる．リスタートではリスタート周期 (Frequency of restart) として m 回の反復数を選び，反復を m 回で打ち切る．次の m 回の反復の初期値として前回の中間値を用いる．m の選択は一意ではなく，少なすぎると収束が遅くなり，多すぎると記憶容量と演算量が過大になる．

GMRES のアルゴリズムは，次のようになる．

1. QR 分解を用いた Hessenberg 行列 $H$ と直交基底 $V$ を与える Arnoldi 過程を計算する．
2. 残差の最小化問題を Hessenberg 行列を用いて解く際に QR 分解を行うが，それには Givens 回転を計算する．$H$ の前の要素に Givens 回転を適用し，次の新しい回転を計算，最後に最小化問題の右辺ベクトル $Z$ に適用する．
3. 必要であれば，解ベクトル $X$ を更新するため，$H$ と $Z$ を用いて最小化問題を解く．

FGMRES(Flexible GMRES) 法は，収束性を高めるために前処理を用いる方法である．前処理には SOR や GMRES 自身など様々な反復法が利用できる．

**Algorithm 3** FGMRES algorithm

m : Number of iterations between each restart (Frequency of restart)
size : Size of a matrix
Iter_Max : Upper limit of GMRES iteration
x(size) : Solution vector
b(size) : right hand side vector
w(size) : Intermediate vector used in the Arnoldi's process to compute H
r(size) : Residual vector r = b - Ax
rm(m+1) : Residual vector of the minimization problem
c(m+1) : Cosine for Givens rotations
s(m+1) : Sine for Givens rotations
h(m+1, m+1) : Matrix for the Hessenberg decomposition
v(size, m+1) : Orthogonal basis for the Krylov subspace
z(size, m+1) : Right-hand side vector for the residual minimization problem
K : Approximately inverse matrix of A
$\epsilon$ : Convergence criteria
$\varepsilon$ : Convergence criteria

Initialize: $w, c, s, r, h, v, z \leftarrow 0$
$x \leftarrow x_0$

**for** $j$ = 1 to Iter_Max **do**
  $r \leftarrow b - Ax$
  $\beta \leftarrow \|r\|$
  **if** ($\beta < \epsilon\|r\|$) **then**
    **return**
  **end if**
  $rm \leftarrow \{\beta, 0, ..., 0\}^T$
  $v^{(1)} \leftarrow r/\beta$

  **for** $i$ = 1 **to** $m$ **do**
    $z^{(i)} \leftarrow K_{i,j}^{-1} v^{(i)}$  *or*  $z^{(i)} \leftarrow v^{(i)}$ (*without preconditioning*)
    $w \leftarrow Az^{(i)}$
    **for** $k$ = 1 **to** $i$ **do**
      $h_{k,i} \leftarrow (w, v^{(k)})$
      $w \leftarrow w - h_{k,i} v^{(k)}$
    **end for**
    $h_{i+1,i} \leftarrow \|w\|$

    **if** ($h_{i+1,i} < \epsilon$) **then**
      **Warning: pseudo-converge**
      $n \leftarrow i - 1$
      **goto jump_1**
    **end if**

    $v^{(i+1)} \leftarrow w/h_{i+1,i}$

**Algorithm 4** FGMRES algorithm (continued)

```
        for k = 2 to i do
            ht ← h_{k-1,i}
            h_{k-1,i} ← c_{k-1} ht + s_{k-1} h_{k,i}
            h_{k,i} ← −s_{k-1} ht + c_{k-1} h_{k,i}
        end for

        δ ← sqrt(h_{i,i}^2 + h_{i+1,i}^2)

        if (δ < ε) then
            n ← i − 1
            goto jump_1
        end if

        c_i ← h_{i,i} / δ
        s_i ← h_{i+1,i} / δ
        h_{i,i} ← c_i h_{i,i} + s_i h_{i+1,i}

        rm_{i+1} ← −s_i r_i
        rm_i ← c_i r_i
        ρ ← |rm_{i+1}|

        if (ρ < ε||b||) then
            n ← i
            goto jump_1
        end if
    end for

    n ← m

    jump_1:

    for j = n downto 2 do
        rm_j ← rm_j / h_{j,j}
        for k = 1 to j − 1 do
            rm_k ← rm_k − rm_j h_{k,j}
        end for
    end for

    rm_1 ← rm_1 / h_{1,1}

    for i = 1 to n do
        x ← x + rm_i z^{(i)}
    end for
end for
```

# 第 5 章

# 境界条件と媒質設定

本章では，CBC ソルバークラスで設定できる境界条件と媒質情報の設定について説明する．まず境界条件と媒質を指定する XML タグの構造について述べた後，流れと熱の境界条件について説明する．3 章で説明した Dirichlet 型と Neumann 型の基本的な境界条件を用いた物理的な境界条件について述べる．また，パラメータの設定については 6 章で詳しく説明する．

## 5.1 パラメータと XML 構造

### 5.1.1 CBC ソルバークラスで指定する境界条件と媒質のパラメータ

境界条件は BC_Table セクション，媒質パラメータは Medium_Table セクションに記述する．

- BC_Table
  - InnerBoundary
  - OuterBoundary
    - Basic_BCs
    - Face_BC
- Medium_Table
  - Fluid
  - Solid

### 5.1.2 BC_Table セクションの構造と境界条件

境界条件を記述する BC_Table セクションは，SphereConfig 内に記述しても良いし，独立したファイルに記述しても良い．独立ファイルとして扱う場合には，DomainInfo セクションで次のように外部ファイルとして記述しておく．

```
<Param name="BCTBL" dtype="STRING" value="xml/boundary.xml"/>
```

value には，sphere を起動するディレクトリからの相対パスとなるように記述しておく．

BC_Table セクションは，内部境界条件 InnerBoundary と外部境界条件 OuterBoundary の 2 つを記述する．内部境界条件と外部境界条件の指定は便宜的に分離しているが，コード中では同じ方法で実装している．

#### 5.1.2.1 OuterBoundary

計算領域の外部境界条件を記述し，次の方針により指定する．

1. 候補となる境界条件の種類を Basic_BCs タグ内にリストアップし，基本リストを作成する．境界条件の基本リストには，ユニークな ID（番号）を割り振る．
2. Face_BC タグ内において，境界条件の基本リストの ID を参照し，計算領域の外部境界の各面における境界条件を指定する．

下記の例では，境界条件の候補として，ID=1, 5, 8 をリストアップしておき，X マイナス方向の外部境界面に流入出境界条件 (ID=5) を与え，Y マイナス方向の外部境界面に壁面境界条件 (ID=1) を与えている．

```
<BC_Table>
    <OuterBoundary>
      <Elem name="Basic_BCs">
        <Elem name="Wall" id="1">
          <Param name="Normal_x"      dtype="REAL"   value="0.0" />
          <Param name="Normal_y"      dtype="REAL"   value="0.0" />
          <Param name="Normal_z"      dtype="REAL"   value="0.0" />
          <param name="specified_type"  dtype="string" value="velocity" />
          <Param name="specified_value"    dtype="REAL"   value="0.0" />
          <param name="profile"    dtype="string" value="constant" />
          <param name="heat_type"  dtype="string" value="adiabatic" />
        </Elem>
        <Elem name="traction_free" id="8">
            <Param name="ambient_temperature"    dtype="REAL"   value="20.0" />
```

```
        </Elem>
        <Elem name="in_out" id="5">
          <Param name="velocity_type" dtype="STRING" value="minmax" />
          <Param name="ambient_temperature"    dtype="REAL"   value="20.0" />
        </Elem>
      </Elem>

      <Elem name="Face_BC">
        <Elem name="X_MINUS" id="5" comment="inout">
          <Param name="Guide_Cell_ID" id="1" />
        </Elem>
        <Elem name="X_PLUS"  id="5" comment="inout">
          <Param name="Guide_Cell_ID" id="1" />
        </Elem>
        <Elem name="Y_MINUS" id="1" comment="wall">
          <Param name="Guide_Cell_ID" id="2" />
        </Elem>
        <Elem name="Y_PLUS"  id="5" comment="inout">
          <Param name="Guide_Cell_ID" id="1" />
        </Elem>
        <Elem name="Z_MINUS" id="5" comment="inout">
          <Param name="Guide_Cell_ID" id="1" />
        </Elem>
        <Elem name="Z_PLUS"  id="5" comment="inout">
          <Param name="Guide_Cell_ID" id="1" />
        </Elem>
      </Elem>
    </OuterBoundary>
</BC_Table>
```

境界条件は，3 章で説明したように Dirichlet 型と Neaumann 型のプリミティブな形式で実装されるが，指定は物理的な状態により指定する．指定できる境界条件の種類を**表 5.1** に示す．熱流れの場合には，壁面境界の場合に細かい指定が可能である．

表 5.1　外部境界で指定できる流れと熱の境界条件の種類

| 流れの境界指定のキーワード | 流れの境界条件 | 熱境界指定のサブキーワード | 熱境界条件 |
|---|---|---|---|
| In_Out | 流入出境界 | Ambient_Temperature | 流入温度指定と対流流出 |
| Outflow | 流出境界 | ← | 対流流出 |
| Periodic | 周期境界 | ← | 周期境界 |
| Specified_Velocity | 流入境界 | Temperature | 流入温度指定 |
| Symmetric | 対称境界 | ← | 対称境界 |
| Traction_Free | 遠方境界 | Ambient_Temperature | 遠方温度指定 |
| Wall | 壁面境界 | Adiabatic | 断熱指定 |
| | | HeatFlux | 熱流束指定 |
| | | HeatTransfer Type_S | 熱伝達係数と表面温度から熱伝達を計算 |
| | | HeatTransfer Type_SF | 強制対流の層流・乱流熱伝達境界 |
| | | HeatTransfer Type_SN | 自然対流の乱流熱伝達境界 |
| | | HeatTransfer Type_B | 固体壁からの放熱条件 |
| | | IsoThermal | 等温指定 |

#### 5.1.2.2　InnerBoundary

計算領域内の境界条件を記述するセクションで，**表 5.2** に示す種類を指定できる．CBC ソルバークラスは，内部境界条件をコンポーネントとして扱う．

表 5.2　内部境界条件（コンポーネント）の種類

| タグ | 指定位置 | 計算モード | 実装形式 | コンポーネントの説明 |
|---|---|---|---|---|
| Specified_Velocity | セル界面 | 流れ・熱流れ | 対流流束 | 速度指定境界[*1] |
| Outflow | セル界面 | 流れ・熱流れ | 対流流束 | 流出境界 |
| Periodic | セル界面 | 流れ・熱流れ | 参照値指定 | 部分周期境界 |
| Pressure_Loss | セル要素 | 流れ・熱流れ | 外力項 | 圧力損失部 |
| Inactive | セル要素 | 流れ・熱流れ | マスク | 不活性化する計算空間内のセル ID を指定 |
| Cell_Monitor | セル要素 | 流れ・熱流れ | - | 物理量のモニター位置の指定[*2] |
| Adiabatic | セル界面 | 熱流れ | 熱流束マスク | 断熱セル指定 |
| Direct_Heat_Flux | セル界面 | 熱流れ | 熱流束 | 熱流束指定 |
| HeatTransfer_B | セル界面 | 固体伝熱 | 熱流束 | 固体壁からの放熱条件 |
| HeatTransfer_S | セル界面 | 熱流れ | 熱流束 | 熱伝達係数と表面温度により計算 |
| HeatTransfer_SF | セル界面 | 熱流れ | 熱流束 | 強制対流の層流・乱流熱伝達境界 |
| HeatTransfer_SN | セル界面 | 熱流れ | 熱流束 | 自然対流の乱流熱伝達境界 |
| IsoThermal | セル界面 | 熱流れ | 熱流束 | 等温面指定 |
| Heat_Source | セル要素 | 熱流れ | 外力項 | 吸発熱指定 |
| Specified_Temperature | セル要素 | 熱流れ | 温度指定 | セルの温度指定 |

```
<BC_Table>
  <InnerBoundary>
      <Elem name="specified_velocity" id="3" comment="inlet">
        <Param name="Normal_x"         dtype="REAL"   value="0.0" />
        <Param name="Normal_y"         dtype="REAL"   value="0.0" />
        <Param name="Normal_z"         dtype="REAL"   value="-1.0" />
        <param name="specified_type"   dtype="string" value="velocity" />
        <Param name="specified_value"  dtype="REAL"   value="2.0" />
        <Param name="def_face"         dtype="INT"    value="10" />
        <param name="profile"          dtype="string" value="harmonic" />
        <Param name="frequency"        dtype="REAL"   value="2.0" />
        <Param name="initial_phase"    dtype="REAL"   value="0.0" />
        <Param name="constant_bias"    dtype="REAL"   value="0.0" />
        <Param name="temperature"      dtype="REAL"   value="35.0" />
      </Elem>
  </InnerBoundary>
</BC_Table>
```

内部境界条件は，外部境界面を除く計算領域内において，セル ID により指定する．同時に，XML の InnerBoundary セクションの ID 番号にその境界条件の属性を記述する．また，内部境界条件で指定する各コンポーネントの個数と実際の解析モデル中のコンポーネントの個数および ID 番号の属性情報は一致している必要がある．**指定できる内部境界条件の数は 30 個が上限**で，**指定境界条件数と媒質数の和は 63 個以下**となる[*1]．Cell_Monitor は境界条件ではないが，同じ仕組みを用いて実装しているのでこのセクションに設けている．

一方，外部境界条件は，計算領域を構成する外部境界面の各面で同じ境界条件とし，XML パラメータにより指定する．

[*1] INFLOW セクションのステートメントで Temperature を指定する．

[*2] Monitor は境界条件ではないが，同じしくみを用いて実装しているので，このセクションに設けている．

[*1] これらの制限は，境界条件を効率よく実装する方法の制約から来るもの．

### 5.1.3　Medium_Table セクションの構造

Medium_Table セクションには，**表 5.3** に示すように媒質 ID の物性値を記述できる．ここで記述する媒質の基本リストは，解析に利用される候補である．各媒質は，固体と流体によって記述しなければならない物性値が異なり，指定できる項目を**表 5.4** に示す．

```
<Medium_Table>
  <Elem name="Fluid" id="1" comment="VirtualFluid">
    <Param name="density" dtype="REAL" value="1.0" />
    <Param name="specific_heat" dtype="REAL" value="1.0" />
    <Param name="thermal_conductivity" dtype="REAL" value="1.0" />
    <Param name="kinematic_viscosity" dtype="REAL" value="1.0" />
    <Param name="viscosity" dtype="REAL" value="1.0" />
    <Param name="sound_of_speed" dtype="REAL" value="1.0" />
    <Param name="volume_expansion" dtype="REAL" value="0.04e-3" />
  </Elem>
  <Elem name="Solid" id="600" comment="Fe">
    <Param name="density" dtype="REAL" value="7870.0" />
    <Param name="specific_heat" dtype="REAL" value="442.0" />
    <Param name="thermal_conductivity" dtype="REAL" value="80.3" />
  </Elem>
</Medium_Table>
```

表 5.3　Medium_Table に記述するパラメータ

| XML キーワード | 説明 |
|---|---|
| Elem name | Fluid または Solid |
| ID | 媒質 ID |
| commnet | 媒質名 |

表 5.4　Medium_Table における物性値の指定

| Fluid のキーワード | 説明 | 単位 |
|---|---|---|
| Density | 密度 | $[kg/m^3]$ |
| Specific_Heat | 定圧比熱 | $[kJ/(kgK)]$ |
| Thermal_Conductivity | 熱伝導率 | $[W/(mK)]$ |
| Kinematic_Viscosity | 動粘性係数 | $[m^2/s]$ |
| Viscosity | 粘性係数 | $[Pa/s]$ |
| Sound_of_Speed | 音速 | $[m/s]$ |
| Volume_Expansion | 体膨張率 | $[1/K]$ |

| Solid のキーワード | 説明 | 単位 |
|---|---|---|
| Density | 密度 | $[kg/m^3]$ |
| Specific_Heat | 定圧比熱 | $[kJ/(kgK)]$ |
| Thermal_Conductivity | 熱伝導率 | $[W/(mK)]$ |

## 5.2　流れ解析の境界条件

本節では，非圧縮性流体の流れの境界条件，つまり速度と圧力の境界条件について説明する．計算領域とコロケート変数配置の変数のインデクスの表記を**図 5.1**，**図 5.2** に示す．


図 5.1　計算領域のインデクス

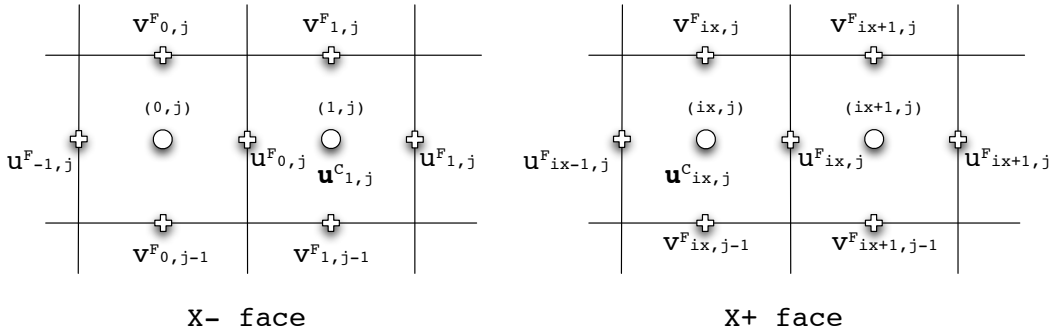



図 5.2　コロケート配置の変数のインデクス．基本変数 ($u_i^C$, $p$, $\theta$) は全てセルセンタ位置に配置され，補助的な速度ベクトル $u_i^F$ がスタガード位置に配置される．

### 5.2.1　流れの境界条件

#### 5.2.1.1　壁面境界

**■外部境界面**　壁面境界条件は，外部境界条件の場合，XML パラメータにより指定する．一方，計算領域内部に存在する壁面はセル ID により指定され，その境界条件は Embedded Boundary Condition Scheme(**??**) の仕組みを用いてスキームに組み込まれる．

外部境界は，壁面速度が時間的に変化する場合と固定の場合があり，指定する境界面の移動速度を与える．ただし，壁面速度ベクトルは壁面と平行なスライド成分のみで，壁面と垂直な成分はゼロである点に注意する．

壁面境界は，Dirichlet 型の速度境界条件を用いてセルフェイス位置の運動量流束を直接与える．外部境界では下記のような XML 入力パラメータで指定する．次の例では，境界条件番号 1 に z 軸に面直で y 方向に 7[m/s] で移動する壁の境界条件を指定している．

```
<OuterBoundary>
  <Elem name="Basic_BCs">
    <Elem name="wall" id="1">
      <Param name="Normal_x"        dtype="REAL"   value="0.0" />
      <Param name="Normal_y"        dtype="REAL"   value="1.0" />
      <Param name="Normal_z"        dtype="REAL"   value="0.0" />
      <param name="Profile"         dtype="STRING" value="Harmonic" />
      <param name="Specified_Type"  dtype="STRING" value="Velocity" />
      <Param name="Specified_Value" dtype="REAL"   value="7.0" />
      <Param name="Frequency"       dtype="REAL"   value="2.0" />
      <Param name="Initial_Phase"   dtype="REAL"   value="0.0" />
      <Param name="Constant_Bias"   dtype="REAL"   value="0.0" />
    </Elem>
```

```
  </Elem>
</OuterBoundary>
```

時間的に変化しない壁面境界の場合には type=constant を指定する．時間変化を伴う速度指定は type=harmonic_oscillation を指定し，**式 (5.1)** の形式の単振動の境界条件を周期や初期位相，切片（初期振幅）と供に与える[*2]．非定常の場合，振幅は velocity で与えられる．

$$v = A \sin(2\pi f t + \phi) + b \tag{5.1}$$

ここで各パラメータは**表 5.5** に対応する．

表 5.5　速度境界条件の単振動パラメータ

| XML キーワード | **式 (5.1)** 中の記号 | 説明 | 単位 |
|---|---|---|---|
| Amplitude | $A$ | 振幅 | $[m/s]$ |
| Frequency | $f$ | 周波数 | $[Hz]$ |
| Initial_Phase | $\phi$ | 初期位相 | [rad] |
| Intercept | $b$ | 切片 | $[m/s]$ |

計算領域内部の壁面境界条件は，EBCS の仕組みを用いてスキームに組み込まれ，セル界面の流束が指定する壁面速度から直接計算される．したがって，計算する流体セルからみて，壁面方向に延びるステンシルのセルの値は参照されず不要である．外部境界の場合には，可視化する場合の利便性を考慮し，出力時にのみガイドセルに値を書き出す．この場合，境界面を中心に速度勾配一定として，速度成分を外挿する．

$$\left.\begin{array}{lll}
x-; & u_{0,j,k} = 2u^g - u_{1,j,k} \\
& v_{0,j,k} = 2v^g - v_{1,j,k} \\
& w_{0,j,k} = 2w^g - w_{1,j,k} \\
x+; & u_{ix+1,j,k} = 2u^g - u_{ix,j,k} \\
& v_{ix+1,j,k} = 2v^g - v_{ix,j,k} \\
& w_{ix+1,j,k} = 2w^g - w_{ix,j,k} \\
y-; & u_{i,0,k} = 2u^g - u_{i,1,k} \\
& v_{i,0,k} = 2v^g - v_{i,1,k} \\
& w_{i,0,k} = 2w^g - w_{i,1,k} \\
y+; & u_{i,jx+1,k} = 2u^g - u_{i,jx,k} \\
& v_{i,jx+1,k} = 2v^g - v_{i,jx,k} \\
& w_{i,jx+1,k} = 2w^g - w_{i,jx,k} \\
z-; & u_{i,j,0} = 2u^g - u_{i,j,1} \\
& v_{i,j,0} = 2v^g - v_{i,j,1} \\
& w_{i,j,0} = 2w^g - w_{i,j,1} \\
z-; & u_{i,j,kx+1} = 2u^g - u_{i,j,kx} \\
& v_{i,j,kx+1} = 2v^g - v_{i,j,kx} \\
& w_{i,j,kx+1} = 2w^g - w_{i,j,kx}
\end{array}\right\} \tag{5.2}$$

[*2] XML の例ではコメントアウトされている．

壁面境界に対する圧力の境界条件は，Navier-Stokes 方程式から Neumann 型の圧力境界条件が得られる．つまり，

$$\frac{\partial p}{\partial x_i}^{n+1} = -\frac{\partial u_i}{\partial t}^{n+1} - \left(u_j - u_j^g\right)\frac{\partial u_i}{\partial x_j} + \frac{1}{Re}\left(\frac{\partial^2 u_i}{\partial x_j^2}\right) \tag{5.3}$$

右辺の空間項は，**2.3** の各アルゴリズムに応じたタイムレベルで評価する．レイノルズ数が小さい場合には粘性項の影響が無視できないが，高レイノルズ数流れにおいては，粘性項の寄与が小さいと仮定し粘性項を省略する場合もある．

$$\frac{\partial p}{\partial x_i}^{n+1} = -\frac{\partial u_i}{\partial t}^{n+1} - \left(u_j - u_j^g\right)\frac{\partial u_i}{\partial x_j} \tag{5.4}$$

**式 (5.3)**，**(5.4)** は，壁面の時間的・空間的な変化がない場合，

$$\frac{\partial p}{\partial x_i}^{n+1} = \frac{1}{Re}\left(\frac{\partial^2 u_i}{\partial x_i^2}\right) \tag{5.5}$$

$$\frac{\partial p}{\partial x_i}^{n+1} = 0 \tag{5.6}$$

と簡単な形式になる．CBC ソルバークラスでは，**(5.6)** の壁面境界条件を用いている．

**■内部領域**　計算領域内部にある固体部分は，任意に現れる特徴があるので，その表面位置で粘着条件が課されるように対流項や粘性項のスキーム中で処理を行う．セル界面が固体壁に接するかどうかは，セルの状態を表すビットフィールドからマスクを生成し，その情報を元に流束を計算する．

#### 5.2.1.2 対称境界

外部境界にのみ用いられる境界条件で，指定する面が対称面であると仮定する．速度については，面直な成分のみ固体壁と同じで，残りはフリーとする．圧力は勾配がゼロとする．

$$
\left.
\begin{array}{lll}
x-; & u_{0,j,k} = 2u^g - u_{1,j,k} \\
& v_{0,j,k} = v_{1,j,k} \\
& w_{0,j,k} = w_{1,j,k} \\
& p_{0,j,k} = p_{1,j,k} \\
x+; & u_{ix+1,j,k} = 2u^g - u_{ix,j,k} \\
& v_{ix+1,j,k} = v_{ix,j,k} \\
& w_{ix+1,j,k} = w_{ix,j,k} \\
& p_{ix+1,j,k} = p_{ix,j,k} \\
y-; & u_{i,0,k} = u_{i,1,k} \\
& v_{i,0,k} = 2v^g - v_{i,1,k} \\
& w_{i,0,k} = w_{i,1,k} \\
& p_{i,0,k} = p_{i,1,k} \\
y+; & u_{i,jx+1,k} = u_{i,jx,k} \\
& v_{i,jx+1,k} = 2v^g - v_{i,jx,k} \\
& w_{i,jx+1,k} = w_{i,jx,k} \\
& p_{i,jx+1,k} = p_{i,jx,k} \\
z-; & u_{i,j,0} = u_{i,j,1} \\
& v_{i,j,0} = v_{i,j,1} \\
& w_{i,j,0} = 2w^g - w_{i,j,1} \\
& p_{i,j,0} = p_{i,j,1} \\
z-; & u_{i,j,kx+1} = u_{i,j,kx} \\
& v_{i,j,kx+1} = v_{i,j,kx} \\
& w_{i,j,kx+1} = 2w^g - w_{i,j,kx} \\
& p_{i,j,kx+1} = p_{i,j,kx}
\end{array}
\right\} \tag{5.7}
$$

次の例では，境界条件番号 20 に対称境界条件を設定する．

```
<OuterBoundary>
  <Elem name="Basic_BCs">
    <Elem name="symmetric" id="20"/>
  </Elem>
</OuterBoundary>
```

#### 5.2.1.3　流出境界

**■外部境界面**　流出境界を指定する場合には，流出方向は既知であり，外部境界面を指定することにより流出方向は決まる．**図 5.3** は X+ 方向の流出境界を示している．セル $(ix, j)$，セル $(ix+1, j)$ ともに流体を指定する．つまり，流出方向の外部領域は流体であるので，ガイドセル領域の ID を Face_BC セクションで流体要素にしておく．次の例では，id=20 は流体要素である．

```
<Elem name="Face_BC">
  <Elem comment="outflow" id="3" name="X_plus">
    <Param id="20" name="Guide_Cell_ID"/>
  </Elem>
</Elem>
```

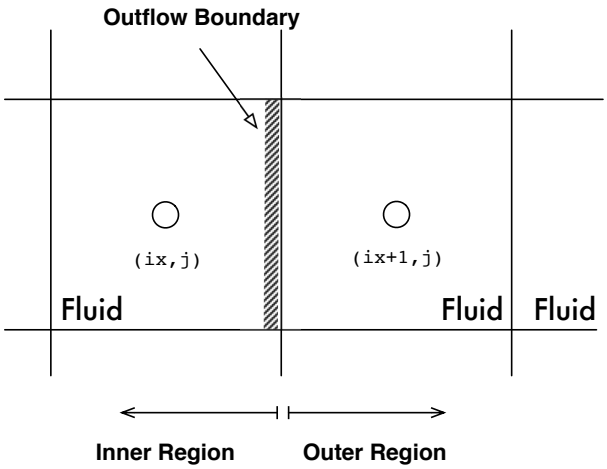



図 5.3　X+ 方向の流出境界の設定.

