ABSTRACT

Title of Thesis:	VERIFICATION TESTS OF MASS CONSERVATION FOR FIREFOAM AND DEVELOPMENT OF A USER'S GUIDE
	Shiyun Wu, Master of Science, 2019
Thesis Directed By:	Professor Arnaud Trouvé Department of Fire Protection Engineering

The objective of this study is to develop basic verification tests for FireFOAM, a large eddy simulation (LES) solver developed by FM Global for fire applications, and based on the general-purpose Computational Fluid Dynamics (CFD) solver called OpenFOAM. These tests will be eventually included in an upcoming User Guide for FireFOAM users. We focus here on a series of tests developed to evaluate global species mass conservation statements. The series includes a two-dimensional helium plume case, a three-dimensional helium plume case and a three-dimensional pool fire case. The two-dimensional helium plume case focuses on the effects of changing the temporal discretization scheme in FireFOAM. The three-dimensional helium plume case focuses on the effects of changing the spatial discretization scheme used to describe the convection terms in the governing equations. Finally, the three-dimensional pool fire case focuses on the effects of changing the number of outer loops used to provide coupling between the governing equations that are solved sequentially. The results of the tests provide valuable insight for FireFOAM users who need to make

numerical choices on the temporal discretization scheme, the spatial discretization scheme and the number of outer loops with little guidance on the impact of these choices.

VERIFICATION TESTS OF MASS CONSERVATION FOR FIREFOAM AND DEVELOPMENT OF A USER'S GUIDE

by

Shiyun Wu

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Master of Science
2019

Acknowledgements

I want to express my gratitude to my advisor Professor Arnaud Trouvé. Thanks for the kind help and patient guides on my research project. Thanks for helping me solve problems I have met in the past.

I want to thank for Dr. Michael J. Gollner and Dr. Peter B. Sunderland for joining my committee.

I want to thank for all the kindness I have received from our research group. Thanks for the help and patience of our research group members, Salman Verma, Rui Xu, Sergio Vargas-Córdoba, Mohamed Ahmed, Talia Fan, Cong Zhang, Alexis Pascal Marchand, Le Van Minh.

I want to thank for faculties, staffs, and students in our Fire Protection Engineering department. Without any one of you, I may not finish my program.

Table of Contents

Acknowledgements ii
Table of Contents iii
List of Tables iv
List of Figures v
Chapter 1:Introduction 1
  Purpose of the Project 1
  Thesis Outline 4
  Introduction of FireFOAM 7
    Fire Modeling 7
    FireFOAM Overview (OpenFOAM) 7
Chapter 2: Development of the Diagnostic 10
  Species Mass Conservation Diagnostic Concept 10
  Data Collection 12
    Left Hand Side Data 12
    Right Hand Side Data 16
  Conclusion 20
Chapter 3: Helium Plume Two-Dimensional Case 21
  Case Description and Setup 21
  Time Scheme 27
  Postprocessing 30
  Results 31
  Discussion and Conclusion 43
Chapter 4: Helium Plume Three-Dimensional Case: 45
  Case Description and Setup 45
  Convetion Scheme 47
  Results 49
    Helium Field Visualization 49
    Plots 53
    Discussion and Conculsion 55
Chapter 5: Pool Fire Case 57
  Case Description and Setup 57
  Mixture Fraction Diagnostic Test 60
  Number of Outer Loops 63
  Visualization of Fire Plume Method 64
  Results 66
  Discussion and Conclusion 68
Chapter 6: Conclusion and Future Work 70
  Conclusion 70
  Future Work 71
Bibliography 72

List of Tables

Table 1.1.1 Verification tests in upcoming FireFOAM Userguide…………………...3
Table 2.1.1 Symbols in Global Species Mass Conservation Equation………………10
Table 3.2.1 Time Scheme Test List……………………………………………….29
Table 4.2.1 Convection Scheme Test List………………………………………...49
Table 5.3.1 Test Number for Fire Case……………………………………………63

List of Figures

Fig. 1.3.2.1 ParaView User Interface……………………………………………………8
Fig. 2.2.1.1 firFOAM-dev content……………………………………………………...12
Fig. 2.2.1.2 *solver* directory content…………………………………………………13
Fig. 2.2.1.3 *include* directory content………………………………………………..13
Fig. 2.2.1.4 Specification of volScalarField: *maker1* in *infoFields.H* file…………...13
Fig. 2.2.1.5 *fireFOAM.C* file before modification…………………………………...14
Fig. 2.2.1.6 *fireFoam.C* file after modification………………………………………14
Fig. 2.2.1.7 *clean.maker* file example………………………………………………..15
Fig. 2.2.1.8 *regions* definition example in *setFieldsDict*…………………………….16
Fig. 2.2.2.1 Specification of surfaceScalarField: phiFu in *infoFieldsOutput.H* file…17
Fig. 2.2.2.2 mass flow rate calculation across a selected surface example in *infoOutput.H* file……………………………………………………………..17
Fig. 2.2.2.3 Definition of surface in *TopoSetDict* file………………………………18
Fig. 2.2.2.4. Specification of *phiFu* Field output for surface *Height100cm*………....19
Fig. 3.1.1 *blockMeshDict* file example………………………………………………21
Fig. 3.1.2 Boundary surface definition example in *blockMeshDict* file……………..23
Fig.3.1.3 Velocity file *U* example……………………………………………………24
Fig. 3.1.4 Computational domain and meshes………………………………………26
Fig. 3.2.1 Temporal scheme: *Euler* in *fvScheme* file………………………………...27
Fig. 3.2.2 Temporal scheme: *backward* in *fvScheme* file……………………………27
Fig. 3.2.3 Turn on CFL number control……………………………………………..28
Fig. 3.4.1 Helium plume field at 440s………………………………………………..31
Fig. 3.4.2 Euler Scheme variable time step at z_2 = 1m………………………………32
Fig. 3.4.3 Euler Scheme: variable time step at z_2 = 2m……………………………...32
Fig. 3.4.4 Euler Scheme: variable time step at z_2 = 3m……………………………...33
Fig. 3.4.5 Euler Scheme: variable time step at z_2 = 4m……………………………...33
Fig. 3.4.6 Euler Scheme: variable time step vs. constant time step at z_2 = 1m……...34
Fig. 3.4.7 Euler Scheme: variable time step vs. constant time step at z_2 = 2m……...35
Fig. 3.4.8 Euler Scheme: variable time step vs. constant time step at z_2 = 3m……...35
Fig. 3.4.9 Euler Scheme: variable time step vs. constant time step at z_2 = 4m……...36
Fig. 3.4.10 Euler scheme with constant time step: Maximum CFL number………...36
Fig. 3.4.11 backward Scheme: variable time step at z_2 = 1m………………………..37
Fig. 3.4.12 backward Scheme: variable time step at z_2 = 2m………………………..37
Fig. 3.4.13 backward Scheme: variable time step at z_2 = 3m………………………..38
Fig. 3.4.14 backward Scheme: variable time step at z_2 = 4m………………………..38
Fig. 3.4.15 backward Scheme: variable time step vs. constant time step at z_2 = 1m...39
Fig. 3.4.16 backward Scheme: variable time step vs. constant time step at z_2 = 2m...39
Fig. 3.4.17 backward Scheme: variable time step vs. constant time step at z_2 = 3m...40
Fig. 3.4.18 backward scheme: variable time step vs. constant time step at z_2 = 4m...40
Fig. 3.4.19 backward scheme with constant time step: Maximum CFL number……41
Fig. 3.4.20 Variable time step: Euler vs. backward scheme at z_2 = 1m……………..41
Fig. 3.4.21 Variable time step: Euler vs. backward scheme at z_2 = 2m……………..42
Fig. 3.4.22 Variable time step: Euler vs. backward scheme at z_2 = 3m……………..42

Fig. 3.4.23 Variable time step: Euler vs. backward scheme at z_2 = 4m……………...43
Fig. 4.1.1 Velocity file *U* example for three-dimensional case……………………...45
Fig. 4.1.2 Visualization of domain by *Points* method in ParaView…………………46
Fig. 4.1.3 Structure of Three-dimensional domain…………………………………..47
Fig. 4.2.1 Convection scheme: *limitedLinear < 1.0 >* definition in *fvShceme* file….48
Fig. 4.2.2 Convection Scheme: *limitedLinear <0.2>* definition in *fvScheme* file…..48
Fig. 4.3.1.1 *Clip* button in ParaView tool bar………………………………………49
Fig. 4.3.1.2 *Properties* section for *Slice* method……………………………………50
Fig. 4.3.1.3 limitedLinear <1.0> helium plume field at time = 10s…………………50
Fig. 4.3.1.4 limitedLinear <0.2> helium plume field at time = 10s…………………51
Fig. 4.3.1.5 *Mapping Data* Section in ParaView……………………………………51
Fig. 4.3.1.6 *Set Range* section in ParaView…………………………………………52
Fig. 4.3.1.7 Edit Color and Legend Parameters……………………………………..52
Fig. 4.3.1.8 Helium Mass Fraction Range…………………………………………...52
Fig. 4.3.2.1 Instantaneous relative error: limitedLinear <0.2> vs.
limitedLinear <1.0> at z_2 =0.4m ……………………………………….53
Fig. 4.3.2.2 Instantaneous relative error: limitedLinear <0.2> vs.
limitedLinear <1.0> at z_2 = 0.8m………………………………………..53
Fig. 4.3.2.3 Instantaneous relative error: limitedLinear <0.2> vs.
limitedLinear <1.0> at z_2 = 1.2m………………………………………..54
Fig. 4.3.2.4 Instantaneous relative error: limitedLinear <0.2> vs.
limitedLinear <1.0> at z_2 = 1.6m………………………………………..54
Fig. 4.3.2.5 root mean square of instantaneous relative error vs. heights for
limitedLinear <0.2> and limitedLinear <1.0>………………………….55
Fig. 5.1.1 *reactions* file example……………………………………………………57
Fig. 5.1.2 *combustionProperties* file Example………………………………………57
Fig. 5.2.1 Comparison between two diagnostics at z_2 = 0.4m………………………61
Fig. 5.2.2 Comparison between two diagnostics at z_2 = 0.8m………………………61
Fig. 5.2.3 Comparison between two diagnostics at z_2 = 1.2m………………………62
Fig. 5.2.4 Comparison between two diagnostics at z_2 = 1.6m………………………62
Fig. 5.3.1 nOuterCorrectors specification in *fvSolution* file………………………63
Fig. 5.4.1 *Mapping Data* Section in ParaView………………………………………64
Fig. 5.4.2 *Choose Preset* Section in ParaView………………………………………65
Fig. 5.4.3 *Background* section in ParaView…………………………………………65
Fig. 5.5.1 Fire Plume at time = 2.5s…………………………………………………66
Fig. 5.5.2 Fire Plume at time = 3s…………………………………………………...66
Fig. 5.5.3 root mean square (rms) of instantaneous relative error vs.
nOuterCorrectors……………………………………………………….67

Chapter 1: Introduction

<u>Purpose of the Project</u>

The objective of this thesis is to develop a series of verification cases, which are about to be included in a user guide, to assess the species mass conservation of FireFOAM.

Validation & verification methods are always used to assess a CFD software. NASA's definition of validation and verification are as follows:

"Validation assessment determines if the computational simulation agrees with physical reality. It examines the science in the models through comparison to experimental results [1]."

"Verification assessment determines if the programming and computational implementation of the conceptual model is correct [1]."

Validation can be assessed by comparing correlations from simulation with correlations from experiments. Researches have been conducted within different aspects to assess the validation of FireFOAM solver's code. In fire spread aspect, Ding, Wang, and Lu (2014) studied the temperature changes and flame spread rate at a selected vertical height on a board with 3 variables, angle of the board to horizontal plane, width of the board, and HRR (heat release rate) generated by the heat source (fire) at the bottom of the board [2]. They found the correlations between temperature changes and flame spread rate with the three variables as mentioned earlier roughly agrees with the correlations from experiments, while some are not perspicuous [2]. With respect to fire plumes, Wang, Chatterjee, and deRis (2010) studied fire plumes and compared correlations from simulation with McCaffrey's experimental results [3]. They have found consistency between simulation and experiments results, and an example can be correlations between flame height and entrainment. [3]. Another study conducted by Maragkos, Rauwoens and Merci (2012) used buoyant helium plume case to assess the validation of two fire dynamics solver FireFOAM and FDS [4]. They found that FireFOAM similar results compared to those found through experimentations in most cases [4]. With respect for wildland fires, Verma (2019) has studied wildland fires' structure with the effects of slope and wind using FireFOAM [5].

According to definition from NASA [1], verification can be understood as if the calculation of CFD codes is correct. In University of Maryland fire modeling group, a series of verification tests have been done before. Vilfayeau's (2015) used differently designed cases, involving with gas, liquid and solid respectively, to assess the verification of FireFOAM solver's codes [6]. According to his results, FireFOAM performs well in these cases [6]. Afterwards, Li (2017) and Córdoba (2018) have studied some typical flow problems, i.e. Lamb-Oseen vortex with co-flow, to assess FireFOAM verification [7][8]. Based on their contributions, there are several cases included in an upcoming user guide for FireFOAM. Although FireFOAM widely applied into fire modeling field, there are not many documents which can give introductions for using FireFOAM. Córdoba (2018) started building a user's guide for FireFOAM in 2018 [8].

Number	**CASE NAME**	**Intent**
1	Lamb Oseen Vortex	(1) grid resolution (2) Temporal discretization scheme
2	Taylor Green Vortex	(1) Spatial discretization scheme
3	Decay of Homogeneous Isotropic	(1) Turbulence model

Table 1.1.1 Verification tests in the upcoming FireFOAM Userguide [6][7][8]

The first case uses laminar flow Lamb Oseen Vortex as an example to evaluate the grid resolution and temporal discretization scheme effects on the accuracy of FireFOAM's solution [8]. The results show that ten meshes across the vortex are required to give an accurate prediction of the vortex variations [8].

The second case uses turbulent flow Taylor Green Vortex as an example to evaluate the dissipative errors of different spatial discretization schemes bringing into simulations [8]. Spatial scheme with lower dissipation is suggested [8].

The third case uses turbulent flow Decay of Homogeneous Isotropic to evaluate the turbulence model [8]. This test compares the performance of turbulence sub-grid models on the predictions of fluid behaviors [8].

These cases test the important parameters in FireFOAM application, which provides valuable guidance on FireFOAM users numerical choices.

Mass conservation, species mass conservation, energy conservation and momentum conservation are the basic physics laws. If the calculations of CFD codes are correct, outcomes should result in conservation of mass, energy and momentum. In this thesis, multiple verification cases are developed to assess species mass conservation of FireFOAM. These cases are to be included in a user guide for FireFOAM.

Thesis Outline

A diagnostic is designed to check if FireFOAM is conserved in mass in global view. In total, three cases are constructed to assess species mass conservation for FireFOAM in different aspects.

The error of species mass conservation associated with solving process is a function of numerical schemes, temporal discretization scheme, and space discretization scheme, and number of outer loops (*nOuterCorrectors*) applied in the simulation. The three cases are to assess these factors respectively. People may be familiar with errors due to the temporal discretization scheme and space discretization scheme but may not be very familiar with errors due to the number of outer loops. The number of outer loops is a parameter in PIMPLE algorithm. The number of outer loops controls the number of iterations in solving the pressure-momentum coupling in the governing equation [9][18]. At each time step, the PIMPLE algorithm calculates the pressure field based on previous velocities field, and then use new pressure field to correct the velocities field, and repeat this process based on number of outer loops [9][18]. The last case of this project is going to evaluate the effects on species mass conservation statement by applying different values of number of outer loops in simulation.

