ADVENTURE_Fluid

Imcompressible thermal fluid analysis

Version: beta - 0.41

User's Manual

May 18, 2005

ADVENTURE Project

Contents

List of Figures

1 Overview

ADVENTURE_Fluid module is the incompressible thermal-fluid analysis module developed in the ADENTURE Project, which has following features;

- The incompressible thermal-fluid analysis codes using the finite element method
- Supporting hexahedral (Q1-P0) elements and tetrahedral (P1-P1) element
- Suitable for the large-scale analysis using the element-by-element algorithm
- Running on all the platforms supporting Message Passing Interface(MPI)
- Consist of the analysis codes, the special pre-processing tools and the data converters

2 Program Features

2.1 Solvers

2.1.1 adventure_fluid_tet

adventure_fluid_tet is the incompressible fluid analysis code for the tetrahedral P1-P1 elements, which is stabilized using SUPG (Streamline-upwind/ Petrov-Galerkin) method and PSPG (Pressure-stabilized/Petrov-Galerkin) method. The Crank-Nicolson method is adopted for the time discritization, and the velocity field and pressure field are solved simultaneously using asynmetric solvers. The supported solvers are Bi-CG STAB method, GPBi-CG method, Bi-CG STAB2 method and GMRES(m) method.

2.1.2 adventure_fluid_hex

adventure_fluid_hex is the incompressible fluid analysis code for the hexahedral Q1-P0 elements, which is stabilized using BTD (Balancing Tensor Diffusivity) method. The MAC (Marker-and-Cell) method is adopted for the time discritization, and the pressure Poisson's equations are solved using CG method.

2.1.3 adventure_thermal_fluid_hex

adventure_thermal_fluid_hex is the incompressible thermal-fluid analysis code for the hexahedral Q1-P0 elements. Almost the same code as *adventure_fluid_hex* except for solving the energy equation and supporting Boussinesq term in the Navier-Stokes equations.

2.2 Pre-Processing Tools

2.2.1 advfluid_pre_cavity

advfluid_pre_cavity is the special pre-processing tool for the lid-driven cubic cavity problem, which supports both hexahedral elemants and tetrahedral elements models. The template file of control data is also generated automatically.

2.2.2 advfluid_pre_cavity_nc

advfluid_pre_cavity_nc is almost the same as *advfluid_pre_cavity* except for the boundary conditions for the natural convection. The data generated by this pre-processing tool is for the *adventure_thermal_fluid_hex*. The template file of control data is also generated automatically.

2.2.3 advfluid_pre_pillar

advfluid_pre_pillar is the special pre-processing tool for the flow around the square pillar in the uniform flow. It is possible to control the length and the number of element in any parts of analysis domain. It is also possible to check the nodes distribution using the simple X11 viewer.

2.3 Data Converters

2.3.1 advfluid_mesh2ucd

advfluid_mesh2ucd converts the mesh data generated by pre-processing tools into the UCD format data of AVS. With this tool, it is possible to check the mesh configurations and the boundary conditions using the AVS Express or the Micro AVS.

2.3.2 advfluid_p_mesh2ucd

advfluid_p_mesh2ucd is almost the same as *advfluid_mesh2ucd* except for that it is for the domain-decomposed data. This tool can convert the decomposed multi data files together.

2.3.3 advfluid_rest2ucd

advfluid_rest2ucd converts the analysis result (restart) data into the UCD format data of AVS. With this tool, it is possible to check the velocity fields, the pressure fields and the temparature fields using the AVS Express or the Micro AVS.

2.3.4 advfluid_p_rest2ucd

advfluid_p_rest2ucd is almost the same as *advfluid_rest2ucd* except for that it is for the domain-decomposed data. This tool can convert the decomposed multi data files together.

2.3.5 advfluid_rest2ucd_nc

advfluid_rest2ucd_nc is almost the same as *advfluid_rest2ucd* except for that it is for the thermal-fluid analysis data. *advfluid_rest2ucd_nc* converts the analysis result (restart) data of *adventure_thermal_fluid_hex* into the UCD format data of AVS.

3 Compilation and Installation

3.1 Libraries

ADVENTURE_Fluid module requires the following libraries:

- ADVENTURE_IO module
- MPI library (mpich, LAM, etc.)