次の例では，基本条件として 3 番に流出境界を登録している．

```
<OuterBoundary>
  <Elem name="Basic_BCs">
    <Elem name="outflow" id="3">
      <Param name="velocity_type" dtype="STRING" value="minmax" />
    </Elem>
  </Elem>
</OuterBoundary>
```

流出境界面における流束は，境界面における流出速度から直接運動量を計算する．**図 5.3** の例は $x$ 方向の流出面なので対流速度は $x$ 方向成分，非移流物理量を $\varphi$ として，

$$\left(\tilde{f}_i\right)_{i+1/2,\,j,\,k} = \left(u_1 - u_1^g\right)_{i+1/2,\,j,\,k} \varphi_{i,\,j,\,k} \tag{5.8}$$

流出速度 $u_1$ は，セル $(i, j, k)$ に対する連続の式から求める[*3]．流出速度は，コンポーネント毎の平均値もしくは最大値と最小値の算術平均を計算している．流出速度の選び方として，流出断面の平均速度や最大値と最小値の算術平均などが提案されており，外部流や噴流のような無限空間の境界面の場合には，**式 (5.9)** の MinMax がよい近似値となる．流出速度の指定は，Outflow_Velocity タグにて average または minmax を指定する．Average

[*3] BCvec_cc.f90 > vibc_c_out_cf_()

は有効セルの平均値をとる．有効セルは物体でブロックされていないセルである．MinMax は，境界面における流速の最大値と最小値の算術平均値を与える．

$$u_{ob} = \frac{1}{2}\{max(u_{ob,j,k}) + min(u_{ob,j,k})\} \tag{5.9}$$

粘性項を評価する場合には，セルフェイス面での流出速度から勾配値が次式のように計算される．

$$\left(\frac{\partial \varphi}{\partial x}\right)_{i+1/2,\,j,k} = \frac{\varphi_{i+1/2,\,j,k} - \varphi_{i,\,j,k}}{h/2} \tag{5.10}$$

コーディングでは格子幅を $h$ としているので，勾配値が同じになるように流出速度を修正する[*4]．

$$\varphi'_{i+1/2,\,j,k} = 2\varphi_{i+1/2,\,j,k} - \varphi_{i,\,j,k} \tag{5.11}$$

実装としては，流出境界が指定された外部境界面には BC Index bcv[] に id=31 がエンコードされている．したがって，id=31 の面に対して，セルセンターの変数に対して，流出境界を考慮した流束計算を行う．具体的には，流出速度が与えられるので，一次風上スキームを適用している[*5]．

**■内部領域**　計算内部領域における流出境界は，**図 5.4** のように，グレー部分，つまり $(i, j) \sim (i+1, j)$ セル間の面が流出境界として指定されている．境界面は，次のように対象となる面を 2 つの ID で挟むことにより指定する．キーとなる ID は固体（この例では id=20）を，def_face に id=210 を指定し，$i+1/2$ 面を流出境界としている．また，指定する id=20 の属性は固体で，id=210 の属性は流体であること．加えて，固体セルは 2 層必要である[*6]．

```
<InnerBoundary>
  <Elem name="outflow" id="20" comment="outflow_sec_1">
    <Param name="def_face"      dtype="INT"    value="210" />
  </Elem>
</InnerBoundary>
```

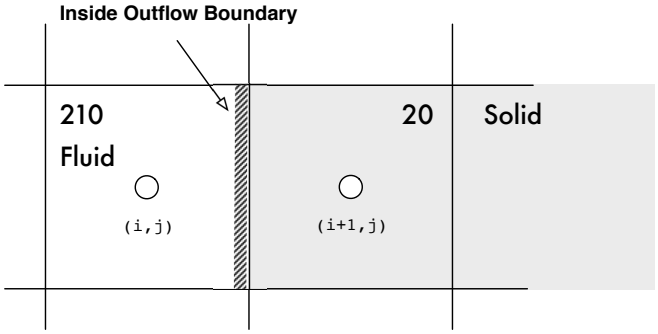



図 5.4　流出境界の設定．流出面の下流セルは固体とする．

流出速度の選び方として，経験上，内部流の場合には Average が流出速度のよい近似値を与える．圧力境界として圧力勾配ゼロを指定している．

[*4] SetBC3D::modifyViscousEE(), SetBC3D::modifyViscousCN() を参照．2013 年 3 月 16 日現在，この修正は行っていない．

[*5] cbc_pvec_vobc(), MUSCL へ変更した方が良い

[*6] これは MUSCL スキームのステンシル長からの要請であるが，実装は未だ upwind

内部境界の流出 flux の計算は，現時点で一次風上となっているので，抜けにくい傾向があるかもしれない．

さて，Fractional Step 法では，予測した疑似速度の発散に対して，連続の式を満足するように圧力場が収束していく．そこで，Poisson 方程式のソース項 $\partial u_i^*/\partial x_i$ に対して，次式のように流出境界面の指定速度が反映されるように修正している[*7]．

流出界面の流出速度には，次式の対流流出条件から時間進行した疑似速度ベクトルを用いる[*8]．

$$\frac{\partial u_i}{\partial t}^* + (u_1)_{i+1/2}\frac{\partial u_i}{\partial x}^n = 0 \tag{5.12}$$

n+1 ステップの速度を計算した後，流出セルフェイス面の速度を修正する．この修正されたセルフェイス速度を用いて，$div(\boldsymbol{u})^{n+1}$ が評価され，収束判定に用いられる[*9]．

$$\frac{\partial u_i}{\partial x_i}^* = u^n_{i+1/2} - u^*_{i-1/2} + v^*_{j+1/2} - v^*_{j-1/2} + w^*_{k+1/2} - w^*_{k-1/2} \tag{5.13}$$

上記のように連続の式から逆に流出境界面速度を求める．しかしながらこの方法では，1 つのセルに 2 つ以上の流出面は設定できない制限が生じる点に注意する．

[*7] SetBC3D::mod_Psrc_VBC(), cbc_div_ibc_() を参照.

[*8] SetBC3D::mod_Pvec_CF(), cbc_vibc_outflow_cf_()

[*9] 収束判定のノルムの種類に $div\,(\mathrm{u})^{n+1}$ が指定されている場合．SetBC3D::modifyCFVelocity() を参照.

#### 5.2.1.4　速度指定境界

速度指定境界は，流入境界として利用される．

**■外部境界面**　流入境界を外部境界面に与える例を示す．

```
<OuterBoundary>
  <Elem id="2" name="Specified_Velocity">
    <Param dtype="REAL"   name="Normal_X"         value="1.0"/>
    <Param dtype="REAL"   name="Normal_Y"         value="0.0"/>
    <Param dtype="REAL"   name="Normal_Z"         value="0.0"/>
    <Param dtype="STRING" name="Specified_Type"   value="Velocity"/>
    <Param dtype="STRING" name="Profile"          value="Constant"/>
    <Param dtype="REAL"   name="Specified_Value" value="5.0"/>
  </Elem>
</OuterBoundary>
```

**図 5.5** において，$w = i - 1/2$ 面に速度 $u_{1,\mathrm{in}}$ を指定する場合を考える．セル $(i, j, k)$ に対する流入条件なので $w$ 面の指定速度は $u_{1,\mathrm{in}} > 0$ になる．

$$\left(\tilde{f}_i\right)_{i-1/2,\,j,k} = u_{1,\mathrm{in}}\,\varphi_{\mathrm{in}} \tag{5.14}$$

速度の流入境界の場合 $\varphi$ は次式となる．

$$\varphi_{\mathrm{in}} = u_{i,\mathrm{in}} + u_i^g \tag{5.15}$$

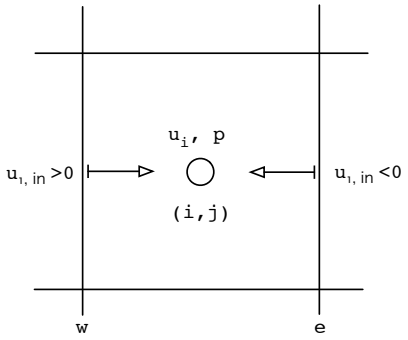



図 5.5　速度指定境界．セル (i, j) は流体セル，流入上流側の (i ± 1, j) セルはガイドセル上で流体を指定する．

速度指定境界条件は，流出条件と同様に流束型の境界条件として実装している[*10]．指定部分で与えられた速度を用いて界面の流束を計算する．このとき，一次風上を用いているので，若干振動が残る．

圧力 Poisson を解く場合の疑似速度の発散の修正は `SetBC3D::mod_Psrc_VBC()`, `cbc_div_obc_vspec_()` で処理される．また，セルフェイス速度は `SetBC3D::mod\_Vec\_CF()`, `cbc\_vobc\_drchlt\_cf\_()` で計算する．

圧力の境界条件は，壁面境界条件と同様に，**式 (5.3)** の Navier-Stokes 方程式から Neumann 型の圧力境界条件が得られる．流入境界における指定速度を $u_{i,in}$ とすると，

$$\left(\frac{\partial p}{\partial x_i}\right)_{in}^{n+1} = -\left(\frac{\partial u_i}{\partial t}\right)_{in}^{n+1} - \left(u_{j,in} - u_j^g\right)\left(\frac{\partial u_i}{\partial x_j}\right)_{in} + \frac{1}{Re}\left(\frac{\partial^2 u_i}{\partial x_j^2}\right)_{in} \tag{5.16}$$

高レイノルズ数流れにおいては，粘性項の寄与が小さいと仮定し粘性項を省略している．

[*10] SetBC3D::mod_Pvec_Flux(), cbc_pvec_vobc_()

**■内部領域**　指定方法として，Dirichlet 型の速度境界条件を用いて，セルフェイス位置の運動量流束を直接与える．計算内部領域における流入境界の XML 入力パラメータは下記のように指定する．

```
<InnerBoundary>
  <Elem name="specified_velocity" id="3" comment="inlet">
    <Param name="Normal_x"        dtype="REAL"   value="0.0" />
    <Param name="Normal_y"        dtype="REAL"   value="0.0" />
    <Param name="Normal_z"        dtype="REAL"   value="-1.0" />
    <Param name="def_face"        dtype="INT"    value="6" />
    <param name="Profile"         dtype="string" value="Harmonic" />
    <param name="Specified_Type"  dtype="string" value="Velocity" />
    <Param name="Specified_Value" dtype="REAL"   value="7.0" />
    <Param name="Frequency"       dtype="REAL"   value="2.0" />
    <Param name="Initial_Phase"   dtype="REAL"   value="0.0" />
    <Param name="Constant_Bias"   dtype="REAL"   value="0.0" />
    <Param name="Temperature"     dtype="REAL"   value="50.0" />
  </Elem>
</InnerBoundary>
```

境界面の指定方法は流出境界と同様である．つまり，流入の上流側のセルには固体属性を与える．前述の XML の指定の例では，id=6 は流体セル，id=3 は固体セルとなる．ベクトルの指定は，法線成分とベクトルの大きさで与える．時間的に変化しない場合には type=constant を指定する．時間変化を伴う速度指定は type=Harmonic を指定し，**式 (5.1)** の形式の単振動の境界条件を周期や初期位相，切片（初期振幅）とともに与える．非定常の場合，振幅は velocity で与えられる．各パラメータは**表 5.5** に対応する．

#### 5.2.1.5 **周期境界**

周期境界条件には，外部領域面に対する周期境界と計算内部領域に設定する部分的な周期境界条件を併用する条件の 2 種類を想定.

**■外部領域面**　外部領域面に対する周期境界条件では，**図 5.1** において，Inner Region の両端の境界が重なる状態を想定している．外部領域面に対する周期境界条件には**表 5.6** に示す 3 つのモードが指定できる.

表 5.6　周期境界条件のモード

| キーワード | モードの説明 |
|---|---|
| Simple_Copy | 周期境界の両端で物理量をガイドセルにコピー. |
| Directional | 圧力差を与える周期境界条件で，上流と下流の境界面を指定. |
| Driver | 計算領域内で部分的な周期境界条件を設定. |

1. Simple_Copy モード
   周期境界条件面の両端で，単純に計算内部領域の値を他方のガイドセルにコピー.
2. Directional モード
   両端で圧力差を与える周期境界条件で，速度や温度については Simple_Copy モードと同じだが，圧力は指定の圧力差を与える．上流側と下流側の設定が必要.
3. Driver モード
   乱流計算などで発達したチャネル流を上流境界として与えるためのしくみで，内部境界条件との組み合わせで利用．Driver モードの説明は内部境界条件を参照.

```
<OuterBoundary>
  <Elem name="Basic_BCs">
    <Elem name="periodic" id="7" >
      <Param name="mode" dtype="string" value="Simple_Copy" />
    </Elem>

    <Elem name="periodic" id="8" >
      <Param name="mode"                dtype="string" value="Directional" />
      <Param name="flow_direction"      dtype="string" value="upstream" />
      <Param name="pressure_difference" dtype="REAL"   value="8.148e-3" />
    </Elem>

    <Elem name="periodic" id="9" >
      <Param name="mode"                dtype="string" value="Directional" />
      <Param name="flow_direction"      dtype="string" value="downstream" />
      <Param name="pressure_difference" dtype="REAL"   value="8.148e-3" />
    </Elem>

    <Elem name="periodic" id="10" >
      <Param name="mode"             dtype="string" value="driver" />
      <Param name="driver_direction" dtype="string" value="x_minus" />
    </Elem>
  </Elem>
</OuterBoundary>
```

Directional モードでは，**表 5.7** に示すパラメータが必要で，Pressure_Difference の値が，Upstream と Downstream で同じ値である必要がある.

表 5.7　Directional モードに必要なパラメータ

| 必要なキーワード | パラメータの説明 |
|---|---|
| Pressure_Difference | 両端にかける圧力差 [$Pa$] |
| Flow_Direction | Upstream（上流面）または Downstream（下流面） |

物理変数 $\varphi$ に対する周期境界条件は，Collocated 変数配置において，各方向について次式のように設定される．適用の制限として，周期境界上には物体と他の境界条件を設定しないこと．

$$
\left\{\begin{array}{lcl}
\varphi_{-2,j,k} &=& \varphi_{ix-2,j,k} \\
\varphi_{-1,j,k} &=& \varphi_{ix-1,j,k} \\
\varphi_{0,j,k} &=& \varphi_{ix,j,k} \\
\varphi_{ix+1,j,k} &=& \varphi_{1,j,k} \\
\varphi_{ix+2,j,k} &=& \varphi_{2,j,k}
\end{array}\right.
\quad
\left\{\begin{array}{lcl}
\varphi_{i,-2,k} &=& \varphi_{i,jx-2,k} \\
\varphi_{i,-1,k} &=& \varphi_{i,jx-1,k} \\
\varphi_{i,0,k} &=& \varphi_{i,jx,k} \\
\varphi_{i,jx+1,k} &=& \varphi_{i,1,k} \\
\varphi_{i,jx+2,k} &=& \varphi_{i,2,k}
\end{array}\right.
\quad
\left\{\begin{array}{lcl}
\varphi_{i,j,-2} &=& \varphi_{i,j,kx-2} \\
\varphi_{i,j,-1} &=& \varphi_{i,j,kx-1} \\
\varphi_{i,j,0} &=& \varphi_{i,j,kx} \\
\varphi_{i,j,kx+1} &=& \varphi_{i,j,1} \\
\varphi_{i,j,kx+2} &=& \varphi_{i,j,2}
\end{array}\right.
\tag{5.17}
$$

#### 5.2.1.6 遠方境界

トラクションフリー条件は，外部境界に対してのみ指定できる．この境界条件は，計算対象の主領域から遠方の挙動を仮定した条件で，計算外部境界において流体の内部応力の法線方向成分がゼロである仮定を用いる．この境界条件は，噴流のエントレインメントの効果などを考慮できる利点があるが，渦が流出するような境界には適用できない．内部応力テンソルを $T_{ij}$, 計算外部境界の外向き法線を $n_j$ とすると，

$$T_{ij} = -p\,\delta_{ij} + \nu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \tag{5.18}$$

$$T_{ij}\,n_j = 0 \tag{5.19}$$

例えば，X 軸の正負方向のトラクション成分を表す式は，次のように導出される．

$$\begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ T_{31} & T_{32} & T_{33} \end{bmatrix} \begin{bmatrix} \pm 1 \\ 0 \\ 0 \end{bmatrix} = \pm \begin{bmatrix} T_{11} \\ T_{21} \\ T_{31} \end{bmatrix} = \pm T_{i1} = 0 \tag{5.20}$$

したがって，各外部境界面に対する解くべき式は次のように表せる．

$$\left.\begin{array}{lll} x & ; & T_{i1} = 0 \\ y & ; & T_{i2} = 0 \\ z & ; & T_{i3} = 0 \end{array}\right\} \tag{5.21}$$

例えば，これから x 方向の式を導くと，**式 (5.22)** のようになる．

$$\left.\begin{aligned} T_{11} &= -p + 2\nu\frac{\partial u_1}{\partial x_1} \\ T_{21} &= \frac{\partial u_2}{\partial x_1} + \frac{\partial u_1}{\partial x_2} \\ T_{31} &= \frac{\partial u_3}{\partial x_1} + \frac{\partial u_1}{\partial x_3} \end{aligned}\right\} \tag{5.22}$$

**図 5.6** では，速度成分を $u$, $v$, $w$ で表し，添え字で格子インデクスを表している．各方向の式を**式 (5.28)** に示す．格子幅を $h$ として $T_{21}$ を外部境界面上で評価すると，

$$\left.\begin{aligned} &\frac{v_{1,j,k} - v_{0,j,k}}{h} + \frac{u_{1/2,j+1/2,k} - u_{1/2,j-1/2,k}}{h} \approx \frac{v_{1,j,k} - v_{0,j,k}}{h} + \frac{1}{2}\frac{u_{1/2,j+1,k} - u_{1/2,j-1,k}}{h} = 0 \\ &v_{0,j,k} = v_{1,j,k} + \frac{1}{2}\left(u_{1/2,j+1,k} - u_{1/2,j-1,k}\right) \end{aligned}\right\} \tag{5.23}$$

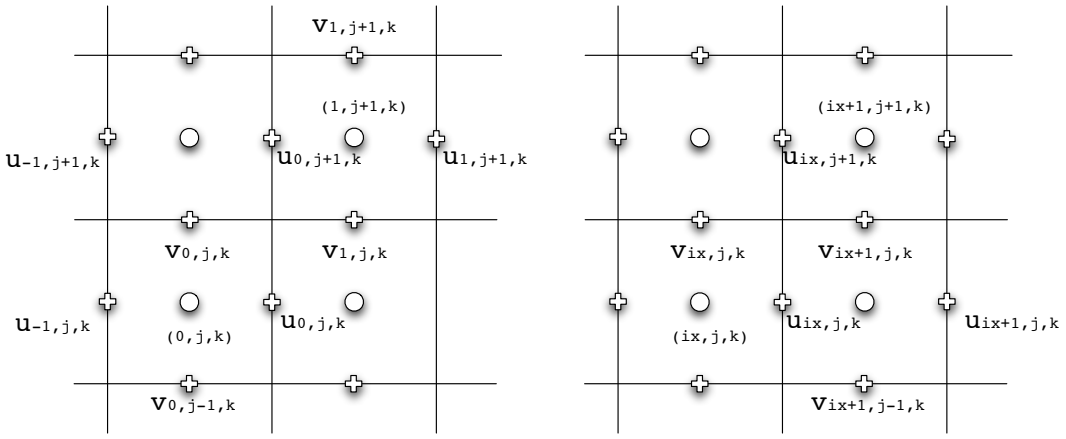
$u$ はセルセンタ位置にあり，固体セルによる影響をマスク関数 $\phi$ で表すと，

$$\phi_{1/2,j,k} = \phi_{0,j,k} \times \phi_{1,j,k} = \begin{cases} 0 & solid \\ 1 & fluid \end{cases} \tag{5.24}$$

$$u_{1/2,j\pm1,k} = (2u_{ref} - u_j)(1 - \phi) + u_{j\pm1}\phi \tag{5.25}$$

同様に，$T_{31}$ から，

$$w_{0,j,k} = w_{1,j,k} + \frac{1}{2}\left(u_{1/2,j,k+1} - u_{1/2,j,k-1}\right) \tag{5.26}$$




(a) X-方向のトラクションフリー条件.　　(b) X+ 方向のトラクションフリー条件.

図 5.6　外部境界のインデクス.

$T_{11}$ は圧力の遠方条件 $p = 0$（基準圧）を考慮して,

$$\frac{\partial u}{\partial x} = 0 \quad \mapsto \quad u_{0,j,k} = u_{1,j,k} \tag{5.27}$$

したがって，各方向の境界条件の式は以下のようになる.

$$\left.\begin{array}{lll}
x-; & v_{0,j,k} = v_{1,j,k} + u_{0,j+1,k} - u_{0,j,k} \\
& w_{0,j,k} = w_{1,j,k} + u_{0,j,k+1} - u_{0,j,k} \\
& u_{-1,j,k} = u_{0,j,k} + v_{0,j,k} - v_{0,j-1,k} + w_{0,j,k} - w_{0,j,k-1} \\
x+; & v_{ix+1,j,k} = v_{ix,j,k} - u_{ix,j+1,k} + u_{ix,j,k} \\
& w_{ix+1,j,k} = w_{ix,j,k} - u_{ix,j,k+1} + u_{ix,j,k} \\
& u_{ix+1,j,k} = u_{ix,j,k} - v_{ix+1,j,k} + v_{ix+1,j-1,k} - w_{ix+1,j,k} + w_{ix+1,j,k-1} \\
y-; & u_{i,0,k} = u_{i,1,k} + v_{i+1,0,k} - v_{i,0,k} \\
& w_{i,0,k} = w_{i,1,k} + v_{i,0,k+1} - v_{i,0,k} \\
& v_{i,-1,k} = v_{i,0,k} + u_{i,0,k} - u_{i-1,0,k} + w_{i,0,k} - w_{i,0,k-1} \\
y+; & u_{i,jx+1,k} = u_{i,jx,k} - v_{i+1,jx,k} + v_{i,jx,k} \\
& w_{i,jx+1,k} = w_{i,jx,k} - v_{i,jx,k+1} + v_{i,jx,k} \\
& v_{i,jx+1,k} = v_{i,jx,k} - u_{i,jx+1,k} + u_{i-1,jx+1,k} - w_{i,jx+1,k} + w_{i,jx+1,k-1} \\
z-; & u_{i,j,0} = u_{i,j,1} + w_{i+1,j,0} - w_{i,j,0} \\
& v_{i,j,0} = v_{i,j,1} + w_{i,j+1,0} - w_{i,j,0} \\
& w_{i,j,-1} = w_{i,j,0} + u_{i,j,0} - u_{i-1,j,0} + v_{i,j,0} - v_{i,j-1,0} \\
z-; & u_{i,j,kx+1} = u_{i,j,kx} - w_{i+1,j,kx} + w_{i,j,kx} \\
& v_{i,j,kx+1} = v_{i,j,kx} - w_{i,j+1,kx} + w_{i,j,kx} \\
& w_{i,j,kx+1} = w_{i,j,kx} - u_{i,j,kx+1} + u_{i-1,j,kx+1} - v_{i,j,kx+1} + v_{i,j-1,kx+1}
\end{array}\right\} \tag{5.28}$$

実装では，壁面上の速度成分はマスク関数でマスクし，速度をゼロにしている.

j=jx の場合に，$u_{0,jx+1,k}$ の値を参照するので，境界条件処理で値を代入しておく必要がある.

次の例では，遠方圧力に $p$ = 0[Pa_g][*11]を指定した境界条件 ID=4 に遠方境界条件を設定する．熱流れの場合には，遠方場における温度を指定.

```
<OuterBoundary>
  <Elem name="Basic_BCs">
    <Elem name="traction_free" id="4">
      <Param name="Ambient_Temperature" dtype="REAL" value="25.0" />
    </Elem>
  </Elem>
</OuterBoundary>
```

[*11] ゲージ圧，Steer > Unit > Pressure で Gauge を指定しておく.

#### 5.2.1.7　流入出境界

振動流のための特殊な境界条件である．指定された外部境界面において，境界流速の符号に応じて流出と遠方条件を切り替える．境界流速には境界面の平均流速を用いる．

下記の例では，流出境界条件に切り替わった場合の対流流出速度のパラメータを Minmax に指定.

```
<OuterBoundary>
  <Elem name="in_out" id="5">
    <Param name="velocity_type" dtype="STRING" value="minmax" />
    <Param name="Ambient_Temperature" dtype="REAL" value="25.0" />
  </Elem>
</OuterBoundary>
```

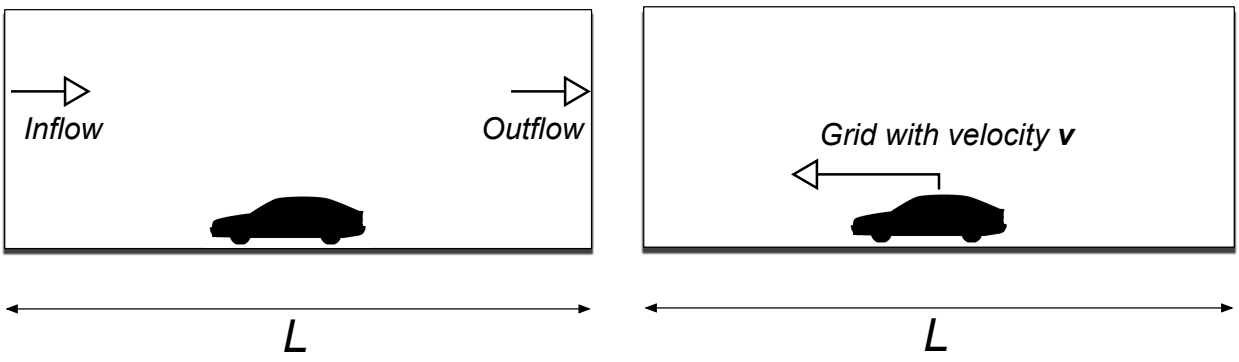
熱流れの場合には，流入時に対応する温度を指定.

### 5.2.2　静止座標系と移動座標系の場合の境界条件

外部流を考える．**図 5.7**(a) の静止座標系と図 5.7(b) の移動座標系のように異なる座標系で物体まわりの流れを計算する場合，境界条件の与え方が異なる．静止座標系は風洞実験に相当し，静止した対象物に対して風をあてる．テストセクション（この場合は計算領域）から出ていく流れが流出風に相当する．一方，移動座標系では対象物に固定した計算格子が対象物とともに静止流体中を移動する．この場合は，計算領域そのものと内部の格子が物体とともに動く．一定速度 $V$ で動いている座標系の添え字を $_M$ とし，静止した座標系の添え字を $_S$ とする．いま，静止座標系で観測される流体の速度を $u_S$ と表すと，同じ速度は移動座標系では $u_M - V$ のように観測される．つまり，

$$u_S = u_M - V \tag{5.29}$$

両者は**式 (5.29)** のガリレイ変換に対して不変である．




(a) 静止座標系　　(b) 移動座標系

図 5.7　静止座標系と移動座標系の観測点の違い

静止座標系において**図 5.7**(a) のような境界条件を与える場合，流入部では $u_0$ を与える．一方，移動座標系では静止流体の条件，つまり $u = 0$, $p = 0$ を想定し，格子速度 $V = -u_0$ を与えると両者は等価になる．

移動座標系の場合注意を要するのが，物体と地面の境界条件である．物体は移動しているので格子速度と同じである．一方，地面は静止している地面と動いている地面の二通りが考えられる．前者は風洞実験で固定地面板に相当し，後者はムービングベルトに相当する．ムービングベルトの場合には物体と格子速度だけ相対速度をもっている．したがって，

$$u_{ground} = \begin{cases} -u_0 & (Stationary\ ground) \\ 0 & (Moving\ ground) \end{cases} \tag{5.30}$$

移動格子の移動速度は Reference_Frame セクションで与える．

### 5.2.3　外力項を用いた境界条件

#### 5.2.3.1　圧力損失・利得を伴う境界条件

熱交換器やファンなどの圧力損失・利得をモデル化した境界条件について説明する．熱交換器は，圧力損失を生じる多孔質物体で，流出方向が法線で与えられる．圧力損失は通過流量（流速）と圧力損失量の関係式が与えられるものとする．一方，ファンは流量と圧力利得が関係式として与えられる．ファンの場合には旋回成分などもあるが，ここでは軸流方向のみを考える．このような流体部品のモデル指定は，セルボリュームに作用する内部境界条件として指定し，コンポーネントを用いて実装する．具体的には，**式 (5.31)** の外力項 $F_i$ として実装する．$\beta$ はセル内部におけるコンポーネントの体積占有率 (Volume Fraction; VF) である．外力項として，**表 5.8** のようなモデルが実装されている．

この境界条件に対応する指定部の平均速度・流量や圧力損失量が history_compo.log にモニタ量として書き出される．

表 5.8　セルボリュームに作用する内部境界条件

| XML キーワード | 境界条件モデル | モニター量 |
|---|---|---|
| Pressure_Loss | 圧力損失モデル | 通過風速，圧力損失量 |

$$\frac{\partial u_i}{\partial t}^{n+1} + \frac{\partial}{\partial x_j}\left(u_i u_j\right) = -\frac{\partial p}{\partial x_i}^{n+1} + (1-\beta)\frac{\partial \tau_{ij}}{\partial x_j} + \beta {F_i}^{n+1} \tag{5.31}$$

■Pressure_Loss　圧力損失モデルは，膜や熱交換器などのモデリングとして利用され，**式 (5.31)** に**式 (5.32)** の実験式を適用する．

$$F_i = -\,sgn\,(u_i)\left(\frac{\Delta p}{\Delta r}\right)^R {n_i}^R \tag{5.32}$$

ここで，$^R$ は圧力損失部を表し，$\Delta p$, $\Delta r$, $n_i$ はそれぞれ圧力損失量，圧力損失部の厚さ，法線方向を表す．圧力損失部の通過ベクトルとは逆方向に圧力損失が発生するモデルとなっている．ただし，**表 5.9** で指定するパラメータ vector が directional でない場合には，速度ベクトルは圧力損失部の流出方向には揃わない．単に，圧力損失が計算された速度ベクトルと逆向きに作用するモデルとなる．

圧力損失パラメータは，圧力損失部の試験結果により，**図 5.8** に示すような実験値が得られる．$\Delta p - V$ の性能線図を $[mmAq - m/s]$ を単位とした場合のパラメータの取得について示す．圧力損失部の圧力損失は大抵二次多項式で近似できる．**図 5.8** のグラフの読みからカーブフィットを行い，**式 (5.33)** に対応する数値 $c_1 - c_4, u_{threshold}$ を得る．ダッシュは有次元を表す．このとき，圧力損失ヘッドの単位に応じて，パラメータは無次元量に変換している．

$$h' = \begin{cases} c_1 u'^2 + c_2 u' + c_3 & (u' \geqq u'_{threshold}) \\ c_4 u'^2 & (u' < u'_{threshold}) \end{cases} \quad [mm] \tag{5.33}$$

**図 5.9** に計算パラメータの取得方法を示す．一般に，低速域のデータは得られない場合が多い．図では，○が測定結果を示し $2[m/s]$ より低速域のデータはない．そこで，測定値を元にカーブフィッティングを行い（図中の緑色の曲線），算出された係数 $c_1 = 0.12321$, $c_2 = 1.2806$, $c_3 = -1.0074$　$(2 - 10\,[m/s])$ を計算パラメータとする．この場合，$h'$ 切片がマイナスになるため，熱交換機の通過速度がゼロに近い場合に急にマイナスの圧力損失（つまり圧力利得）が発生し，実際の現象とは異なり計算上好ましくない．そこで**式 (5.33)** に示すようにある閾値で曲線を切り替える．ここでは，測定された最小速度 $u_{threshold} = 2\,[m/s]$ を閾値として，図中の C4 のカーブ

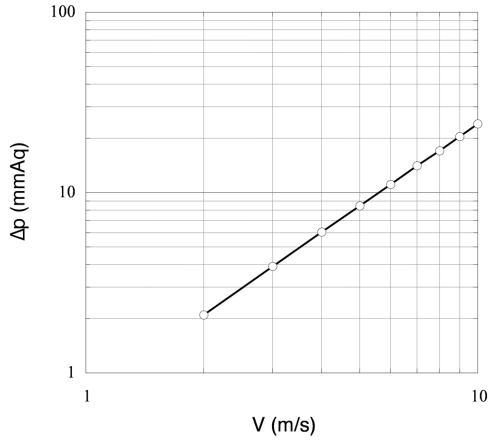



図 5.8　$\Delta p - V$ 性能線図（対数表示）

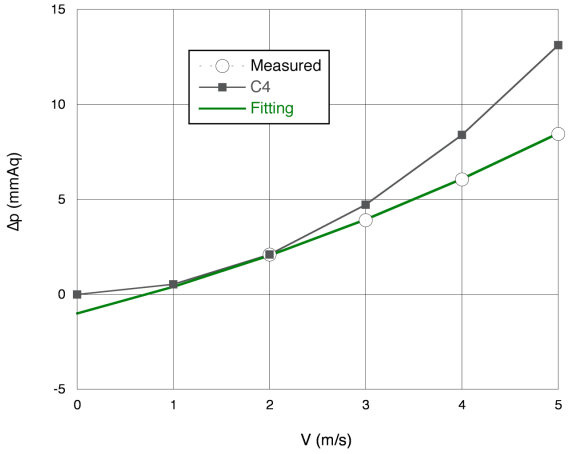



図 5.9　パラメータの取得（**図 5.8** と同じものを線形表示）

$c_4 = 0.525$　$(0 - 2\,[m/s])$ で切り替える．圧力損失部厚さは実務での観点から単位を $[mm]$ で指定するので，注意のこと．詳細については，3.4.1 を参照.