Chapter 1: Introduction

Chapter 2: Development of species mass conservation diagnostic

Chapter 3: Assessment of time scheme effects on mass (species) conservation.

Time scheme: (1) Euler time scheme; (2) backward time scheme

Species mass conservation assessment starts from a two-dimensional buoyant helium plume case. Helium is injected into computational domain through a burner. This is done to assess the effects of time scheme species mass conservation.

Chapter 4: Assessment of convection scheme effects on species mass conservation.

Convection Scheme: (1) limitedLinear <1>; (2) limitedLinear <0.2>

In this chapter, the assessment moves from two dimensional to three dimensional. The convection scheme limitedLinear <coefficient>is commonly applied in FireFOAM. Users can select the coefficient in a range from 0 to 1. According to Verma’s studies [5], limtedLinear <0> is a central difference scheme; and limitedLinear <0.2> and limitedLinear <1.0> are both blends of central difference scheme and upwind scheme. This case will assess how two convection schemes: limitedLinear <1.0> and limitedLinear <0.2> affect species mass conservation statement. The limitedLinear <0.2> is used in previous studies by the University of Maryland (UMD) fire modeling group.

Chapter 5: Assessment of *nOuterCorrectors* effects on mass (species) conservation.

This case introduces fire into the simulation. This case is to assess the effects of *nOuterCorrectors* (number of outer loops), a parameter in PIMPLM algorithm, on the accuracy of species mass conservation statement for FireFOAM.

As mentioned above, the *nOuterCorrectors* in PIMPLE algorithm is associated with iteratively solving coupling pressure and momentum governing equations [9][18]. At last, the pressure field and velocities field are only appropriately satisfied. There is a residual error. As the number of iterations increase, the residual error will decrease.

FireFOAM solves coupling governing equations separately and sequentially. These coupling governing equations conclude couplings between total mass conservation, species mass conservation, momentum conservation and energy conservation. Similar as solving pressure and momentum coupling equation, if the coupling between each individual governing is weak, results will have high residual errors. The parameter, *nOuterCorrectors,* is the number of the iterations in this process. It can be used to control the level of coupling between solutions of the individual governing equations and thereby reduce the residual errors. Therefore, as *nOuterCorrectors* increases, the residual error will decrease.

This case is to assess the effects of different *nOuterCorrectors* values on the accuracy of species mass conservation statement for FireFOAM.

Chapter 6: Conclusion and Future Work

Introduction of FireFOAM

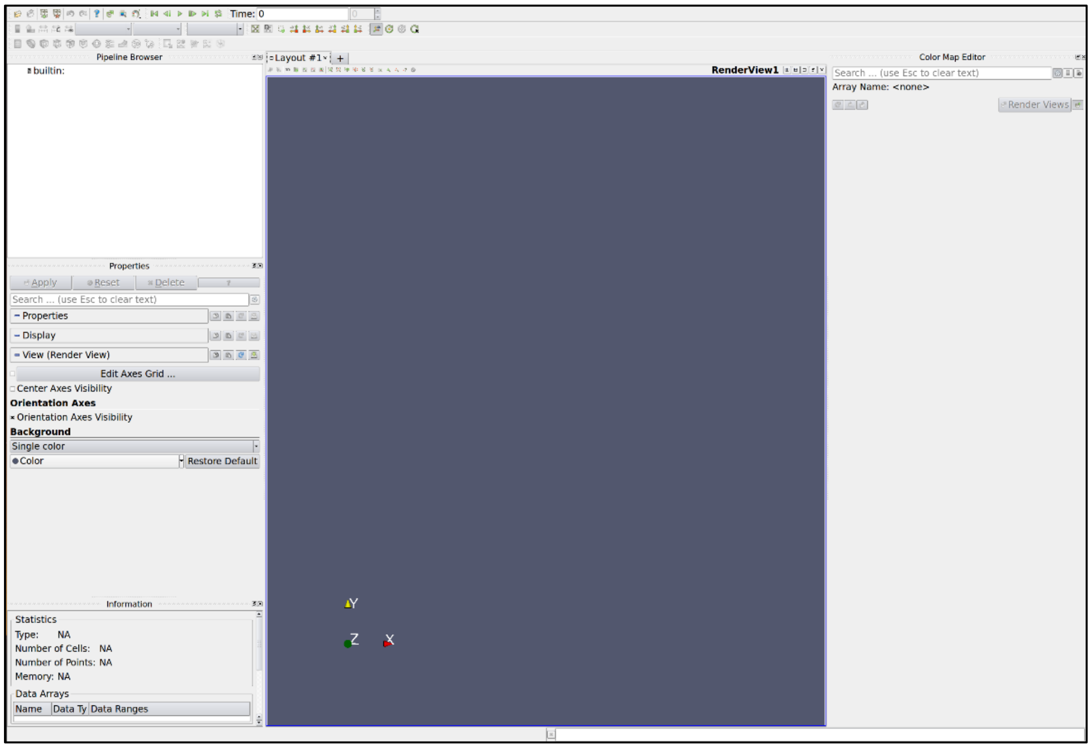
Fire Modeling

Fire is a common hazard as other disasters like hurricanes in real life. It may bring treats to people's lives and their properties. Occasionally, it may negatively impact ecosystems due to some the materials used to extinguish fires. Therefore, it is necessary to understand fire behaviors to prevent or reduce the hazardous level of fires. To better understand fire behaviors, researchers have conducted many experiments over the past few decades. With the development of technology, it allows people to take advantage of computer power to simulate and understand fluids problems through Computational Fluid Dynamics (CFD) software. Generally, CFD software solves Naiver-Strokes governing equations numerically and give predictions on fluid's behaviors. Aiming at simulating fires, FireFOAM, a fire dynamics solver, was developed by FM Global.

FireFOAM Overview (OpenFOAM)

FireFOAM is a Large Eddy Simulation (LES) fire dynamics solver, which performs relying on OpenFOAM libraries. Written by objected-oriented programming language C++, OpenFOAM is a license-free and open sources CFD software whose applications can be categorized into two main types: solvers and utilities [10][11]. There are multiple solvers aiming at solving different types of fluids problems available in the OpenFOAM libraries [11]. FireFOAM is an example of these solvers specifically aimed to solving fires problems. In addition, an example of utilities can be *blockMesh* utility, which is used for generating mesh.

As the OpenFOAM Userguide stated [11], people usually classify simulation process into three steps, including preprocessing step, simulating, and postprocessing. Preprocessing is generally the setting-up step before starting the simulation. Setting-up can include defining initial and boundary conditions etc. Simulation can be run on local computers or on multiple processors of servers, which depend on the computational cost of the simulation. FireFOAM allows parallel simulations as well. Postprocessing is the step when users visualize simulation results and analyze data. Postprocessing can be performed either after simulation or during simulation. FireFOAM users can use ParaView as a visualization tool. ParaView can be used to check mesh and structure of computational domains; Furthermore, ParaView also enables users to visualize and observe species fields and temperature fields etc. Detailed information about ParaView can be found in [12].



Statistics
Type: NA
Number of Cells: NA
Number of Points: NA
Memory: NA
Data Arrays

Name	Data Ty	Data Ranges

Fig.1.3.2.1 ParaView User Interface

FireFOAM can generate data in text file format. Users may use third party tools to plot their data. MATLAB, Python and Excel are common tools used to plot data.

A typical case for FireFOAM (OpenFOAM) generally contains three directories, time directories, constant directories and system directories [13].As OpenFOAM Userguide Chapter 4 introduced [13], *time* directory stores different fields at each time step, and the 0 directory is especially for defining the initial conditions and boundary condition at time equal zero; *constant* directory contains files in which users can specify species properties, turbulence properties and reactions etc.; *system* directory is where users can specify parameters involved with solutions of how solver solving governing equations.

OpenFOAM is mostly based on Linux operating system. The directories structure would have the characters of directories under Linux operating system. OpenFOAM directories have a tree structure character. The example of showing location of constant directory can be: /FireFOAM_example_case/constant. In Linux operating systems, many operations can be done by typing commands in terminal. For example, if users define a computational domain and mesh and wish to generate mesh, users may type *blockMesh* command in the terminal under FireFOAM_example_case directory (/FireFOAM_example_case).

Chapter 2: Development of the Diagnostic

Species Mass Conservation Diagnostic Concept:

This diagnostic is designed to access species mass conservation of solver FireFOAM in global view. The global view species mass conservation statement can be explained as following. For an arbitrary control volume (C.V), the change of mass in C.V within a time interval equals the amount of net mass flow across the C.V boundary (by convection and diffusion) plus source term (species mass generated by chemical reactions) in C.V. Following equation can mathematically explain the concept.

$$\frac{d}{dt}\left(\iiint_{CV} \rho Y_k \, dV\right) = - \oiint_{CS} \rho Y_k (\vec{u} - \vec{v}) \cdot \vec{n} \, dS + \oiint_{CS} \rho D (\nabla Y_k \cdot \vec{n}) dS \; + \iiint_{CV} s_k dV \qquad \text{eqn. 2.1.1}$$

Symbol	Meaning
ρ	density
Y_k	Mass fraction of specie k
$\overrightarrow{u_k}$	Velocity of specie k
$\vec{v}$	Bulk flow velocity
$\vec{n}$	Surface direction vector
D	Diffusion coefficient
s_k	Source term
CV	Control Volume
CS	Control Surface

Table 2.1.1 Symbols in Global Species Mass Conservation Equation

As the global species mass conservation equation showing, the left hand side includes unsteady term, and the right hand side of the equation includes convection term, diffusion term, and source term. For buoyant helium flow, there are no chemical reactions occurred in the control volume (no source term). Therefore, the equation above can be simplified as follow:

$$\frac{d}{dt}\left(\iiint_{CV} \rho Y_k \, dV\right) = -\oiint_{CS} \rho Y_k (\vec{u} - \vec{v}) \cdot \vec{n} \, dS + \oiint_{CS} \rho D (\nabla Y_k \cdot \vec{n}) dS \qquad \text{eqn. 2.1.2}$$

After simplification, the left terms in right hand side are convection term and diffusion term.

The species mass conservation assessment diagnostic method is developed based on the simplified equation. It will collect data for left hand side of the equation (LHS) and right hand side of the equation (RHS) respectively.

Define relative error in the following equation:

$$LHS - RHS = residual \qquad \text{eqn. 2.1.3}$$

If mass conserved, LHS - RHS == 0. Considering about numerical errors occurring in simulation, the residual may not be exactly equal to 0. The residual will be normalized by inlet mass quantity to get the relative error.

$$\frac{LHS - RHS}{\dot{m}_{inlet}} = relative\ error \qquad \text{eqn. 2.1.4}$$

Data Collection:

Left Hand Side Data

As the above equation indicates, left hand side of equation includes the rate of mass changes (unsteady term) in the fixed control volume. The diagnostic is not to build a function to calculate rate of mass changes in control volume directly. FireFOAM can generated mass fraction fields, velocity fields and density fields at each time step. The diagnostic takes an advantage of the density fields and mass fraction fields and calculate the mass in a selected control volume at each time step. This an example of postprocessing during simulation. FireFOAM allows users to specify time schemes in case files. Depending on the time scheme applied in simulation, users can calculate unsteady term.

The way to tell FireFOAM to do calculation as desired is adding codes in specific files. The FireFOAM version used in this assessment is *fireFoam-dev*. Under fireFoam-dev directory, users can find a sub-directory called *solver.*

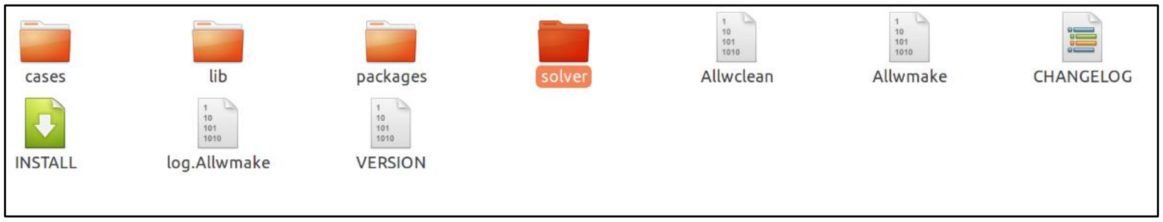


Fig 2.2.1.1. *fireFoam-dev* content

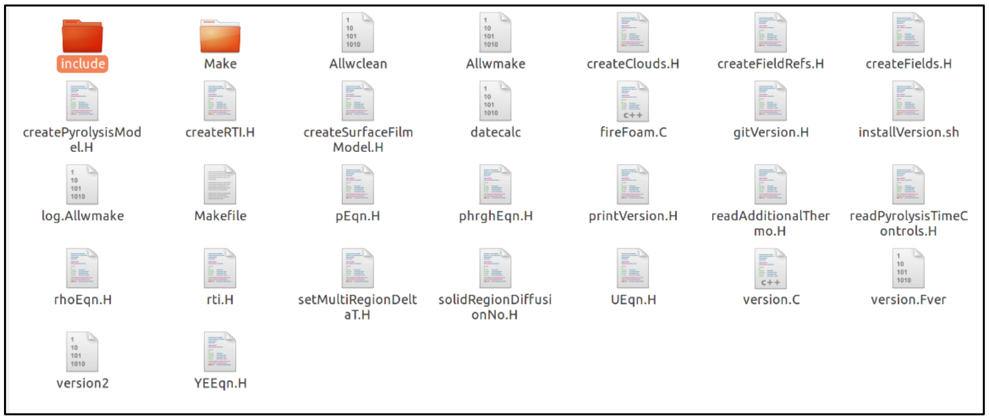
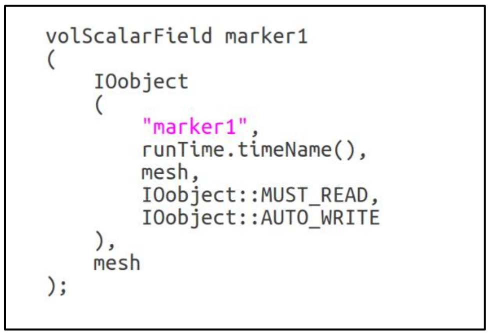


Fig 2.2.1.2. *solver* directory content

Firstly, a piece of code is added in *infoFieldsOutput.H* file. Users can find this file in the *include* directory under solver directory. The purpose is to create a *volScalarField* called *marker1*. Users can customize the name, but the name should be consistent with the name in later files.

Fig.2.2.1.3 include directory content



```
volScalarField marker1
(
    IOobject
    (
        "marker1",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);
```

Fig. 2.2.1.4 Specification of volScalarField: *marker1 in infoFieldsOutput.H* file

In the *solver* directory, users can find the *fireFoam.C* file, where the second piece of code will be added. The purpose is to use mass fraction field and density field generated by FireFOAM to calculate the mass in selected control volume at each time step. The following figures will show where and how to add the codes.