3.2 Compilation

ADVENTURE_Fluid module supports configure script. To buid the module, just type the following command on the almost all the systems.

```
% ./configure

% make
```

3.3 Installation

For installing all the program into $(HOME)/ADVERNTURE/bin, just type as follows;

```
% make install
```

On the other hand, for deleting all the programs from $(HOME)/ADVENTURE/bin and clear the executable modules and object modules from source tree, just type as follow;

```
% make clean
```

4 Execution of the Programs

4.1 Solvers

4.1.1 adventure_fluid_tet

The usage of adventure_fluid_tet is as follows;

```
% mpirun -np <np> adventure_fluid_tet <log> <ctrl> <mesh> <rest_out> [<rest_in>]
  <np>       : <np>  (number of processors)
  <log>      : <log>  (log file name)
  <ctrl>     : <ctrl>  (control file name)
  <mesh>     : <mesh>_<np>.adv  (mesh file name)
  <rest_out> : <rest_out>_<num_steps>_<np>.adv  (restart output file name)
  <rest_in>  : <rest_in>_<np>.adv  (restart input file name)
```

Refer to the hints listed below;

- Even in the case of running on single processor, it is necessary to use "mpirun -np 1".
- Assign an arbitrary file name for the "log file name".
- Assign a control file described in the next chapter for the "control file name".
- Assign a mesh file described in the next chapter for the "mesh file name". At that time, exclude the domain number "_<np>" and file extension ".adv" from the "mesh file name".
- Assign an arbitrary file name for the "restart output file name". The file name output practically is with the number of steps "_<num_steps>", the domain number "_<np>" and the file extension ".adv".
- In the case of restart computing, assign "restart input file name". At that time, exclude the domain number "_<np>" and file extension ".adv" from the "restart input file name".

4.1.2 adventure_fluid_hex

The usage of adventure_fluid_hex is as follows;

```
% mpirun -np <np> adventure_fluid_hex <mesh> <ctrl> <rest_out> [<rest_in>]
  <np>       : <np>  (number of processors)
  <mesh>     : <mesh>_<np>.adv  (mesh file name)
  <ctrl>     : <ctrl>  (control file name)
  <rest_out> : <restout>_<num_steps>_<np>.adv  (restart output file name)
  <rest_in>  : <rest_in>_<np>.adv  (restart input file name)
```

Refer to the hints listed below;

- Even in the case of running on single processor, it is necessary to use "mpirun -np 1".
- Assign a control file described in the next chapter for the "control file name".
- Assign a mesh file described in the next chapter for the "mesh file name". At that time, exclude the domain number "_<np>" and file extension ".adv" from the "mesh file name".
- Assign an arbitrary file name for the "restart output file name". The file name output practically is with the number of steps "_<num_steps>", the domain number "_<np>" and the file extension ".adv".
- In the case of restart computing, assign "restart input file name". At that time, exclude the domain number "_<np>" and file extension ".adv" from the "restart input file name".
- The adventure_fluid_hex is not able to output log file, therefore you shoud use redirect ">&" at the end of comand line.

4.1.3 adventure_thermal_fluid_hex

The usage of adventure_thermal_fluid_hex is as follows;

```
% mpirun -np <np> adventure_thermal_fluid_hex <mesh> <ctrl> <rest_out> [<rest_in>]
  <np>       : <np>  (number of processors)
  <mesh>     : <mesh>_<np>.adv  (mesh file name)
  <ctrl>     : <ctrl>  (control file name)
  <rest_out> : <restout>_<num_steps>_<np>.adv  (restart output file name)
  <rest_in>  : <rest_in>_<np>.adv  (restart input file name)
```

Refer to the hints listed below;

- All the arguments is the same as the ones of the adventure_fluid_hex.
- The format of control file is different. (see next chapter for detail)

4.2 Pre-Processing Tools

4.2.1 advfluid_pre_cavity

The usage of advfluid_pre_cavity is as follows;

```
% advfluid_pre_cavity <tet | hex> <num_division_per_direction> <write file name>
```

Refer to the hints listed below;

- To the first argument, assign either "tet"(tetrahedron) or "hex"(hexahedron).

- To the second argument, assign the number of divison in one direction of cubic. When you assign N, the number of elements is $5N^3$ in the case of tetrahedral element, N^3 in the case of hexahedral element, and the number of nodes is $(N+1)^3$ in the both cases
- To the third argumentassign an arbitrary file name.

4.2.2 advfluid_pre_cavity_nc

The usage of advfluid_pre_cavity_nc is as follows;

```
% advfluid_pre_cavity_nc <num_division_per_direction> <write file name>
```

Refer to the hints listed below;

- This pre-processing tool is only for adventure_thermal_fluid_hex.
- To the first argument, assign the number of divison in one direction of cubic. When you assign N, the number of elements is N^3 and the number of nodes is $(N+1)^3$
- To the second argument, assign an arbitrary file name.