次の例では，境界条件番号 50 に圧力損失条件を設定する．ここで各パラメータは**表 5.9** に対応する．

```
<Elem name="Pressure_Loss" id="50" comment="radiator">
  <Param name="Normal_x"    dtype="REAL"   value="5.0" />
  <Param name="Normal_y"    dtype="REAL"   value="0.0" />
  <Param name="Normal_z"    dtype="REAL"   value="-1.0" />
  <Param name="c1"          dtype="REAL"   value="2.0" />
  <Param name="c2"          dtype="REAL"   value="10.0" />
  <Param name="c3"          dtype="REAL"   value="0.0" />
  <Param name="c4"          dtype="REAL"   value="0.0" />
  <Param name="u_threshold" dtype="REAL"   value="0.5" />
  <Param name="thickness"   dtype="REAL"   value="0.05" />
  <Param name="unit"        dtype="STRING" value="mmAq" />
  <Param name="vector"      dtype="STRING" value="directional" />
<\Elem>
```

表 5.9　圧力損失モデルのパラメータ

| XML キーワード | パラメータの説明 |
|---|---|
| Normal_x | 圧力損失部の法線ベクトルの x 方向成分　法線は単位ベクトル |
| Normal_y | 圧力損失部の法線ベクトルの y 方向成分　法線は単位ベクトル |
| Normal_z | 圧力損失部の法線ベクトルの z 方向成分　法線は単位ベクトル |
| c1 | 圧力損失部の圧力損失係数 c1　$[mmAq \mid mmHg \mid Pa]$ |
| c2 | 圧力損失部の圧力損失係数 c2　$[mmAq \mid mmHg \mid Pa]$ |
| c3 | 圧力損失部の圧力損失係数 c3　$[mmAq \mid mmHg \mid Pa]$ |
| c4 | 圧力損失部の圧力損失係数 c4　$[mmAq \mid mmHg \mid Pa]$ |
| u_threshold | 圧力損失カーブの切り替え速度 $u_{threshold}$ $[m/s]$ |
| thickness | 圧力損失部の厚さ　$[mm]$ |
| unit | 圧力損失 $\Delta p - V$ 線図のヘッドの単位　$[mmAq \mid mmHg \mid Pa \mid Non - Dimension]$ [*7] |
| vector | 速度ベクトルの法線方向への強制　$[Directional \mid Non - Directional]$ |

[*7] mmAq は水（300K, p=101.325kPa）996.62 $[kg/m^3]$，mmHg は水銀（300K）13538 $[kg/m^3]$ をプログラム中でハードコード.

## 5.3　熱解析の境界条件

非圧縮性流体近似されたエネルギー式，つまりパッシブスカラー型の温度輸送方程式について，温度の境界条件を説明する．流れの境界条件と同様に，まず，外部境界条件を 5.3.1 で説明し，その後内部境界条件を 5.3.2 で説明する．内部境界条件には熱交換機での利用を想定した放熱境界条件を含む．

### 5.3.1　外部境界条件の指定

熱の外部境界条件は，6.1.2 で詳しく述べているが，断熱条件はマスクにより温度勾配ゼロの条件が直接反復式中に取り込まれる．また，熱伝導条件は拡散項で表現され多媒質の場合に物性値依存で熱移動が計算される．これらの 2 つ以外の境界条件は明示的に指定する．

**表??**に示す熱の外部境界条件について例示する．熱の境界条件は，全て有次元でパラメータを与える．

■Dirichlet　指定面のガイドセルに指定温度を与える．次の例では，境界条件番号 40 として，20.0 [°C or K] の Dirichlet 境界条件を設定する．温度の単位は Unit_Temperature に従う．

```
<Param name="dirichlet" dtype="REAL" value="20.0" id="40"/>
```

■Insulation　指定面で断熱条件を指定する．断熱条件の実効的な実装は，内部境界条件の断熱指定によるので，ここでは単にガイドセルへの値のコピーを行っているだけである．次の例では，境界条件番号 30 に断熱条件 $\partial\theta/\partial x_i = 0$ を指定する．

```
<Param name="Insulation" dtype="REAL" value="" id="30"/>
```

■Outflow　指定面に流出境界を指定する．流出境界条件は，速度と同様に**式 (5.34)** の対流流出条件を用いている．

$$\frac{\partial\theta^{n+1}}{\partial t} + u_{ob}\frac{\partial\theta^{n}}{\partial x} = 0 \tag{5.34}$$

$u_{ob}$ は流出境界上の流出速度で，速度の場合と同様である．例えば，X+ 方向のガイドセルの値を決める場合には，**式 (5.35)** のように 2 段階で値を決めていく．

$$\left.\begin{aligned} \theta^{n+1}_{ix+1} &= \theta^{n}_{ix+1} - \Delta t\, u_{ob}\,\frac{\theta^{n}_{ix+1} - \theta^{n}_{ix}}{\Delta x} \\ \theta^{n+1}_{ix+2} &= \theta^{n}_{ix+2} - \Delta t\, u_{ob}\,\frac{\theta^{n}_{ix+2} - \theta^{n}_{ix+1}}{\Delta x} \end{aligned}\right\} \tag{5.35}$$

次の例では，境界条件番号 40 として，対流流出境界条件を設定する．

```
<Param name="Outflow" dtype="REAL" value="" id="40"/>
```

■Periodic　指定面で次の周期境界条件を指定する．

$$\left.\begin{aligned} \theta_0 &= \theta_{ix} \\ \theta_{ix+1} &= \theta_1 \end{aligned}\right\} \qquad \left.\begin{aligned} \theta_0 &= \theta_{jx} \\ \theta_{jx+1} &= \theta_1 \end{aligned}\right\} \qquad \left.\begin{aligned} \theta_0 &= \theta_{kx} \\ \theta_{kx+1} &= \theta_1 \end{aligned}\right\} \tag{5.36}$$

次の例では，境界条件番号 40 として周期境界条件を設定する．

```
<Param name="Periodic" dtype="REAL" value="" id="40"/>
```

■Symmetric　指定面に対称境界条件を指定する．対称条件は断熱と同じであるため，Insulation を用いている．次の例では，境界条件番号 40 として，対称境界条件を設定する．

```
<Param name="Symmetric" dtype="REAL" value="" id="40"/>
```

### 5.3.2　内部境界条件の指定

CBC ソルバークラスにおける熱の内部境界条件は，**表 5.2** に示すコンポーネントとして実装されている．指定できる熱境界条件には，**表 5.10** に示すようにセルフェイスとセルボリュームに対する指定方法がある．

表 5.10　熱の内部境界条件

| コンポーネント | タイプ | 境界条件 | モニター量 |
|---|---|---|---|
| Heat_Face | Adiabatic | 断熱境界 | なし |
| | Direct_Flux | 熱流束境界 | 熱流束 [$W/m^2$] |
| | Heat_Transfer_B | 熱伝達境界 type B | 熱流束 [$W/m^2$] |
| | Heat_Transfer_N | 熱伝達境界 type N | 熱流束 [$W/m^2$] |
| | Heat_Transfer_S | 熱伝達境界 type S | 熱流束 [$W/m^2$] |
| | Heat_Transfer_SF | 熱伝達境界 type SF | 熱流束 [$W/m^2$] |
| | Heat_Transfer_SN | 熱伝達境界 type SN | 熱流束 [$W/m^2$] |
| | Iso_Thermal | 等温境界 | 熱流束 [$W/m^2$] |
| | Radiation | 輻射境界[*1] | |
| Heat_Volume | Const_Temperature | 一定温度 | なし |
| | Heat_Generation | 吸発熱体 | なし |

#### 5.3.2.1　セルフェイスに対する境界条件

セルフェイスの指定方法は，ボクセルモデル作成時にセルに与える ID により指定する．つまり，境界条件に指定する ID と同時に指定される def_face タグで挟まれるセルフェイスが対象となる．

■Adiabatic　指定面に温度を直接指定する．6.1.2.2 で説明する断熱マスクにより，計算負荷を抑えて実装している．次の例では，ID=610 と ID=600 で挟まれる面に断熱境界を与える．

```
<Elem name="Adiabatic" id="600" comment="insulator">
  <Param name="def_face" dtype="INT" value="610"/>
</Elem>
```

■Direct_Flux　指定面に熱流束を直接指定する．モニター量として，指定部分の熱流束の和をログ出力する．次の例では，ID=78 と ID=610 で挟まれる面に熱流束 0.7[$W/m^2$] を適用する．

```
<Elem name="Direct_Flux" id="78" comment="direct_1">
  <Param name="Heat_Flux" dtype="REAL" value="0.7"/>
  <Param name="def_face" dtype="INT" value="610"/>
</Elem>
```

[*1] Radiation は 2013 年 3 月 16 日未実装.

■Heat_Transfer_B　熱伝達係数とバルク温度を与え，熱流束を計算する．固体の熱移動のみを解く場合の境界条件として利用する．次の例では，ID=60 に 10 度の温度が与えられ，ID=1 と挟まれる面に熱伝達係数 $H = 0.2\,[W/(m^2K)]$ が与えられる．

```
<Elem name="Heat_Transfer_B" id="60"  comment="HF_3">
  <Param name="def_face"    dtype="INT"    value="1" />
  <Param name="Bulk_Temperature" dtype="REAL" value="10.0"/>
  <Param name="Coef_of_Heat_Transfer" dtype="REAL" value="0.2"/>
</Elem>
```

■Heat_Transfer_N　固体表面セルの温度は計算によって決まる値を用い，熱伝達係数 ($H > 0$) のみを与え計算する．モニター量として，指定部分の熱流束の和をログ出力する．次の例では，ID=68 と ID=1 で挟まれる面に境界条件を設定する．

```
<Elem name="Heat_Transfer_N" id="68"  comment="HF_1">
  <Param name="Coef_of_Heat_Transfer" dtype="REAL" value="1.2e-2"/>
  <Param name="def_face"    dtype="INT"    value="1" />
</Elem>
```

■Heat_Transfer_S　想定する表面温度と熱伝達係数を与える．熱流体計算で固体側を解かず，流体のみを解く場合の境界条件として用いる．モニター量として，指定部分の熱流束の和をログ出力する．

次の例では，ID=67 と ID=1 で挟まれる面に境界条件を設定する．表面温度に 80[°C or K] と熱伝達率 H=0.012$[W/(m^2K)]$ を与えて計算する．

```
<Elem name="Heat_Transfer_S" id="67"  comment="HF_2">
  <Param name="Surface_Temperature" dtype="REAL" value="80.0"/>
  <Param name="Coef_of_Heat_Transfer" dtype="REAL" value="0.012"/>
  <Param name="def_face"    dtype="INT"    value="1" />
</Elem>
```

■Heat_Transfer_SF　流体のみを解く場合に，強制対流熱伝達の実験式を組み込んだ熱伝達境界条件を与える．モニター量として，指定部分の熱流束の和をログ出力する．

次の例では，ID=67 と ID=1 で挟まれる面に境界条件を設定する．表面温度に 80[°C or K] を与えて計算する．熱伝達流束を計算するときの温度差として，隣接セル温度との温度差を用いている．

```
<Elem name="Heat_Transfer_SF" id="60"  comment="HF_3">
  <Param name="def_face"    dtype="INT"    value="1" />
  <Param name="type"    dtype="STRING"    value="LOCAL_TEMPERATURE" />
  <Param name="Surface_Temperature" dtype="REAL" value="80.0"/>
  <Param name="alpha" dtype="REAL" value="0.037"/>
  <Param name="beta"  dtype="REAL" value="0.8"/>
  <Param name="gamma" dtype="REAL" value="0.333333"/>
```

■Heat_Transfer_SN　流体のみを解く場合に，自然対流熱伝達の実験式を組み込んだ熱伝達境界条件を与える．モニター量として，指定部分の熱流束の和をログ出力する．

次の例では，ID=67 と ID=1 で挟まれる面に境界条件を設定する．表面温度に 80[°C or K] を与えて計算する．熱伝達流束を計算するときの温度差として，バルク温度との温度差を用いている．

```
<Elem name="Heat_Transfer_SN" id="67"  comment="HF_2">
  <Param name="def_face"    dtype="INT"    value="1" />
  <Param name="type"    dtype="STRING"    value="BULK_TEMPERATURE" />
  <Param name="Surface_Temperature" dtype="REAL" value="80.0"/>
  <Param name="vertival_laminar_alpha" dtype="REAL" value="0.59"/>
  <Param name="vertival_laminar_beta"  dtype="REAL" value="0.25"/>
  <Param name="vertival_turbulent_alpha" dtype="REAL" value="0.1"/>
  <Param name="vertival_turbulent_beta"  dtype="REAL" value="0.3333333"/>
```

```
  <Param name="vertival_ra_critial" dtype="REAL" value="1.0e9"/>
  <Param name="lower_laminar_alpha" dtype="REAL" value="0.27"/>
  <Param name="lower_laminar_beta"  dtype="REAL" value="0.25"/>
  <Param name="lower_turbulent_alpha" dtype="REAL" value="0.27"/>
  <Param name="lower_turbulent_beta"  dtype="REAL" value="0.25"/>
  <Param name="lower_ra_critial" dtype="REAL" value="1.0e9"/>
</Elem>
```

■Iso_Thermal　指定面で温度が一定となる境界条件で，面温度を一定に保つような熱流束が発生する．等温境界は，固体－流体界面と固体－固体界面の 2 つの場合が考えられる．次の例では，ID=70 と ID=1 に挟まれる面を 30 度の等温面として扱うことを指定している．

```
<Elem name="Iso_Thermal" id="70"  comment="iso_1">
  <Param name="def_face"    dtype="INT"    value="1" />
  <Param name="Temperature" dtype="REAL" value="30.0"/>
</Elem>
```

#### 5.3.2.2　セルボリュームに対する境界条件

■Const_Temperature　指定温度を与える．

```
<Elem name="Heat_Volume" id="1">
  <Elem name="Const_Temperature" id="247" comment="Tape">
    <Param name="temperature" dtype="REAL" value="45.0"/>
  </Elem>
</Elem>
```

■Heat_Generation　**表 5.11** に示すように，発熱量または発熱密度を指定セルに与えることができる．発熱量を指定した場合には，該当 ID の体積を前処理で計算し，発熱密度に変換する．次の例では，ID=246 に 10[$W/m^3$] の発熱量を与えている．

```
<Elem name="Heat_Volume" id="1">
  <Elem name="Heat_Generation" id="246" comment="Desktop">
    <Param name="heat_generation_density" dtype="REAL" value="10.0"/>
  </Elem>
</Elem>
```

表 5.11　発熱セルの指定方法

| XML キーワード | パラメータの種類 | 単位 |
|---|---|---|
| Heat_Release_Value | 発熱量 | [$W$] |
| Heat_Generation_Density | 発熱密度 | [$W/m^3$] |

#### 5.3.2.3　外部境界における内部境界条件の取り扱い

計算領域外部境界においては，内部境界条件が優先する．このため，例えば，X-の外部境界面で断熱境界を指定しておき，同時に一部分を内部境界で上書きすることができる．

# 第 6 章

# 多媒質の熱流体解析

この章では，流れと熱の解析方法，特に固体熱伝導と温度の移流拡散方程式の連成を考慮した定式化，および熱境界条件の導入方法について解説する．まず，単媒質の場合の熱流動について述べ，その後，多媒質の場合について述べる．

本章では，熱流動を媒質数により次のように分類して説明する．

- 単媒質
  解くべき流体（固体）の媒質数が 1 つのみの場合で，セルの物性値 $\rho$, $C_p$, $\lambda$ は一定である．ただし，境界条件の媒質は複数あってもよい．
- 多媒質
  解くべき媒質数が複数ある場合で，流動とともに媒質位置が変化する．つまり，物性値 $\rho$, $C_p$, $\lambda$ がセル毎に変化するため，物性値の配列が必要となる．流体の媒質は一つのみの場合であっても，解くべき固体要素があれば共役熱移動問題となる．また，固体の多媒質もこのカテゴリで扱う．

# 6.1　単媒質の熱流動現象

## 6.1.1　支配方程式

### 6.1.1.1　熱伝導方程式と移流拡散方程式

固体の熱伝導の支配方程式 [18] は,

$$\rho' C \frac{\partial \theta'}{\partial t'} = \frac{\partial}{\partial x_i'} \left( \lambda \frac{\partial \theta'}{\partial x_i'} \right) + Q' \tag{6.1}$$

| | | |
|---|---|---|
| $\rho'$ | $[kg/m^3]$ | density |
| $C$ | $[J/(kg\,K)]$ | specific heat |
| $\theta'$ | $[K]$ | temperature |
| $\lambda$ | $[W/(m\,K)]$ | heat conductivity |
| $Q'$ | $[W/m^3]$ | heat source |
| $t'$ | $[s]$ | time |
| ${u_j}'$ | $[m/s]$ | velocity component |
| ${x_j}'$ | $[m]$ | coordinate axis |

エネルギー方程式から非圧縮近似をして得られるパッシブスカラの移流拡散方程式は,

$$\rho' C \left[ \frac{\partial \theta'}{\partial t'} + \frac{\partial}{\partial x_i'} \left( {u'}_i \theta' \right) \right] = \frac{\partial}{\partial x_i'} \left( \lambda \frac{\partial \theta'}{\partial x_i'} \right) + Q' \tag{6.2}$$

である.

### 6.1.1.2　単媒質の場合の支配方程式

有次元での計算は有効桁数の制約から桁落ちなどの問題が生じやすいため，$\theta \sim \mathrm{O}(1)$ となるようにできるだけ無次元化して解きたい．**式 (6.2)** は次の代表値により無次元化される.

$$\left.\begin{aligned} u_i' &= u_0' u_i \\ \theta' &= {\theta_0}' + \Delta\theta' \theta \\ t' &= (L'/{u_0}')\, t \\ x_i' &= L' x_i \end{aligned}\right\} \tag{6.3}$$

$$\frac{\partial \theta}{\partial t} + \frac{\partial}{\partial x_i} \left( u_i \theta \right) = \frac{1}{Pe} \frac{\partial}{\partial x_i} \frac{\partial \theta}{\partial x_i} + \Theta_V \tag{6.4}$$

$$\frac{1}{Pe} = \frac{\lambda}{\rho' C} \frac{1}{{u_0}' L'} \tag{6.5}$$

$$\Theta_V = \frac{Q'}{\rho' C} \frac{L'}{{u_0}' \Delta\theta'} \tag{6.6}$$

対流項がない拡散場のみ，つまり熱伝導方程式の場合には，一般に熱輸送の時間スケール，代表速度は熱流の伝播速度に相当すると考え,

$${u_0}' = \frac{\alpha}{L'} \tag{6.7}$$

とスケーリングされる．この代表速度に基づき，Fourier 数 $F$ を用いて均質場の熱輸送現象に適した無次元化が行われる．

$$t = \frac{{u_0}'}{L'}t' = \frac{\alpha}{{L'}^2}t' \equiv F \tag{6.8}$$

$$\frac{\partial\theta}{\partial F} = \frac{\partial}{\partial x_i}\left(\frac{\partial\theta}{\partial x_i}\right) + W, \qquad W = \frac{{L'}^2}{\lambda\Delta\theta'}Q' \tag{6.9}$$

このとき，無次元化の代表時間スケール ${t_0}'$ は，

$${t_0}' = \frac{{L'}^2}{\alpha} \qquad [s] \tag{6.10}$$

固体熱伝導を解く場合には，**式 (6.7)** の代表速度を用いて**式 (6.4)**~**(6.6)** を無次元化し，**式 (6.9)** と等価な解を得る．ただし，対流項の流体速度はゼロである．

## 6.1.2　熱境界条件の種類と離散式への埋め込み

### 6.1.2.1　熱境界条件の種類と実装の指針

C3D ソルバーでは，セル面に対して以下の熱境界条件を導入する．また，**式 (6.1)** の熱源 $Q'$ は，セル体積に作用する境界条件として実装する．

$$\left.\begin{array}{ll} Adiabatic & q'_A = 0 \\ Thermal\,conductivity & q'_C = -\lambda\dfrac{\partial\theta'}{\partial x'_i} \\ Thermal\,transmission & q'_T = -H(\theta'_s - \theta'_\infty) \\ Thermal\,radiation & q'_R = \sigma\varepsilon\left({\theta'_s}^4 - {\theta'_\infty}^4\right) \\ Isothermal & q'_{ISO} \\ Direct & q'_D \end{array}\right\} = q'_{BC} \quad [W/m^2] \tag{6.11}$$

断熱条件は計算領域中の任意の場所で現れる可能性が可能性が高く，そのたびに境界条件処理を行うのはコストがかかる．これを回避するため，温度勾配をゼロにするマスク関数を基礎式中に導入することにより実装する．また，輻射はメカニズムが異なるので，他の熱境界条件の実装とは別に扱う．

残りのセル面に対する境界条件について，時間進行，あるいは反復過程での境界条件の導入方法を考える．空間インデクスのループを回し，その後に境界条件を設定すると，1 ステップ/ループの遅れが生じる．境界条件のずれは，収束性の低下，あるいは解が異なる方向へ時間発展する可能性がある．また，ゴーストセルを導入した境界条件の設定は，圧力の場合と同様に多価の問題に厳密に対応する実装は難しい．できるだけ現象にコンシステントな境界条件の導入を考えると，境界条件の導入レベルには以下の 3 つが考えられる．

1. 反復ループ内で，境界条件の時間遅れが無いように取り扱う．最も精確な方法は支配方程式に整合するように陰的にとり扱う方法であるが，実装は煩雑になる．
2. 一反復の遅れを許容すると，一反復前の物理量を用いて境界流束の評価を陽的に行え，実装が簡単になる．この場合，収束すると，ほぼ正しい境界条件になることが期待できる．
3. 1 タイムステップ前の値を使う．1 ステップの間の変化量が小さければ問題はなく，定常流解析では有効である．ソース項などに用いると，安定な傾向である．

ここでは，2) の方法を採用し，ゴーストセルを使わず，簡単な形式かつ演算量の少ない方法でスキーム中に埋め込む方法を用いる．

熱境界条件を明示的に与えない固体-流体セル面[*1]は熱伝導面として扱う．熱伝導形式の境界条件は，基礎方程式中の拡散項で表現されているので，特別な境界条件を必要としない．

境界条件実装の方針をまとめると，以下のようになる．

- 境界熱流束 $q'_{BC}$ の具体的な評価は，一反復前の値を用いて計算する．
- 断熱は他の条件と排他的に扱い，基礎式に組み込む．
- 熱伝導，熱伝達，等温，熱流束は各形式間で排他的に扱う．
- 輻射は独立に考える．

これらのルールから，

$$q'_{BC} = [(q'_{ISO} | q_T' | q_C' | q_D') + q_R'] | q_A' \tag{6.12}$$

#### 6.1.2.2　スキームへの境界条件の埋め込み

**式 (6.12)** の境界条件をスキーム中に埋め込む実装方法として，境界条件マスク $\gamma_{BC}$ と断熱マスク $\gamma_A$ を導入する．境界条件マスクは通常の計算による熱流束と境界条件による熱流束を区別するために用いる．

$$\gamma_{BC} = \begin{cases} 0 & ... \quad BC \\ 1 & ... \quad Non-BC \end{cases} \tag{6.13}$$

$$\gamma_A = \begin{cases} 0 & ... \quad Adiabatic \\ 1 & ... \quad Non-adiabatic \end{cases} \tag{6.14}$$

マスク関数を導入すると，拡散熱流束は次のように表記できる．

$$q'_i = [\gamma_A \gamma_{BC} q' + \gamma_A (1-\gamma_{BC}) q'_{BC}]_i \tag{6.15}$$

マスクパターンと熱流束の対応は**表 6.1** のようになる．

表 6.1　境界条件マスクによる熱境界条件パターン

| $\gamma_A$ | $\gamma_{BC}$ | $q'$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | $q'_{BC}$ |
| 1 | 1 | $q'$ |

**式 (6.2)** に Euler 陽解法を適用し，空間項を有限体積的に評価すると，

$$\rho' C \left[ \int \frac{\partial \theta'}{\partial t'} dV' + \int \frac{\partial}{\partial x'_i} (u_i' \theta') dV' \right] = - \int \frac{\partial q'_i}{\partial x'_i} dV' + \int Q' dV' \tag{6.16}$$

半離散的に示すと，

$$\rho' C \frac{\theta'^{n+1} - \theta'^n}{\Delta t'} = - \frac{\rho' C}{h'} \sum_{j=1}^{\dim \times 2} (u' \theta')_j n'_j - \frac{1}{h'} \sum_{j=1}^{\dim \times 2} q'_j n'_j + Q' \tag{6.17}$$

[*1] 固体壁面の種類として，断熱壁面，熱伝導壁面，熱伝達壁面，等温壁面，熱流束指定面を設定できる．

ここで，$ds'$ は法線成分をもつ面素，$n'_j$ はセルの外向き単位法線ベクトルである．右辺第二項を**式 (6.15)** を用いて書き下すと，

$$-\frac{1}{h'}\sum_{j=1}^{\dim\times 2} q'_j n'_j = -\frac{1}{h'}\sum_{j=1}^{\dim\times 2}(-1)^j\left(\gamma_A\gamma_{BC}q'\right)_j - \frac{1}{h'}\sum_{j=1}^{\dim\times 2}(-1)^j\left\{\gamma_A\left(1-\gamma_{BC}\right)q'_{BC}\right\}_j \tag{6.18}$$

添字 $j$ は**図 6.1** に示すように，着目するセル P を囲むセル（二次元では４つ，三次元では６つ）との界面を意味する．記号 {w, e, s, n, b, t} が $j = 1 \sim 6$ に対応する．一方，添字 $J$ は，{W, E, S, N, B, T} における物理量を表す．**式 (6.18)** の右辺第一項を書き下すと，

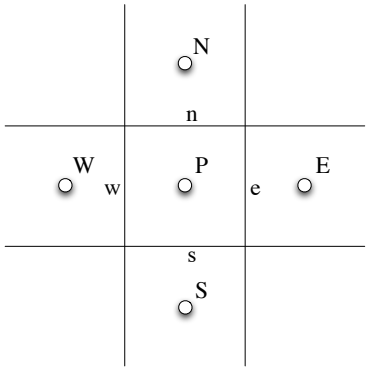



図 6.1　更新セルと参照セルのインデクス

$$-\frac{1}{h'}\sum_{j=1}^{\dim\times 2}(-1)^j\left(\gamma_A\gamma_{BC}q'\right)_j = \frac{1}{h'^2}\sum_{j=1}^{\dim\times 2}\left(\gamma_A\gamma_{BC}\lambda\right)_j\left(\theta'_J-\theta'_P\right) = \frac{1}{h'^2}\left[\sum_{j=1}^{\dim\times 2}\left(\gamma_A\gamma_{BC}\lambda\right)_j\theta'_J - \theta'_P\sum_{j=1}^{\dim\times 2}\left(\gamma_A\gamma_{BC}\lambda\right)_j\right] \tag{6.19}$$

したがって，**式 (6.17)** は

$$\begin{aligned}\rho' C\frac{\theta'^{\,n+1}-\theta'^{\,n}}{\Delta t'} = &-\frac{\rho' C}{h'}\sum_{j=1}^{\dim\times 2}\left(u'\theta'\right)_j n'_j + \frac{1}{h'^2}\left[\sum_{j=1}^{\dim\times 2}\left(\gamma_A\gamma_{BC}\lambda\right)_j\theta'_J - \theta'_P\sum_{j=1}^{\dim\times 2}\left(\gamma_A\gamma_{BC}\lambda\right)_j\right]\\ &-\frac{1}{h'}\sum_{j=1}^{\dim\times 2}(-1)^j\left\{\gamma_A\left(1-\gamma_{BC}\right)q'_{BC}\right\}_j + Q'\end{aligned} \tag{6.20}$$