Step 1: Find following code in *fireFoam.C* file.

```
runTime.write();

Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
    << "  ClockTime = " << runTime.elapsedClockTime() << " s"; // kvm
```

Fig. 2.2.1.5 *fireFoam.C* file before modification

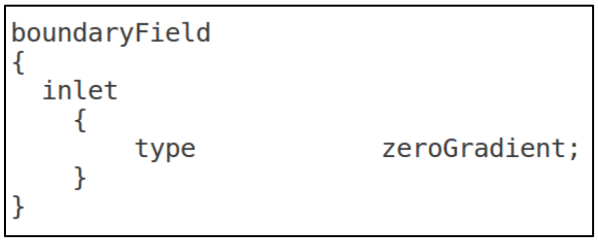
Step 2: Insert following piece of code after "runtime.write();" , but before "Infor<< ….".

```
runTime.write();

os << runTime.timeName() << " , " << fvc::domainIntegrate(marker1*rho*fu).value()
                         << " , " << fvc::domainIntegrate(marker2*rho*fu).value()
                         << " , " << fvc::domainIntegrate(marker3*rho*fu).value()
                         << " , " << fvc::domainIntegrate(marker4*rho*fu).value()
                         << " , " << fvc::domainIntegrate(marker5*rho*fu).value()
                         << " , " << fvc::domainIntegrate(marker6*rho*fu).value()
                         << " , " << fvc::domainIntegrate(marker7*rho*fu).value()
                         << " , " << fvc::domainIntegrate(marker8*rho*fu).value()
                         << " , " << fvc::domainIntegrate(marker9*rho*fu).value()
                         << " , " << fvc::domainIntegrate(marker10*rho*fu).value()
                         << endl;

Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
    << "  ClockTime = " << runTime.elapsedClockTime() << " s";
```

Fig. 2.2.1.6 *fireFoam.C* file after modification

Users need to compile codes after any modifications. The compiling process can check bugs in codes, and make sure added codes work for later simulation. To compile code, users can type *wmake* in terminal under the */fireFoam-dev/solver* directory.

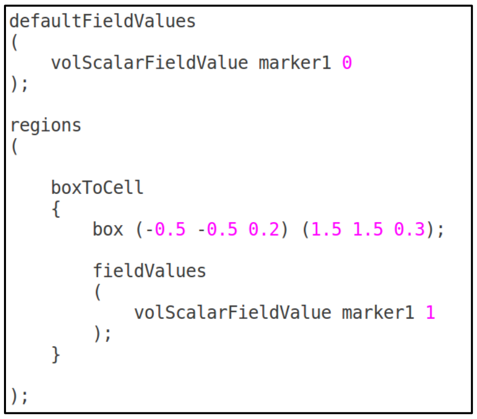
Except for modifications in /fireFoam-dev/solver directory, users also need to define two files in case files. These two files are *clean.marker* file in *0* directory, and *setFieldsDict* in system directory respectively. In this file, users need to define boundary condition for each boundary. An example of definition shows as below.



```
boundaryField
{
  inlet
    {
        type             zeroGradient;
    }
}
```

Fig. 2.2.1.7 *clean.maker* file example

The *setFieldsDict* file is the place where users can define the location (coordinate) of the control volume. The coordinate of the eight points for control volume can be specified in parentheses of regions. The following figure is an example of definition in regions in the *setFieldsDict* file. In x-axis direction, the selected region starts from x = - 0.5 to y = 1.5; In y-axis direction, the selected region starts from y = - 0.5 to 1.5; and in z-axis direction, the selected region starts from z = 0.2 to 0.3. The selected region means that all meshes in this region will be counted into data collection. In *defaultFieldValues* parentheses, 0 is Boolean type value in C++ language. It means False here, while in *fieldValues* parentheses, 1 means True. Besides, users should be careful about the usage of braces "{ }" and parentheses "( )" in the file. Also, users also need to pay attention to the semicolons after some parentheses. Missing of any one of these will cause a fatal error when running a simulation. (Fatal error means simulation will stop when FireFOAM finds the incorrect definition).



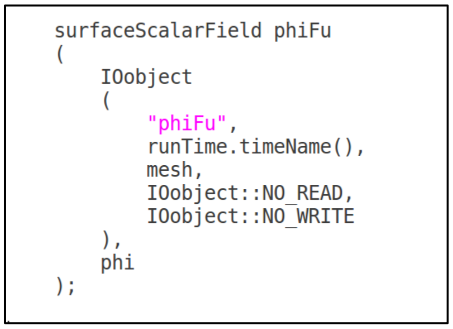
```
defaultFieldValues
(
    volScalarFieldValue marker1 0
);

regions
(

    boxToCell
    {
        box (-0.5 -0.5 0.2) (1.5 1.5 0.3);

        fieldValues
        (
            volScalarFieldValue marker1 1
        );
    }

);
```

Fig. 2.2.1.8 *regions* definition example in *setFieldsDict*

Right Hand Side Data

In the simplified species mass conservation eqn. 2.1.2, the right-hand side (RHS) of the equation is net mass flow across the control volume boundary. The diagnostic uses the density fields, mass fraction fields, and velocity fields and calculate mass flow rate across control volume surfaces at each time step. Then, users can calculate the sum of the mass flow rate for all boundary surface to get the right hand side of the equation. Similarly, we need to tell FireFOAM to calculate mass flow rate across surfaces.

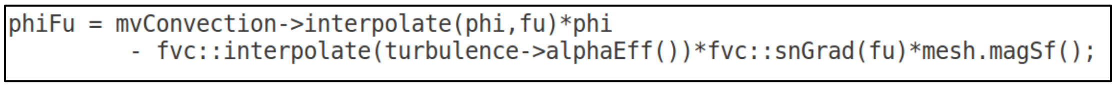
Firstly, define a *surfaceScalarField* called *phiFu* in *infoFieldsOutput.H* file. This file locates under *include* directory.



```
surfaceScalarField phiFu
(
    IOobject
    (
        "phiFu",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::NO_WRITE
    ),
    phi
);
```

Fig. 2.2.2.1 Specification of surfaceScalarField: phiFu in *infoFieldsOutput.H* file

Afterwards, we need to translate the mathematic expression of RHS of eqn. 2.1.2 to C++ language. Users can add a piece of code as following figure shows in *infoOutput.H* file. Users can find this file under *include* directory.

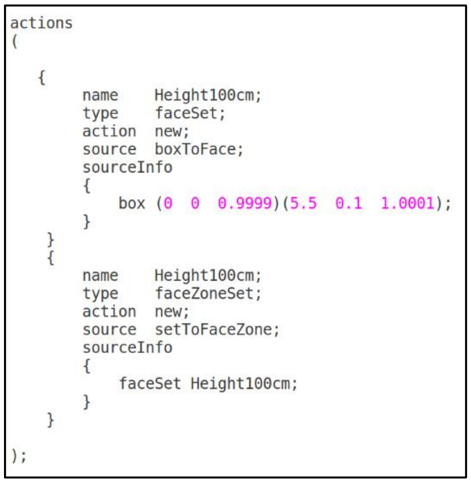


```
phiFu = mvConvection->interpolate(phi,fu)*phi
        - fvc::interpolate(turbulence->alphaEff())*fvc::snGrad(fu)*mesh.magSf();
```

Fig. 2.2.2.2 mass flow rate calculation across a selected surface example in *infoOutput.H* file

After modification, users should compile codes by typing *wmake* in terminal under the directory of */fireFoam-dev/solver*.

Above operations will ask FireFOAM to calculate the mass flow rate across a selected surface. However, the surfaces have not been chosen yet. Users can define surfaces in the *topoSetDict* file, which is under the system directory. An example of the surface definition is as below.



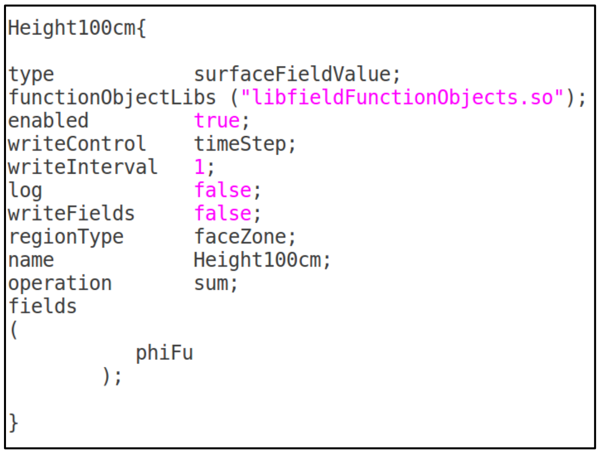
```
actions
(

   {
        name    Height100cm;
        type    faceSet;
        action  new;
        source  boxToFace;
        sourceInfo
        {
            box (0  0  0.9999)(5.5  0.1  1.0001);
        }
    }
    {
        name    Height100cm;
        type    faceZoneSet;
        action  new;
        source  setToFaceZone;
        sourceInfo
        {
            faceSet Height100cm;
        }
    }

);
```

Fig. 2.2.2.3 Definition of surface in *TopoSetDict* file

In *topoSetDict* file, users can customize the name of new defined surfaces in the *actions* parentheses. There are two braces under the *action* parentheses. In the first braces, users should be careful about the definition of *type* and *source.* They should belong to *faceSet* and *boxToFace* categories respectively. In the *sourceInfo* sub-braces, it is where users can locate the new defined surface. The line of " *box (0   0   0.9999) (5.5   0.1   1.0001)* " stands for selected region is from x = 0 to x = 5.5 in x-axis; and from y = 0 to y = 0.1 and from z = 0.9999 to z = 1.0001 in z-axis. It is easy to see that two z values are close. Here, we are using a method that define a new surface which is under a selected box. The defined surface is the sum of meshes' boundary at same height. To avoid counting more than one surface into calculation, users can choose two z values that are really close as example shows. If using two same z value, FireFOAM would not understand, and simulation will meet fatal error.

In the second braces under actions parentheses, the name of new defined surfaces should be consistent with previous name defined in first braces. Users also need to pay attention to the categories of *type* and *source*, which are *faceZoneSet* and *setToFaceZone* respectively.

After defining surfaces in *topoSetDict*, users need to take another step in *controlDict* file in case files. It is to tell FireFOAM to output calculation results of *phiFu* field at each time step. The *controlDict* file locates in *system* directory. The example of specification is as below.



```
Height100cm{

type            surfaceFieldValue;
functionObjectLibs ("libfieldFunctionObjects.so");
enabled         true;
writeControl    timeStep;
writeInterval   1;
log             false;
writeFields     false;
regionType      faceZone;
name            Height100cm;
operation       sum;
fields
(
           phiFu
        );

}
```

Fig. 2.2.2.4. Specification of *phiFu* Field output for surface *Height100cm*

In controlDict file, users should the same surface name. Under the surface name (Height100cm) braces, there are some definitions user might need to pay attention to. The sort of *writeControl* is *timeStep*, which means that FireFOAM will output data every time step. The time step is also specified in controlDict file. It can be either constant time step or variable time step. Variable time step is controlled by Courant Friedriches Lewy number.

Conclusion:

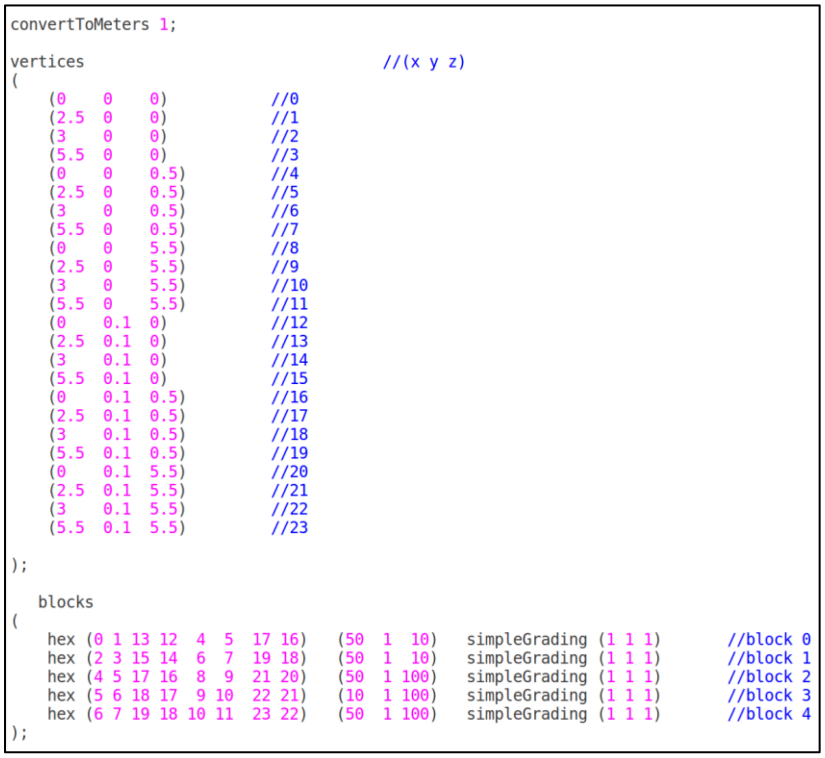
This chapter introduces the process of modifying FireFOAM solver code to develop a diagnostic method for mass (species) conservation assessment. The concept of this diagnostic is to ask FireFOAM to collect data for left hand side and right hand side for eqn. 2.1.2 separately. Besides, these data are not directly applied in species mass conservation assessment. Users still need to postprocess these data.

The development of species mass conservation diagnostic indicates that users can get an access to FireFOAM solver codes. This is one of the advantages of fire dynamics solver FireFOAM. FireFOAM does not set limitations in running simulation within unchangeable codes. Users can customize solver to meet their numerical prediction needs.

Chapter 3: Helium Plume 2-Dimensional Case

Case Discription and Setup:

Species mass conservation assessment starts from a buoyant helium plume in a two-dimensional computational domain. The computational domain is 5.5 meters in length, 5.5 meters in height, and 0.1 meter in depth, with a 0.5m×0.5m×0.1m burner. Helium flow will be injected into domain through the burner. Width of 5.5m and height of 5.5m can make sure helium has enough space to develop as a plume. In OpenFOAM, if the width of domain is equal to width of a single mesh, the simulation is considered as two-dimensional case. In this case, *blockMeshDict* file is used to construct the computational domain and define boundary surfaces.

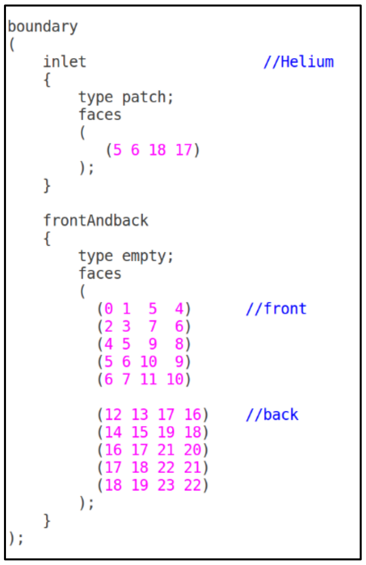


```
convertToMeters 1;

vertices                                    //(x y z)
(
    (0    0    0)           //0
    (2.5  0    0)           //1
    (3    0    0)           //2
    (5.5  0    0)           //3
    (0    0    0.5)         //4
    (2.5  0    0.5)         //5
    (3    0    0.5)         //6
    (5.5  0    0.5)         //7
    (0    0    5.5)         //8
    (2.5  0    5.5)         //9
    (3    0    5.5)         //10
    (5.5  0    5.5)         //11
    (0    0.1  0)           //12
    (2.5  0.1  0)           //13
    (3    0.1  0)           //14
    (5.5  0.1  0)           //15
    (0    0.1  0.5)         //16
    (2.5  0.1  0.5)         //17
    (3    0.1  0.5)         //18
    (5.5  0.1  0.5)         //19
    (0    0.1  5.5)         //20
    (2.5  0.1  5.5)         //21
    (3    0.1  5.5)         //22
    (5.5  0.1  5.5)         //23

);

   blocks
(
    hex (0 1 13 12  4  5  17 16)   (50  1  10)   simpleGrading (1 1 1)       //block 0
    hex (2 3 15 14  6  7  19 18)   (50  1  10)   simpleGrading (1 1 1)       //block 1
    hex (4 5 17 16  8  9  21 20)   (50  1 100)   simpleGrading (1 1 1)       //block 2
    hex (5 6 18 17  9 10  22 21)   (10  1 100)   simpleGrading (1 1 1)       //block 3
    hex (6 7 19 18 10 11  23 22)   (50  1 100)   simpleGrading (1 1 1)       //block 4
);
```

Fig. 3.1.1 *blockMeshDict* file example

In the *vertices* parentheses, users can define coordinates for each points. After determining coordinates for points, users can define *blocks*. The *blocks* are the components to assemble the simulation domain. Example shows there are 5 blocks composing the computational domain. The parenthesis in front of simpleGrading is where users can define number of mesh in x, y and z axis of each block. It is easy to observe that there is only 1 mesh in y axis of all 5 blocks. This is consistent with previous description of 2-dimensional domain setup method.