4.2.3 advfluid_pre_pillar

The usage of advfluid_pre_pillar is as follows;

```
% advfluid_pre_pillar <filename>
```

Refer to the hints listed below;

- Assign an arbitrary file name to the argument.
- When you run advfluid_pre_pillar, you are required to input 5 parameters of model size. Input the appropriate values refer to the fig.1
- Then, you are required to input 3 parameters of the element divison in the X-Y plane. Input the appropriate values refer to the fig.1 again. itemAfter checking the node distribution in the X-Y plane, input “y” for next step.
- Next, you are required to input 2 parameters of the element divison in the X-Z plane. Input the appropriate values refer to the fig.1 again.
- After checking the node distribution in the X-Z plane, input “y” for generating mesh data filen

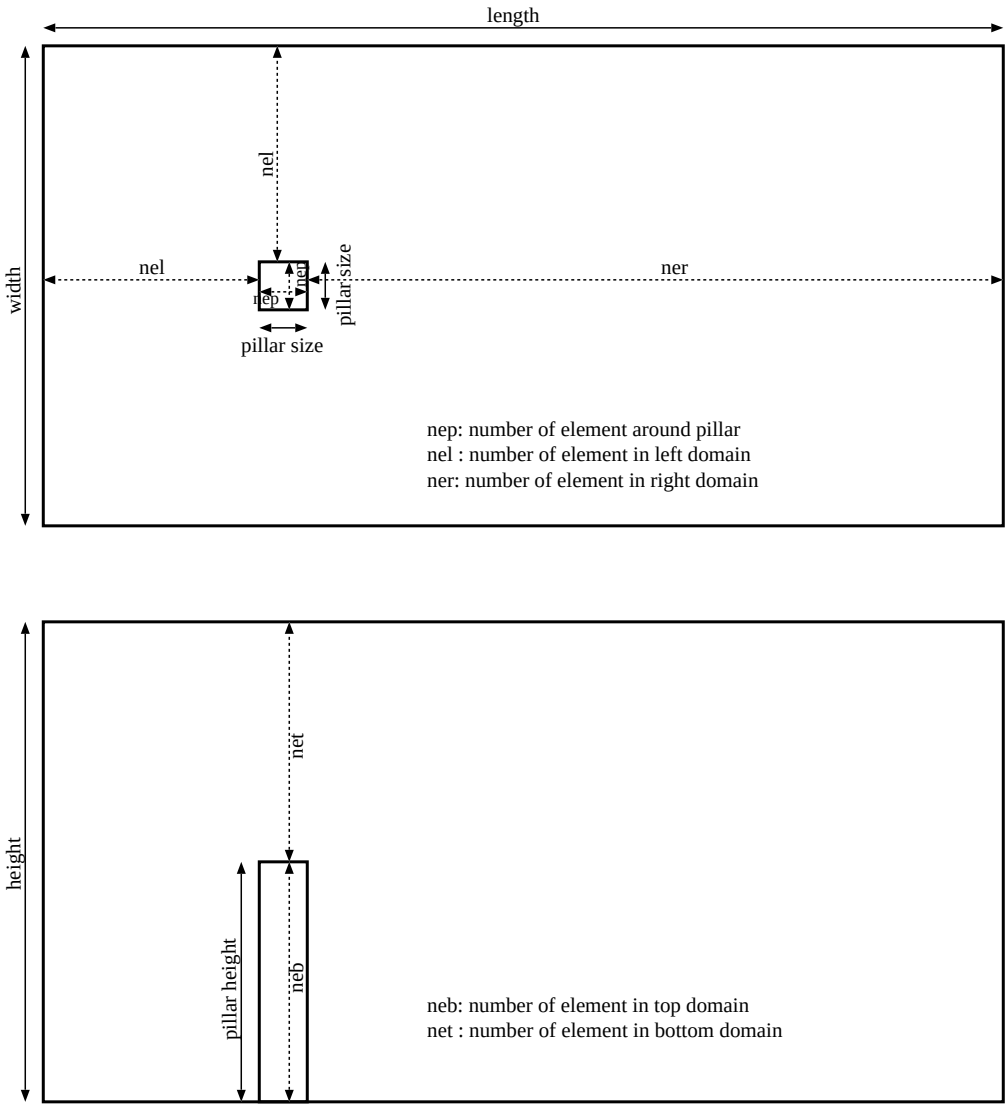

Figure 1: Parameters for advfluid_pre_pillar

4.3 Data Converters

4.3.1 advfluid_mesh2ucd

The usage of advfluid_mesh2ucd is as follows;

```
% advfluid_mesh2ucd [-M] <mesh_file> <ucd_file>
  -M          : Micro AVS mode(including I.C and B.C data)
  <mesh_file> : input mesh file name(<mesh_file>)
  <ucd_file>  : output UCD file name(<ucd_file>.inp)
```