無次元形式で表示すると，

$$\begin{aligned}\frac{\theta^{n+1}-\theta^n}{\Delta t} = &-\frac{\rho C}{h}\sum_{j=1}^{\dim\times 2}\left(u\theta\right)_j n_j\\ &+\frac{1}{h^2 Pe}\left[\sum_{j=1}^{\dim\times 2}\left(\gamma_{A\,j}\gamma_{BC\,j}\theta_J\right)-\theta_P\sum_{j=1}^{\dim\times 2}\gamma_{A\,j}\gamma_{BC\,j}\right] - \frac{1}{h}\sum_{j=1}^{\dim\times 2}(-1)^j\left\{\left(1-\gamma_A\gamma_{BC}\right)q_{BC}\right\}_j + \Theta_V\end{aligned} \tag{6.21}$$

ただし，

$$q_{BC} = \frac{q'_{BC}}{u'_0\Delta\theta'}\frac{1}{\rho' C} \tag{6.22}$$

である．このとき，無次元化パラメータに用いた $\rho'$, $C$ は解くべき媒質の物性値であることに注意する．C3D ソルバークラスでは，XML 入力ファイルにおいて，Reference セクションの Ref_ID タグでボクセル ID を指定する．この ID は，Steer セクションの Model_Setting タグで媒質テーブルと対応づけられ，基礎式で解くべき媒質を示す．つまり，**式 (6.22)** の $\rho'$, $C$ の媒質を示している．

### 6.1.3　時間発展方程式

基礎方程式 **(6.4)** の対流項を $F$ でまとめて表し，時間進行スキームを切り替えるパラメータ $\alpha$ を用いると，

$$\frac{\partial \theta^{n+1}}{\partial t} = (1-\alpha) F^{n} + \alpha F^{n+1} + \Theta_V \tag{6.23}$$

$$\alpha = \begin{cases} 0 & Euler\,Explicit \\ 1/2 & Crank\,Nicolson \\ 1 & Euler\,Implicit \end{cases} \tag{6.24}$$

ソース項の時制については任意性を残しておく．一般に時刻について線形化し，$\Theta_V^n$ とする方が安定である．以下の議論では，拡散項の時間進行に着目して説明する．

#### 6.1.3.1　Euler **陽解法**

**式 (6.21)** より，

$$\begin{aligned} \frac{\theta_P^{n+1} - \theta_P^{n}}{\Delta t} = & -\frac{\rho C}{h} \sum_{j=1}^{\dim\times 2} (u\theta)_j n_j + \frac{1}{h^2 Pe} \left[ \sum_{j=1}^{\dim\times 2} \left( \gamma_{A\,j} \gamma_{BC\,j} \theta_J \right) - \theta_P \sum_{j=1}^{\dim\times 2} \gamma_{A\,j} \gamma_{BC\,j} \right]^n \\ & - \frac{1}{h} \sum_{j=1}^{\dim\times 2} (-1)^j \left\{ (1 - \gamma_A \gamma_{BC})\, q_{BC} \right\}_j^n + \Theta_V^n \end{aligned} \tag{6.25}$$

陽解法の場合には，次の拡散数による数値安定性の点に注意する．

$$\Delta t \le \frac{1}{2dim} h^2 Pe \tag{6.26}$$

### 6.1.4　セル界面に対する熱境界条件

本節で述べる熱境界条件を**式 (6.20)** の $q'_{BC}$ に代入することにより，多様な熱境界条件を導入することができる．熱流束の次元は，

$$q'_{BC} \sim -\lambda \frac{\partial \theta'}{\partial x_i'} \qquad [W/m^2] \tag{6.27}$$

境界条件に必要なパラメータは多媒質への拡張も考慮し，有次元での入力とする．熱流束形式の境界条件は，前の時刻あるいは前の反復ステップの値を用いて，流束の形式で計算しておき，その値を反復過程で参照しながら計算する．

#### 6.1.4.1　**熱流束境界** $q_D$

何らかの物理モデルにより，熱流束 $q'\,[W/m^2]$ が直接与えられる場合に用いる．セル界面における熱流束 $q'_{i+1/2} > 0$ が与えられた場合を考える．熱流の方向が正なので，セル $i$ は冷却，セル $i+1$ は加熱となる．単媒質の場合，

$$q_D = \frac{q'_D}{u_0' \Delta\theta'} \frac{1}{\rho' C} \tag{6.28}$$

ここで $\rho' C$ は解くべきセルの物性値で，単媒質の場合には予め定数として与えられる．次の例では，ID=78 と ID=1 で挟まれる面に $q'_D = 2.0\,[W/m^2]$ の熱流束を与えている．

```
<Elem name="Direct" id="78" comment="direct_1">
  <Param name="def_face"  dtype="INT"  value="1" />
  <Param name="Heat_Flux" dtype="REAL" value="2.0"/>
</Elem>
```

#### 6.1.4.2　熱伝達境界 $q_T$

熱伝達形式の境界熱流束は，下記のように書ける．

$$q'_T = -H(\theta'_{sf} - \theta'_{\infty}) \tag{6.29}$$

| | | |
|---|---|---|
| $H$ | $[W/(m^2K)]$ | Coefficient of heat transfer |
| $\theta'_{sf}$ | $[K]$ | Surface temperature of solid |
| $\theta'_{\infty}$ | $[K]$ | Temperature at outer boundary layer |

**図 6.2** において，セル P における熱量の収支を考える．セル P は固体壁 S との界面で正の熱流束が与えられたときにセルの温度が上昇する．一般に外部流の場合には，温度境界層外縁における参照温度 $\theta'_{\infty}$ は，セル幅が温度境界層よりも厚い場合を想定しているので，P 点の温度としてよい．しかしながら，複雑な流動となる内部流の場合には参照温度自体の定義に一意性がない．参照温度として外部流のように P 点をとると，温度差が小さくなる場合に熱伝達による熱流束が小さくなる．一方で，基準温度を想定すると，局所的な温度分布に関係なく一定量の熱伝達流束を与えることになる点に注意する．

固体表面における温度が必要な場合には，等間隔格子上で，

$$\theta'_{sf} = \frac{\lambda_S \theta_S + \lambda_P \theta_P}{\lambda_S + \lambda_P} \tag{6.30}$$

となる．**表 6.4** に示すように固体の熱伝導率は流体よりも大きい ($\lambda_{solid} \gg \lambda_{fluid}$) ため，一般に $\theta'_{sf} \cong \theta'_{solid}$ のように考えてよい．

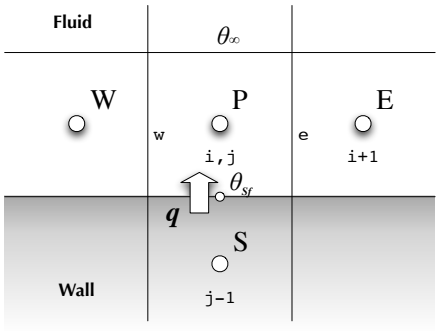



図 6.2　固体壁面での熱伝達

指定面における熱伝達境界条件の実装方法として，次の 3 つの場合が考えられる．

**タイプ** N)　熱伝達係数を与え，計算結果の固体表面温度から熱流束を計算する．共役熱移動の場合に利用する．

**タイプ** S)　表面温度と熱伝達係数を与え，熱流束を計算する．流体のみを解く場合に利用する．

**タイプ** B)　熱伝達係数とバルク温度を与え，熱流束を計算する．固体の熱移動のみを解く場合の境界条件として利用する．

**タイプ** SNB/SNL)　自然対流の乱流熱伝達の実験式を実装したもの．流体のみを解く場合に利用し，タイプ S を拡張している．SNB は温度差の定義にバルク温度を用い，SNL は隣接セルの値を用いた実装である．

**タイプ** SFB/SFL)　強制対流の層流・乱流熱伝達の実験式を実装したもの．流体のみを解く場合に利用し，タイプ S を拡張している．SFB は温度差の定義にバルク温度を用い，SFL は隣接セルの値を用いた実装である．

**■Nusselt 数と Stanton 数**　まず，ヌセルト数との関連を示す．ヌセルト数の定義は，熱伝導に対する熱伝達の比なので，

$$Nu = \frac{HL'}{\lambda}, \qquad \left.\begin{array}{ll} H & [W/(m^2K)] \\ \lambda & [W/(mK)] \\ L' & [m] \end{array}\right\} \tag{6.31}$$

フーリエの法則とニュートンの冷却則から，

$$H = \lambda\left(\frac{\partial\theta'}{\partial n'}\right)_{n'=0}\Bigg/\Delta\theta' = \frac{\lambda\Delta\theta'}{L'\Delta\theta'}\left(\frac{\partial\theta}{\partial n}\right)_{n=0} = \frac{\lambda}{L'}\left(\frac{\partial\theta}{\partial n}\right)_{n=0} \tag{6.32}$$

$$Nu = \left(\frac{\partial\theta}{\partial n}\right)_{n=0} \tag{6.33}$$

一方，熱流束を表す**式 (6.27)** の無次元形式は，**式 (6.28)** となり，これは Stanton 数を表す．つまり，

$$q = \frac{q'}{u'_0\Delta\theta'}\frac{1}{\rho'C} \quad \equiv \quad St = \frac{Nu}{RePr} \tag{6.34}$$

**■タイプ** N　タイプ N の計算は固体表面セルの温度は計算によって決まる値を用い，熱伝達係数（$H > 0$）のみで計算できる．**図 6.3** には，固体と流体の相対的な位置が異なる (a), (b) 2 つのパターンを示している．いま，セル境界において，正の熱流束 $q'_{j+1/2} > 0$ を仮定する．つまり，

$$q'_{j+1/2} = -H(\theta'_{j+1} - \theta'_j) > 0 \quad \Longleftrightarrow \quad \theta'_j > \theta'_{j+1} \tag{6.35}$$

$q'_{j+1/2} > 0$ なので，j セルは冷却, j+1 セルは加熱となる．次に，セル界面において負の熱流束 $q'_{j+1/2} < 0$ を仮定すると，

$$q'_{j+1/2} = -H(\theta'_{j+1} - \theta'_j) < 0 \quad \Longleftrightarrow \quad \theta'_j < \theta'_{j+1} \tag{6.36}$$

したがって，熱伝達形式の境界熱流束 $q'_T$ は，常に次式で計算できる．

$$q'_{T,\,j+1/2} = -H(\theta'_{j+1} - \theta'_j) \tag{6.37}$$

**式 (6.22)** を用いて無次元化すると，

$$q_{T,\,j+1/2} = -\frac{1}{\rho'C}\frac{H}{u_{0'}}(\theta_{j+1} - \theta_j) \tag{6.38}$$

ここで $\rho'C$ は，単媒質の場合には代表媒質の物性値である．

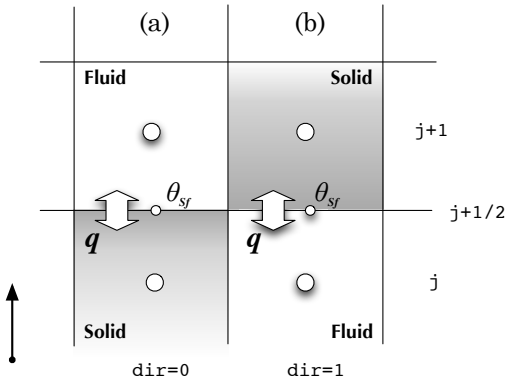



図 6.3　熱伝達境界タイプ N/S のパターン．方向フラグ dir は**式 (6.39)** で用いるインデクス j に対する相対的な位置を示す．

```
<Elem name="Heat_Transfer_N" id="68"  comment="HF_1">
  <Param name="def_face"    dtype="INT"    value="1" />
  <Param name="Coef_of_Heat_Transfer" dtype="REAL" value="1.2e-2"/>
</Elem>
```

**■タイプ S**　この境界条件は想定する表面温度と熱伝達係数を与える．熱流体計算で固体側を解かず，流体のみを解く場合の境界条件として用いる．**図 6.3** を参照して，参照温度を外側 1 点目にとる場合，パターン (a) の $dir = 0$ のときに法線が正になるようにインデクス計算を行うと，

$$q_{T,\,j+1/2} \;=\; -\frac{1}{\rho' C}\frac{H}{u_{0'}}\left(\theta_{j+1-dir} - \theta_{sf}\right)(1-2dir) \tag{6.39}$$

また，参照温度を基準温度にとる場合，

$$q_{T,\,j+1/2} \;=\; -\frac{1}{\rho' C}\frac{H}{u_{0'}}\left(\theta_{0} - \theta_{sf}\right)(1-2dir) \tag{6.40}$$

タイプ S の実装は，**式 (6.40)** である．

次の例では，ID=67 に 80 度の温度が与えられ，ID=1 と挟まれる面に熱伝達係数 $H = 0.012\,[W/(m^2K)]$ が与えられる．

```
<Elem name="Heat_Transfer_S" id="67"  comment="HF_2">
  <Param name="def_face"    dtype="INT"    value="1" />
  <Param name="Surface_Temperature" dtype="REAL" value="80.0"/>
  <Param name="Coef_of_Heat_Transfer" dtype="REAL" value="1.2e-2"/>
</Elem>
```

**■タイプ B**　この境界条件は固体熱伝導を解く場合の境界条件である．つまり，固体セルのみを解き，流体セルは解かない場合に用い，流体の参照温度と熱伝達係数を指定する．計算式は**式 (6.41)** を用いる．参照する流体のインデクスは，**図 6.4** のように，タイプ N/S と参照方向が逆になっている．パターン (a) の場合に熱伝達面からの法線は正，(b) の場合に法線が負になるように，方向を表すインデクス計算を $2dir - 1$ として，

$$q_{T,\,j+1/2} \;=\; \frac{1}{\rho' C}\frac{H}{u_{0'}}\left(\theta_{j+1-dir} - \theta_{\infty}\right)(2dir-1) \tag{6.41}$$

次の例では，ID=60 に 10 度の温度が与えられ，ID=1 と挟まれる面に熱伝達係数 $H = 0.2\,[W/(m^2K)]$ が与えられる．

```
<Elem name="Heat_Transfer_B" id="60"  comment="HF_3">
```

```
  <Param name="def_face"    dtype="INT"    value="1" />
  <Param name="Bulk_Temperature" dtype="REAL" value="10.0"/>
  <Param name="Coef_of_Heat_Transfer" dtype="REAL" value="0.2"/>
</Elem>
```

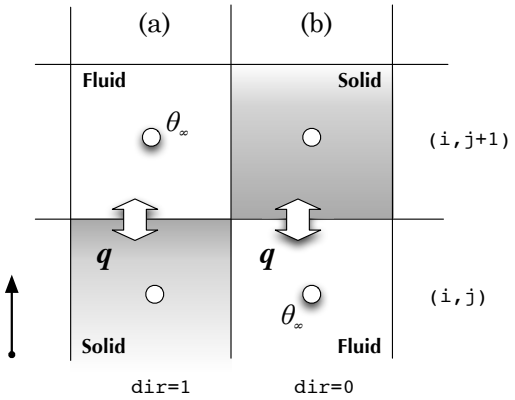



図 6.4　熱伝達境界 B のパターン.

**■タイプ** SNB/SNL　文献 [18] には，平板に対する自然対流の層流と乱流の熱伝達に関する近似式が説明されている．雰囲気流体の温度に比べ加熱面の温度が非常に高い場合，平板が長くなると境界層が不安定になり，ほぼ $Ra > 10^9$ で層流から乱流へ遷移する．垂直平板に関する平均熱伝達 ($\overline{Nu_L}$，代表長 $L$) は次式で整理される．

$$\left.\begin{array}{ll} \text{層流} & \overline{Nu_L} = 0.59 Ra_L^{1/4} \quad (10^4 < Ra_L < 10^9) \\ \text{乱流} & \overline{Nu_L} = 0.10 Ra_L^{1/3} \quad (10^9 < Ra_L < 10^{13}) \end{array}\right\} \tag{6.42}$$

一方，水平平板の場合には，加熱面が上面と下面にある場合で雰囲気流体の挙動が異なるため，**式 (6.43)** のように整理されている．

$$\left.\begin{array}{ll} \text{上面加熱} & \overline{Nu_L} = 0.54 Ra_L^{1/4} \quad (10^4 < Ra_L < 10^7) \\ \text{上面加熱} & \overline{Nu_L} = 0.15 Ra_L^{1/3} \quad (10^7 < Ra_L < 10^{11}) \\ \text{下面加熱} & \overline{Nu_L} = 0.27 Ra_L^{1/4} \quad (10^5 < Ra_L < 10^{10}) \end{array}\right\} \tag{6.43}$$

上記の自然対流熱伝達の整理式を**図 6.5** に示す.

これらの整理式を見ると，大まかには垂直面の実験式を用いて水平面の上面を近似できると考える．そこで，水平面の上面と垂直面，水平面の下面の 2 つのパターンについて，層流と乱流の実験式を用いることにする．境界条件のパラメータで与える値を**表 6.2** に示す．標準的には，**式 (6.42)**, **式 (6.43)** の値を用いる．

次に，タイプ SN の熱伝達境界条件の実装について説明する．**式 (6.42)**，**式 (6.43)** は形式的に次式のように表せる．

$$\overline{Nu_L} = \alpha {Ra_L}^{\beta} \tag{6.44}$$

**式 (6.31)** と**式 (6.44)** から,

$$H = \alpha Ra_L^{\beta} \frac{\lambda}{L'} \tag{6.45}$$

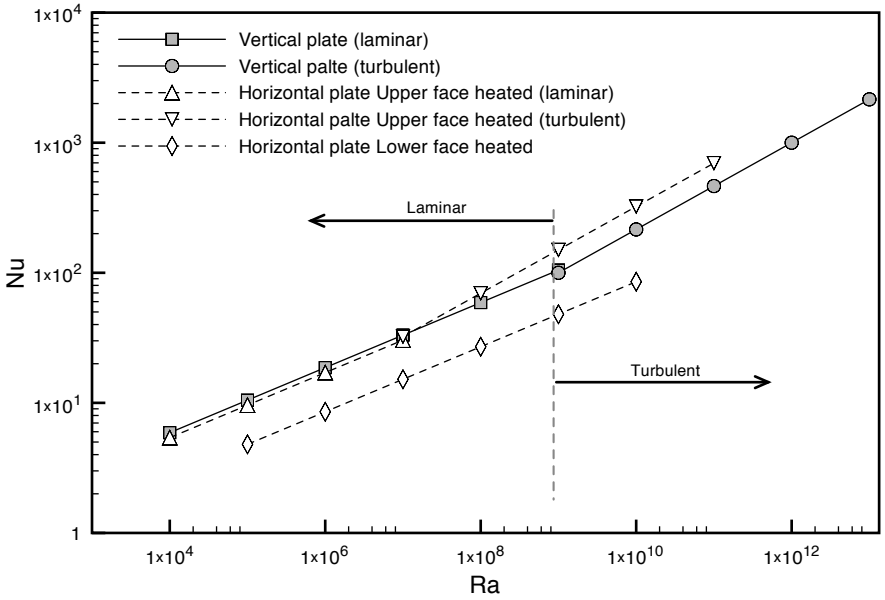



図 6.5　自然対流熱伝達の整理式 [18]

表 6.2　自然対流熱伝達のパラメータ

| パラメータタグ | 記号の意味 |
|---|---|
| vertival_laminar_alpha | 垂直平板の層流時の係数 $\alpha$ |
| vertival_laminar_beta | 垂直平板の層流時の係数 $\beta$ |
| vertival_turbulent_alpha | 垂直平板の乱流時の係数 $\alpha$ |
| vertival_turbulent_beta | 垂直平板の乱流時の係数 $\beta$ |
| vertival_Ra_critial | 垂直平板の臨界 Ra 数 $Ra_L$ |
| lower_laminar_alpha | 水平平板（下面）の層流時の係数 $\alpha$ |
| lower_laminar_beta | 水平平板（下面）の層流時の係数 $\beta$ |
| lower_turbulent_alpha | 水平平板（下面）の乱流時の係数 $\alpha$ |
| lower_turbulent_beta | 水平平板（下面）の乱流時の係数 $\beta$ |
| lower_Ra_critial | 水平平板（下面）の臨界 Ra 数 $Ra_L$ |
| type | BULK_TEMPERATURE or LOCAL_TEMPERATURE |

**式 (6.29)** と**式 (6.22)** から，

$$q = -\frac{q'}{u_0'\Delta\theta'\rho'C} = -\frac{H(\theta_{sf} - \theta_\infty)}{u_0'\rho'C} = -\frac{\lambda(\theta_{sf} - \theta_\infty)}{u_0'L'\rho'C}\alpha Ra_L^\beta \tag{6.46}$$

ここで，水平面の上下面では熱伝達が大きく異なるので，z 軸方向が重力方向と仮定して，上下面で選択的な実装を行う．上下面の選択は**図 6.3** から，

$$\left.\begin{array}{ll} \text{上面加熱} & dir = 0 \\ \text{下面加熱} & dir = 1 \end{array}\right\} \tag{6.47}$$

SNL　参照点をひとつ外側の点にとるタイプ SNL の場合，パラメータに LOCAL_TEMPERATURE を指定する．**式 (6.47)** の関係を用いて，**式 (6.39)** に倣い，

$$q_{T,\,j+1/2} \;=\; -\frac{\lambda\left(\theta_{j+1-dir}-\theta_{sf}\right)}{u_0' L' \rho' C}\,\alpha Ra_L^{\beta}\,(1-2dir) \tag{6.48}$$

SNB　参照点を基準温度 (BaseTmp) にとる SNB の場合，BULK_TEMPERATURE を指定する．

$$q_{T,\,j+1/2} \;=\; -\frac{\lambda\left(\theta_{0}-\theta_{sf}\right)}{u_0' L' \rho' C}\,\alpha Ra_L^{\beta}\,(1-2dir) \tag{6.49}$$

次の例では，ID=67 に 80 度の温度が与えられ，ID=1 と挟まれる面に**式 (6.42)**，**(6.43)** の熱伝達係数が与えられる．

```
<Elem name="Heat_Transfer_SN" id="67"  comment="HF_2">
  <Param name="def_face"    dtype="INT"    value="1" />
  <Param name="type"    dtype="STRING"    value="BULK_TEMPERATURE" />
  <Param name="Surface_Temperature" dtype="REAL" value="80.0"/>
  <Param name="vertival_laminar_alpha" dtype="REAL" value="0.59"/>
  <Param name="vertival_laminar_beta"  dtype="REAL" value="0.25"/>
  <Param name="vertival_turbulent_alpha" dtype="REAL" value="0.1"/>
  <Param name="vertival_turbulent_beta"  dtype="REAL" value="0.3333333"/>
  <Param name="vertival_ra_critial" dtype="REAL" value="1.0e9"/>
  <Param name="lower_laminar_alpha" dtype="REAL" value="0.27"/>
  <Param name="lower_laminar_beta"  dtype="REAL" value="0.25"/>
  <Param name="lower_turbulent_alpha" dtype="REAL" value="0.27"/>
  <Param name="lower_turbulent_beta"  dtype="REAL" value="0.25"/>
  <Param name="lower_ra_critial" dtype="REAL" value="1.0e9"/>
</Elem>
```

**■タイプ** SFB/SFL　強制対流熱伝達の実験式を組み込んだ熱伝達境界条件を与える．文献 [18] から，平板に対する発達した強制対流の乱流熱伝達は，実験による摩擦係数の測定結果とチルトン-コルバーンのアナロジーを用い，温度一定で平板が遷移長さよりも十分に大きいと仮定すると，**式 (6.50)** のように表せる．**図 6.6** には幾つかの $Pr$ 数に対する熱伝達の様子を示す．

$$\overline{Nu_L} \;=\; 0.037 Re_L^{4/5} Pr^{1/3} \tag{6.50}$$

熱伝達係数の形式を次のように表し，タイプ SN と同様に書き下すと，

$$H \;=\; \alpha Re_L^{\beta} Pr^{\gamma}\,\frac{\lambda}{L'} \tag{6.51}$$

タイプ SN と同様に，**式 (6.51)** のパラメータを**表 6.3** で与える．

表 6.3　強制対流熱伝達のパラメータ

| パラメータタグ | 記号の意味 |
|---|---|
| alpha | 係数 $\alpha$ |
| beta | 係数 $\beta$ |
| gamma | 係数 $\gamma$ |
| type | BULK_TEMPERATURE or LOCAL_TEMPERATURE |

次に，タイプ SF の熱伝達境界条件の実装について説明する．**式 (6.51)** は温度差の取り方により，次の 2 つのタイプがある．

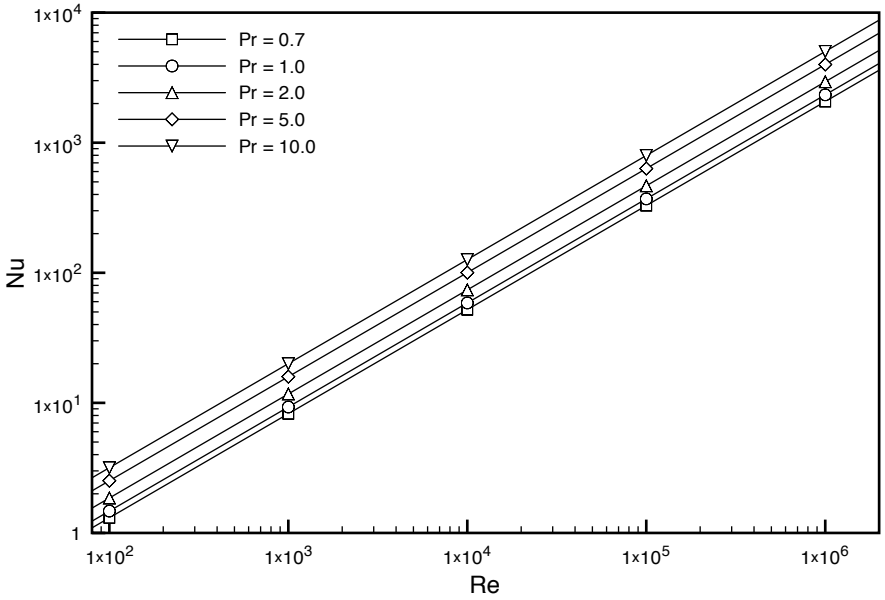



図 6.6　強制対流熱伝達 [18]

SFL　参照点を一つ外側にとる場合，パラメータに LOCAL_TEMPERATURE を指定する．

$$q_{T,\,j+1/2} \;=\; -\frac{\lambda\left(\theta_{j+1-dir}-\theta_{sf}\right)}{u_0' L' \rho' C}\,\alpha Re_L^{\beta} Pr^{\gamma}\;(1-2dir) \tag{6.52}$$

SFB　参照点を基準温度 (BaseTmp) にとる SFB の場合，BULK_TEMPERATURE を指定する．

$$q_{T,\,j+1/2} \;=\; -\frac{\lambda\left(\theta_{0}-\theta_{sf}\right)}{u_0' L' \rho' C}\,\alpha Re_L^{\beta} Pr^{\gamma}\;(1-2dir) \tag{6.53}$$

次の例では，ID=60 に 80 度の温度が与えられ，ID=1 と挟まれる面に**式 (6.50)** の熱伝達係数が与えられる．

```
<Elem name="Heat_Transfer_SF" id="60"  comment="HF_3">
  <Param name="def_face"    dtype="INT"    value="1" />
  <Param name="type"    dtype="STRING"    value="LOCAL_TEMPERATURE" />
  <Param name="Surface_Temperature" dtype="REAL" value="80.0"/>
  <Param name="alpha" dtype="REAL" value="0.037"/>
  <Param name="beta"  dtype="REAL" value="0.8"/>
  <Param name="gamma" dtype="REAL" value="0.333333"/>
</Elem>
```

#### 6.1.4.3　**等温境界** $q_{ISO}$

指定面で温度が一定となる境界条件で，面温度を一定に保つような熱流束が発生する．ここで扱う等温境界は，固体－流体界面と固体－固体界面の 2 つの場合が考えられる．境界面は，IsoThermal の ID と def_face により挟まれる面となる．

**図 6.7** において，色塗りの Solid セルは IsoThermal の ID がつけられたセルである．(a), (b) の各パターンは IsoThermal セルの位置が異なる．dir は前処理段階で設定される方向インデクスである．(a) の場合は，界面と温度指定の Solid セルのインデクスが両方とも j であるので，dir=0 とする．このとき，IsoThermal セルからの法線方向は正である．一方，(b) の場合は，界面インデクスは j であるが，等温指定の Solid セルは j+1 なので，dir=1 とする．このとき，IsoThermal セルからの法線方向は負である．

(a) のパターン（dir=0）を考える．セル j は固体セルで，表面温度は常に指定温度である．計算する方のセル j+1 側

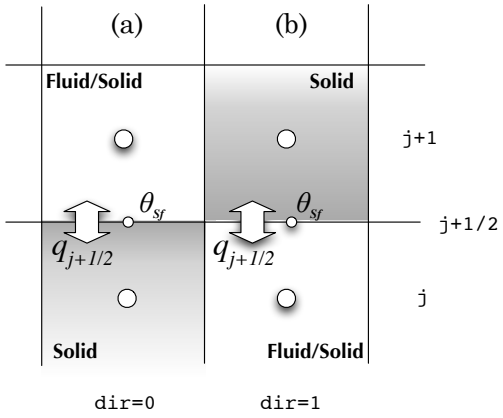



図 6.7　等温境界のパターン．方向フラグ dir は IsoThermal セルの位置と法線方向を示す．

からの熱流束を考えると，片側近似により，

$$q'_{ISO,\ j+1/2}=-\lambda_{j+1}\frac{\theta'_{j+1}-\theta'_{sf}}{h'/2} \tag{6.54}$$