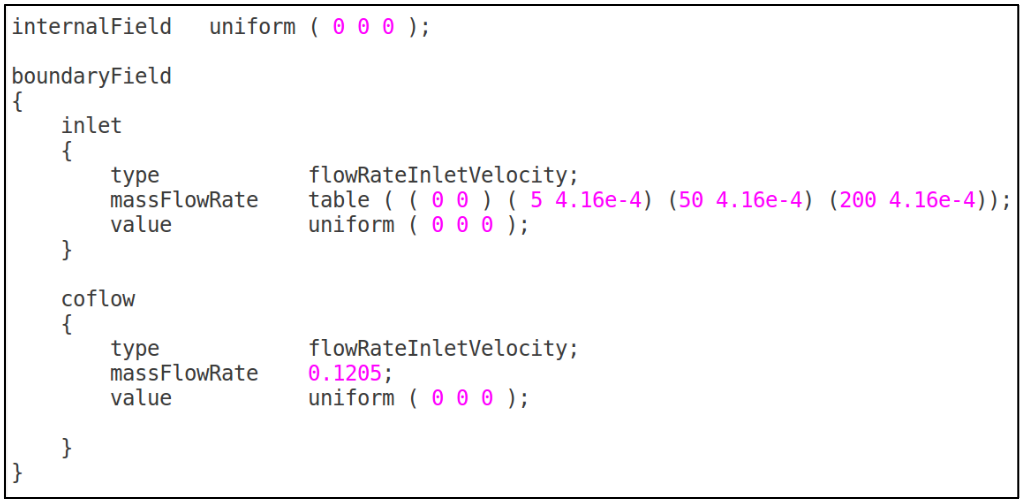
Except for editing *vertices* and *blocks* in blockMeshDict, users can also define boundary surface in this file. An example of definition of boundary condition in *blockMeshDict* file is as follow. Users should pay attention definition of *frontAndback* boundary surface. In this case, the width of domain equals to width of a mesh in y direction. Therefore, x direction and z direction are considered. However, there are still two surfaces to define, front surface and back surface. As the example indicates, user can use type of *empty* to define these surfaces. If users want to detail boundary condition in other files, users may classify the boundary type into *patch* type. *Patch* is a normal boundary type in OpenFOAM.



```
boundary
(
    inlet                   //Helium
    {
        type patch;
        faces
        (
           (5 6 18 17)
        );
    }

    frontAndback
    {
        type empty;
        faces
        (
          (0 1  5  4)      //front
          (2 3  7  6)
          (4 5  9  8)
          (5 6 10  9)
          (6 7 11 10)

          (12 13 17 16)    //back
          (14 15 19 18)
          (16 17 21 20)
          (17 18 22 21)
          (18 19 23 22)
        );
    }
);
```

Fig. 3.1.2 Boundary surface definition example in *blockMeshDict* file

In time *0* directory, users can define initial conditions of simulation. In this case, helium is injecting through the burner with a velocity of 0.05m/s, and surrounded by air coflow with a velocity of 0.4m/s. The aim of higher velocity of air coflow is to prevent helium flowing out of domain through left and right walls. Therefore, vertical direction is the only direction to be considered. In other words, net mass flow across boundary for this case only considers surfaces vertical to z direction, which simplifies the assessment process. The following is an example of *U* file in *0* directory.



```
internalField   uniform ( 0 0 0 );

boundaryField
{
    inlet
    {
        type            flowRateInletVelocity;
        massFlowRate    table ( ( 0 0 ) ( 5 4.16e-4) (50 4.16e-4) (200 4.16e-4));
        value           uniform ( 0 0 0 );
    }

    coflow
    {
        type            flowRateInletVelocity;
        massFlowRate    0.1205;
        value           uniform ( 0 0 0 );

    }
}
```

Fig.3.1.3 Velocity file *U* example

User may define helium velocity properties in the inlet braces. The *flowRateInletVeloctiy* is a method to define inject flow in OpenFOAM. The velocity of helium is 0.05m/s, which is 4.16×10^{-4} kg/s by using velocity of helium to multiply area of burner(0.5m×0.1m) and density of helium(0.1664kg/m^3). In the *massFlowRate* line, users may define different injecting mass flow rate of helium flow at different time interval. As above figure shows, from 0s to 5s, helium injecting mass flow rate equals 4.16×10^{-4} kg/s; from 5s to 50s, it equals to 4.16×10^{-4} kg/s; from 50s to 200s, it equals to 4.16×10^{-4} kg/s; And after 200s, it all equals 4.16×10^{-4} kg/s. The aim is to delay injection of helium flow to make sure air coflow arrives at the height of burner before helium injection. Users may specify different mass flow rate at different time interval to meet their simulation needs.

As above discussion said, the assessment only considers z axis direction, which will simplify the assessment process. For this reason, the species mass conservation equation can be simplified as:

$$\frac{d}{dt}\left(\int_{z_1}^{z_2}\rho Y_{He} dV\right) = \dot{m_{He}}(z_1) - \dot{m_{He}}(z_2) \qquad \text{eqn. 3.1.1}$$

The left hand side of the equation is the unsteady term of species mass conservation equation. The area of cross section ($S_{cross\_section}$), which is perpendicular to z-axis, is a constant at different height (above burner) in the domain. The z becomes only variable in the unsteady term. Consequently, the volume integral can be simplified to line integral. This equation uses a simpler to represent mass flow rate across boundary surface. In the above equation, $\dot{m_{He}}(z_1)$ means that mass flow into control volume through cross section surface at z = z_1; and $\dot{m_{He}}(z_2)$ means that mass flow out control volume through cross section surface at z = z_2. This case selects the height of helium inlet as z_1, and z_2 varies within computational domain's height. Then the equation is reformatted as:

$$\frac{d}{dt}\left(\int_{inlet}^{z_2}\rho Y_{He}\, S_{cross\_section} dz\right) = \dot{m_{He}}(inlet) - \dot{m_{He}}(z_2) \qquad \text{eqn. 3.1.2}$$

As discussed before, users can use setFieldsDict to locate control volume for left hand side data collection, and use topoSetDict to define cross section at different height for right hand side data collection.

After preparation of case file, user can type *blockMesh* command in terminal under case file directory to generate mesh. People sometimes may want to store mesh generation process for future review. Then, users may type *blockMesh | tee log.blockMesh* to generate a log file for the process. This method applied to other commands as well. Next step is to visualize computational domain and check meshes.

Paraview will help in the step. To open paraview, users can simply type *paraFoam&* in terminal under same case file directory. If there is a very minor error in mesh generation process, it will be very hard to find. A command called *checkMesh* will make its contributions in this situation.

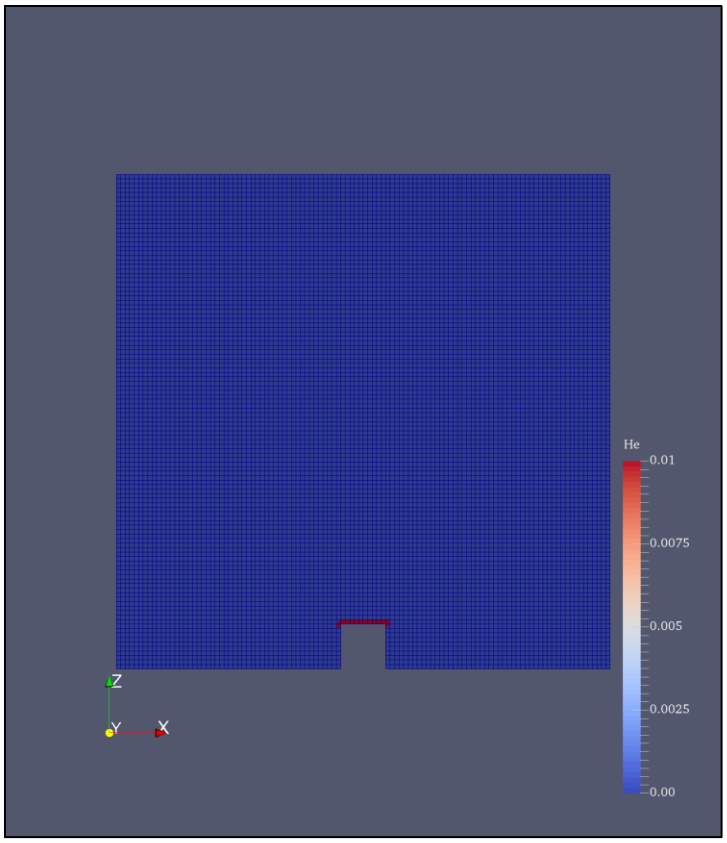


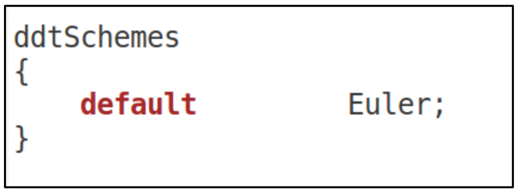
Fig. 3.1.4 Computational domain and meshes

To apply new defined surfaces in *topoSetDict* in simulation, users need to type *topoSet* in terminal under case directory; To apply defined control volume in *setFieldsDict* in simulation, users need to type *cp -rf 0/clean.marker 0/marker1* and *setFields* orderly in terminal.

After finishing these preprocessing steps, users can type *fireFoam* in terminal under case directory to run simulation. If users want to run simulation parallelly, users can specify number of subdomains and method of decomposing in *decomposeParDict*, then use *decomposePar* command to generate subdomains.

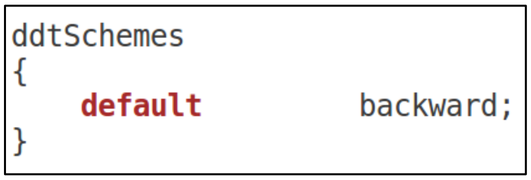
<u>*Time Scheme:*</u>

This aim of the 2-dimensional helium plume case is to check how different time schemes affect numerical error. Two types of temporal schemes are tested in this case, first order accuracy scheme: *Euler*, and second order accuracy scheme: *backward*. To change time scheme, users can go to *fvScheme* file under system directory in case files.



```
ddtSchemes
{
    default         Euler;
}
```

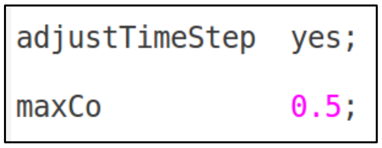
Fig. 3.2.1 Temporal scheme: *Euler* in *fvScheme* file



```
ddtSchemes
{
    default         backward;
}
```

Fig. 3.2.2 Temporal scheme: *backward* in *fvScheme* file

FireFoam supports running simulation with either constant time step or variable time step. Users can define a constant time step. Variable time step is controlled by Courant Friedriches Lewy number. The empirical value normally used in FireFoam application is 0.5. Users can go to controlDict to turn on CFL controlled variable time step.



```
adjustTimeStep  yes;

maxCo           0.5;
```

Fig. 3.2.3 Turn on CFL number control

As the diagnostic will help to collect data for mass in control volume at each time step, the data need to be postprocessed based on the time scheme applied. Euler is a first order accuracy scheme, which indicates that unsteady term can be calculated numerically as:

$$\frac{\partial m}{\partial t} = \frac{m_t - m_{t-\Delta t}}{\Delta t}$$

eqn. 3.2.3

Users can find detailed explanation of Euler scheme equation in OpenCFD: UserGuide time schemes [14]. The equation here is written in terms of *m* for consistent with previous content purpose. This applied to both constant time step and variable time step. As the diagnostic will collect Δt, and m_t at every time step, then users can calculate the unsteady term based on the above equation. Users can write a Matlab script to postprocess data.

For second order accuracy scheme, there are two different equations applied to constant time step and variable time step respectively in OpenFOAM.

Constant time step (Δt = constant):

$$\frac{\partial m}{\partial t} = \frac{\frac{3}{2}m_t - 2m_{t-\Delta t} + \frac{1}{2}m_{t-2\Delta t}}{\Delta t}$$

eqn. 3.2.4

Variable Time Step (CFL number controlled):

$$\frac{\partial m}{\partial t} = \frac{1}{\Delta t}\left\{\left(1+\frac{\Delta t}{\Delta t+\Delta t'}\right)m_t - \left(1+\frac{\Delta t}{\Delta t'}\right)m_{t-\Delta t} + \left(\frac{\Delta t^2}{\Delta t'(\Delta t+\Delta t')}\right)m_{t-\Delta t-\Delta t'}\right\}$$

eqn.3.2.5

Users may find backward formula in OpenCFD: Userguide [14] and information for variable time step in Moukalled, Mangani's and Darwish's book [15]. Verma (2019) has organized and corrected the information of constant time step and variable time step for backward scheme in his studies [5].

The purpose of this test is to check how different temporal schemes affect the accuracy of simulation results. Simulation with two different time schemes and two different time step types are performed for the checking process.

Temporal Scheme	Variable time step	Constant time step
Euler	(1) CFL = 0.5 controlled	(2) $\Delta t = 0.01s$
Backward	(3) CFL = 0.5 controlled	(4) $\Delta t = 0.01s$

Table 3.2.1 Time Scheme Test List

(Hints: The convection scheme: limitedLinear <1.0> and nOuterCorrectors = 5 are applied in all tests).

The constant time step applied has same scale of variable time step. And the results and analysis will be presented in next chapter.

Postprocessing

To check the effects of time schemes on the accuracy of species mass conservation statement, we need to calculate instantaneous the relative error. Using the simplified species mass conservation eqn. 3.1.2:

$$\text{LHS} = \frac{d}{dt}\left(\int_{inlet}^{z_2} \rho Y_{He}\, S_{cross\_section} dz\right) = \quad \left(\frac{\partial m_{He}}{\partial t}\right)_{z_2} \qquad \text{eqn.3.3.1}$$

$$RHS = \dot{m_{He}}(inlet) \; - \; \dot{m_{He}}(z_2) \qquad \text{eqn.3.3.2}$$

According to eqn. 2.1.3 and eqn. 2.1.4 in chapter 2:

$$\text{Instantaneous relative error} \; = \frac{LHS - RHS}{\dot{m_{He}}(inlet)} \qquad \text{eqn. 3.3.3}$$

The diagnostic will collect m_{He} in control volume at each time step. In the second section of this chapter, the equations for Euler and backward scheme has been introduced. Depending on the time scheme applied, the LHS in eqn. 3.3.3 can be calculated.