Refer to the hints listed below;

- "-M" option is required to output the boundary conditions and initial conditions. This option is essential for the Micro AVS.
- To the `<mesh_file>`, assign mesh data file name to be converted.
- To the `<ucd_file>`, assign UCD format data file name. The file extension `.inp` is attached automatically.

4.3.2 advfluid_p_mesh2ucd

The usage of advfluid_p_mesh2ucd is as follows;

```
% advfluid_p_mesh2ucd [-M] <np> <mesh_file> <ucd_file>
  -M          : Micro AVS mode(including I.C and B.C data)
  <np>        : number of PEs(number of domain)
  <mesh_file> : input mesh file name(<mesh_file>_<n>.adv)
  <ucd_file>  : output UCD file name(<ucd_file>_<n>.inp)
```

Refer to the hints listed below;

- "-M" option is required to output the boundary conditions and initial conditions. This option is essential for the Micro AVS.
- Assign the number of domains to the `<np>`.
- To the `<mesh_file>`, assign mesh data file name to be converted. At that time, exclude the domain number `_<np>` and file extension `.adv` from the mesh file name.
- To the `<ucd_file>`, assign UCD format data file name. The domain number and the file extension `.inp` are attached automatically.

4.3.3 advfluid_rest2ucd

The usage of advfluid_rest2ucd is as follows;

```
% advfluid_rest2ucd <mesh_file> <rest_file> <ucd_file>
  <mesh_file> : input mesh file name(<mesh_file>)
  <rest_file> : input restart file name(<rest_file>)
  <ucd_file>  : output UCD file name(<ucd_file>.inp)
```

Refer to the hints listed below;

- To the `<mesh_file>`, assign mesh data file name to be converted.
- To the `<ucd_file>`, assign UCD format data file name. The file extension `.inp` is attached automatically.

4.3.4 advfluid_p_rest2ucd

The usage of advfluid_p_rest2ucd is as follows;

```
% advfluid_p_rest2ucd <np> <mesh_file> <rest_file> <ucd_file>
  <np>        : number of PEs(number of domain)
  <mesh_file> : input mesh file name(<mesh_file>_<n>.adv)
  <rest_file> : input restart file name(<rest_file>_<n>.adv)
  <ucd_file>  : output UCD file name(<ucd_file>_<n>.inp)
```

Refer to the hints listed below;

- Assign the number of domains to the `<np>`.
- To the `<mesh_file>`, assign mesh data file name to be converted. At that time, exclude the domain number `_<np>` and file extension `.adv` from the mesh file name.
- To the `<rest_file>`, assign restart data file name to be converted. At that time, exclude the domain number `_<np>` and file extension `.adv` from the mesh file name.
- To the `<ucd_file>`, assign UCD format data file name. The domain number and the file extension `.inp` are attached automatically.

4.3.5 advfluid_rest2ucd_nc

The usage of advfluid_rest2ucd_nc is as follows;

```
% advfluid_rest2ucd_nc <mesh_file> <rest_file> <ucd_file>
  <mesh_file> : input mesh file name(<mesh_file>)
  <rest_file> : input restart file name(<rest_file>)
  <ucd_file>  : output UCD file name(<ucd_file>.inp)
```

Refer to the hints listed below;

- To the `<mesh_file>`, assign mesh data file name to be converted.
- To the `<rest_file>`, assign restart data file name to be converted.
- To the `<ucd_file>`, assign UCD format data file name. The file extension `.inp` is attached automatically.

5 Data File Format

The format of mesh data file and restart data file are written in "Document" form defined in the ADVENTURE_IO. Otherwise, the control data file is written in text form.

5.1 Mesh Data

The mesh data file contains the element data, the node data, the initial condition data and the boundary condition data. Each data has more than one "Document", which are written in the following formats.