無次元化すると，

$$q_{ISO,\ j+1/2} \sim \frac{q'_{ISO}}{u'_0\Delta\theta'}\frac{1}{\rho' C} = -\frac{1}{u'_0\Delta\theta'}\frac{\lambda_{j+1}}{\rho' C}\frac{\theta'_{j+1}-\theta'_{sf}}{h'/2} = -\frac{2}{u'_0 L'}\frac{\lambda_{j+1}}{\rho' C}\frac{\theta_{j+1}-\theta_{sf}}{h} \tag{6.55}$$

一方，(b) のパターン（dir=1）は，セル j+1 は固体セルで常に指定温度である．

$$q'_{ISO,\ j+1/2}=-\lambda_j\frac{\theta'_{sf}-\theta'_j}{h'/2} \tag{6.56}$$

$$q_{ISO,\ j+1/2} \sim \frac{q'_{ISO}}{u'_0\Delta\theta'}\frac{1}{\rho' C} = -\frac{1}{u'_0\Delta\theta'}\frac{\lambda_j}{\rho' C}\frac{\theta'_{sf}-\theta'_j}{h'/2} = -\frac{2}{u'_0 L'}\frac{\lambda_j}{\rho' C}\frac{\theta_{sf}-\theta_j}{h} \tag{6.57}$$

計算する流体または固体セルは $j+1-dir$ で表されること，基準セルに注意してパターン (a), (b) をまとめると，単媒質の場合，

$$q_{ISO,\ j+1/2} = -\frac{2}{u'_0 L'}\frac{\lambda_{j+1-dir}}{\rho' C}\frac{\theta_{j+1-dir}-\theta_{sf}}{h}(1-2dir) \tag{6.58}$$

ここで $1-2dir$ はパターン (a) のとき，つまり $dir=0$ のときに法線が正になるように構成している．

次の例では，ID=70 と ID=1 に挟まれる面を 30 度の等温面として扱うことを指定している．

```
<Elem name="Iso_Thermal" id="70"  comment="iso_1">
  <Param name="def_face"    dtype="INT"    value="1" />
  <Param name="Temperature" dtype="REAL" value="30.0"/>
</Elem>
```

#### 6.1.4.4 **輻射境界** $q_R$

熱輻射の計算を行う場合に用いる．輻射境界条件は，形態係数の計算を行い輻射流束の収支計算を行った後，その熱流束を直接熱流束境界として与える．

```
<Elem name="Heat_Face" id="0">
  <Elem name="Radiation" id="75"  comment="rad_1">
    <Param name="def_face"    dtype="INT"    value="1" />
    <Param name="epsilon" dtype="REAL" value="0.2"/>
    <Param name="projection" dtype="REAL" value="0.6"/>
  </Elem>
</Elem>
```

#### 6.1.4.5 **断熱境界** $q_A$

断熱境界 $q_A$ は空間におけるセル数の出現頻度が多くなることが予想されるため，他の境界条件とは異なり，断熱マスクにより離散式中に組み込んでいる．このため，モデル作成時に ID で指定することになる．断熱面の境界条件は，Adiabatic の指定により導入する．Adiabatic の指定は，id と def_face で指定される面を断熱境界と解釈する．下記の例では，ID=40 と ID=1 で挟まれる面が断熱面となる．

```
<Elem name="Adiabatic" id="40" comment="insulation">
  <Param name="def_face"    dtype="INT"    value="1" />
</Elem>
```

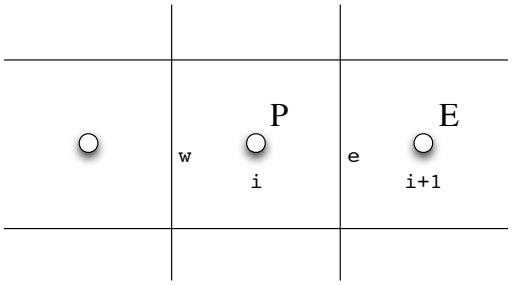



図 6.8　セルインデックスとセル界面の指定

**図 6.8** において，$e$ 面（$i + 1/2$ 位置のセル面）が断熱面の場合，断熱フラグはセル要素に対してスタガード変数配置と同じインデクスで表す．今，$i$ 方向の $e$ 面が断熱面とすると，**図??**の BC Index の 18 ビット目を 0 にして，BC の情報をエンコードする．断熱条件は，断熱マスク式 (6.14) により実装される．

### 6.1.5 セルボリュームに対する熱境界条件の実装

#### 6.1.5.1 **発熱源**

発熱・吸熱の境界条件は，セルボリュームに対して与える．単位体積あたりの発熱量 $Q'$ [$W/m^3$] が与えられると，発熱項は**式 (6.6)** により無次元化される．ボクセルモデルの ID に対して，XML ファイルで熱源を発熱量 (heat release value) または発熱密度 (heat generation density) により指定する．この例では，ID=246 に対して，10.0 [$W/m^3$] の発熱密度を与えている．

```
<Elem name="Heat_Volume" id="1">
  <Elem name="Heat_Generation" id="246" comment="Desktop">
    <Param name="heat_generation_density" dtype="REAL" value="10.0"/>
  </Elem>
```

```
</Elem>
```

熱源項は，安定性のため一時刻前の値を用いて線形化する．ここで注意すべき点は，発熱源の媒質の物性値の指定である（**式 (6.59)** 中の $\rho'$, $C$)）．単媒質の場合には，指定された媒質の物性値を用いる．現物が複数の媒質から構成されている場合には適切な物性値を推定する必要がある．

$$\theta^{n+1} = \theta^n + \Delta t\, RHS + \Delta t\, \Theta_V^n, \qquad \Theta_V^n = \frac{Q'^n}{\rho' C} \frac{L'}{u_0' \Delta\theta'} \tag{6.59}$$

#### 6.1.5.2　定温熱源

温度を指定する場合には，強制的に指定セルの温度を与える．

```
<Elem name="Heat_Volume" id="1">
  <Elem name="Const_Temperature" id="247" comment="Tape">
    <Param name="temperature" dtype="REAL" value="45.0"/>
  </Elem>
</Elem>
```

#### 6.1.5.3　熱交換器モデル

## 6.2　多媒質の熱流動現象

多媒質の熱流動現象とは，文字通り，複数の媒質にわたる熱流動現象である．流体のみに限ると，多相流や多成分流などがこの範疇に入る．同時に，固体熱伝導現象における多媒質間の熱移動も含む．

### 6.2.1　固体熱伝導と流体の熱移動現象のスケーリング

水とアルミニウムの場合の熱移動現象を考える．水のバルクにおける拡散係数を系の代表的な量と考える．**表 6.4** の熱拡散係数の値から，水とアルミニウムの熱拡散速度の比は，$0.143/96.8 = 1.47 \times 10^{-3}$ である．つまり，水の熱拡散速度はアルミニウムの $10^{-3}$ 倍になり，熱の伝わり方が非常に遅い．この時間スケールが大きく異なる点は計算効率の上で大きな問題点である．一方，鉄と空気の場合には，熱拡散の速度は同じオーダであるので，流動側の代表速度でスケーリングしても問題ない．

表 6.4　物性値の例

| | $\lambda$ $[W/(mK)]$ | $\rho$ $[kg/m^3]$ | $C$ $[J/(kgK)]$ | $\alpha$ $\times 10^{-6}\,[m^2/s]$ |
|---|---|---|---|---|
| Air | $25.7 \times 10^{-3}$ | 1.2 | 1007 | 21.2 |
| Water | $598 \times 10^{-3}$ | 998.2 | 4182 | 0.143 |
| Fe | 80.3 | 7870.0 | 442 | 22.7 |
| Al | 237.0 | 2688.0 | 905 | 96.8 |

固体と流動の熱移動現象を同時に扱う共役熱移動問題を解く場合，対象とする現象によって，以下の 2 つのアプローチが考えられる．

- 流体と固体の方程式に対して同じスケーリングを行い，場を統一的に解く．時間進行に関して境界部での熱流束の計算は正確にできるので，非定常現象を扱うことができる．この場合，現象の時定数が異なると非効率になる．
- 流体と固体側でスケーリングを個別に行い，それぞれ別の方程式を境界で整合する熱流束を授受しながら解き進める．時間スケールが異なることに起因して，合理的な熱流束の授受（境界条件）が必要となる．

対象とする作動流体と固体の物性値と流速によっては，時間スケールの違いから非効率な計算となることがある．そこで，始めに流体の時間スケールで解く方法，その後個別の時間スケールで解く解法について述べる．

#### 6.2.1.1　多媒質の場合の支配方程式

本来は，保存量であるエネルギーを基本変数にした保存系の方程式を解けばよいが，非圧縮領域では分離解法の方が低コストであること，種々のモデリング手法が適用しやすいことから非圧縮近似による非保存のパッシブスカラ方程式を解く．単媒質の場合と同様に，有次元での計算は有効桁数の制約から桁落ちなどの問題が生じるため，$\theta \sim \mathrm{O}(1)$ となるようにできるだけ無次元化して解きたい．支配方程式は**式 (6.4)** であるが，多媒質中における輸送現象は物性値が局所的に異なるため，単媒質の場合の**式 (2.18)** のように単純な形式での無次元化は難しい．そこで，物理熱流束の保存性，離散化時の界面での連続性に注意する．拡散項に対して，多媒質では単一のペクレ数を導入した**式 (6.4)** のような無次元化ができない．したがって，離散化時の桁落ちなどの点を考慮して，半無次元的な実装を行う．**式 (6.2)** から，

$$\frac{\partial \theta}{\partial t} + \frac{\partial}{\partial x_i}(u_i \theta) = \frac{1}{\rho' C}\left[\frac{\partial}{\partial x_i}\left(\lambda \frac{\partial \theta}{\partial x_i}\right)\right]\frac{1}{u_0' L'} + \Theta_V \tag{6.60}$$

**式 (6.60)** は拡散項の計算に物理熱流束の温度勾配のみを無次元化した半無次元的な熱流束を用いている．この半無次元熱流束は，物理熱流束とは $1/(u_0' L')$ の定数倍だけ異なるが，セル界面で連続である．拡散項の係数 $1/(\rho' C)$ はセル中心の温度と結びつく熱容量であるので熱流束の計算とは定義点が別であることに注意する．拡散項は次節のモデリングに従う．

#### 6.2.1.2　拡散項のモデリング

バイナリボクセルを用いて熱伝導を解く場合，物体境界がセル界面に位置していない場合，あるいは，異なる熱伝導率をもつ媒質界面における熱流束の評価が問題となる．対処法として，熱流束を調和平均で表す方法を用いる [19]．**図 6.9** において，セル内で物性値は一定とする．セル P と E の間の界面は，中点 $e$ ではなく $e^*$ の場合を考える．セル界面 $e^*$ における熱流束 $q'_{e^*}$ は，

$$q'_{e^*} = q'_{e^+} = q'_{e^-} \tag{6.61}$$

$$\begin{cases} q'_{e+} = -\lambda_E(\theta'_E - \theta'_{e*})/\Delta x'_{e+} \\ q'_{e-} = -\lambda_P(\theta'_{e*} - \theta'_P)/\Delta x'_{e-} \end{cases} \tag{6.62}$$

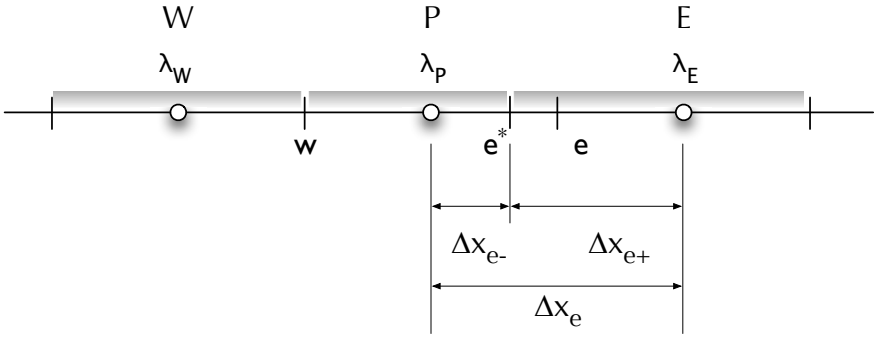



図 6.9　調和平均を用いた拡散項の近似

これより，熱流束と界面温度は以下のようになる．

$$q'_{e*} = -\left(\frac{\Delta x'_{e+}}{\lambda_E} + \frac{\Delta x'_{e-}}{\lambda_P}\right)^{-1}(\theta'_E - \theta'_P) \tag{6.63}$$

$$\theta'_{e*} = \frac{\varphi_E{}'\theta'_E+\varphi_P{}'\theta'_P}{\varphi_E{}'+\varphi_P{}'} \tag{6.64}$$

ただし，$\varphi'_E = \lambda_E/\Delta x'_{e+}$，　$\varphi'_P = \lambda_P/\Delta x'_{e-}$ である．

ここで，セル界面が中点 $e$ の場合を考えてみると，$h'$ をボクセル幅として，

$$\Delta x'_{e+} = \Delta x'_{e-} = h'/2 \tag{6.65}$$

$$q'_{e*} = -2\left(\frac{1}{\lambda_E}+\frac{1}{\lambda_P}\right)^{-1}\frac{\theta'_E-\theta'_P}{h'} \tag{6.66}$$

これは，均質媒質のとき拡散流束の標準的な差分近似と同じである．さらに $\lambda_E \gg \lambda_P$ の場合は $\lambda_E$ の値が大きいので，**式 (6.66)** は

$$q'_{e*} = -\lambda_P\frac{\theta'_E-\theta'_P}{h'/2} \tag{6.67}$$

となる．**式 (6.67)** は，界面 $e^*$ 付近でセル E の温度を $\theta_E$ と近似した上で片側差分により勾配を近似したことを表している．界面での熱伝導率として調和平均を用いる点に注意する．

$$\lambda_{e*} = 2\frac{\lambda_E\lambda_P}{\lambda_E+\lambda_P} \tag{6.68}$$

この方法により，熱拡散係数が異なる媒質間の熱移動を合理的な近似により表現する．

多媒質の場合は，

$$\frac{\theta^{n+1}-\theta^n}{\Delta t} = \frac{1}{\rho' C}\left[\frac{1}{h^2}\left\{\sum_{j=1}^{\dim\times 2}(\gamma\lambda)_j\theta_J - \theta_P\sum_{j=1}^{\dim\times 2}(\gamma\lambda)_j\right\}\right]\frac{1}{u_0{}'L'} - \frac{1}{\rho' C}\left[\frac{1}{h}\sum_{j=1}^{\dim\times 2}(-1)^j\{(1-\gamma)\,q'_{BC}\}_j\right]\frac{1}{u_0{}'\Delta\theta'} + \Theta_V \tag{6.69}$$

ここで $\rho' C$ は解くべきセルの物性値なので，前処理時に方向インデクスを保存しておく．境界面を挟む 2 つのセルのうちどちらが固体要素であるかは，前処理でそのインデクスを特定している．熱伝達境界面をもつインデクスの小さい方のセル（つまり**図 6.3** の (a), (b) の両方のパターンともセル j）に熱伝達境界の attribute ID を保持するようにしている．

**図 6.3** において，熱境界条件を昇順にサーチし，まずインデクス j を得る．この場合，j+1/2 面が熱伝達形式の境界流束となるので，インデクス j に対して j または j+1 のどちらかが参照側の固体となる．(a) の場合，j 要素が固体なので，j を基準として固体要素の方向を示す値 dir=0 となる．一方 (b) の場合，固体要素は j+1 なので j から見ると，dir=1 である．

多媒質（Multi-Medium, MM）の場合には，セル (i, j) に対する固体要素の参照は (i, j+dir) のように参照できる．一方，流体が単媒質（Single-Medium, SM）の場合には，熱伝達境界条件として与える部分の固体の物性に依存するのみであり，この情報はコンポーネントテーブルから読み取る．

# 第 7 章

# ドライバ

本章では，発達したチャネル流などの解析を実施する際に用いられるドライバ部分の実装について説明する．

## 7.1　ドライバセクション

ドライバセクションは，乱流計算などで発達した管路内の流れを流入条件として与える場合に用いられる．**図 7.1** に想定する利用形態を示す．図では左側から流れが流入し，ドライバセクションで発達流を形成し，内部領域での計算結果を評価する．流れを有限の領域で発達させるために，ドライバセクションで周期境界を適用し，発達した流れを下流の内部領域に支配方程式を満たしながら接続することを考える．


図 7.1　ドライバセクションを含む流れ解析.

### 7.1.1　ドライバセクションにおける周期境界の実装

**図 7.2** にドライバセクションの取り扱いを示す．ドライバセクションはインデクス `i=1~ip` の領域で，流出部のインデクスが `i=ip` である．ドライバセクションの目的は，発達した流れを形成し，計算する内部領域へと導くことである．その際に支配方程式を満たすことが重要となる．

外部境界に対する周期境界であれば，単に，ドライバセクションの前後で計算した物理量を相互に交換すればよい．ここで注意すべき点は，セル `ip,ip+1` は計算内点にあり，数値流束 $f_{ip+1/2}$ をドライバセクションと内部領域で一致させる必要があることである．単にデータを交換することは難しく，例えば一旦物理量を待避させ，その間に周期処理を行い，その後待避した物理量を戻すという処理などが必要になる．つまり，特殊な操作をしないと通常のステンシル型の計算スキームでは境界条件を実装しにくい．この点は，処理の複雑さによる処理コストの増加と並列計算時には同期待ちを生じるため，演算性能の低下につながる．

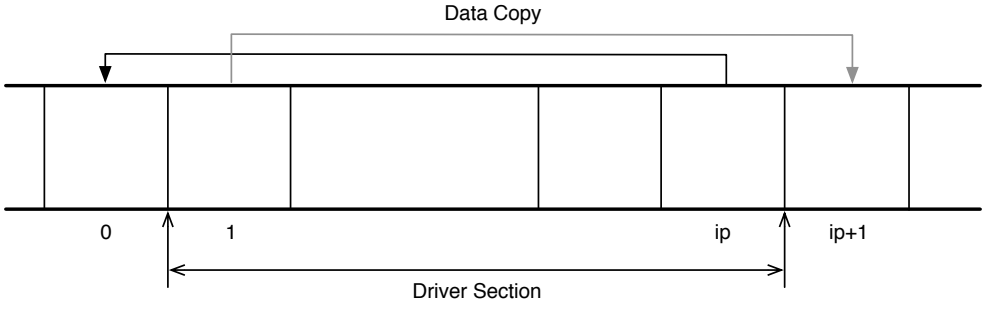



図 7.2　ドライバセクションにおける周期境界処理.

そこで，本来の目的に立ち返り，完全な周期境界でなく，あくまで流れを発達させることに重点をおき，なおかつ処理コストが低い方法を考える．実装する方法は，以下のようになる．

- ドライバセクションでは，計算領域内部に用いる部分的な周期境界条件を実装する．
- ドライバセクションの形状的な条件として，流入部と流出部の断面形状は同一で，座標軸に対して面直であること．必ず計算空間を構成する 6 面のどれかに接続していること．ドライバセクションは複数あっても良い．
- 周期条件は，流出面から流入面への one way とする．計算は，ドライバセクションと内部領域にわたり，同じスキームにより計算する．このため，セル界面における数値流束は保存する．このとき，流入面へとコピーされる

物理量は下流領域の影響を受けた値となるが，支配方程式を満たした値であるので，これを流入側に与えても問題はない．

- ドライバセクションは圧力駆動とする．つまり，ドライバセクションの両端に圧力差を与え，速度はフリーとする[*1]．

#### 7.1.1.1　速度

速度の場合は，利用する空間スキームのステンシルに応じてデータをコピーする．

#### 7.1.1.2　圧力

圧力の場合は，与える圧力差を $p_d$ として，次式のように圧力値をコピーする．

$$p_0 = p_{ip} + p_d \tag{7.1}$$

#### 7.1.1.3　パッシブスカラ

非圧縮の温度のようなパッシブスカラ値については，ドライバセクション流出部の状況によらず，常に流入部で指定値を与える．

### 7.1.2　ボクセルモデルの指定

**図 7.3** にドライバセクションを指定する場合のボクセル ID の例を示す．ドライバセクションの内部とは別の ID で出口断面を与える点に注意する．

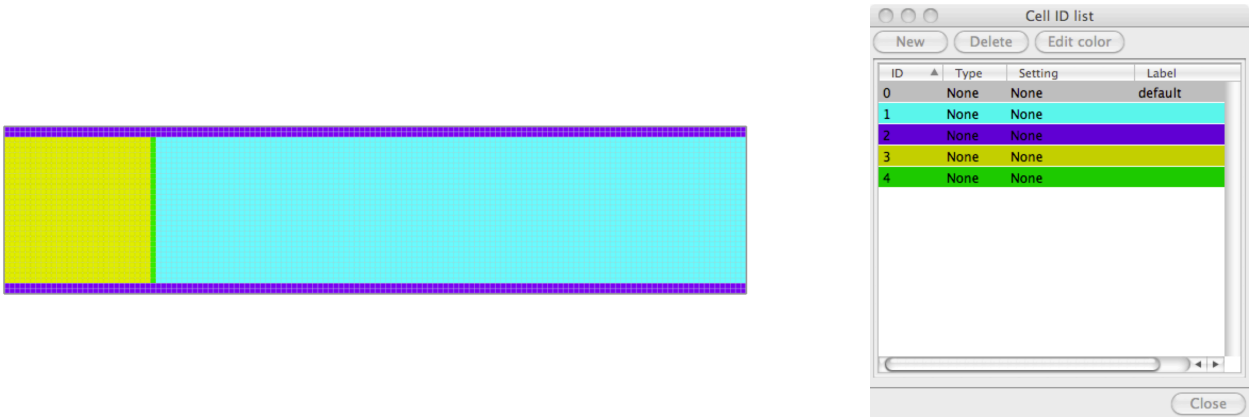



図 7.3　チャネル流のボクセル設定．ID=1:内部流体，ID=2:固体セル，ID=3:ドライバセクション（流体），ID=4:ドライバセクションの流出断面（流体）．

[*1] このため，計算内部領域の背圧によって流量が変化する可能性もある．その場合には，フィードバック機構を組み込み流量調整をする必要がある．

### 7.1.3　境界の指定方法

**図 7.3** を例に，境界条件の指定方法を示す．ドライバセクションを利用するためには，内部境界と外部境界の両方の指定が必要になる．

#### 7.1.3.1　内部境界

```
<InnerBoundary>
  <Elem comment="inner_driver" id="4" name="Periodic">
    <Param dtype="STRING" name="Upstream_Direction"  value="X_minus"/>
    <Param dtype="REAL"   name="Pressure_Difference" value="1.636e-4"/>
    <Param name="def_face"  dtype="INT"  value="1" />
  </Elem>
</InnerBoundary>
```

内部境界で指定する Upstream_Direction と外部境界で指定する Driver_Direction の方向は同じであること．

#### 7.1.3.2　外部境界

```
<OuterBoundary>
  <Elem id="10" name="Periodic">
    <Param dtype="STRING" name="Mode" value="Driver"/>
    <Param dtype="STRING" name="Driver_Direction" value="X_minus"/>
    <Param dtype="int"    name="Driver_Lid_Index" value="28"/>
  </Elem>
</OuterBoundary>
```

Driver_Lid_Index には，流入方向の空間インデクス値を指定する．この値がわからない場合には，適当な値をいれておき，CBC ソルバを実行すると，コンポーネント情報の部分に下記のように表示されるので，その値を入力する．この例の場合には，X_Minus 方向がドライバの方向なので，i_st の値を入力する．

```
>> Component Information

[Periodic]
 no            Label  ID  i_st  i_ed  j_st  j_ed  k_st  k_ed  Pressure Difference [Pa]/[-]  Driver
  1     inner_driver   4    28    28     2    29     2    29     1.63600e-04 / 3.21221e-01      X-
```

# 第 8 章

# 物理量のモニタリングクラス

本章では，計算中の任意点の物理量をモニターする仕組みについて解説する．

物理量モニタリング機能は，ユーザが指定した位置で指定した物理量をファイルに出力する機能である．位置の指定には，ボクセルモデルの ID で指定する方法と XML パラメータファイルで指定する 2 種類がある．ここでは内部実装について説明する．

## 8.1　物理量モニタリング機能 API

物理量モニタリング機能は，全モニタグループを管理する MonitorList クラス，個々のモニターグループを管理する MonitorCompo クラス，各モニタ点でのサンプリング機能を提供する Sampling クラスより構成される．

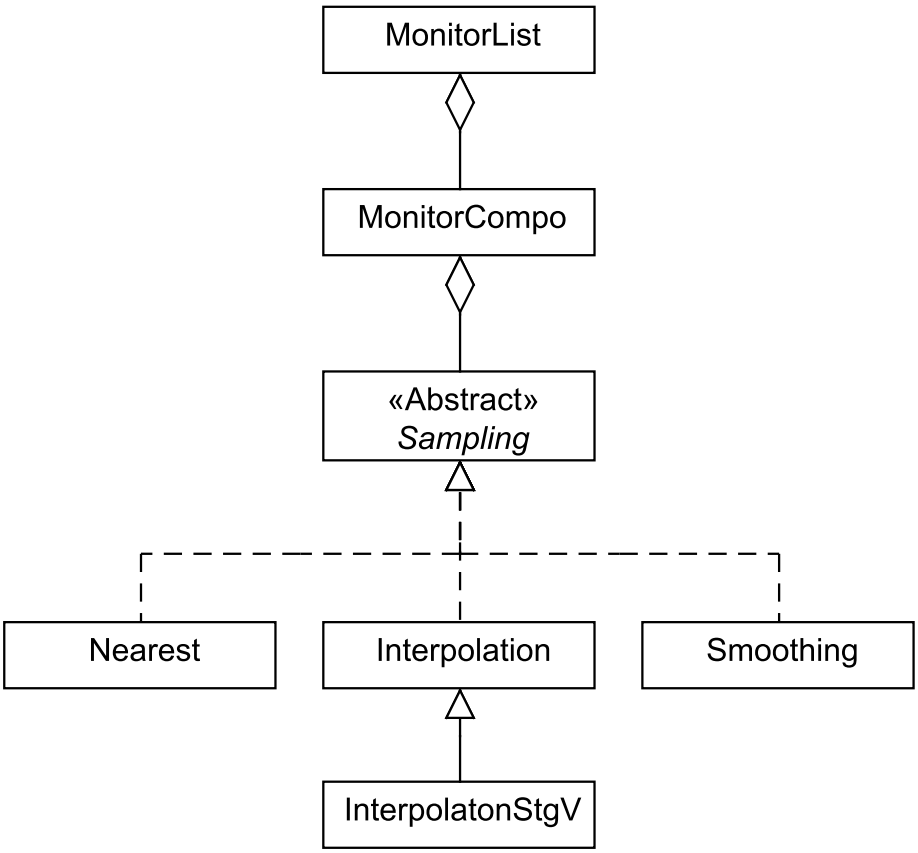



図 8.1　物理量モニタリング機能のクラス図

MonitorList クラスおよび MonitorCompo クラスは，Parallel_Node クラスを継承している．Sampling クラスは抽象クラスで，各モニタ点での実際のサンプリング計算は，Sampling クラスから継承された Nearest クラス，Interpolation クラス，Smoothing クラスが担当する．ソルバーから直接インスタンスされ操作されるのは，MonitorList クラスのみである．

### 8.1.1　MonitorList クラス公開メソッド

**■必須パラメータのコピー**

モニタリング機能の管理およびサンプリング計算に必要なパラメータをコピーする．

| | |
|---|---|
| bcd | BCindex ID 配列 |
| g_org | グローバル領域基点座標 |
| g_lbx | 領域サイズ |
| org | ローカル領域基点座標 |
| dx | セル幅 |
| lbx | 領域サイズ |
| rs | ローカルセルサイズ |
| gc | ガイドセル数 |
| refVelocity | 代表速度 |
| baseTemp | 基準温度 |
| diffTemp | 代表温度差 |
| refDensity | 代表密度 |
| refLength | 代表長さ |
| basePrs | 基準圧力 |
| modeUnit | 出力単位の指定フラグ（有次元，無次元） |
| unitTemp | 温度の単位 |
| modePrecision | 精度のフラグ（単精度，倍精度） |
| unitPrs | 圧力の単位 |
| v00 | 座標系移動速度と成分 |

```
void setControlVars(unsigned* bcd, SKL_REAL g_org[3], SKL_REAL g_lbx[3],
                    SKL_REAL org[3], SKL_REAL dx[3], SKL_REAL lbx[3],
                    unsigned rs[3], unsigned gc,
                    SKL_REAL refVelocity, SKL_REAL baseTemp, SKL_REAL diffTemp,
                    SKL_REAL refDensity, SKL_REAL refLength, SKL_REAL basePrs,
                    unsigned modeUnit, unsigned unitTemp,
                    unsigned modePrecision, unsigned unitPrs)
```

**■参照速度の指定**

引数 v00[4] には，参照格子系の速度と速度成分をもつ配列を指定する．

```
void set_V00(SKL_REAL v00[4])
```

**■出力タイプの設定**

引数 type には，MonitorList::GATHER(単一ファイル出力) または MonitorList::DISTRIBUTE(分散出力) を指定する．

```
void setOutputType(OutputType type)
```

**■point_set グループの登録**

ラベル文字列 labelStr，モニタ対象物理量文字列の vector variables，サンプリング方法文字列 methodStr，サンプリングモード文字列 modeStr，MonitorPoint 構造体の vector pointSet により point_set グループを登録する．

```
void setPointSet(const char* labelStr, vector<string>& variables,
                 const char* methodStr, const char* modeStr,
                 vector<MonitorCompo::MonitorPoint>& pointSet)
```