The diagnostic will collect $\dot{m_{He}}(inlet)$ and $\dot{m_{He}}(z_2)$ separately. The RHS in eqn. 3.3.3 can be calculated.

Therefore, the instantaneous relative error can be calculated using eqn. 3.3.3.

Results:

Duration of all four simulations is 500 seconds. As eqn. 3.1.2 states, z_2 is the only variable in the equation. Therefore, selected z_2 are 1m, 1.5m, 2m, 3m, and 4m. Assessment starts from Euler time scheme with variable time step.

Following figure is the helium field at 440 seconds. Users may use ParaView to visualize helium field same as visualizing computational domain and meshes. But users need to select *He* field.

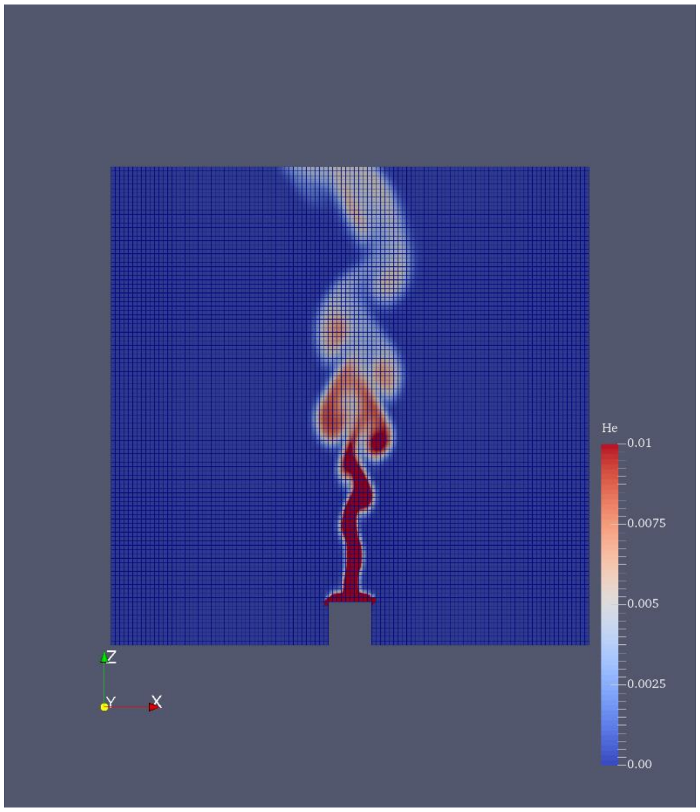


Fig. 3.4.1 Helium plume field at 440s

The results are represented by relative error vs. simulation time. The y-axis represents that relative error. It is the residual between LHS and RHS normalized by value of mass flow rate of Helium at inlet. The x- axis is simulation time in seconds.

(1) Euler, variable time step (CFL = 0.5):

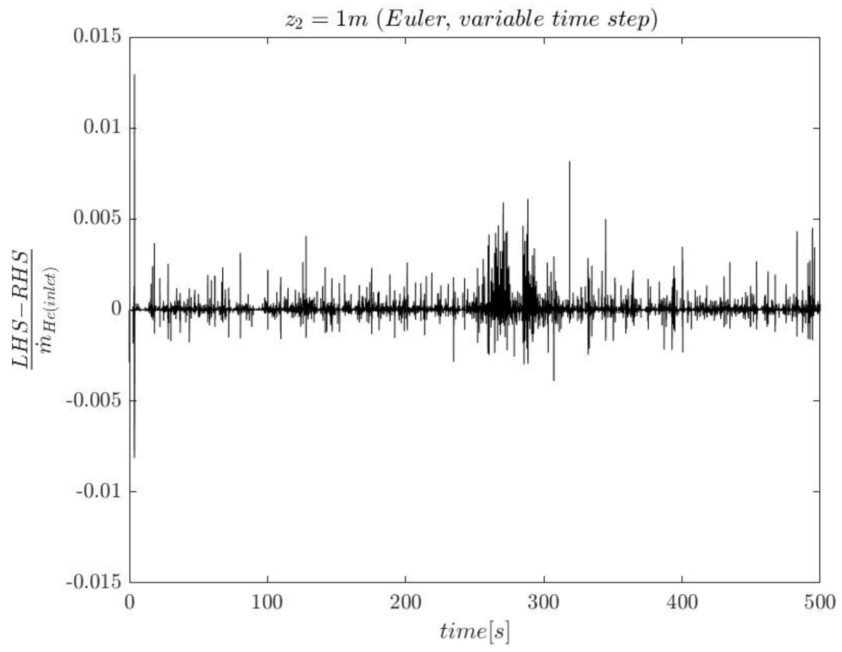


Fig. 3.4.2 Euler Scheme variable time step at z_2 = 1m

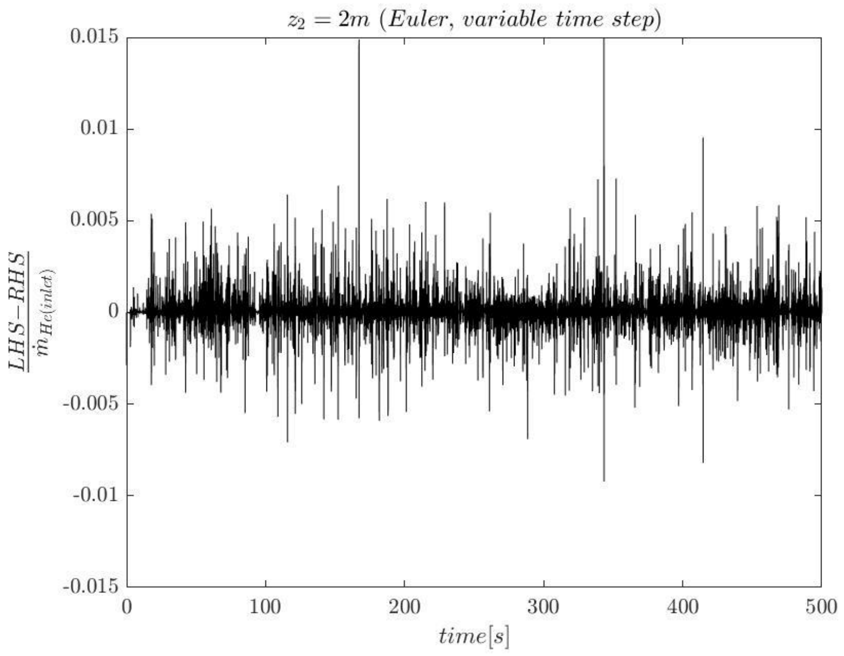


Fig. 3.4.3 Euler Scheme: variable time step at z_2 = 2m

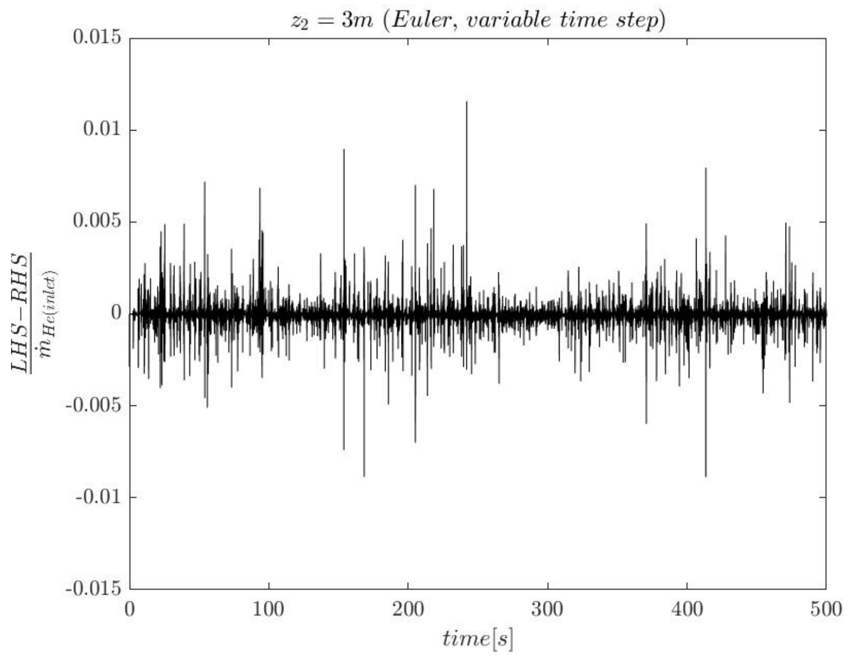


Fig. 3.4.4 Euler Scheme: variable time step at $z_2 = 3m$

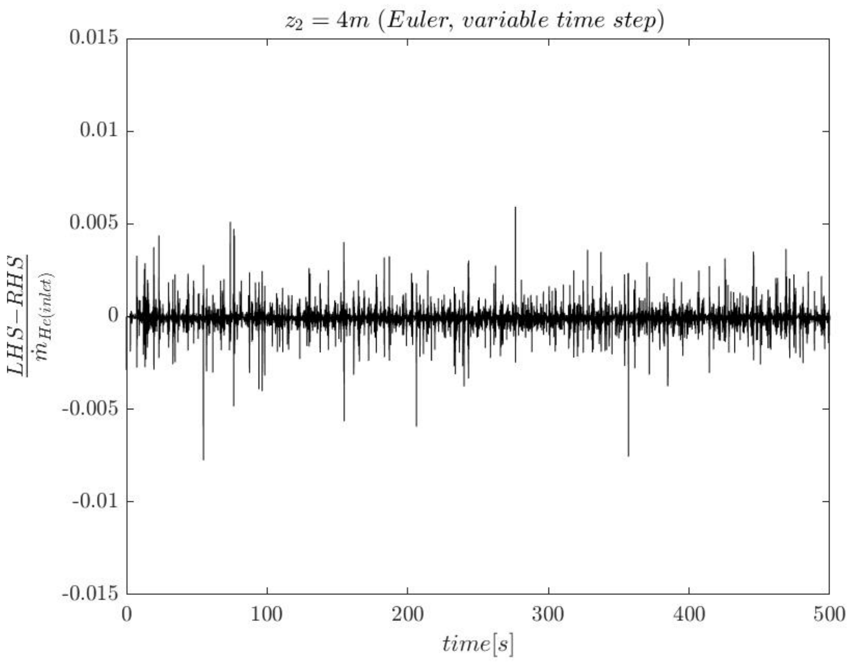


Fig. 3.4.5 Euler Scheme: variable time step at $z_2 = 4m$

As above results show, the scale of instantaneous relative error is most below 1% for every elevation, except for some numerical errors.

Following figures are comparing results for variable time step and constant time step.