5.1.1 Tetrahedral (P1-P1) Elements

The "Document" of tetrahedral (P1-P1) elements data is written in the format below;

```
[Properties]

content_type            =       "Element"
num_items               =       <num_element>
num_items_orig          =       <num_element_before_decomposed>
format                  =       "i4i4i4i4"
num_node_per_element    =       4
element_type            =       "3DLinearTetrahedron"
dimension               =       3
index_byte              =       4

[Mass data] (element-node connectivity)

nop[0][0], nop[0][1], nop[0][2], nop[0][3]
nop[1][0], nop[1][1], nop[1][2], nop[1][3]
...
(The number of element x 4 data of int type are listed in this order.)
```

5.1.2 Hexahedral (Q1-P0) Elements

The "Document" of hexahedral (Q1-P0) elements data is written in the format below;

```
[Properties]

content_type            =       "Element"
num_items               =       <num_element>
num_items_orig          =       <num_element_before_decomposed>
format                  =       "i4i4i4i4i4i4i4i4"
num_node_per_element    =       8
```

```
element_type            =       "3DLinearHexahedron"
dimension               =       3
index_byte              =       4

[Mass data] (element-node connectivity)

nop[0][0], nop[0][1], nop[0][2], nop[0][3],
  nop[0][4],nop[0][5], nop[0][6], nop[0][7]
nop[1][0], nop[1][1], nop[1][2], nop[1][3],
  nop[1][4], nop[1][5], nop[1][6], nop[1][7]
...
(The number of element x 8 data of int type are listed in this order.)
```

5.1.3 Nodes

The "Document" of nodes data is written in the format below;

```
[Properties]

content_type            =       "Node"
num_items               =       <num_node>
num_items_orig          =       <num_node_before_decomposed>
format                  =       "f8f8f8"
dimension               =       3
index_byte              =       4

[Mass data] (coordinates of nodes)

x[0], y[0], z[0]
x[1], y[1], z[1]
...
(The number of node x 3 data of double type are listed in this order.)
```

5.1.4 Initial Conditions

The "Document" of initial condition is written in the format below; (Note that this is the case of attaching a uniform velocity field in the whole analysis domain.)

```
[Properties]

content_type            =       "FEGenericAttribute"
num_items               =       1
format                  =       "f8f8f8"
```

```
fega_type               =       "AllNodeConstant"
label                   =       "VelocityIC"
coordinate              =       7
dd_option               =       "dirichlet"
index_byte              =       4

[Mass data] (initial velocity field)

u, v, w
```

5.1.5 Boundary Conditions : No-slip boundary

The "Document" of the no-slip boundary condition is written in the format below;

```
[Properties]

content_type            =       "FEGenericAttribute"
num_items               =       <num_node>
format                  =       "f8f8f8"
fega_type               =       "NodeConstant"
label                   =       "VelocityBC"
coordinate              =       7
dd_option               =       "dirichlet"
index_byte              =       4

[Mass data] (value of velocity, the list of assigned nodes)

0.0, 0.0, 0.0
n[0], n[1], ...
(The <num_node> data of int type are listed in this order.)
```

5.1.6 Boundary Conditions : Velocity boundary

The "Document" of the velocity boundary condition is written in the format below;

```
[Properties]

content_type            =       "FEGenericAttribute"
num_items               =       <num_node>
format                  =       "f8f8f8"
fega_type               =       "NodeConstant"
label                   =       "VelocityBC"
coordinate              =       7
```

```
dd_option               =       "dirichlet"
index_byte              =       4

[Mass data] (value of velocity, the list of assigned nodes)

u, v, w
n[0], n[1], ...
(The <num_node> data of int type are listed in this order.)
```

5.2 Restart Data

The restart data file contains the velocity field and the pressure field. The temparature field data is also included in the restart data of adventure_thermal_fluid_hex. Each data has more than one "Document", which are written in the following formats.

5.2.1 Velocity Field

The "Document" of the velocity field is written in the format below;

```
[Properties]

content_type            =       FEGenericAttribute
num_items               =       <num_node>
format                  =       f8f8f8
fega_type               =       AllNodeVariable
label                   =       VelocityRestartData
index_byte              =       4
num_steps               =       <time_steps>
time                    =       <time>

[Mass data] (velocity field)

u[0], v[0], w[0]
u[1], v[1], w[1]
...
(The number of node x 3 data of double type are listed in this order.)
```