MonitorPoint 構造体は，個々のモニタ点に対応した構造体である．

```
struct MonitorPoint {
  Vec3r crd;      ///< モニタ点座標
  string label;   ///< モニタ点ラベル（コメント）
  MonitorPoint(const SKL_REAL v[3], const char* str) : crd(v), label(str) {}
  ~MonitorPoint() {}
};
```

**■line グループの登録**

ラベル文字列 labelStr，モニタ対象物理量文字列の vector variables，サンプリング方法文字列 methodStr，サンプリングモード文字列 modeStr，線分端点座標 from,to，分割数 nDivision により line グループを登録する．

```
void setLine(const char* labelStr, vector<string>& variables,
             const char* methodStr, const char* modeStr,
             SKL_REAL from[3], SKL_REAL to[3], int nDivision)
```

**■内部境界条件としてのモニタ指定の有無を調査**

サイズ nBC のコンポーネント配列 cmp を調べ，内部境界条件としてのモニタ指定があれば true を返す．

```
bool hasCellMonitor(CompoList* cmp, int nBC)
```

**■内部境界条件としてのモニタ領域を登録**

サイズ nBC のコンポーネント配列 cmp を参照して，内部境界条件として指定されたモニタ領域を登録する．

```
void setInnerBoundary(CompoList* cmp, int nBC)
```

**■サンプリング元となるデータ配列の登録**

速度変数配列 v，圧力変数配列 p，温度変数配列 t を登録．

```
void setDataPtrs(SKL_REAL* v, SKL_REAL* p, SKL_REAL* t=NULL)
```

**■モニタ点位置情報の出力**

セル ID 配列 id に対して，point_set グループおよび line グループのモニタ点を含むセルに対して ID=255 を書き込む．

```
void write_ID(int* id)
```

**■モニタリング情報を出力**
ファイルポインタ fp に，登録されたモニタリング設定情報を出力する．

```
void printMonitorInfo(FILE* fp)
```

**■出力ファイルのオープン**
文字列 str に XML コンフィギュレーションファイルの OutputData 要素の log_sampling で指定されたファイル名を渡すことにより，各グループ用，各ノード用 (分散出力時) のファイル名が生成され，オープンされる．このメソッドは，出力タイプによらず，全プロセスから呼んでも問題ない．

```
void openFile(const char* str)
```

**■サンプリング計算** (point_set, line)
以下のメソッドにより，point_set グループおよび line グループの各モニタ点でサンプリング計算を行う．

```
void sampling()
```

**■サンプリング計算 (内部境界条件)**
内部境界条件として指定されたモニタ領域のサンプリングには次のメソッドを用いる．このメソッドで，サンプリング結果のノード 0 への集計，モニタ領域内での平均，速度の場合は法線ベクトルとの内積を計算し，コンポーネント cmp への格納までを行う．

```
void samplingInnerBoundary()
```

**■サンプリング結果の出力**
サンプリング時のステップ数 step およびソルバー内部時間 tm とともに，サンプリング結果をファイルに出力する．単一ファイル出力の場合も，このメソッド内部で集約計算を行うため，全プロセスから呼ぶ必要がある．

```
void print(unsigned step, SKL_REAL tm)
```

**■出力ファイルのクローズ**
オープンしていた全ての出力ファイルをクローズする．出力タイプによらず，全プロセスから呼んでも問題ない．

```
void closeFile()
```

## 8.2　ソルバーへの組み込み方法

MonitorList クラスを利用するためには，ヘッダファイル Monitor.h をインクルードする．

### 8.2.1　初期化

以下の 8 ステップが必要:

1. インスタンス化
2. setParallelInfo メソッドによる並列情報の設定
3. setControlVars メソッドによる必須パラメータのコピー
4. setPointSet メソッド，setLine メソッドによる point_set グループと line グループの登録 (これらの操作は，Control::setMonitor メソッド内で行われる)
5. write_ID メソッドにより，モニタ点位置のセル ID に 255 を書き出す
6. setInnerBoundary メソッドによる内部境界条件指定のモニタ領域の登録
7. openFile メソッドによるモニタ結果出力ファイルのオープン
8. setDataPtrs メソッドによるサンプリング元データ配列の登録

```
Control       C;
Parallel_Info pn;
CompoList*    cmp;
…
// (1) インスタンス化
MonitorList MO;
…

// (2) 並列情報設定
MO.setParallelInfo(pn);

// (3) 必須パラメータのコピー
MO.setControlVars(bcd, G_org, G_Lbx, C.org, C.dx, C.Lbx, size, guide,
             C.RefVelocity, C.BaseTemp, C.DiffTemp, C.RefDensity, C.RefLength,
             C.BasePrs, C.Unit.Param, C.Unit.Temp, C.Mode.Precision, C.Unit.Prs);

// (4) Control::setMonitor メソッドの中で，XML ファイルをパースして，
//     setPointSet メソッドにより point_set グループを，
//     setLine メソッドにより line グループを登録している
C.setMonitor(&MO);

// (5) モニタ点位置のセル ID に 255 を書き出す
MO.write_ID(mid);

// (6) 内部境界条件指定のモニタ領域の登録
MO.setInnerBoundary(cmp, C.NoBC);

// (7) モニタ結果出力ファイルのオープン
//    (出力タイプによらず全プロセスで openFile メソッドを呼ぶ)
MO.openFile(C.HistoryMonitorName);

// (8) サンプリング元となるデータ配列の登録
if (C.isHeatProblem()) {
  MO.setDataPtrs(dc_v->GetData(), dc_p->GetData(), dc_t->GetData());
} else {
  MO.setDataPtrs(dc_v->GetData(), dc_p->GetData());
}
```

### 8.2.2　サンプリングおよびファイル出力

非定常計算の場合には，タイムステップループ中に格子の移動速度が変化する場合もあるので，参照速度を各タイムステップ内でモニタークラスに渡す．

```
// SKL_REAL v00[4] で宣言
MO.set_V00(v00);
```

point_set グループおよび line グループに対するサンプリングには sampling メソッドを用いる．サンプリング結果の出力には，全プロセスから print メソッドを呼ぶ．

```
// サンプリング
MO.sampling();

// ファイル出力
// (出力タイプによらず全プロセスで print メソッドを呼ぶ)
MO.print(m_currentStep, (SKL_REAL)SklGetTotalTime());
```

内部境界条件指定によるモニタ領域のサンプリングには samplingInnerBoundary メソッドを用いる．サンプリング結果の出力は，History::printHistoryCompo メソッドが担当する．

```
// サンプリング
MO.samplingInnerBoundary();

// ファイル出力は History クラスが担当
if (pn.ID == 0) H.printHistoryCompo(fp_c, this, cmp, &C);
```

### 8.2.3　サンプリング方法の追加・改良方法

nearest, interpolation, smoothing の各サンプリング方法は，抽象クラス Sampling を継承した，Nearest クラス，Interpolation クラス，Smoothing クラスでそれぞれ実装されている．

サンプリング方法の追加・改良するには，Sampling クラスを継承した新たなクラスを作成する方法と，既存の 3 クラス (Nearest,Interpolation,Smoothing) から継承した新クラスを作成する方法がある．

### 8.2.4　Sampling クラスを継承する方法

Sampling クラスを継承したクラスでは，次の 5 つメソッドを実装する必要がある．

```
// 速度をサンプリング
Vec3r samplingVelocity(const SKL_REAL* v);

// 圧力をサンプリング
SKL_REAL samplingPressure(const SKL_REAL* p);

// 温度をサンプリング
SKL_REAL samplingTemperature(const SKL_REAL* t);

// 全圧をサンプリング
SKL_REAL samplingTotalPressure(const SKL_REAL* v, const SKL_REAL* p);

// 渦度をサンプリング
Vec3r samplingVorticity(const SKL_REAL* v);
```

この時，Sampling クラスで定義された以下の protected メソッドを利用することができる．

**■全圧の計算**
速度 v，圧力 p から計算した全圧値を返す.

```
SKL_REAL calcTotalPressure(const Vec3r v, SKL_REAL p)
```

**■渦度の計算**
速度変数配列 v から，セル位置 index での渦度を計算して返す.

```
Vec3r calcVorticity(const SKL_REAL* v, Vec3i index)
```

**■セルインデックスのシフト**
近傍セルのインデックスを求めるための，セルインデックス index を指定方向にシフトさせる shift1〜shift7 の 7 メソッド，shift_xm〜shift_zp の 6 メソッドが提供される.

```
/// セルインデックスを (1,0,0) 方向にシフト
Vec3i shift1(Vec3i index)
…
…
/// セルインデックスを (1,1,1) 方向にシフト
Vec3i shift7(Vec3i index)

/// セルインデックスを-x 方向にシフト
Vec3i shift_xm(Vec3i index)
…
…
/// セルインデックスを +z 方向にシフト
Vec3i shift_zp(Vec3i index)
```

**■セルの状態 (流体/固体) を調べる**
セル index が流体なら true, 固体なら false を返す.

```
bool isFluid(Vec3i index)
```

**■セルでのスカラー値を取得**
スカラー配列 s のセル index での値を返す.

```
SKL_REAL getScalar(const SKL_REAL* s, Vec3i index)
```

**■セルでのベクトル値を取得**
ベクトル配列 v のセル index での値を返す.

```
Vec3r getVector(const SKL_REAL* v, Vec3i index)
```

### 8.2.5　既存クラス (Nearest,Interpolation,Smoothing) を継承する方法

Nearest クラス，Interpolation クラス，Smoothing クラスのいずれかを継承し，その一部のメソッドを上書きすることにより，基底クラスのサンプリング方法を改良することができる．

現バージョンのソースコード (Sampling.h, Sampling.C) には，既存クラスからの継承のサンプルとして，InterpolationStgV クラスを収録してある．InterpolationStgV クラスは，Interpolation クラスを継承して，スタガード配置の速度変数配列に対応させたものである．

### 8.2.6　端点処理

線分の端点が計算対象領域境界上にある場合には，そのモニタ点がガイドセル側に属すると判断されることがあり，sampling_mode=fluid と指定したにもかかわらずモニタ点を含むセルが固体セルであるため，警告メッセージ「`*skip(unexpected solid)*`」が出力されることがある．

この現象を防止するために，line グループ指定時の線分端点座標を，常に実際の計算対象領域よりわずかに小さい領域にクリッピングする仕様としている．

# 第 9 章

# 計算性能モニタクラス

本章では，性能測定モジュールの機能について説明する．

## 9.1　計算性能測定機能

計算性能測定機能は，ソルバー実行時に，プログラムコード内の測定対象部分の実行時間を測定し，その統計情報を出力する機能を提供する．

各測定箇所は，整数値キー番号により識別される．測定箇所はキー番号以外に，次のプロパティを持つ．

**ラベル**　統計情報出力時のラベル文字列
**測定対象タイプ**　「計算時間」または「通信時間」
**排他測定フラグ**　「排他測定」または「非排他測定」

**■測定対象タイプ**　測定を終了する stop メソッドには，オプションとして測定区間の「計算量」をユーザが自己申告できる機能がある．測定対象タイプが計算時間の場合には，この申告量を浮動小数点演算量と解釈し，統計情報出力時に計算速度 (FLOPS 値) を出力する．一方，測定対象タイプが通信時間の場合には，申告量をバイト単位の通信量と解釈し，統計情報出力時に通信速度 (Byte/s 単位) を出力する．

**■排他測定と非排他測定**　測定箇所は，排他測定フラグの真偽により，「排他測定」と「非排他測定」とに分類される．排他測定は，「(ほぼ) 完成されたプログラムの実行時の状況を把握するために，プログラムコードを排他的な領域に分割し，各領域の実行時間を測定する」という使用方法を想定している．排他測定では，統計情報出力時に，「全排他測定箇所の合計実行時間」に対する「個々の排他測定箇所の実行時間」の割合も出力する．

一方，非排他測定は，プログラムのデバッグあるいはチューニング時に，一時的に興味のある対象領域の実行時間を把握するのに使用することを想定している．そのため，非排他測定区間は，排他測定区間や他の非排他測定区間と重ねることも可能で，自由に設置できる．

**■並列実行時の制限**　並列実行時には，それぞれの排他測定区間は，並列計算ノードによらず同一回数だけ実行されるものと仮定している．計算ノード毎に実行回数が異なっていた場合には，後述する print メソッド実行時には，該当区間の統計情報は出力されない．

一方，非排他測定区間の設置には制限はなく，並列計算ノード毎に実行回数が異なっていても問題ない．非排他測定区間の測定結果の出力には printDetail メソッドを使用する．

### 9.1.1　API 説明

計算性能測定機能は，個々の測定箇所毎に時間測定を担当する PerfWatch クラスと，全 PerfWatch を管理して統計情報の計算と出力を担当する PerfMonitor クラスにより提供される．両クラスとも，Parallel_Node クラスを継承している．

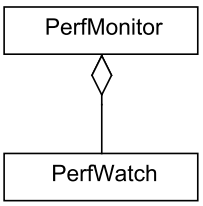



図 9.1　計算性能測定機能 クラス図

PerfWatch クラスは，全て PerfMonitor クラスをとおして生成され操作されるので，ソルバーから直接 PerfWatch クラスの API を呼ぶことはない．

#### 9.1.1.1　PerfMonitor クラス公開メソッド

**■初期化**

測定区間数 (nWatch) ＋ 1 個の測定時計 (PerfWatch クラス) をインスタンスして，全計算時間用測定時計をスタートさせる．測定区間を指定するキー番号は，0 以上 nWatch 未満である必要がある．

```
void initialize(unsigned nWatch)
```

**■測定区間にプロパティを設定**

キー番号 key で識別される測定区間に，ラベル文字列 label, 測定対象タイプ type, 排他測定フラグ (true=排他測定/false=非排他測定) を設定.

```
void setProperties(unsigned key, const string& label, Type type,
                   bool exclusive=true)
```

なお，測定対象タイプ type の指定には，以下の定数を使用する．

**通信**　PerfMonitor::COMM

**計算**　PerfMonitor::CALC

**■測定スタート**

キー番号 key で識別される測定区間の始まりを指定する．

```
void start(unsigned key)
```

**■測定ストップ**

キー番号 key で識別される測定区間の終わりを指定する．オプションとして,「タスク」あたりの計算量/通信量 (バイト単位)flopPerTask，実行「タスク」数 iterationCount を申告できる．

```
void stop(unsigned key, SKL_REAL flopPerTask=0.0, unsigned iterationCount=1)
```

**■測定結果の集約**

測定結果情報をノード 0 に集約し，統計情報を計算する．また，初期化時にスタートさせた全計算時間用測定をストップさせる．

```
void gather()
```

**■測定結果の出力**

排他測定区間についての統計情報を出力する．ノード 0 以外では, 呼び出されてもなにもしない.

```
void print(FILE* fp)
```

**■詳細な測定結果の出力**

非排他測定区間も含めて，詳細な統計情報を出力する．ノード 0 以外では, 呼び出されてもなにもしない.

```
void printDetail(FILE* fp)
```

### 9.1.2　PerfMonitor クラスの使用方法

PerfMonitor クラスを使用するには，ヘッダファイル PerfMonitor.h インクルードする必要がある．

**■初期化**

以下の 8 ステップが必要:

1. インスタンス化
2. 測定番号識別キー番号定数の用意
3. setParallelInfo メソッドによる並列情報の設定
4. initialize メソッドによる測定区間数の登録
5. setProperties メソッドによる各測定区間へのプロパティの設定

```
Parallel_Info pn;
…
// (1) インスタンス化
PerfMonitor PM
…
// (2) 測定番号識別キー番号定数
enum timing_key {
  tm_sec1,
  tm_sec2,
  …
  tm_secN,
  tm_END_,   // ターミネータ（＝測定区間総数）
};

…
// (3) 並列情報設定
PM.setParallelInfo(pn);

// (4) 測定区間数の登録
PM.initialize(tm_END_);

// (5) 各測定区間にプロパティを設定
PM.setProperties(tm_sec1, "SEC1", PerfMonitor::CALC);       //計算，排他
PM.setProperties(tm_sec2, "SEC2", PerfMonitor::COMM);       //通信，排他
PM.setProperties(tm_sec3, "SEC3", PerfMonitor::CALC, false);//計算，非排他
…
```

**■測定区間の指定**　単純な指定では，測定対象区間を同一キーを指定した start メソッドと stop メソッドで挟む．

```
PM.start(tm_sec1);

/* 測定対象区間 */

PM.stop(tm_sec1);
```

以下の例では，測定対象区間内で浮動小数計算量 flop のブロックを nStep 回実行していることを申告してる．

```
PM.start(tm_sec1);

for (int i = 0; i < nStep; i++) {
  /* このブロックの浮動小数計算量が flop */
}

PM.stop(tm_sec1, flop, nStep);
```

**■測定結果の集計・出力**

測定結果出力メソッドを呼び出す前に，全ノードで集計メソッド gather を呼び出す必要がある．出力メソッドは print および printDetail は，ノード 0 から呼ばれた時のみ指定されたファイルへ測定結果情報を出力する．

```
// 測定結果の集計（gathre メソッドは全ノードで呼ぶこと）
PM.gather();

// コンソールへの測定結果の出力
// （print と printDetail メソッドはノード 0 以外は呼ばれても何もしない）
PM.print(stdout);
PM.printDetail(stdout);

if (p.ID == 0) {
  FILE* fp = fopen("timing.txt", "w");

  // ファイルへの測定結果の出力
  // （print と printDetail メソッドは，ノード 0 のみから呼び出しても OK）
  PM.print(fp);
  PM.printDetail(fp);

  fclose(fp);
}
```

**■クラス変数 TimingLevel による測定レベル制御**

クラス変数 TimingLevel の内容を

$$\text{TimingLevel} = \begin{cases} 0 & \text{測定なし} \\ 1 & \text{排他測定のみ実行} \\ 2 & \text{非排他測定も実行} \end{cases}$$

と設定することにより，ヘッダファイル PerfMonitor.h 内で定義された 2 つのマクロ PM_TIMING__, PM_TIMING_DETAIL__をとおして，ソルバー実行時の測定レベルを制御できる．

```
#define PM_TIMING__          if (PerfMonitor::TimingLevel > 0)
#define PM_TIMING_DETAIL__   if (PerfMonitor::TimingLevel > 1)
```

```
  …
PM_TIMING__ { MO.start(...); }

  /* 排他測定区間 */

PM_TIMING__ { MO.stop(...); }
  …
  …
PM_TIMING_DETAIL_ { MO.start(...); }

  /*  非排他測定区間  */

PM_TIMING_DETAIL_ { MO.stop(...);
```

### 9.1.3　統計情報出力内容

**■print メソッド出力内容**

「Report of Timing Statistics」に続いて

Total execution time
　ノード 0 における，initialize メソッド呼び出しから gather メソッド呼び出しまでの時間
Total time of measured sections

全排他測定区間における「積算実行時間のノード間平均」の合計

「Statistics per MPI process [Node Average]」に続いて，各排他測定区間毎に

call
: 測定回数 (「start メソッド〜stop メソッド」ペアの呼び出し回数)

accumulated time
: 積算実行時間 (avr[sec] はノード間平均値，avr[%] は「Total time of measured sections」値に対する平均値の割合，sdv[sec] は標準偏差，avr/call[sec] は 1 測定あたりの平均値)

flop | messages[Bytes]
: 積算浮動小数演算量または通信量 (avr はノード間平均値，speed は積算実行時間のノード間平均値より求めた計算速度または通信速度)

なお，排他測定区間における測定回数がノード毎に異なっていた場合には，その排他測定区間に関する統計情報の計算は行われず，「NA」と出力される．

■printDetail **メソッド出力内容**　各測定区間，各ノード毎に，以下を出力

accm[s]
: 積算実行時間

accm[%]
: 積算実行時間の「Total time of measured sections」値に対する割合

waiting[s]
: 待ち時間 (「積算実行時間のノード間最大値」と「積算実行時間」の差)

accm/call[s]
: 1 測定あたりの実行時間

flop|msg
: 積算浮動小数演算量または通信量 (バイト単位)

speed
: 積算実行時間より求めた計算速度または通信速度

なお，非排他測定区間については，ラベル文字列をその前に「*」をつけて表示する．

# 第 10 章

# データ保持クラス

本章では，測定データを基に得られた実験式を条件とするような場合に利用するデータ保持機能について説明する．

## 10.1　時系列データ保持機能

時系列データ保持機能では，ソルバー実行開始時に，指定されたファイル (入力データファイル) より時系列データ (データセット) を読み込み，ソルバー実行中にその値を参照できる機能を提供する．

入力データセットは時系列だけでなく，速度値に対する圧力損失値のように，定義域が時間以外の物理量の場合にも対応可能である．以下では便宜上，参照時にキーとなる量を「時刻」，参照したい量を「値」と記すことにする．

データ値は，入力データセット毎に付けられたラベルと，時刻をキーにして以下の規則により参照される．

- 指定された時刻のデータがデータセット内に存在した場合，その時刻での値
- 指定された時刻がデータセット内の最小時刻より小さい場合，最小時刻データの値
- 指定された時刻がデータセット内の最大時刻より大きい場合，最大時刻データの値
- その他の場合，指定された時刻を挟む 2 つの時刻データから，線形補間により値を計算

### 10.1.1　XML コンフィギュレーションファイルによる入力データファイル指定方法

XML コンフィギュレーションファイルの Parameter セクションの Data_Holder 要素内に，データセット数分の Param 要素を記述する．各 Param 要素には，name 属性にデータの内容を表すラベル文字列を，文字列型の value 属性にデータファイルのパスを指定する．ソルバー内部では，このラベル文字列によりデータセットの識別を行う．

```
<Parameter>
  …
  <Elem name="Data_Holder">
    <Param name="label1" dtype="STRING" value="data/data1.dat" />
    <Param name="label2" dtype="STRING" value="data/data2.dat" />
  </Elem>
  …
</Parameter>
```

上例では，data ディレクトリの 2 つのデータファイル data1.dat, data2.dat を，それぞれ「label1」「label2」というラベルを付けて指定している．

### 10.1.2　入力データファイルのフォーマット

入力データファイルは，テキストファイルとし，各行に一組ずつ時刻と値を次の規則に従って記述する．

- 時刻と値は，空白またはカンマで区切るものとする．
- 空白とは 1 つ以上の，半角スペースまたはタブである．
- 時刻の前，値の後ろ，カンマの前後に，空白を入れることもできる．
- 空行および「#」で始まるコメント行を任意の場所に挿入できる．
- 時刻と値の後ろに，さらに空白またはカンマを挟んでレコードが続く場合，それらは無視される．

なお，時刻は昇順にソートされている必要はないが，同一時刻のデータが複数存在する場合にはエラーとして扱い，ソルバーの実行を停止する．

```
#
# 1 行に 2 つの実数「時刻」と「値」を空白またはカンマ区切りで記述
#
  時刻１    値１   # 空白区切り
  時刻２, 値２     # カンマ区切り
  時刻３,  値３   # カンマ＋空白区切り
   …         …
  時刻ｎ    値ｎ
```

### 10.1.3　API 説明

時系列データ保持機能は，個々のデータセットの内容を保持管理する DataHolder クラスと，DataHolder クラス全体を管理する DataHolderManager クラスからなる．なお，DataHolderManager クラスは，Parallel_Node クラスから継承されている．

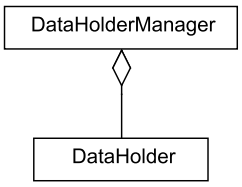



図 10.1　時系列データ保持機能 クラス図

#### 10.1.3.1　DataHolderManager クラス公開メソッド

**■DataHolder の取得**

ラベル label を指定して，対応するデータセットを保持管理している DataHolder クラスオブジェクトへのポインタを返す．対応するデータセットがない場合には 0 を返す．

```
DataHolder* getDataHolder(const string& label)
```

**■DataHolder 数の取得**

管理下にある DataHolder 数を返す．

```
unsigned getNumDataHolder() const
```

**■DataHolder 情報の出力**

管理下にある全 DataHolder の情報を出力する．

```
void printInfo(FILE* fp) const
```

**■DataHolder 情報の出力 (デバッグ用)**

管理下にある全 DataHolder の情報 (全データ内容を含む) を出力する．

```
void printInfoDebug(FILE* fp) const
```

**■XML ファイルポインタの受け取り**

XML コンフィギュレーションファイル (SklCfg::SklSolverConfig クラス) へのポインタを受け取る．ポインタの内容が不正な場合，false を返す．

```
bool receiveCfgPtr(SlkCfg::SklSolverConfig* cfg)
```

**■データファイル読み込み**

XML コンフィギュレーションファイルをパースし，入力データファイルを読み込み，内部に DataHolder を生成し登録する．

```
void readData()
```

#### 10.1.3.2　DataHolder クラス公開メソッド

**■時刻を指定して値を取得**

指定された時刻 t に対して，格納データを参照して，線形補間により求めた値を返す．指定された時刻が入力データの定義域外の場合には，最初あるいは最後の (最も近い時刻での) 値を返す．

```
SKL_REAL getValue(SKL_REAL t) const
SKL_REAL get_value(SKL_REAL t) const
```

**■データ値のスケーリング**

指定された倍率で，全時刻での値をスケーリングする．データ値の単位変換などでの使用を想定している．

```
void scale(SKL_REAL ratio)
```

**■時刻の最小値・最大値の取得**

時刻 (データの定義域変数) の最小値・最大値を返す．

```
SKL_REAL getMin() const
SKL_REAL getMax() const
```

**■格納データ情報の出力**　格納データ情報 (ラベル，ファイル名，データ数，時刻の最小値・最大値) を出力する．

```
void printInfo(FILE* fp) const
```

**■格納データ情報の出力 (デバッグ用)**　全格納データを出力する．

```
void printInfoDebug(FILE* fp) const
```

### 10.1.4　ソルバーへの組み込み方法

DataHolder クラスおよび DataHolderManager クラスを利用するには，ヘッダファイル DataHolder.h をインクルードする必要がある．

**■初期化**

以下の 4 ステップが必要:

1. インスタンス化
2. setParallelInfo メソッドによる並列情報の設定
3. receiveCfgPtr メソッドによる XML コンフィギュレーションファイルポインタの設定
4. readData メソッドによる入力データファイルの読み込み

```
Parallel_Info pn;
SklCfg::SklSolverConfig* m_solvCfg;
...
// (1) インスタンス化
DataHolderManager DH;
...
// (2) 並列情報設定
```

```
DH.setParallelInfo(pn);

// (3) XML コンフィギュレーションファイルポインタ設定
if (!DH.receiveCfgPtr(m_solvCfg)) assert(0);

// (4) 入力データファイル読み込み
DH.readData();
```

**■データセットの参照**

以下の例のように，DataHolder クラスのポインタ変数を通して使う.

```
DataHolderManager DH;
…
/* ここで DH の初期化 */
…
// データセット「label1」へのポインタを取得
DataHolder* data1 = DH.getDataHolder("label1");

// エラー処理
if (!data1) {
  printf("Error: no DataHolder labeled 'label1'.\n");
  …
}

SKL_REAL t = …;  // 時刻

// 時刻 t での値を取得
SKL_REAL x = data1->getVal(t);
```

また，次の例のように，DataHolderManager のみを通して使うことも可能.

```
DataHolderManager DH;
…
/* ここで DH の初期化 */
…
SKL_REAL t = …;  // 時刻

// データセット「label1」の時刻 t での値を取得
SKL_REAL x = DH.getDataHolder("label1")->getVal(t);
```

# 第 11 章

# コンポーネントの実装

本章では，CBC ソルバークラスにおけるコンポーネントの実装について説明する．

## 11.1　内部境界条件の実装のしくみ

### 11.1.1　BC Index

計算空間の各セルおよびセルの各フェイスへの境界条件処理は，機能性と計算効率の点で注意深く実装すべき部分である．空間スキームのカーネル部分と比べて，条件分岐などが多くベクトル化，SIMD 化が利きにくいため，計算量を最小化する工夫が必要となる．境界条件の空間的存在範囲は局所的な傾向であり，その計算は並列処理時のロードバランスを崩す要因なので，この点にも気をつける必要がある．CBC の実装方針として，以下のポリシを基本とする．

1. 局所的な処理
   境界条件の存在範囲を探索する領域を最小化するため，BV 情報を利用する．
2. 演算量を最小化
   前処理可能な部分は，保存メモリ量も少なくするようにして保持する．
3. ２パス計算
   通常の空間演算処理を行い，境界処理が必要な部分だけ計算し，修正量を加える．
4. 高性能計算
   通常の空間演算処理においても，計算空間内の任意部分に現れる固体壁面の処理を効率よく行う工夫が必要になる．また，反復計算において，係数行列のロードコストを削減することも高速計算に必要になる．

CBC ソルバでは，様々な境界条件の情報を纏めて管理し，計算時に境界条件を効率的に実装するため，**図 11.1** および**図 11.2** に示す BC Index，CompoList クラス，MaterialList クラスを利用する．BC Index は，計算処理に必要なフラグ情報を 4 バイトのビットフィールドに機能を持たせたフラグをコンパクトに割り当て，領域空間における境界条件の設定のために利用する．CompoList クラスは境界条件の種類やパラメータなどの情報，MaterialList クラスはセル属性を保持する．

bcd[] は計算空間を構成する各セルに対して割り当てられた配列で，**表 11.1** に示すように，共通的に利用する情報を保持する．Component および Material は，それぞれ，CompoList と MaterialList クラスオブジェクト配列のエントリを格納する領域である．bh2[] でも CompoList へのエントリをエンコードするが，bcd[[] は全てのコンポーネントを保持する．一方，bh2[] では熱境界条件に必要なコンポーネントのみ保持する．Cell ID はセルに割り当てられている ID を保持する．この ID はユーザが作成するボクセルモデルにより指定される．Volume Fraction はセル内における流体の体積占有率を 0 ~ 255 の値で量子化した値を保持する．Forcing は？．State はセルの状態を指定するフラグで，排他的である．Active は対象セルが計算に有効かどうかを指定し，Kind_of_Solver の指定するモードにより異なる．