(2) Euler: compare variable time step and constant time step

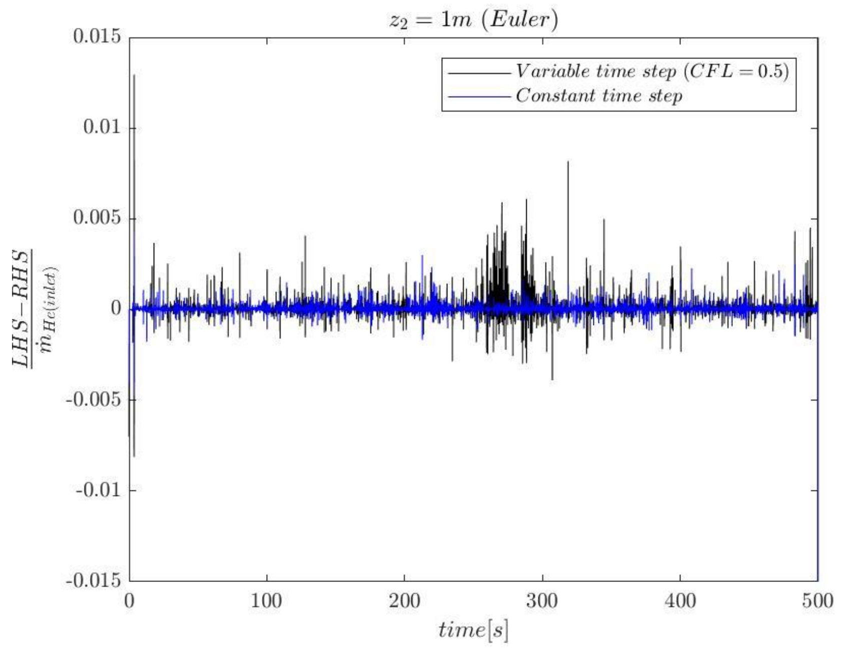


Fig. 3.4.6 Euler Scheme: variable time step vs. constant time step at z_2 = 1m

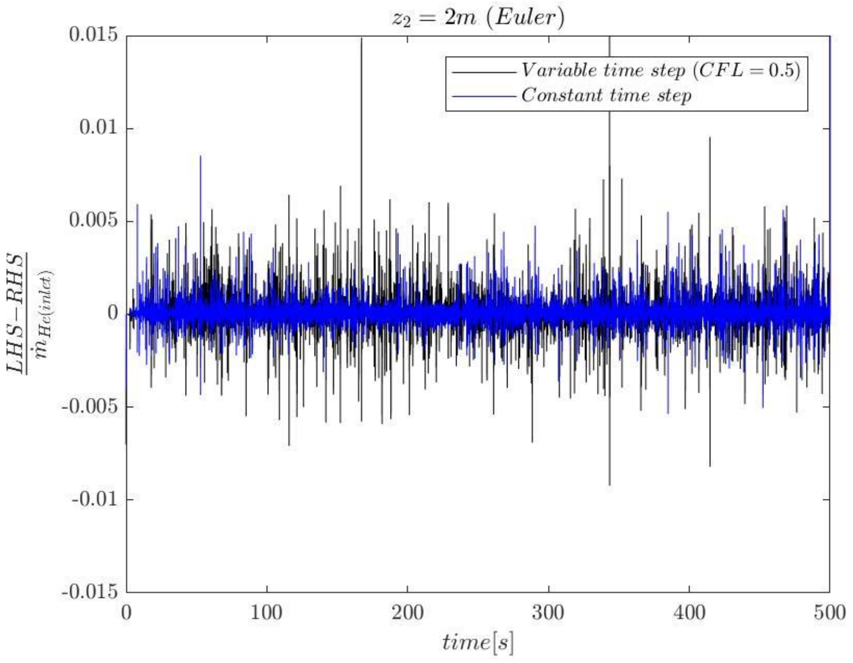


Fig. 3.4.7 Euler Scheme: variable time step vs. constant time step at z_2 = 2m

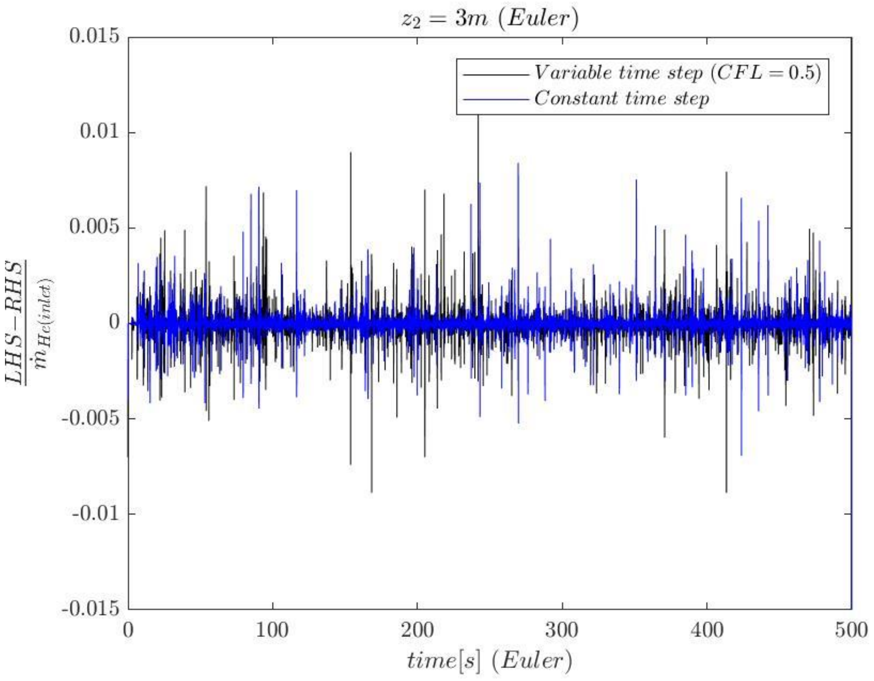


Fig. 3.4.8 Euler Scheme: variable time step vs. constant time step at z_2 = 3m

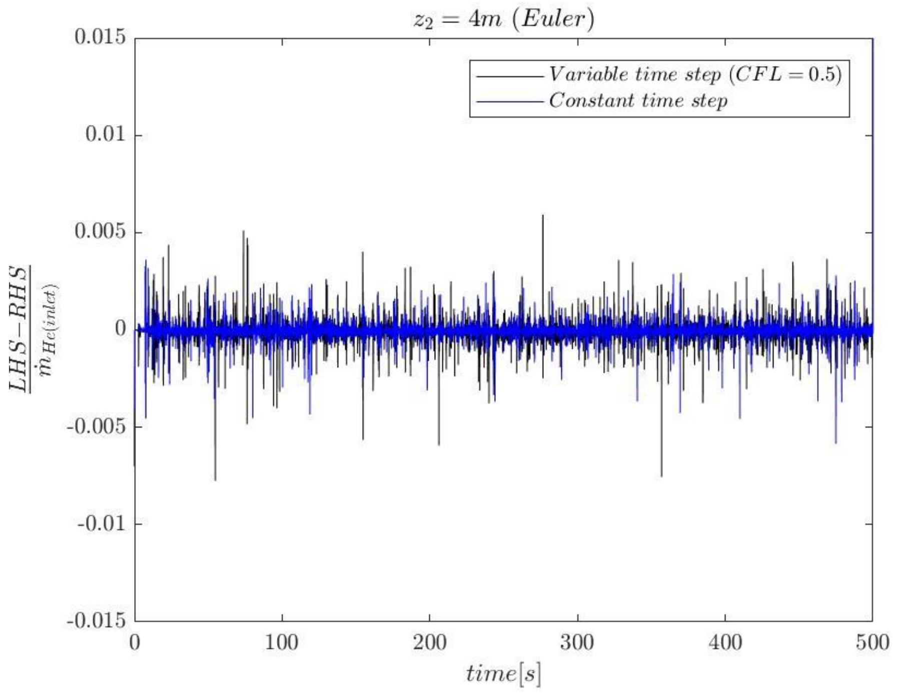


Fig. 3.4.9 Euler Scheme: variable time step vs. constant time step at z_2 = 4m

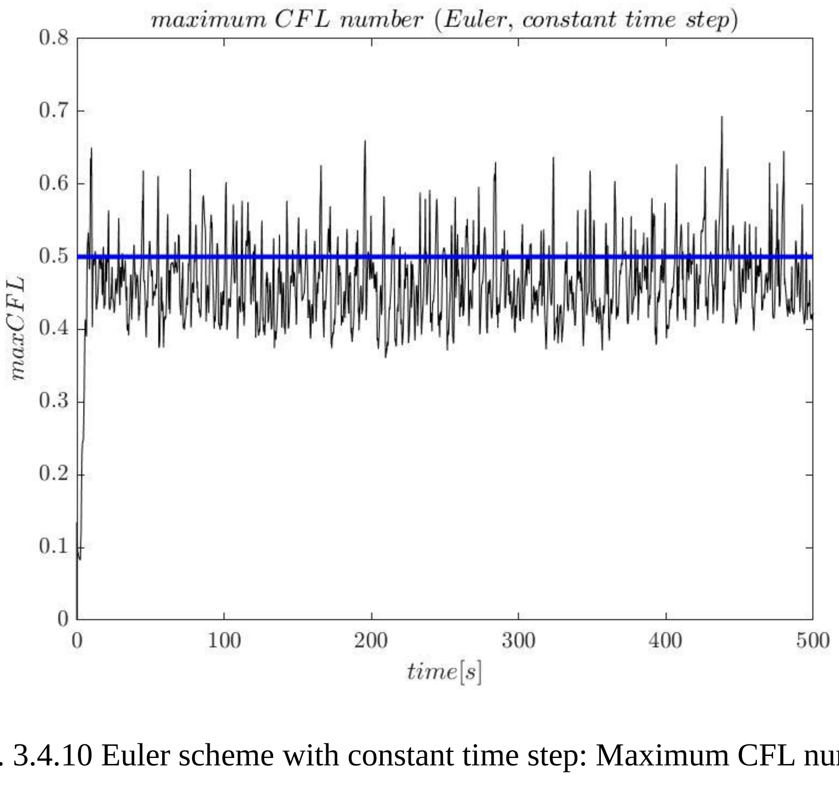


Fig. 3.4.10 Euler scheme with constant time step: Maximum CFL number

(3) backward, variable time step (CFL = 0.5):

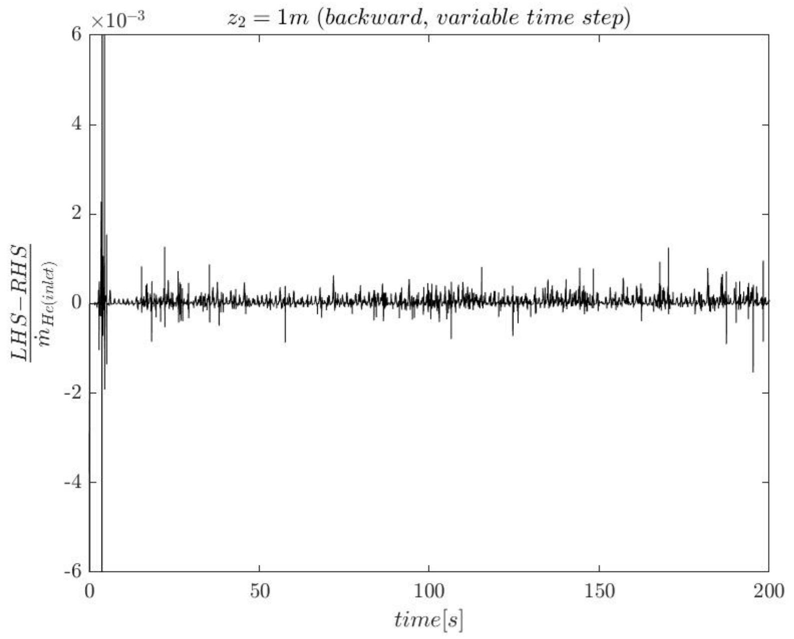


Fig. 3.4.11 backward Scheme: variable time step at z_2 = 1m

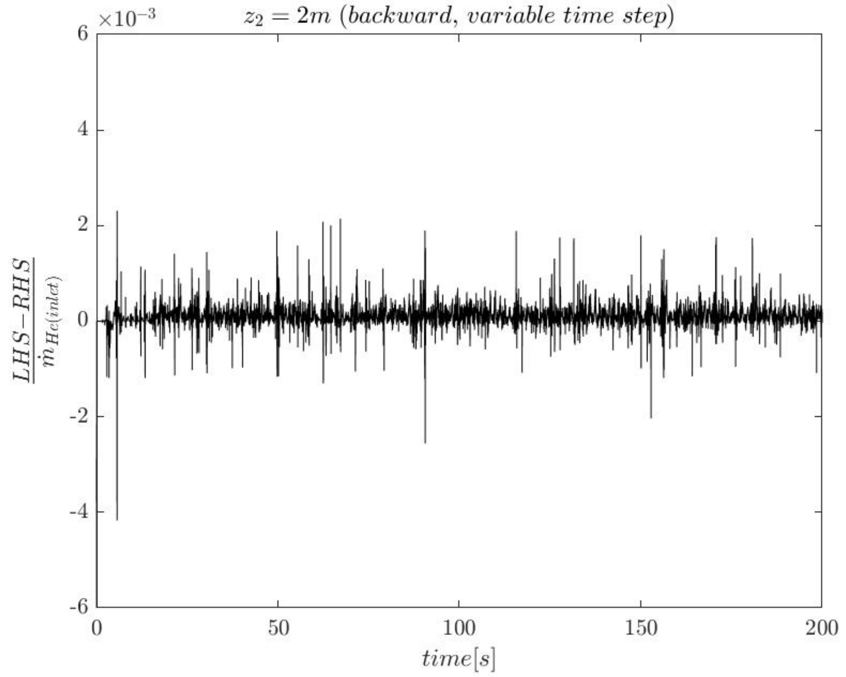


Fig. 3.4.12 backward Scheme: variable time step at z_2 = 2m

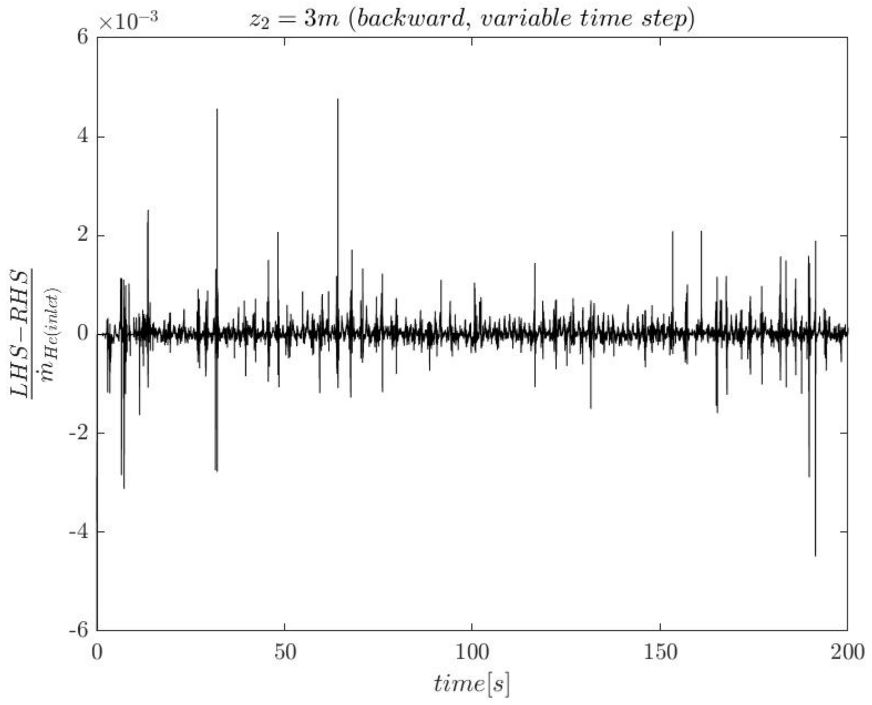


Fig. 3.4.13 backward Scheme: variable time step at z_2 = 3m

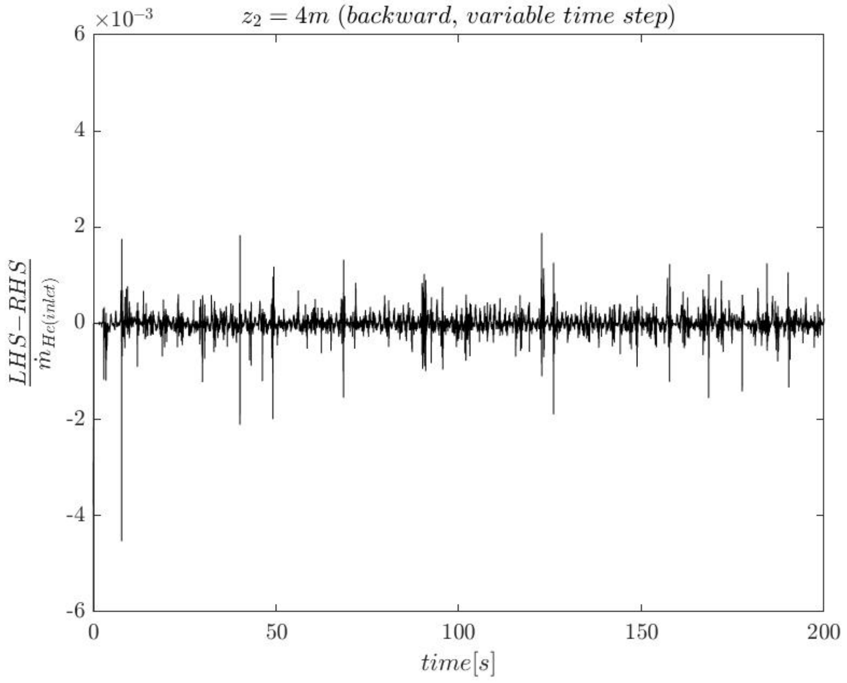


Fig. 3.4.14 backward Scheme: variable time step at z_2 = 4m

(4) backward: compare constant time step and variable time step

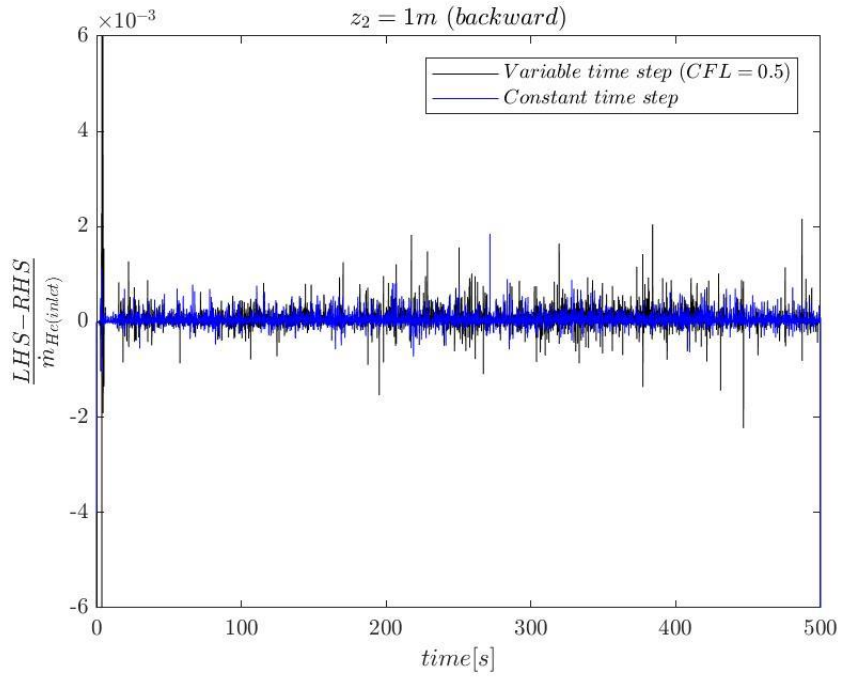


Fig. 3.4.15 backward Scheme: variable time step vs. constant time step at z_2 = 1m