5.2.2 Pressure Field (P1-P1 Elements)

The "Document" of the pressure field of P1-P1 elements is written in the format below;

```
[Properties]
```

```
content_type            =       FEGenericAttribute
num_items               =       <num_node>
format                  =       f8
fega_type               =       AllNodeVariable
label                   =       PressureRestartData
index_byte              =       4
num_steps               =       <time_steps>
time                    =       <time>

[Mass data] (pressure field)

p[0], p[1], ...
(The number of node x 1 data of double type are listed in this order.)
```

5.2.3 Pressure Field (Q1-P0 Elements)

The "Document" of the pressure field of Q1-P0 elements is written in the format below;

```
[Properties]

content_type            =       FEGenericAttribute
num_items               =       <num_element>
format                  =       f8
fega_type               =       AllElementVariable
label                   =       PressureRestartData
index_byte              =       4
num_steps               =       <time_steps>
time                    =       <time>

[Mass data] (pressure field)

p[0], p[1], ...
(The number of element x 1 data of double type are listed in this order.)
```

5.2.4 Temparature Field

The "Document" of the temparature field is written in the format below;

```
[Properties]

content_type            =       FEGenericAttribute
num_items               =       <num_node>
format                  =       f8
```

```
fega_type               =       AllNodeVariable
label                   =       TemparatureRestartData
index_byte              =       4
num_steps               =       <time_steps>
time                    =       <time>

[Mass data] (temparature field)
t[0], t[1], ...
(The number of node x 1 data of double type are listed in this order.)
```

5.3 Control Data

In the control data file, the analysis conditions (e.g. the Reynolds number and the time increment) are written in text form. The format of control data file is depend on the kind of solver, as follows;

5.3.1 adventure_fluid_tet

The control data for adventure_fluid_tet is written in the format below;

```
[Example of contorl data for the adventure_fluid_tet]

dt              1.000000e-02
t_end           1.000000e+00
mu              0.01
rho             1.0
solver_type     1
max_cg          1000
eps_cg          1.000000e-10
diag_scaling_do 1
rest_in_do      0
rest_intvl      100
```

Each variables mean as follows;

```
dt              time increment
t_end           total analysis time
mu              viscosity of fluid
rho             density of fluid
solver_type     0:Bi-CGSTAB, 1:GPBi-CG, 2:Bi-CGSTAB2, m>=3:GMRES(m)
max_cg          maximum number of iterations of Bi-CGSTAB method
eps_cg          criteria of convergence for Bi-CGSTAB method
diag_scaling_do 0:without diagonal scaling, 1:with diagonal scaling
rest_in_do      flag for restart computing (0;No, 1;Yes)
rest_intvl      interval to output restart data (time steps)
```

5.3.2 adventure_fluid_hex

The control data for adventure_fluid_hex is written in the format below;

```
[Example of contorl data for the adventure_fluid_hex]

dt              2.000000e-02
t_end           1.000000e+01
max_cg          1000
eps_cg          1.000000e-6
eps_mac         1.000000e-6
rest_in_do      0
rest_intvl      1000
re              100
hourg_level     2
```

Each variables mean as follows;

```
dt              time increment
t_end           total analysis time
max_cg          maximum number of iterations of CG method
eps_cg          criteria of convergence for CG method
eps_mac         criteria of convergence for MAC method
rest_in_do      flag for restart computing (0;No, 1;Yes)
rest_intvl      interval to output restart data (time steps)
re              Reynolds number
hourg_level     level of hourglass controler(0;nothing, 1;Gresho 2;Okuda et al.)
```

5.3.3 adventure_thermal_fluid_hex

The control data for adventure_thermal_fluid_hex is written in the format below;

```
[Example of contorl data for the adventure_thermal_fluid_hex]

dt              2.000000e-02
t_end           1.000000e+01
max_cg          1000
eps_cg          1.000000e-6
eps_mac         1.000000e-6
rest_in_do      0
rest_intvl      1000
ra              1.000000e+5
pr              7.100000e+0
hourg_level     2
```

Each variables mean as follows;

```
dt              time increment
t_end           total analysis time
max_cg          maximum number of iterations of CG method
eps_cg          criteria of convergence for CG method
eps_mac         criteria of convergence for MAC method
rest_in_do      flag for restart computing (0;No, 1;Yes)
rest_intvl      interval to output restart data (time steps)
ra              Rayleigh number
pr              Prandtl number
hourg_level     level of hourglass controler(0;nothing, 1;Gresho 2;Okuda et al.)
```

6 Examples

6.1 Lid-driven Cubic Cavity Flow

In this section, the procedures of solving the lid-driven cubic cavity flow problem are shown using adventure_fluid_tet, which is the incompressible fluid solver for the tetrahedral (P1-P1) elements.