表 11.1　BC Index bcd[] に割り当てるビットフラグ（4 バイト幅）

| Bit | Flag | Property | Explanation |
|---|---|---|---|
| 31 | Active | 0-Invalid / 1-Valid | 対象セルが計算に有効かどうかを指定するフラグ |
| 30 | State | 0-Solid / 1-Fluid | セルの状態を指定するフラグ（排他的） |
| 28 | Forcing | | |
| 20~27 | Volume Fraction | 0-255 | 8bit で量子化された体積率 [0,1] |
| 12~19 | Cell ID | 1-255 | 8bit 幅の正数 |
| 6~11 | Material | 1-63 | MaterialList[] へのエントリ（6bit 幅） |
| 0~5 | Component | 1-63 | CompoList[] へのエントリ（6bit 幅） |

bcp[] は**表 11.2** に示すように，基本的に圧力計算に必要な情報を保持している．No_Sface は粘性項の修正に利用するための係数で実験的な実装である．Facing_?は，壁法則の計算などに利用するための情報で各セル方向に壁面があるかないかを示す．BC_Ndag_?は反復計算時に利用する行列の対角要素の係数である．BC_Ndag_?は同じく非対角要素の係数である．BC_N_?と BC_D_?はぞれぞれ，Neumann 条件と Dirichlet 条件のフラグであり，両者は同時に 0 ではあり

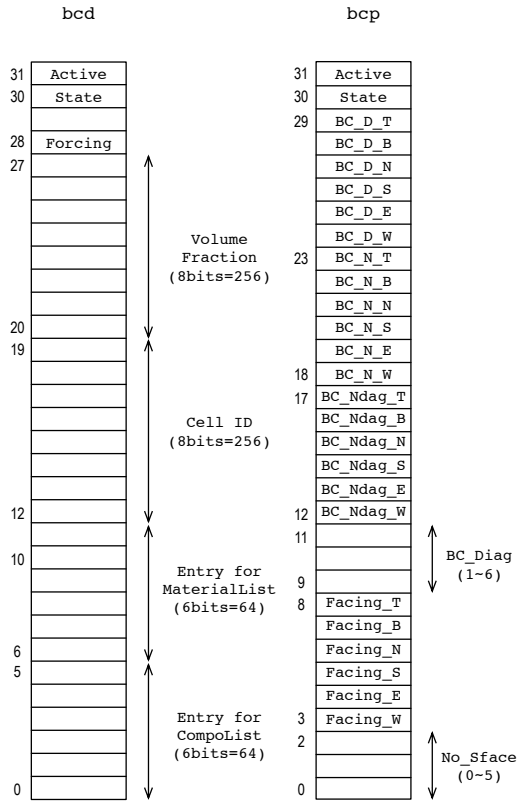



図 11.1　境界条件実装に用いる BC Index. bcd[] は共通要素，bcp[] は圧力計算に用いる．

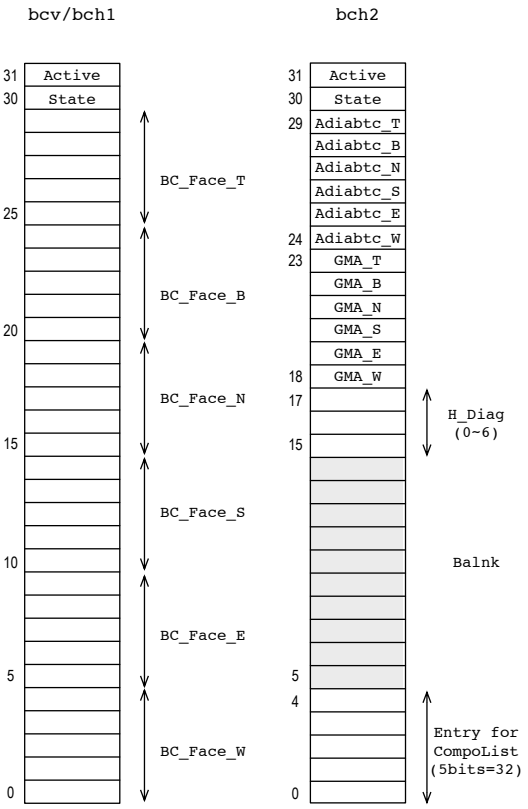



図 11.2　境界条件実装に用いる BC Index. bcv[] は速度計算，bch1, bch2[] は温度計算に用いる．

えない.

表 11.2　BC Index bcp[] に割り当てるビットフラグ（4 バイト幅）

| Bit | Flag | Property | Explanation |
|---|---|---|---|
| 31 | Active | 0-Invalid / 1-Valid | 対象セルが計算に有効かどうかを指定するフラグ |
| 30 | State | 0-Solid / 1-Fluid | セルの状態を指定するフラグ（排他的） |
| 24~29 | BC_D_? | 0-Dirichlet / 1-Non Dirichlet(Default) | 各方向のセルフェイスの Dirichlet 境界フラグ |
| 18~23 | BC_N_? | 0-Neumann / 1-Non Neumann(Default) | 各方向のセルフェイスの Neumann 境界フラグ |
| 12~17 | BC_Ndag_? | 0-境界条件あり / 1-境界条件なし | 各セルフェイス方向の係数 |
| 9~11 | BC_Diag | [1-6] | 対角要素の係数 |
| 3~8 | Facing_? | 0-壁面なし / 1-壁面あり | 各セルフェイス方向の壁面の有無 (wall location) |
| 0~2 | No_Sface | [0-5] | セルの壁面の面数 |

**表 11.3** に示す bcv[] は速度の計算に用いる境界条件情報である．各方向に 5bit 幅[*1]をもつ．0 は境界条件がなく，標準スキームでの流束計算を行う．1-30 は境界条件で CompoList[] へのエントリを示す．31 は外部境界条件であることを示す.

表 11.3　BC Index bcv[] に割り当てるビットフラグ（4 バイト幅）

| Bit | Flag | Property | Explanation |
|---|---|---|---|
| 31 | Active | 0-Invalid / 1-Valid | 対象セルが計算に有効かどうかを指定するフラグ |
| 30 | State | 0-Solid / 1-Fluid | セルの状態を指定するフラグ（排他的） |
| 25~29 | BC_Face_T | 0-BC なし / 1-30 エントリ / 31-外部 BC | 各セル T 方向の BC |
| 20~24 | BC_Face_B | 0-BC なし / 1-30 エントリ / 31-外部 BC | 各セル B 方向の BC |
| 15~19 | BC_Face_N | 0-BC なし / 1-30 エントリ / 31-外部 BC | 各セル N 方向の BC |
| 10~14 | BC_Face_S | 0-BC なし / 1-30 エントリ / 31-外部 BC | 各セル S 方向の BC |
| 5~9 | BC_Face_E | 0-BC なし / 1-30 エントリ / 31-外部 BC | 各セル E 方向の BC |
| 0~4 | BC_Face_W | 0-BC なし / 1-30 エントリ / 31-外部 BC | 各セル W 方向の BC |

温度の計算に用いる境界条件情報のひとつ bch1[] を**表 11.4** に示す．各方向に 5bit 幅をもつ．0 は境界条件がなく，標準スキームでの熱流束計算を行う．1-30 は境界条件で CompoList[] へのエントリを示す．速度の場合と同様に，外部境界面上はコンポーネントの指定は行わない.

表 11.4　BC Index bch1[] に割り当てるビットフラグ（4 バイト幅）

| Bit | Flag | Property | Explanation |
|---|---|---|---|
| 31 | Active | 0-Invalid / 1-Valid | 対象セルが計算に有効かどうかを指定するフラグ |
| 30 | State | 0-Solid / 1-Fluid | セルの状態を指定するフラグ（排他的） |
| 25~29 | BC_Face_T | 0-BC なし / 1-30 エントリ | 各セル T 方向の BC |
| 20~24 | BC_Face_B | 0-BC なし / 1-30 エントリ | 各セル B 方向の BC |
| 15~19 | BC_Face_N | 0-BC なし / 1-30 エントリ | 各セル N 方向の BC |
| 10~14 | BC_Face_S | 0-BC なし / 1-30 エントリ | 各セル S 方向の BC |
| 5~9 | BC_Face_E | 0-BC なし / 1-30 エントリ | 各セル E 方向の BC |
| 0~4 | BC_Face_W | 0-BC なし / 1-30 エントリ | 各セル W 方向の BC |

[*1] 6 方向 $\times 5bit < 4B$

表 11.5　BC Index bch2[] に割り当てるビットフラグ（4 バイト幅）

| Bit | Flag | Property | Explanation |
|---|---|---|---|
| 31 | Active | 0-Invalid / 1-Valid | 対象セルが計算に有効かどうかを指定するフラグ |
| 30 | State | 0-Solid / 1-Fluid | セルの状態を指定するフラグ（排他的） |
| 24~29 | Adiabatic_? | 0-断熱 / 1-非断熱 | 各フェイスの断熱指定状況 |
| 18~23 | GMA_? | 0-BC あり / 1-BC なし | 各フェイスの境界条件の有無 |
| 15~17 | H_Diag | [1-6] | 行列の対角要素の係数 |
| 5~15 | Blank | | 未使用 |
| 0~4 | CompoList | 1~31 | 境界条件のエントリのみ |

### 11.1.2　コンポーネント

内部境界条件は，CBC ソルバークラスのコンポーネントという概念で実装している．コンポーネントは，ボクセルモデル内に設定された ID の位置や付随する条件などを管理する機能である．コンポーネントリストに登録された ID は，境界条件やモニタ指定位置，媒質などの要素として利用される．

#### 11.1.2.1　内部境界条件の扱い

熱流動解析の場合の境界条件を内部境界条件として指定する場合，通常，固体面側を基準に境界条件を与える．これは，流体中で固体面が境界条件となることが多く，指定に便利なためである．したがって，内部境界条件の指定時にキーとなる ID には固体セルを指定する．一方で，固体熱伝導を解く場合には，熱流動解析とは逆に，流体面側を基準に境界条件を与える．

前処理の手順を以下に簡単に示し，後で詳述する．

1. セル ID の存在範囲の探索
   ボクセルモデルの ID を探索して，コンポーネントの存在範囲の BV 情報を得る．このため `getLocalCmpIdx()` 内で，V-Sphere のメソッド `GetBndIndexExtGc()` を用いてコンポーネントが存在する範囲の BV(Bounding Vox) を探索し，各ノード内のローカルインデクス情報を取得する．つぎに，得られた BV 情報から一層だけ拡大したインデクスを `SklSolverCBC::getEnlargedIndex()` で計算する（11.1.3 参照）．一層拡大して BV 情報を取得する理由は，キーとなる ID は固体セルであるが，境界条件の対象となるセルは固体 ID に接する流体セルであるので，そのセルまで含めるためである．グローバルインデクスはローカルインデクスから計算する．グローバルなインデクスはソルバークラスのメンバ変数（`int* GC_bv`）で管理し，ローカルなインデクスはコンポーネントクラスのメンバ変数（`st[3], ed[3]`）で管理する．

2. ビットフラグのエンコード
   BC Index は，計算空間の各セルに対して unsigned int (4bytes) の変数が割当てられている．この中に境界条件処理に関する情報をビットフィールドにエンコードする．BC Index 導入の目的は，諸情報をコンパクトに保持し，(1) メモリ領域の節約，(2) ロードコストを減らすことである．BC Index にエンコードされるフラグは，境界条件設定時の空間ループ内の演算コストが小さくなるように実装する．境界条件処理時のロードコストの削減が主目的であるので，**図 11.1**，**図 11.2** に示すように，必要に応じて幾種類かの BC Index を併用する．

3. コンポーネント BV 情報の再構築
   1 の処理で，コンポーネントの BV は実際のコンポーネントの存在範囲の 1 つ外側を示していた．2 の処理で境界条件に必要な情報をエンコードし，実際に探索する範囲が確定した．そこで，実行時の探索範囲を局所的に抑え，実行時間を短縮するため，BV インデクスを再度サーチし確定する．探索には，境界条件処理で利用する探索アルゴリズム（BCindex 情報を利用）を用いて，`BinaryVoxel::resizeCompoBV()` により範囲を定める．

再検索処理は，セルフェイス resizeBVface() とセル要素 resizeBVcell() に対する 2 種類の処理がある．処理手順は，setBCIndexP(),setBCIndexV(),setBCIndexH() に倣う．ここでは，ローカルインデクスとグローバルインデクスの両方を更新している．

### 11.1.3　ID の探索処理

コンポーネントとして保持する ID について，ID の存在範囲（BV, Bounding Vox）の算出方法について述べる．ID の BV 情報となる条件は，**キー ID が存在する BV を一層拡大した矩形範囲が自領域内にあること**である．パターンとして，**図 11.3** に示すようなケースが考えられる．領域内の最外側の層に対する処理を考えなければならないため，ガイドセル上の ID を考慮する必要がある．

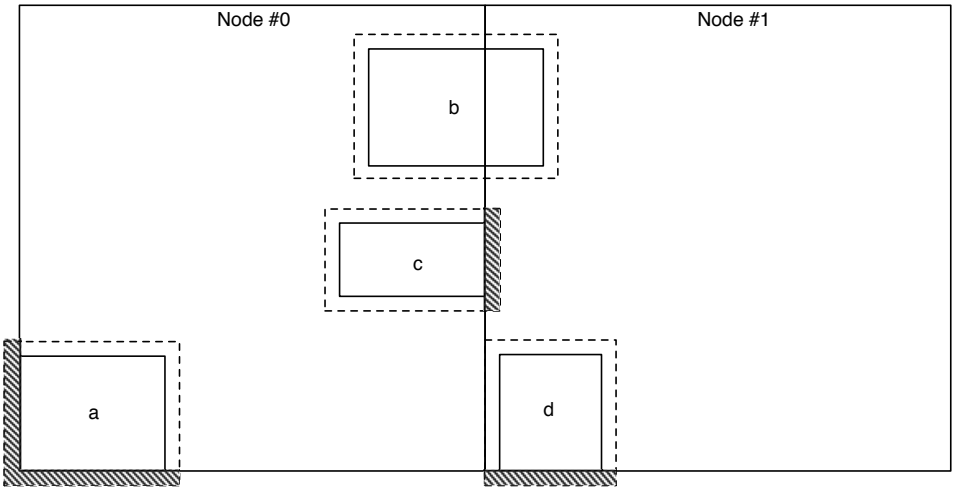



図 11.3　計算領域内に存在する ID のパターンと BV 情報の計算．図中の a, d のハッチング部分は計算領域外なので，BV の対象外となる．一方，c のハッチング部分は，キー ID は領域#0 内，つまり領域 #1 から見るとガイドセル上にあるが，一層拡大した領域は#1 の領域内にあるため，BV の対象となる．

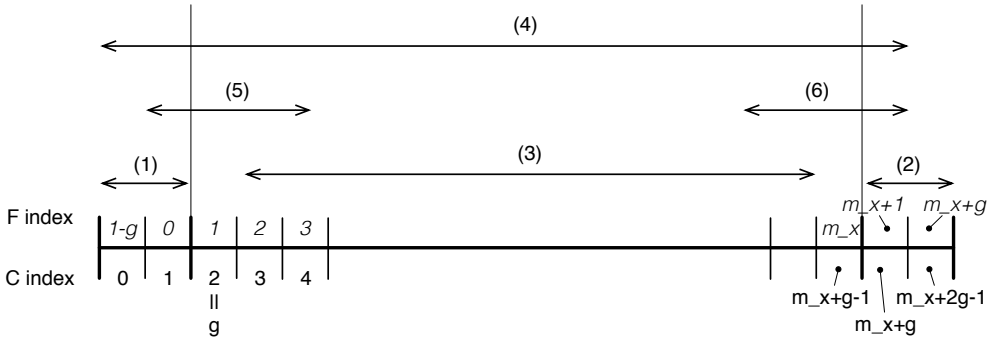



図 11.4　一次元方向の ID の分布パターン．図ではガイドセル (g=2) と仮定している．処理は (1) ~ (6) の順番に行う．C index から F index への変換は，$F_{idx} = C_{idx} + 1 - g$

.

一次元方向の ID のパターンを**図 11.4** に示す．最初に，対象 ID のみが存在する BV 情報は，V-Sphere のメソッド GetBndIndexExtGc() で拡張オプションの引数 0 を渡して, ローカルなインデクス情報を得る．GetBndIndexExtGc() は，各方向の開始インデクス st_i，st_j，st_k とコンポーネントの長さ len_i，len_j，len_k を返す．このインデクス情報を初期値として，下記の各パターンに対して，一層拡大したインデクスを求め，所望のインデクス範囲 st，ed を得る．以下，i 方向について示す．インデクス st，ed は Fortran インデクスで保持する．終了インデクスは C インデクスで次のように表せる．

```
ed_i = st_i + len_i - 1;
```

#### 11.1.3.1 基本処理

各ノード間のガイドセルを考慮した基本処理について示す．以下の処理は，基本的に逐次の場合．並列時は，計算領域内か領域外かの判定が必要．便宜的に次の変数を導入しておく．m_x は三次元配列の各要素方向の最大値（計算内部領域）である．

```
n_st = st_i - 1; // 一層外側へ拡大
n_ed = ed_i + 1; // 一層外側へ拡大
max_c1 = m_x + g;
```

また，以下のコンテキストで nID[] には隣接 6 方向の各ノードのランク番号が入っており，隣接ノードがなければ負値が入っている．

(1) BV が X-方向のガイドセル内のみにある場合
x+ 側に一層拡大し，その値が自領域内であれば，1 層分だけ存在する．

```
if ( ed_i < guide ) {
  if( nID[MINUS_DIR] < 0 ){ // 計算領域の外部面に接する場合は，対象外
    m_st = 0;
    m_ed = 0;
  }
  else { // 計算領域内部にある場合（並列時）
    if ( n_ed == guide ) { // ガイドセル 1 層外側の場合
      m_st = 1; // F index
      m_ed = 1; // F index
    }
    else {
      m_st = 0;
      m_ed = 0;
    }
  }
}
```

(2) BV が X+ 方向のガイドセル内のみにある場合
x-側に一層拡大し，その値が自領域内であれば，1 層分だけ存在する．

```
if ( st_i >= max_c1 ) {
  if( nID[PLUS_DIR] < 0 ){ // 計算領域の外部面に接する場合は，対象外
    m_st = 0;
    m_ed = 0;
  }
  else {
    if ( n_st == (max_c1 - 1) ) { // ガイドセル 1 層外側の場合
      m_st = m_x; // F index
      m_ed = m_x; // F index
    }
    else {
      m_st = 0;
      m_ed = 0;
    }
  }
}
```

(3) BV が内部領域のみにある場合
開始点の処理は，x-側に一層拡大しその値が自領域内の端の場合，開始インデクスは領域内の最小インデクス．それ以外はマイナス 1．終点の処理は，x+ 側に一層拡大し，その値が自領域内の端の場合，終了インデクスは領域内の最大インデクス．それ以外はプラス 1．

```
if ( (st_i >= guide) && (ed_i < max_c1) ) {
  if ( st_i == guide ) { // 最外層セル
    m_st = 1; // F index
  }
  else {
    m_st = n_st + 1 - guide; // F index
  }

  if ( ed_i == (max_c1 - 1) ) { // 最外層セル
    m_ed = m_x; // F index
  }
  else { // 内部
    m_ed = n_ed + 1 - guide; // F index
  }
}
```

(4)　BV が両方向のガイドセルにまたがる場合

```
if ( (st_i < guide) && (ed_i >= max_c1) ) {
  m_st = 1; // F index
  m_ed = m_x; // F index
}
```

(5)　BV が X-方向のガイドセルから内部領域にある場合
開始点は領域内の最小インデクス．終点は端点か，あるいは内部の可能性．

```
if ( (st_i < guide) && (ed_i < max_c1) ) {
  m_st = 1; // F index

  if ( ed_i == (max_c1 - 1) ) { // 最外層セル
    m_ed = m_x; // F index
  }
  else { // 内部
    m_ed = n_ed + 1 - guide; // F index
  }
}
```

(6)　BV が X+ 方向のガイドセルから内部領域にある場合
終点は領域内の最大インデクス．開始点は端点か，あるいは内部の可能性．

```
if ( (st_i < max_c1) && (ed_i >= max_c1) ) {
  m_ed = m_x; // F index

  if ( st_i == guide ) { // 端点
    m_st = 1; // F index
  }
  else { // 内部
    m_st = n_st + 1 - guide; // F index
  }
}
```

上記のメソッドを各方向から利用できるように `getEnlargedIndex()` を実装する．

### 11.1.4　ビットフラグのエンコード処理

BC Index のビットフィールドへのエンコード処理を具体的に示す．

1. bcd へのエンコード（1）　`BinaryVoxel::setBCIndex_base1()`
   BC Index の配列 bcd[] は，ソルバ中で共通的に利用するフラグを集めている．ガイドセルを含む全領域に対して処理を行う．まず，state を流体で初期化．次に，ボクセルモデルのセル ID を 8bit でエンコードする[*2]．コンポーネントリストに登録された境界条件と媒質に対して，媒質情報にアクセスするための MaterialList[] へのエントリをエンコードする．最後に，各セルの媒質エントリをたどり，MaterialList[] の内容から流体か固体かを判断し，state をエンコードする．

2. bcd[] へのエンコード（2）　`BinaryVoxel::setBCIndex_base2()`
   先に設定した state をみて，`find_isolated_Fcell()` で孤立した流体セルを固体セルへ変更する．これは，計算の不安定性を取り除くための処理で，大局的にみて影響は小さい．この処理では，置換する固体セルは隣接固体 ID の最頻値とする．
   次に，`encActive()` で不活性セルの設定を行う．不活性セルは，Kind_of_Solver のパラメータにより異なる処理となる．
   - FLOW_ONLY, THERMAL_FLOW, THERMAL_FLOW_NATURAL
     流体セルのみが計算に有効で，固体セルは不活性とする．
   - SOLID_CONDUCTION
     固体セルのみが計算に有効で，流体セルは不活性とする．
   - CONJUGATE_HEAT_TRANSFER
     流体・固体セルともに有効.

   さらに，コンポーネントで指定された不活性セルを不活性化する（Active bit を INACTIVE にする）．不活性セルの指定は，流体・固体セル両方可能.

3. bcp[] へのエンコード　`BinaryVoxel::setBCIndexP()`
   bcd[] から active, state の両フラグをコピーし，残りのビットフラグを (1) で初期化する．

   - 内部領域の処理　`encPbit_Wall_Inside()`
     内部領域にある流体セルのうち，固体セルに隣接する面のノイマンフラグ `BC_N_?`をゼロにする．外部境界面に接するセル面は後で処理．つぎに，全内部領域について，固体セルに隣接する流体セルに wall location フラグを付与する．この処理は，`cbc_pvec_()` などの関数で壁法則の計算に利用する．

   - 外部境界面の処理　`BinaryVoxel::encPbit_OBC()`
     外部境界面が Dirichlet か Neumann かを指定し，対応するフラグ `BC_D_?`または `BC_N_?`をエンコードする．外部境界面が壁面の場合にのみ，方向フラグを付与する．

   - コンポーネントのエンコード
     コンポーネントの境界条件の種類毎に処理の内容は異なる．
     - SPEC_VEL, SPEC_VEL_WH, OUTFLOW
       テストセルが流体セルで Def_Face ID が指定されている場合，かつ，テストセルの隣接セルが固体でキー ID の場合，テストセルの対象面に Neumann フラグをエンコードし，bcd[] に ComponentList[] へのエントリを登録する．
     - PERIODIC, CELL_MONITOR, IBM_DF, HEX, FAN, DARCY
       bcd[] に ComponentList[] へのエントリを登録する処理のみ行う．

   - 対角・非対角要素の係数のエンコード　`encPbit()`
     Neumann フラグと Dirichlet フラグが同時に設定されていないことを確認．反復行列の非対角要素/対角要素をエンコードする．対角要素がゼロの場合はエラー．

[*2] ID=0 は使わないので，ID=1~255 の範囲.

4. bcv[] へのエンコード `BinaryVoxel::setBCIndexV()`
   速度境界条件の種類をセルの各面にエンコードする．各面に 5bit を割り当てる．(0) は境界条件なし，(31) は外部境界条件，それ以外の (1~30) はコンポーネントのエントリを示す．まず，bcd[] から active, state の両フラグをコピーする．
   - 外部境界面の境界条件のエンコード `BinaryVoxel::encVbit_OBC()`
     この処理は, 外部境界面に接するセルが流体の場合を対象とする．外部境界条件の実装には, 流束形式と境界値形式の 2 種類がある．流束形式は OBC_SYMMETRIC, OBC_WALL, OBC_SPEC_VEL, OBC_OUTFLOW で，境界値形式は OBC_TRC_FREE, OBC_PERIODIC, OBC_IN_OUT である．流束形式の場合には，セルフェイスに外部境界条件フラグ (31) をエンコードする[*3]．OBC_SYMMETRIC, OBC_WALL のときガイドセルが固体であることをチェックし，OBC_SPEC_VEL, OBC_OUTFLOW のとき対象セルが流体の場合のみガイドセルが流体であることをチェックする．
     境界値形式の OBC_TRC_FREE, OBC_IN_OUT の場合, 対象面が流体であることのみチェックし, セルフェイスに外部境界条件フラグはエンコードしない．つまり, 通常スキームによる計算となる．OBC_PERIODIC の場合には，外部境界条件フラグのエンコードも流体面のチェックも行わない．
   - 内部境界条件のコンポーネントへのエンコード `BinaryVoxel::encVbit_IBC()`
     コンポーネントが SPEC_VEL_WH, SPEC_VEL, OUTFLOW の場合，キー ID は固体セル，Def_Face ID に流体セルが指定される．したがって，テストセルが流体で Def_Face ID が指定され，かつ，テストセルの隣接セルが固体でキー ID の場合にのみ，エンコード処理を行う．また，流体セルの流入出面にはコンポーネントのエントリをエンコードし，同時に bcp[] の壁面方向を表す wall location フラグを off にする．これは，境界条件付与部分の固体面は，壁法則の計算を行わないようにするためである．
     内部境界条件の流入出は，ボクセルモデルでは固体セルにより流入出面が規定されている．流入出境界条件処理で高次風上スキーム時の運動量流束の正確な計算のため，指定されたキー ID のセルを流体の状態にしておく．
   - セルの固体面数のエンコード `encPbit_No_SolidFace()`
     bcp[] にセルの 6 面のうち，固体面の数をエンコードする[*4]．もし，6 面とも固体面の場合にはエラーで停止する．

5. bch1, bch2 へのエンコード `BinaryVoxel::setBCIndexH()`
   温度境界条件の情報をエンコードする．bcd[] から active, state の両フラグをコピーする．
   - 断熱フラグの初期化
     bch2[] の断熱フラグを非断熱 (1) に初期化する．
   - Kind_of_Solver のパラメータに応じて，Solid-Fluid 面の断熱処理
     THERMAL_FLOW, THERMAL_FLOW_NATURAL の場合，Fluid セルのうち Solid セルに接する面について，`setAmask_Thermal()` で bch2[] の断熱フラグを断熱 (0) にする．SOLID_CONDUCTION の場合，Solid セルのうち Fluid セルに接する面について，`setAmask_Solid()` で bch2[] の断熱フラグを断熱 (0) にする．
   - 対称境界面に断熱マスクをセット
     対称境界が指定された外部境界面に対して，`encAmask_SymtrcBC()` で bch2[] の断熱フラグを断熱 (0) にする．
   - 不活性セルの場合の断熱マスク処理

[*3] 運動量 flux の計算で，セルフェイスが境界条件 (!=0) の場合には，その流束の寄与をゼロにしておいて，後ほど境界条件による流束を加算する．
[*4] 粘性項の表面積補正のための実験的実装.