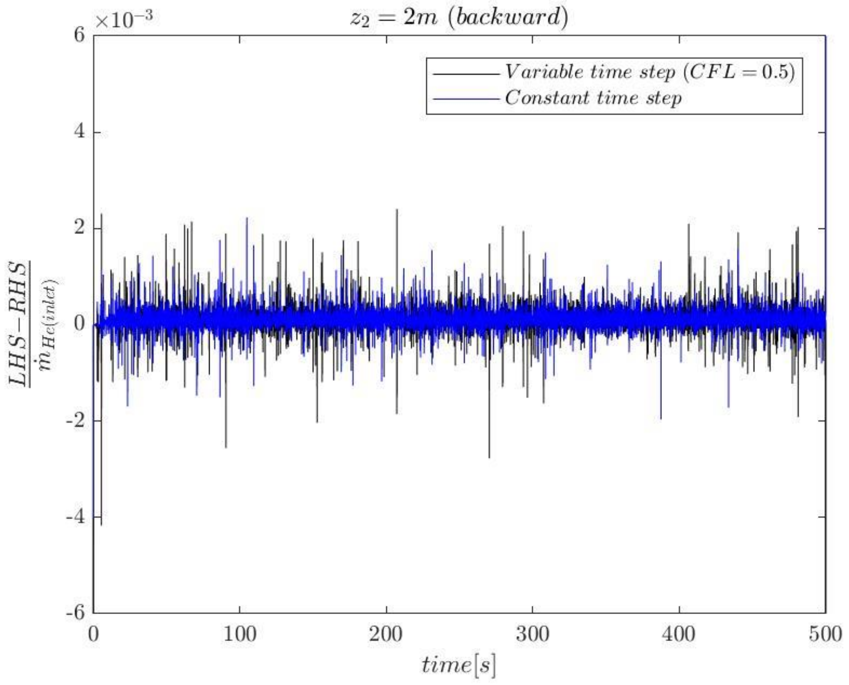


Fig. 3.4.16 backward Scheme: variable time step vs. constant time step at z_2 = 2m

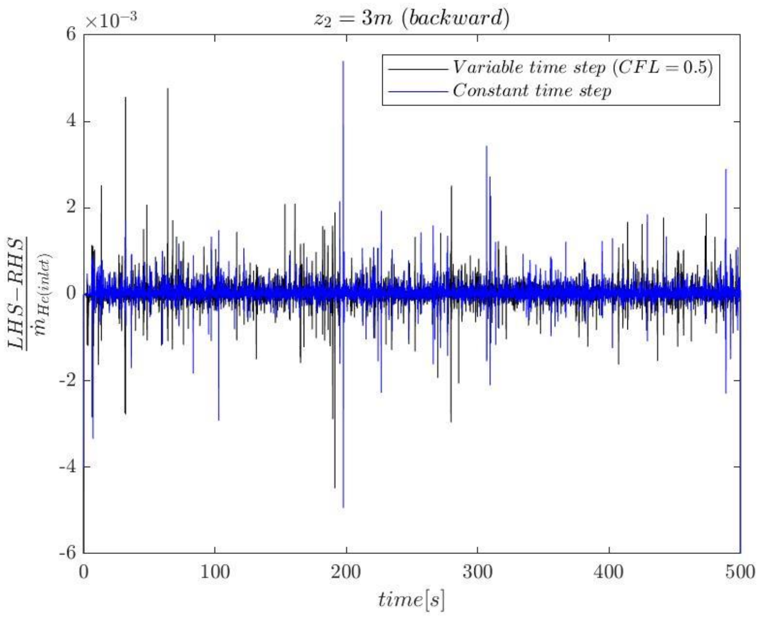


Fig. 3.4.17 backward Scheme: variable time step vs. constant time step at z_2 = 3m

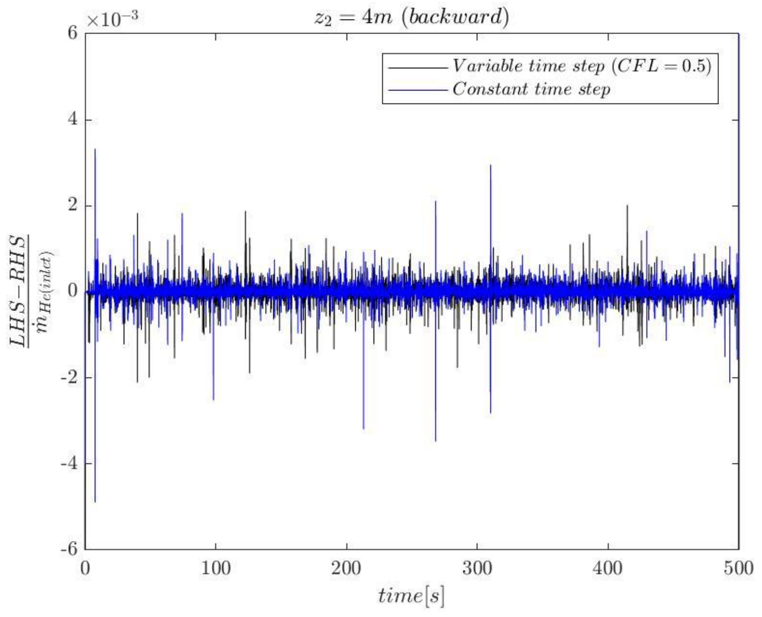


Fig. 3.4.18 backward scheme: variable time step vs. constant time step at z_2 = 4m

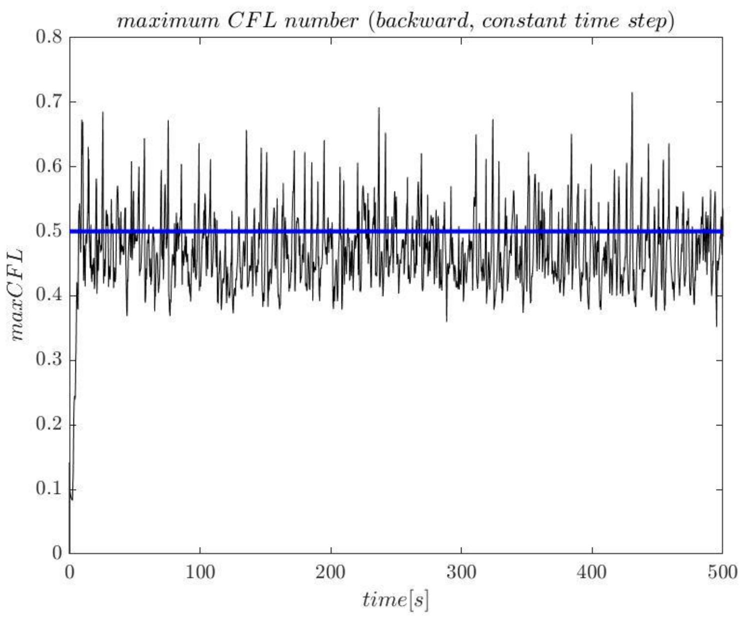


Fig. 3.4.19 backward scheme with constant time step: Maximum CFL number

(5) Compare Euler and Backward (variable time step):

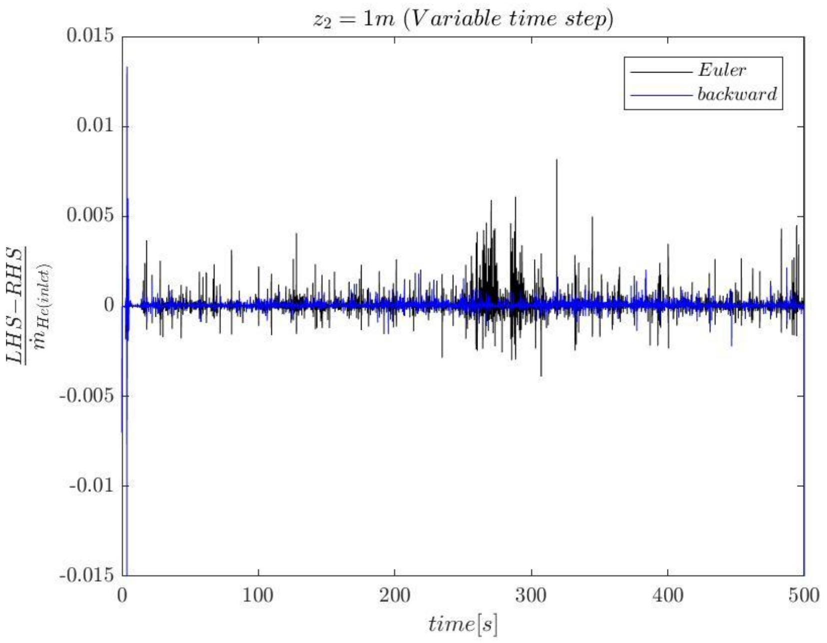


Fig. 3.4.20 Variable time step: Euler vs. backward scheme at z_2 = 1m

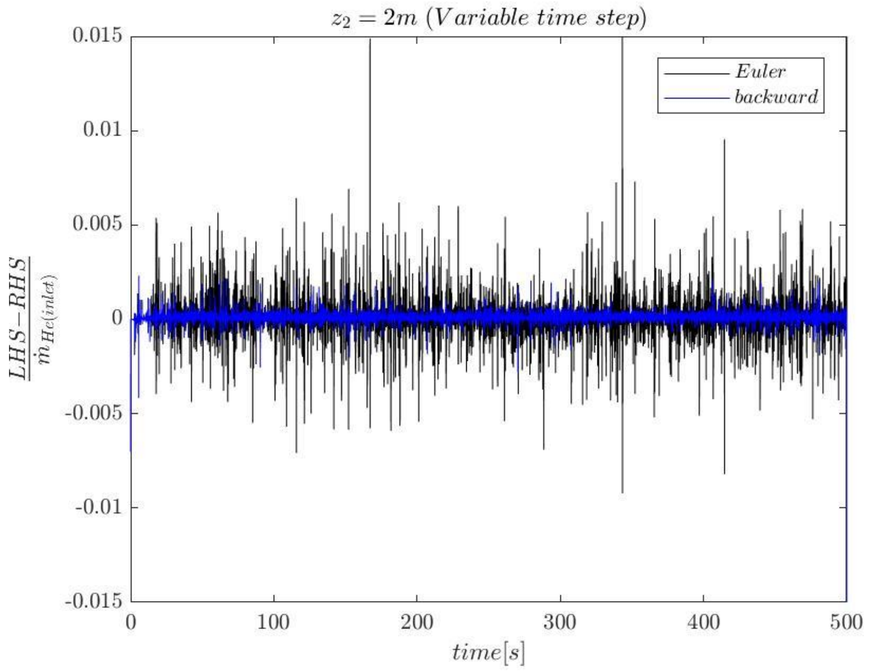


Fig. 3.4.21 Variable time step: Euler vs. backward scheme at z_2 = 2m

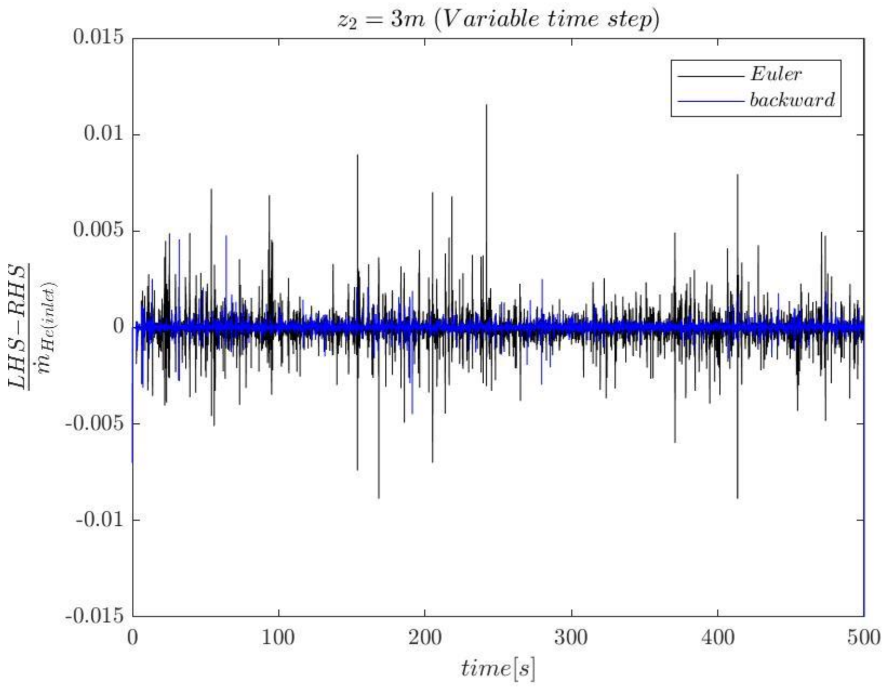


Fig. 3.4.22 Variable time step: Euler vs. backward scheme at z_2 = 3m

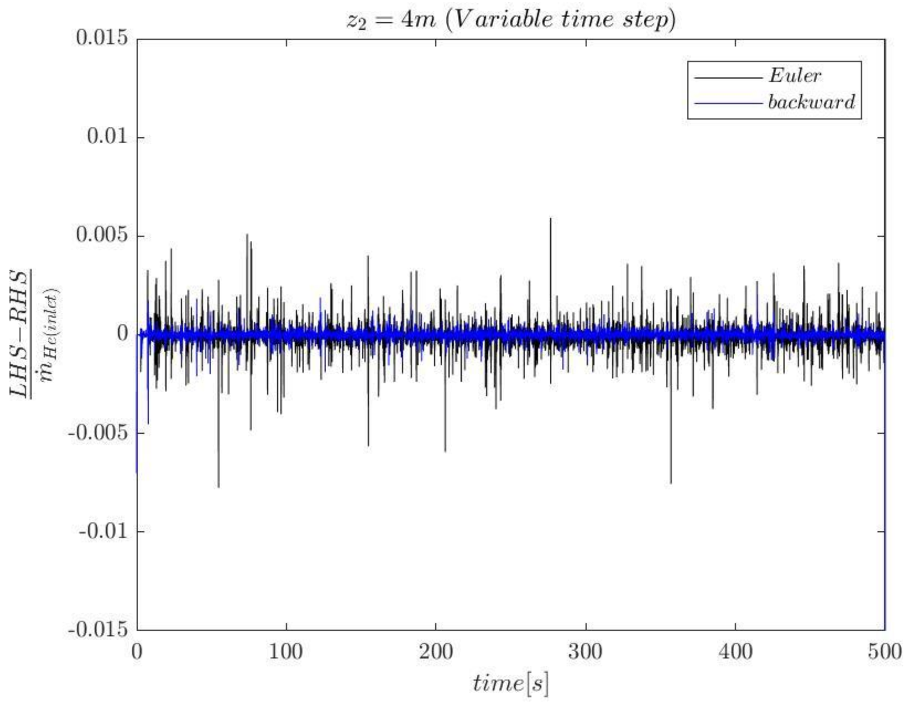


Fig. 3.4.23 Variable time step: Euler vs. backward scheme at z_2 = 4m

Discussion and Conclusion:

Four tests are performed to assess the effect of different temporal schemes on species mass conservation. Results of Euler scheme with either constant time step or variable time step have a scale less than 1%. Results of backward scheme with either constant time step or variable time step have a scale less than 0.2%. Simulation with constant time step have a comparable results scale with variable time step.