6.1.1 Analysis Procedures

To solve the lid-driven cubic cavity flow problem, follow the steps shown below;

1. Using the pre-processing tool, generate mesh data.
   `% advfluid_pre_cavity tet 8 tet8`
   In this case, the cubic is divided into 8x8x8 elements.

2. Check the generated mesh data using the data converter.
   `% advfluid_mesh2ucd -M tet8 tet8`
   This command generates UCD format data file (tet8.inp) for AVS. (see fig.3)

3. Edit the control data file (sample.ctrl), which is generated together with the mesh data file(tet8). (see fig.2)
   `% vi sample.ctrl`

```
                dt              2.500000e-02
                t_end           2.500000e+00
                mu              1.000000e-02
                rho             1.000000e+00
solver_type     1
                max_cg          1000
                eps_cg          1.000000e-08
diag_scalig_do  1
                rest_in_do      0
                rest_intvl      50
```

Figure 2: Control data for lid-driven cubic cavity flow

4. Without the domain decomposition method, i.e. using single processor, it is necessary to change the name of mesh data file by hand. In particular, put the symbolic link as follows;
   `% ln -s tet8 tet8_0.adv`

5. Run the analysis code.
   `% mpirun -np 1 adventure_fluid_tet log sample.ctrl tet8 tet8`
   Note that it is necessary to use "mpirun" even in the case of using single processor.

6. If you use the control data in fig.2, the restart data file is generated every 50 time steps. Therefore, there are two restart data files after the analysis is finished.
   `tet8_000050_0.adv`
   `tet8_000100_0.adv`

7. Convert the restart data file into the UCD format data file.
   `% advfluid_rest2ucd tet8 tet8_000050_0.adv tet8_000050`
   `% advfluid_rest2ucd tet8 tet8_000100_0.adv tet8_000100`
   This command generates the UCD format data files as follows;
   `tet8_000050.inp` (result at the 50 time steps)
   `tet8_000100.inp` (result at the 100 time steps) (fig.4)

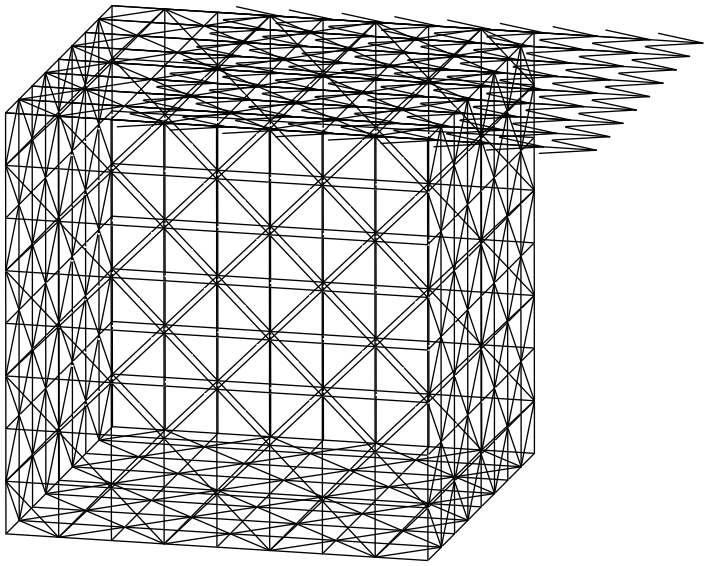

Figure 3: Analyasis model of 3-dimensional cavity flow

6.2 Lid-driven Cubic Cavity Flow (Parallel Processing)

In this section, the procedures of solving the lid-driven cubic cavity flow problem in parallel processing with the domain decomposition method are shown using adventure_fluid_tet, which is the incompressible fluid solver for the tetrahedral (P1-P1) elements.