コンポーネントで指定された不活性セル ID に対して，setAmask_InActive() でセルを Inactive にする．流体セル，固体セルに関わらず，隣接セルが Inactive 指定の場合には断熱にするが，自セルが指定 ID 以外のときに限る．つまり，Inactive に指定したセルと接するセルの隣接面を断熱にする．ただし，隣接セルは非 Inactive 指定であること．これは，Inactive 指定セルが塊の時には，その内部はラプラスを解き，平均場にするため．

- コンポーネントのエンコード
  各コンポーネント毎の処理を行う．

  - 断熱条件 encQface()
    テストセルが Def_Face で，隣接セルがキー ID のセルで挟まれる面に対して，bch1[] に CompoList のエントリをエンコードし，bch2[] に断熱条件 (0) をエンコード．

  - 熱流束指定 encQface()
    テストセルが Def_Face で，隣接セルがキー ID のセルで挟まれる面に対して，bch1[] に CompoList のエントリをエンコードし，bch2[] に非断熱条件 (1) をエンコード．

  - 流入・流出指定 encQfaceSVO()
    コンポーネントが SPEC_VEL_WH, OUTFLOW の場合，キー ID は固体セル，Def_Face ID に流体セルが指定される．テストセルが Def_Face ID で指定され，かつ，隣接セルがキー ID の場合にのみ，bch1[] に CompoList のエントリをエンコードし，bch2[] に非断熱条件 (1) をエンコードする．BinaryVoxel::encVbit_IBC() の場合とは異なり，セルの状態をチェックしない．流入出境界条件処理で高次風上スキーム時の対流熱流束の正確な計算のため，指定されたキー ID のセルを流体の状態にしておく．

  - 熱伝達指定 Type_S encQfaceHT_S()
    Kind_of_Solver が THERMAL_FLOW または THERMAL_FLOW_NATURAL の場合に，コンポーネント HeatTransfer_S, HeatTransfer_SN，HeatTransfer_SF を指定できる．キー ID は固体セル，Def_Face ID に流体セルが指定される．テストセルが Def_Face ID かつ流体で指定され，かつ，隣接セルがキー ID かつ固体の場合にのみ，bch1[] に CompoList のエントリをエンコードし，bch2[] に非断熱条件 (1) をエンコードする．

  - 熱伝達指定 Type_B encQfaceHT_B()
    Kind_of_Solver が CONJUGATE_HEAT_TRANSFER または SOLID_CONDUCTION の場合に，コンポーネント HeatTransfer_B を指定できる．キー ID は固体セル，Def_Face ID に流体セルが指定される．テストセルがキー ID かつ固体で指定され，かつ，隣接セルが Def_Face かつ流体の場合にのみ，bch1[] に CompoList のエントリをエンコードし，bch2[] に非断熱条件 (1) をエンコードする．

  - 等温指定 encQfaceISO_SF/SS()
    等温境界条件は，流体と固体，あるいは固体と固体で挟まれる面に対して適用可能である．流体と固体セルの間で指定される等温境界は，Kind_of_Solver が THERMAL_FLOW または THERMAL_FLOW_NATURAL の場合に用いられる．このとき，キー ID に固体セル，Def_Face ID には流体セルが指定される．テストセルが Def_Face ID かつ流体で指定され，かつ，隣接セルがキー ID かつ固体の場合に，bch1[] に CompoList のエントリをエンコードし，bch2[] に非断熱条件 (1) をエンコードする．

    一方，固体セル間の面で指定される等温境界は，Kind_of_Solver が CONJUGATE_HEAT_TRANSFER または SOLID_CONDUCTION の場合に用いられる．このとき，キー ID セルと Def_Face ID セルは両方とも固体セルである．テストセルがキー ID かつ固体で，Def_Face ID（固体セル）と挟まれる面に対して，bch1[] に CompoList のエントリをエンコードし，bch2[] に非断熱条件 (1) をエンコードする．

- 熱源指定
  熱源のタイプとして，Heat_Source と Constant_Temperature の 2 種類の指定方法がある．これらの境界条件は，体積要素に対する指定である．テストセルが指定される ID の場合，bch2[] に CompoList のエントリをエンコードする．

- 熱境界フラグの設定 encHbit()
  GMA_?のフラグをセットする．まず，(1) で初期化．bch1[] の各セルフェイスフラグ BC_FACE_?をみて，境界条件があれば bch2[] の GMA_?を (0) にする．その後，対角要素の係数を断熱マスクを考慮して計算し，H_DIAG に対角要素の係数 (0~ 6) をストアする．もし，対角要素が (0) であればゼロ割防止のために (1) を入れておく．対角要素がゼロのセルは境界条件や断熱セルで囲まれたセルに相当する．

BC Index bx[] は図 11.5 を参照して下記のコードのように利用する．この例では速度の境界条件について説明している．まず，setInnerVBCface() で境界条件のループ n を回し，それがフェイス面に対する速度境界条件であれば (isVBCF())，処理を行う．次に，cmp[n].getType() の値により，n 番目に登録された境界条件の種類を特定する．その後，個別の処理を行う．setDirichletVInside() では，指定された面に速度成分を与える処理を行っている．空間ループを回し，bx[i,j,k] の値がエンコードされた境界条件の格納番号，および方向と一致する場合に目的の処理を行う．このとき，必要に応じて MaterialList に登録された物性値を利用する．

```
/**
 @fn void SetBC3D::setInnerVBCface(SKL_REAL* v, unsigned* bx, SKL_REAL v00, ...)
 @brief 速度の内部境界をセットする
 @param v 速度ベクトル
 @param bx BCindex
 @param v00 基準速度情報
 @param tm 時刻
 @param arrangement 格子配置
 */
void SetBC3D::setInnerVBCface(SKL_REAL* v, unsigned* bx, SKL_REAL v00,
                                      SKL_REAL tm, const char* arrangement)
{
  for (unsigned n=1; n<=NoBC; n++) {
    if ( cmp[n].isVBCF() ) {

      switch ( cmp[n].getType() ) {
        case DIRICHLET_INSIDE:
case DIRICHLET_CTMP:
          if ( !strcasecmp("staggered", arrangement)) {
            if ( cmp[n].usw != DRC_V_SHO ) {
                  cmp[n].val[0] = setDirichletV_Inside_S(v, bx, n, v00);
            }
            else {
                  cmp[n].val[0] = setSHO_Inside_S(v, bx, n, tm, v00);
            }
          }
          else if ( !strcasecmp("collocated", arrangement)) {
            ;
          }
          else {
            stmpd_printf("Invalid keyword\n");
          }
          break;
```

```
        default:
          break;
      }

    }
  }
}

/**
 @fn SKL_REAL SetBC3D::setDirichletVInside(SKL_REAL* v, unsigned* bx, unsigned n, SKL_REAL v00)
 @brief 内部領域の速度のディリクレ境界をセットする
 @param v 速度ベクトル
 @param bx BCindex
 @param n 境界条件コンポーネントのエントリ番号
 @param v00 参照速度
 @retval 無次元平均流速
 @note
    - 内部境界はセル面直を仮定
 */
SKL_REAL SetBC3D::setDirichletV_Inside_S(SKL_REAL* v, unsigned* bx, unsigned n, SKL_REAL v00)
{
  unsigned i, j, k, s;
  SKL_REAL va=0.0;
  SKL_REAL vel, vec[3];

  vel   = cmp[n].N1.Velocity * v00;

  vec[0]= cmp[n].nv[0]*vel;
  vec[1]= cmp[n].nv[1]*vel;
  vec[2]= cmp[n].nv[2]*vel;

  for (k=cmp[n].ci.st[2]; k<=cmp[n].ci.ed[2]; k++) {
    for (j=cmp[n].ci.st[1]; j<=cmp[n].ci.ed[1]; j++) {
      for (i=cmp[n].ci.st[0]; i<=cmp[n].ci.ed[0]; i++) {
        s = bx[ SklUtil::getFindexS3D(size, guide, i, j, k) ];
        if ( FBUtility::chkBCOrder(s, n) ) {
          if ( FBUtility::chkShift(s, VFACE_I) ) {
            va += v[SklUtil::getFindexV3D(size, guide, i, j, k, 0)] = vec[0];
          }
          if ( FBUtility::chkShift(s, VFACE_J) ) {
            va += v[SklUtil::getFindexV3D(size, guide, i, j, k, 1)] = vec[1];
          }
          if ( FBUtility::chkShift(s, VFACE_K) ) {
            va += v[SklUtil::getFindexV3D(size, guide, i, j, k, 2)] = vec[2];
          }
        }
      }
    }
  }
  va = va * dh*dh / (SKL_REAL)cmp[n].area;
  return (va);
}
```

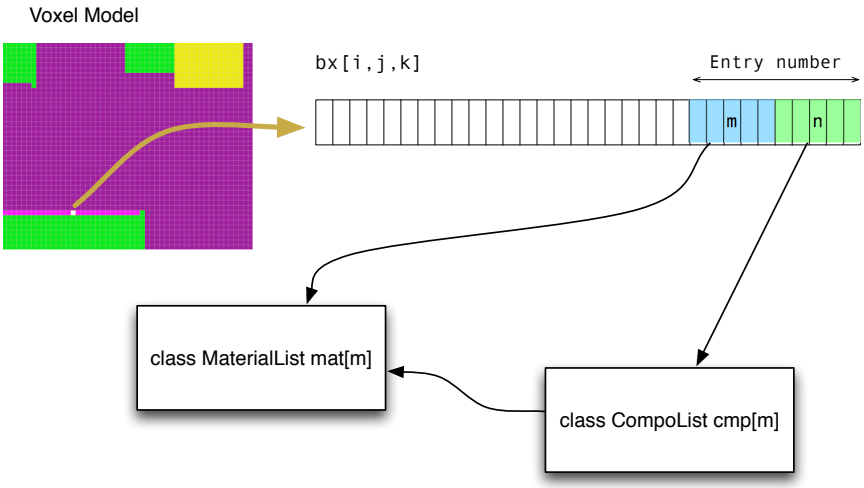



図 11.5　BC Index からのビットフラグのデコード

### 11.1.5　セルフェイスに対する速度境界条件

スタガード変数配置における各速度成分に対して値を指定する．例として，図 11.6 に示すようなバイナリボクセルモデルにおいて，矢印方向の速度成分を与えることを考える．ID として，空間に 1, 3 が指定されている．このとき，矢印の方向ベクトルは (0,1,0) で ID=3 と ID=1 が接する面であるので，以下のような XML で大きさ 0.58 の一定な速度成分を指定する．

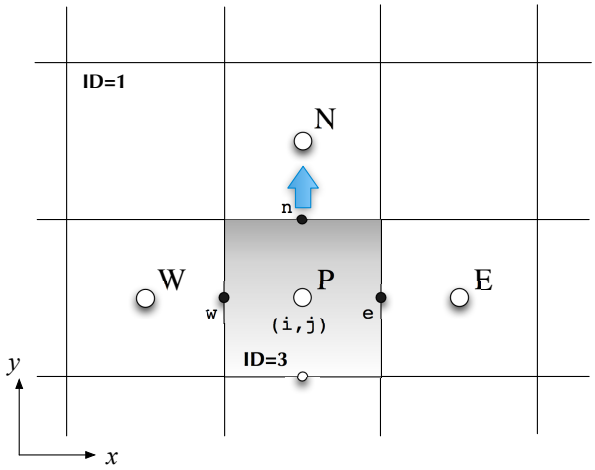



図 11.6　セルフェイスにおける速度境界条件

```
<InnerBoundary>
  <Elem name="Vec_Face" id="1">
    <Elem name="Dirichlet" id="3" comment="source">
      <Param name="Norm_x"      dtype="REAL"   value="0.0" />
      <Param name="Norm_y"      dtype="REAL"   value="1.0" />
      <Param name="Norm_z"      dtype="REAL"   value="0.0" />
      <Param name="velocity"    dtype="REAL"   value="0.58" />
      <Param name="def_face"    dtype="INT"    value="1" />
      <param name="type"        dtype="string" value="constant" />
```

```
    </Elem>
  </Elem>
</InnerBoundary>
```

### 11.1.6　セルフェイスに対する熱境界条件

熱境界条件をコンポーネントとして与える場合，基本的に，固体面に与える．**図 11.7** に示すように，固体面から流体側へ法線を考えて，法線方向が指定値の正の値とする．つまり，流体側への熱移動が正の方向となるように考える．

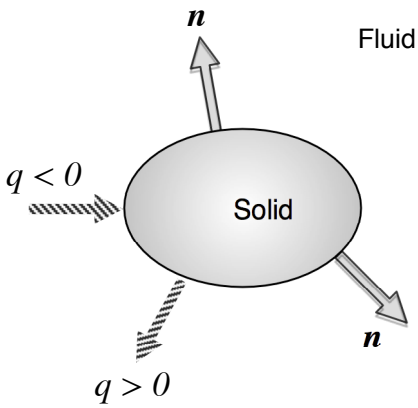



図 11.7　流体-固体界面における熱境界条件の指定方法

この考え方に基づくと，Solver_Property の Kind_of_Solver の指定モードによって，熱境界条件の働きが異なる．Thermal_Flow と Thermal_Flow_Natural の場合は，熱流動計算で流体の温度のみが意味を持ち，固体部分は計算対象とはならず不活性セルとして扱う．したがって，流体-固体の境界面で与える熱境界条件（熱流束）は，流体セル側のみに指定する．

一方，Solid_Conduction の場合は固体部分の熱伝導のみを解くので，流体部分は不活性セルとして扱う．したがって，流体-固体の境界面で与える熱境界条件は，固体セル側のみに指定する．

また，Conjugate_Heat_Transfer の場合には流体と固体の熱移動を計算する．したがって，流体-固体の境界面で与える熱境界条件は，流体セルと固体セルの両方で指定する．

以下の各境界指定で述べるように，ID と Def_Face タグの組み合わせにより，流体-固体の境界面を指定する．このとき，ID には固体の ID を，Def_Face には流体または固体の ID を指定することを基本とする[*5]．

境界条件の処理手順は次のとおり．

1. `SetBC3D::InnerTBCface()`
   拡散項に関するセルフェイスの熱境界条件処理．コンポーネントのループを回し，対象境界条件を特定し，各処理に分岐．熱流束指定の場合には `SetBC3D::ps_IBC_Heatflux()` で境界条件を計算．戻り値をコンポーネントのモニタ保持変数に代入．

2. `SetBC3D::ps_IBC_Heatflux()`
   与える熱流束を有次元から無次元に変換．各コンポーネントリストの BV ループ範囲のセルに対して，セルを構成する各面に境界条件が指定されているかどうかをテストする．対象となる面に対しては，熱流束を保持する配列 `qbc` に値をストアする．スタガードのインデクスに注意．セルの W 面における正値

[*5] ID に固体を指定する理由は，固体面は大抵の場合局所的なので，参照範囲を小さく抑えて効率的な計算をするため．

($q \geq 0$) は流入で，E 面における正値 ($q \geq 0$) は流出となる．この関数内で allreduce() をかけ，無次元熱流束を集約，返値にする．

上記の処理を行うための前処理として，次の処理を行っておく．

1. `BinaryVoxel::setBCIndexH()`
   熱境界条件のビットインデクスをエンコードする．
   - まず，セルの各面の断熱マスク（6 面分）を非断熱 (1) に初期化する．
   - モードに応じて，流体-固体界面を断熱 (0) にする．
     - THERMAL_FLOW と THERMAL_FLOW_NATURAL の場合, `BinaryVoxel::setAmask_Thermal()` で，S-F 面の F 側を断熱にする．
     - SOLID_CONDUCTION の場合は，`BinaryVoxel::setAmask_Solid()` で，S-F 面の S 側を断熱にする．
     - CONJUGATE_HEAT_TRANSFER の場合には，固体セルも流体セルも両方とも計算するので，断熱処理はしない．
   - 対称境界面に断熱マスクをセット
   - 不活性セルの場合の断熱マスク処理
   - 媒質コンポーネントに対して，element をセット
   - 具体的なエンコード処理
   - 係数行列の値をビットでエンコード
     `BinaryVoxel::encHbit()` で，`bh2[]` の $\gamma$ を 1 で初期化．`bh1[]` を調べて，セルの各面に境界条件がセットされていなければ (!=0)，`bh2[]` の各面で $\gamma = 1$ とする．対角要素の係数は $\sum_j = 0^5 (\gamma_{BC}\gamma_A)_j$ として，断熱マスクを考慮.
2.

ボクセルモデルのセルフェイスに対する熱境界条件 QBC_Face の具体的な種類は，XML ファイルに記述された内容を基にして CompoList クラスのオブジェクトのメンバー変数，compo[n].type にエンコードする．

図**??**の BC Index の 17 ~ 22 ビットにおいて，セルの 6 面に対して BC の種類を判別するフラグを持たせるのは，隣接セルとの重複があるため冗長であるが，BC Index の値をロードした後ビット演算だけで済む．よって，メモリアクセスが少ないので効率が良い．セル P の全てのプラス方向の面に対してエンコード処理を行い，その後プラス方向の面の情報を隣のセルのマイナス方向にコピーする．

BC フラグの値は，下記の疑似コードのように利用する．

```
/**
 @fn void FBUtility::getBCgamma (unsigned idx, SKL_REAL* a)
 @brief γの値から係数を配列で返す
 @retval γ
 @param idx BCindex
 @param a 係数配列
 */
```

```
void FBUtility::getBCgamma(unsigned idx, SKL_REAL* a)
{
  a[0] =(SKL_REAL)( (idx >> GMA_W) & 0x1 );
  a[1] =(SKL_REAL)( (idx >> GMA_E) & 0x1 );
  a[2] =(SKL_REAL)( (idx >> GMA_S) & 0x1 );
  a[3] =(SKL_REAL)( (idx >> GMA_N) & 0x1 );
  a[4] =(SKL_REAL)( (idx >> GMA_B) & 0x1 );
  a[5] =(SKL_REAL)( (idx >> GMA_T) & 0x1 );
}
```

QBC_Face の具体的な境界条件は，前述の XML ファイルに記述された内容を基に判断する．次節以降で具体的な境界条件について述べる．

### 11.1.7　外力を用いた境界条件の実装

3.4.1 で説明した外力項による境界条件について，その実装を説明する．

### 11.1.8　Immersed Boundary を用いた境界条件の実装

3.5 で述べた Immersed Boundary Method の実装について説明する．計算する格子系としては，Staggered と Collocated の 2 種類がある．そのため，前処理では格子配置に応じて内部処理が異なる．図 11.8 に示すように，変数配置により指定したセル領域内にかかる速度定義点が異なる．このため，Staggered ではセル境界に与える境界条件 VBCF で記述し，Collocated ではセルセンターに与える境界条件 VBCC で記述する．Staggered の場合には，セルに与えられた Direct_Forcing の ID を見つけると，そのセルの両端にあるセル界面位置の速度定義点を Forcing の対象とする．これに対して，Collocated の場合には，セルセンターの速度定義点のみが Forcing の対象となる．

解法で述べたように，Direct Forcing は速度の強制で実装するので，内部的にはディリクレ型速度境界条件を使って実装する．また，Direct Forcing は流体セルに対して有効であるので，Active_Bit は ON にしておく．

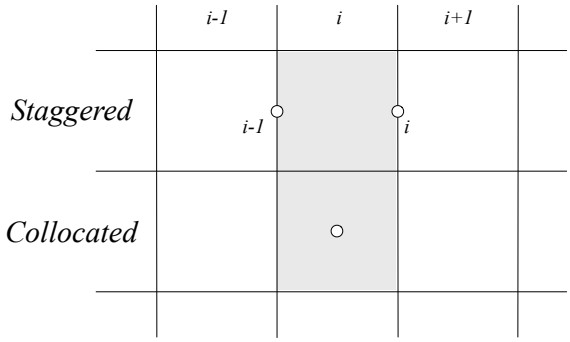



図 11.8　Direct Forcning のセル指定と強制対象となる速度位置の関係

### 11.1.9　V-Sphere コンポーネントを用いた実装

全計算領域の一部に存在する内部境界条件要素（コンポーネント）の実装に V-Sphere により提供される Component 機能を用いる．この機能には，並列処理時のサブドメイン間のデータ通信や管理が含まれ，均等分割とマルチボックス分割の両方に対応する．

1. 全計算領域のボクセル情報の初期化
   `SklSolverC3D::VoxelInitialize()`

2. Bounding Box のグローバル情報の取得
   `SklSolverC3D::getGlobalCmpIdx_SubVoxInit()`
3. コンポーネント領域の初期化と登録
   `SklSolverC3D::getGlobalCmpIdx_SubVoxInit()`
4. コンポーネントへのデータクラスの追加
   `SklSolverC3D::allocArray_forcing (long &total, int para_key)`
   dc_bsi（スカラ）, dc_cfr（ベクトル）のデータクラスをコンポーネントでインスタンスする．
5. コンポーネントの存在するセルに体積率を与える
   `BinaryVoxel::setCmpFractionBV()` dc_cfr に体積率を保持する．

## 11.2　境界条件

境界条件の本体は主に Fortran90 で記述している．Fortran コードは，C++ の名前空間内では C ソースコードと同様に名前なし空間として扱われるので，FortranFuncC3D.h 内にプロトタイプを記述しておく．

### 11.2.1　外部境界条件の実装

外部境界条件は，計算領域の外周に設定する境界条件である．各節で，速度，圧力，温度に対する境界条件を説明する．Fortran コードは，SetBC3D クラスに実装されたメソッド setOuterPBC, SetOuterTBC, setOuterVBC 内でハンドリングされている．Fortran コードには，たとえば流出境界条件ならば，計算領域の $X\pm$ 面，$Y\pm$ 面，$Z\pm$ 面の各面に対する処理が書かれている．このサブルーチンをコールするときに，方向と必要なパラメータをセットすることで必要な面の境界条件を設定する．

```
#define X_MINUS 0
#define X_PLUS  1
#define Y_MINUS 2
#define Y_PLUS  3
#define Z_MINUS 4
#define Z_PLUS  5
```

## 11.3　温度輸送方程式の実装

### 11.3.1　Euler 陽解法

式 (6.25) を $\Delta$ 形式にして解く．

$$\theta^{n+1} \;=\; \theta^{n} \;+\; \Delta\theta^{n+1} \tag{11.1}$$

$$\Delta\theta^{n+1} \;=\; \Delta t\, f\,(u_i,\, \theta) \tag{11.2}$$

式 (11.2) の $f\,(u_i,\, \theta)$ は式 (6.25) の右辺である．`SklSolverCBC::PS_E_CBS()` メソッド内では，次のように実装している．

1. 対流熱流束の計算
   `cbc_tfx_cnv_uwd_()` または `cbc_tfx_cnv_muscl_()` で対流熱流束を求める．流入境界条件を指定している場合には，計算した対流熱流束に上書きする形で与えている．

2. 対流項の増分
   ConvectionEE() で対流項の増分を計算する.
3. 境界条件

# 第 12 章

# Appendix

## 12.1　回転行列

### 12.1.1　回転行列の性質

回転行列の一般的な性質について述べる [20,21]．ある座標 $\boldsymbol{p}$ が原点を通る回転軸によって回転して $\boldsymbol{q}$ に移動する状態変化は，回転行列 $\boldsymbol{T}$ を用いて，

$$\boldsymbol{q} = \boldsymbol{T}\boldsymbol{p} \tag{12.1}$$

ただし，

$$\boldsymbol{T} = \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix} = (\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}) \tag{12.2}$$

また，行列 $\boldsymbol{T}$ は，

$$\boldsymbol{T} = \boldsymbol{a}\boldsymbol{i} + \boldsymbol{b}\boldsymbol{j} + \boldsymbol{c}\boldsymbol{k} \tag{12.3}$$

つまり，ある座標軸を表すベクトル $(\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k})$ を，それぞれベクトル $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c})$ に変換することを意味している[*1]．

### 12.1.2　回転の合成

三次元空間における任意の回転は，座標軸周りに 3 回回転させる．各軸周りの回転は座標軸の正方向に向かって右回りを回転の正方向とする．回転の順序は任意であるが，通常 $x$ 軸，$y$ 軸，$z$ 軸の順序でこれをオイラー回転という．各軸周りの回転行列をそれぞれ $\boldsymbol{T}_x, \boldsymbol{T}_y, \boldsymbol{T}_z$ とすると，

$$\boldsymbol{T}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix} \tag{12.4}$$

$$\boldsymbol{T}_y = \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{bmatrix} \tag{12.5}$$

$$\boldsymbol{T}_z = \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{12.6}$$

合成された回転行列は，

$$\boldsymbol{T}_{xyz} = \boldsymbol{T}_z\boldsymbol{T}_y\boldsymbol{T}_x = \begin{bmatrix} C_yC_z & S_xS_yC_z - C_xS_z & C_xS_yC_z + S_xS_z \\ C_yS_z & S_xS_yS_z + C_xC_z & C_xS_yS_z - S_xC_z \\ -S_y & S_xC_y & C_xC_y \end{bmatrix} \tag{12.7}$$

ここで，例えば，$C_x$ は $\cos\theta_x$ を表す．

行列積は一般に $\boldsymbol{AB} \neq \boldsymbol{BA}$ なので，行列を作用させる順番で答えが異なる．したがって，回転の順番を変えると，同じ角度であっても異なる位置に変換されることになるので注意．

[*1] ここで，扱うベクトルは右手系とする．

また，逆変換の回転行列は，

$$\boldsymbol{T}_{zyx} = \boldsymbol{T}_x \boldsymbol{T}_y \boldsymbol{T}_z = \begin{bmatrix} C_y C_z & C_y S_z & -S_y \\ S_x S_y C_z - C_x S_z & S_x S_y S_z + C_x C_z & S_x C_y \\ C_x S_y C_z + S_x S_z & C_x S_y S_z - S_x C_z & C_x C_y \end{bmatrix} \tag{12.8}$$

### 12.1.3　座標軸まわりの回転角度

**式 (12.2)** に示した行列要素から座標軸ごとの回転角度を求める．$\theta_x$, $\theta_y$, $\theta_z$ をそれぞれ $\alpha$, $\beta$, $\gamma$ とする．

1. コンポーネントの流出方向の指定ベクトル $\vec{n}$ が z 軸の単位ベクトル $\vec{e}_3$ と作る角度 $\alpha$ を求める．$\vec{n}$ を yz 面へ射影したベクトルと $\vec{e}_3$ と内積をとり，方向角を求める．射影ベクトルの大きさがゼロの場合には回転角はゼロとする．
2. 回転角 ($\alpha$, 0.0, 0.0) でベクトル $\vec{n}$ を回転し，xz 面へ射影したベクトルと z 軸の単位ベクトルと作る角度 $\beta$ を求める．ベクトル $\vec{n}$ が x 軸と y 軸の両方から見て $\vec{e}_3$ と反対の場合にだけ，y 軸回りのみ回転する．
3. 矩形形状の場合には，さらに方向ベクトルの向きも合わせる．回転角 ($\alpha$, $\beta$, 0.0) でベクトル $\vec{n}$ を回転すると，xy 平面へ射影されたベクトルとなる．この射影されたベクトルと x 軸の単位ベクトル $\vec{e}_1$ と作る角度 $\gamma$ を求める．

# 第 13 章

# アップデート情報

本ユーザガイドのアップデート情報について記す.

Version 0.7.0　　2012/3/3
- 体積率をもつコンポーネントの前処理を加筆.

Version 0.6.8　　2011/9/17
- EBCS スキームをバイナリ近似と面近似に整理.
- 対流項に minmod スキームを追加.
- 距離情報を用いた形状近似を追加.

Version 0.6.7　　2011/9/3
- ファンモデル，前処理の記述追加.
- Appendix の追加.

Version 0.6.6　　2011/7/31
- LES 解析，アルゴリズム追加.

Version 0.6.5　　2011/7/28
- 外力タイプの境界条件の整理.

Version 0.6.4　　2011/7/22
- コンポーネントの実装を変更.

Version 0.6.3　　2011/7/11
- FOCUS スパコンへのインストールを追記.
- インストールの内容を `cbc_ug.pdf` の追加情報の形にする.

Version 0.6.2　　2011/7/9
- CBC ユーザガイドと重複部分を削除.

Version 0.6.1　　2011/6/30
- コンポーネントの実装の章を設ける.

Version 0.6.0　　2011/2/10
- C3D User Guide をベースに編集開始.

# 参考文献


[1] http://accc.riken.go.jp/ricc.html.

[2] http://www.j-focus.or.jp/index.html.

[3] 小竹進, 土方邦夫, 松本洋一郎. 熱流体ハンドブック　現象と支配方程式. 丸善, 1994.

[4] http://www.gs.niigata-u.ac.jp/ kimlab/lecture/GasConstant.html.

[5] 中山顕, 桑原不二朗, 許国良. 熱流体力学 -基礎から数値シミュレーションまで-. 共立出版, 2002.

[6] 梶島岳夫. 乱流の数値シミュレーション (第一版). 養賢堂, 1999.

[7] 稲垣昌英. LES による工学問題の解明. 日本流体力学会 数値流体力学部門 Web 会誌, Vol. 11, No. 2, pp. 51–58, 2007.

[8] M. Germano, U. Piomelli, P. Moin, and W. H. Cabot. A dynamic subgrid-scale eddy viscosity model. *Physics of Fluids (A)*, Vol. 3, No. 7, pp. 1760–1765, 1991.

[9] S. Ghosal, T. Lund, P. Moin, and K. Aksevoll. A dynamic localization model for large-eddy simulation of turbulent flows. *Journal of Fluid Mechanics*, Vol. 286, pp. 229–255, 1995.

[10] 日本機会学会（編）. 計算力学ハンドブック [第 2 巻 差分法・有限体積法 (熱流体編)]. 丸善, 2006.

[11] 中村育雄. 流体解析ハンドブック. 共立出版, 1998.

[12] 藤井孝蔵. 流体力学の数値計算法. 東京大学出版会, 1994.

[13] 赤坂啓, 小野謙二. 複雑形状に対するボクセル法に基づく非圧縮流れ解析の境界条件の実装方法. *Transaction of JSCES*, No. 20060024, 2006.

[14] 赤坂啓, 小野謙二. 非水密なポリゴン要素で構成された任意形状周りの直交格子を用いた非圧縮粘性流れ解析. 日本機械学会論文集 B 編, Vol. 76, No. 764, pp. 536–545, 2010.

[15] Xu-Dong Liu, Ronald P. Fedkiw, and Myungjoo Kang. A boundary condition capturing method for poisson ’s equation on irregular domains. *Journal of Computational Physics*, Vol. 160, pp. 151–178, 2000.

[16] J. Mohd-Yusof. Combined immersed boundaries/b-splines methods for simulations of flows in complex geometries. *CTR Annual Research Briefs*, pp. 317–327, 1997.

[17] Jacques Mohcine Bahi, Sylvain Contassot-Vivier, and Raphaël Couturier. *Parallel Iterative Algorithms*. Chapman & Hall/CRC, 2007.

[18] 庄司正弘. 伝熱工学. 東京大学出版会, 1995.

[19] Suhas V. Patankar. *Numerical Heat Transfer and Fluid Flow*. Hemisphere Publishing Corp., 1980.

[20] 島田静雄. CAD・CG のための基礎数学. 共立出版, 2000.

[21] 大石進一, 牧野光則. グラフィクス. 日本評論社, 1994.