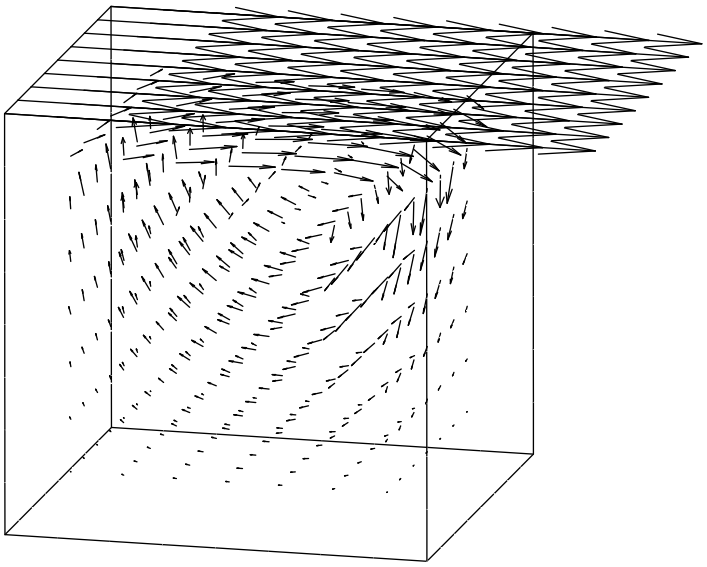

Figure 4: Analyasis result of 3-dimensional cavity flow

6.2.1 Analysis Procedures

To solve the lid-driven cubic cavity flow problem in parallel processing, follow the steps shown below;

1. Using the pre-processing tool, generate mesh data.
   `% advfluid_pre_cavity tet 8 tet8`
   In this case, the cubic is divided into 8x8x8 elements.

2. Decompose the domain using the ADVENTURE_Metis.
   At first, make the data output directory (out) for the ADVENTURE_Metis.
   `% mkdir out`
   Then, decompose the domain using the adventure_metis with "`-DYS`" option.
   `% mpirun -np 2 adventure_metis -DYS tet8 tet8`
   This command generates the decomposed mesh data files as follows;
   `out/tet8_0.adv`
   `out/tet8_1.adv`

3. Check the generated mesh data using the data converter.
   `% advfluid_p_mesh2ucd -M 2 out/tet8 out/tet8`
   This command generates UCD format data files as follows;
   `out/tet8_0.inp`
   `out/tet8_1.inp`

4. Edit the control data file (sample.ctrl), which is generated together with the mesh data file(tet8). (see fig.2)

5. Run the analysis code.
   `% mpirun -np 2 adventure_fluid_tet log sample.ctrl out/tet8 out/tet8`

6. If you use the control data in fig.2, the restart data file is generated every 50 time steps. Therefore, there are four restart data files after the analysis is finished.
   `out/tet8_000050_0.adv`
   `out/tet8_000050_1.adv`
   `out/tet8_000100_0.adv`
   `out/tet8_000050_1.adv`

7. Convert the restart data file into the UCD format data file.
   `% advfluid_p_rest2ucd tet8 tet8_000050 tet8_000050`
   `% advfluid_p_rest2ucd tet8 tet8_000100 tet8_000100`
   This command generates the UCD format data files as follows;
   `tet8_000050_0.inp` (result at the 50 time steps (domain 0))
   `tet8_000050_1.inp` (result at the 50 time steps (domain 1))
   `tet8_000100_0.inp` (result at the 100 time steps (domain 0))
   `tet8_000100_1.inp` (result at the 100 time steps (domain 1))

References

[1] ADVENTURE Project Home Page, http://adventure.q.t.u-tokyo.ac.jp/

[2] Yagawa, G., Nakabayashi, Y. and Okuda, H., "Large-Scale Finite Element Fluid Analysis by Massively Parallel Processors", *Parallel Computing*, Vol.23, pp.1365-1377 (1997).

[3] Nakabayashi,Y., Okuda,H. and Yagawa,G., "Parallel Finite Element Fluid Analysis on an Element-by-Element Basis",*Computational Mechanics*, Vol.18, pp.377-382 (1996).

[4] Okuda,H. and Yagawa,G., "A One-point Quadrature Technique with a New Hourglass Controller for Large Scale Finite Element Fluid Analysis.", *Proc. of 2nd Japan-US Symposium on FEM in Large-Scale CFD*, pp.125-128 (1994).

[5] Gresho,P.M., Chan,S.T., Lee,R.L. and Up-son,C.D., "A Modified Finite Element Method for Solving the Time-Dependent Incompressible Navier-Stokes Equations by a Fractional Step Method.", *Int. J. Num. Meth. in Fluids*, Vol.4, pp.557-598 (1984).

[6] Tezduyar,T.E., Aliabadi,S., Behr,M., Johnson,A., Kalro,V. and Litke,M., "Flow Simulation and High Performance Computing", *Computational Mechanics*, Vol.18, pp.397-412 (1995).

[7] Kalro,V., Aliabadi,S., Garrard,W., Tezduyar,T.E., Mittal,S. and Stein,K., "Parallel Finite Element Simulation of Large Ram-Air Parachutes", *Compt. Meth. Appl. Mech. Fluids* (1997).