Euler with variable time step is controlled by CFL number. Generally, people will choose 0.5 as a standard value. Euler with constant time step chooses a same scale of time step using in variable time step, which is Δt = 0.01s. Plot of maximum CFL vs. time also indicates the average of maximum CFL number for constant time step test is around 0.5. This makes two results comparable. By plotting two results in one figure, it will be easier to see their differences. As figures showing, results for Euler scheme with constant time step or variable time step have a similar scale, though they are not exactly match. The results for backward scheme also follow same pattern.

There are two time schemes using in the simulation, Euler and backward. Euler is first order accuracy time scheme, while backward is second order accuracy scheme. It is necessary to compare the difference between these two schemes. Results for backward scheme have a scale less than 0.2%, while results for Euler scheme have a scale less than 1%. This can roughly tell that backward can produce less error in simulation. To be clearer, plot results for two schemes in the same figure. The figures show the same pattern. The results for backward scheme are all less than the scale of Euler scheme.

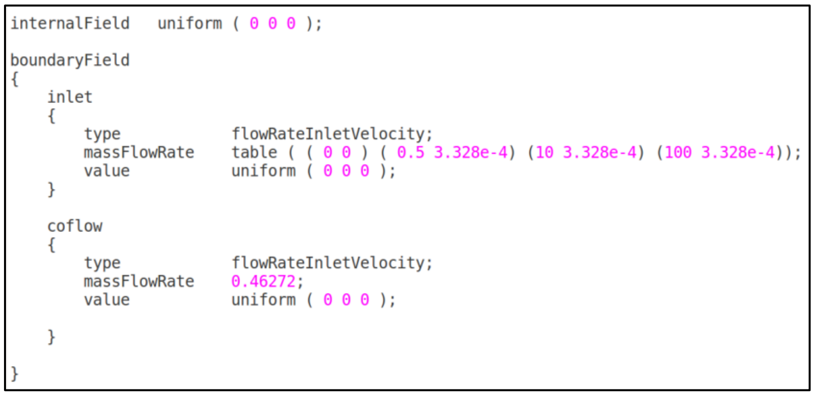
In conclusion, constant time step mode or variable time step mode will give comparable results. The errors, which Euler scheme and backward scheme will bring, are both acceptable. However, the backward scheme can result in less scale error in simulation. Another thing needs to be mentioned is that Euler scheme will need less computing time, while using backward scheme will need more computing time. FireFOAM is a LES solver, which requires a second order time scheme. For the following assessment tests, backward time scheme with variable time step will be applied into simulation.

Chapter 4: Helium Plume Three-Dimensional Case:

Case Description and Setup

Species mass conservation assessment starts from a two-dimensional case. To make it more realistic, next step is to construct a case in three-dimensional. If keeping last case's dimension but using 5.5 meter in depth, the number of computational meshes will be much higher than before, which will result in expensively computational cost. Therefore, helium plume three-dimensional case uses a shrinking computational domain. The domain is 1 meter in length, 2 meter in height and 1 meter in depth with a 0.2m×0.2m×0.2m burner in the center of domain. Users may use *blockMeshDict* to construct three-dimensional domain as in two-dimensional case.

The helium is injecting from the burner with a 0.05m/s velocity, and surrounding by air coflow with 0.4m/s velocity, which are same as 2 dimensional case. In terms of mass flow rate, helium flow is 3.328×10^{-4} kg/s, and air coflow is 0.46272 kg/s. Users may go to *U* file to change the definition of inlet and coflow velocity.



```
internalField   uniform ( 0 0 0 );

boundaryField
{
    inlet
    {
        type            flowRateInletVelocity;
        massFlowRate    table ( ( 0 0 ) ( 0.5 3.328e-4) (10 3.328e-4) (100 3.328e-4));
        value           uniform ( 0 0 0 );
    }

    coflow
    {
        type            flowRateInletVelocity;
        massFlowRate    0.46272;
        value           uniform ( 0 0 0 );

    }
}
```

Fig. 4.1.1 Velocity file *U* example for three-dimensional case

With a smaller computational domain, the selected tested heights (z_2) are different from two-dimensional case. Users may use *topoSetDict* to define new surfaces, and *setFieldsDict* to define location of control volume as 2-D case. The backward time scheme with variable time step applied in this case. Users can go to *fvScheme* to specify the temporal schemes. Besides, the aim of this three-dimensional case is to test convection schemes. Definition of convection schemes locates in *fvScheme* file. This will be discussed later.

To visualize the three-dimension domain, users can go to user terminal to open ParaView. The structure of burner is hidden in the center of domain. If users want to check if definition in *blockMeshDict* file locates the burner correctly, users may visualize domain using *Points* method. Users can find it in the tool bar at top of ParaView user interface.

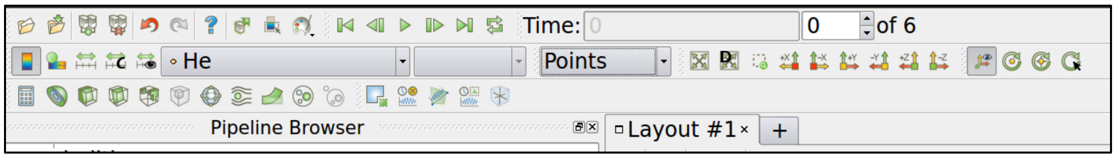


Fig. 4.1.2 Visualization of domain by *Points* method in ParaView

Using the visualizing method, users can see the structure of computational domain and burner are as below.

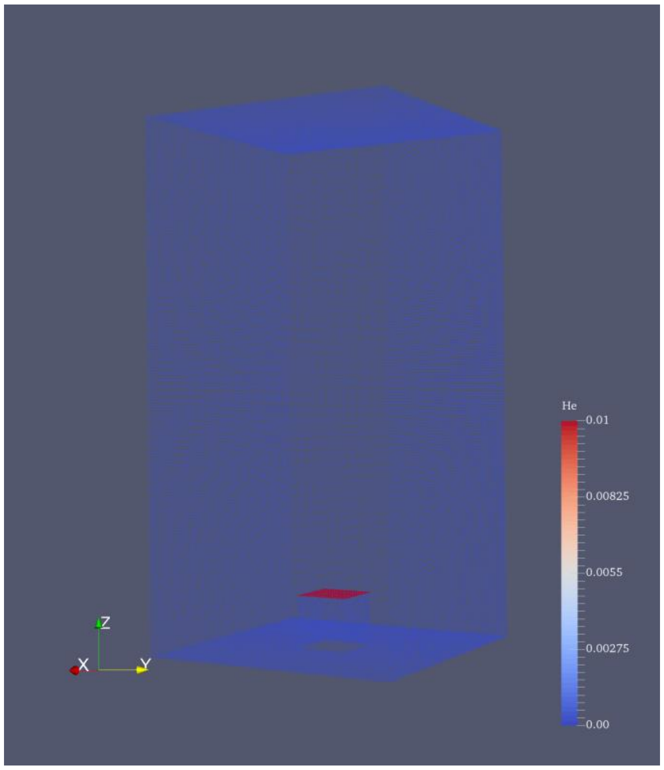
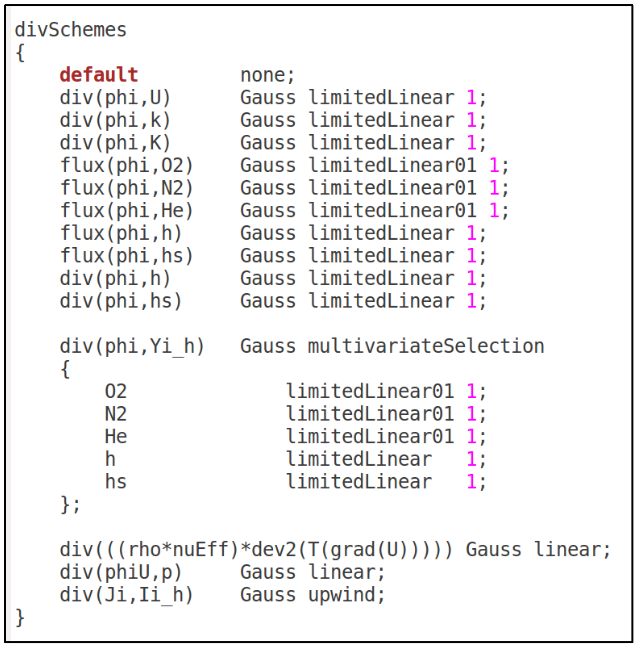


Fig. 4.1.3 Structure of Three-dimensional domain

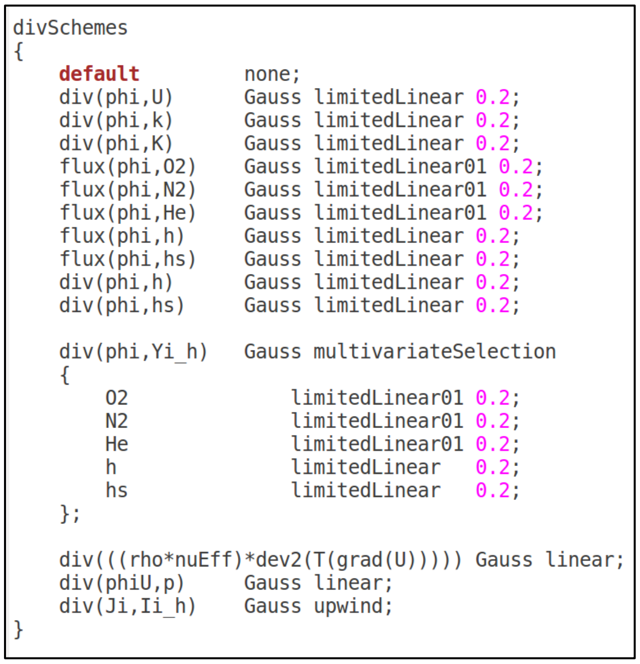
Convection Scheme:

The purpose of this case is to assess the convection scheme *limitedLinear* with two different coefficients. As the diagnostic equation states, the right hand side includes convection term and diffusion term. This case is to check how convection scheme: limitedLinear <1.0> and limtedLinear <0.2> affect species mass conservation. To change the convection scheme, users may go to the *fvScheme* file, where to define time scheme previously.



```
divSchemes
{
    default         none;
    div(phi,U)      Gauss limitedLinear 1;
    div(phi,k)      Gauss limitedLinear 1;
    div(phi,K)      Gauss limitedLinear 1;
    flux(phi,O2)    Gauss limitedLinear01 1;
    flux(phi,N2)    Gauss limitedLinear01 1;
    flux(phi,He)    Gauss limitedLinear01 1;
    flux(phi,h)     Gauss limitedLinear 1;
    flux(phi,hs)    Gauss limitedLinear 1;
    div(phi,h)      Gauss limitedLinear 1;
    div(phi,hs)     Gauss limitedLinear 1;

    div(phi,Yi_h)   Gauss multivariateSelection
    {
        O2              limitedLinear01 1;
        N2              limitedLinear01 1;
        He              limitedLinear01 1;
        h               limitedLinear   1;
        hs              limitedLinear   1;
    };

    div(((rho*nuEff)*dev2(T(grad(U))))) Gauss linear;
    div(phiU,p)     Gauss linear;
    div(Ji,Ii_h)    Gauss upwind;
}
```

Fig. 4.2.1 Convection scheme: *limitedLinear < 1.0 >* definition in *fvShceme* file



```
divSchemes
{
    default         none;
    div(phi,U)      Gauss limitedLinear 0.2;
    div(phi,k)      Gauss limitedLinear 0.2;
    div(phi,K)      Gauss limitedLinear 0.2;
    flux(phi,O2)    Gauss limitedLinear01 0.2;
    flux(phi,N2)    Gauss limitedLinear01 0.2;
    flux(phi,He)    Gauss limitedLinear01 0.2;
    flux(phi,h)     Gauss limitedLinear 0.2;
    flux(phi,hs)    Gauss limitedLinear 0.2;
    div(phi,h)      Gauss limitedLinear 0.2;
    div(phi,hs)     Gauss limitedLinear 0.2;

    div(phi,Yi_h)   Gauss multivariateSelection
    {
        O2              limitedLinear01 0.2;
        N2              limitedLinear01 0.2;
        He              limitedLinear01 0.2;
        h               limitedLinear   0.2;
        hs              limitedLinear   0.2;
    };

    div(((rho*nuEff)*dev2(T(grad(U))))) Gauss linear;
    div(phiU,p)     Gauss linear;
    div(Ji,Ii_h)    Gauss upwind;
}
```

Fig. 4.2.2 Convection Scheme: *limitedLinear <0.2>* definition in *fvScheme* file

Users may find detailed information about limitedLinear convection scheme in OpenFOAM Userguide Chapter 4 [13]. Also, Verma's studies have described limitedLinear convection scheme [5]. List of Assessment tests is as below.

Test Number	Convection Scheme
(1)	limitedLinear <1.0>
(2)	limitedLinear <0.2>

Table. 4.2.1 Convection Scheme Test List

Hints: The backward scheme with variable time step and nOuterCorrectors = 5 are applied into these tests.

The three-dimensional case is computationally expensive than two-dimensional case. Therefore, the duration for helium plume case is shrinking to 10s.

Results

Helium Field Visualization:

When moving to a three-dimensional case, users may not view the fields in the computational domain directly, there is another method need to be used to visualized helium plume field. In the tool bar at the top of ParaView user interface, users can find a button called *Clip*.

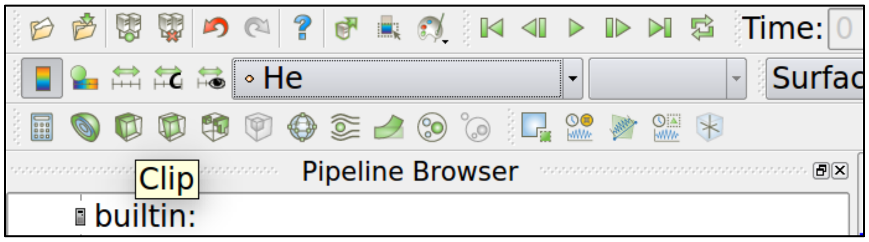
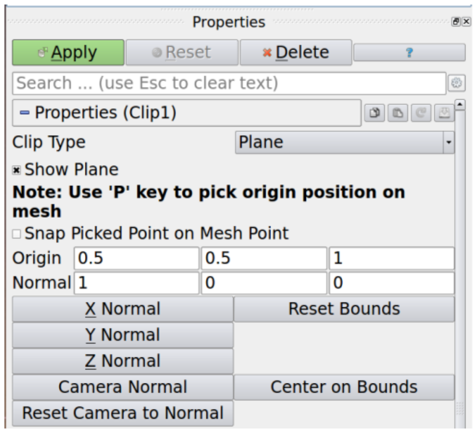


Fig. 4.3.1.1 *Clip* button in ParaView tool bar

Click the *Clip* button and select region in *Propeties* section. The example shows the slice locates in the center of computational domain. After finishing selecting region, click the *Apply* button.



Clip Type: Plane

Show Plane

Note: Use 'P' key to pick origin position on mesh

Snap Picked Point on Mesh Point

Origin	0.5	0.5	1
Normal	1	0	0

Fig. 4.3.1.2 *Properties* section for *Slice* method

After applying *Clip* visualization method, users can view helium plume field in the domain. Followings are helium plume field at time = 10s using limitedLinear <1.0> and limitedLinear <0.2> convection scheme respectively.

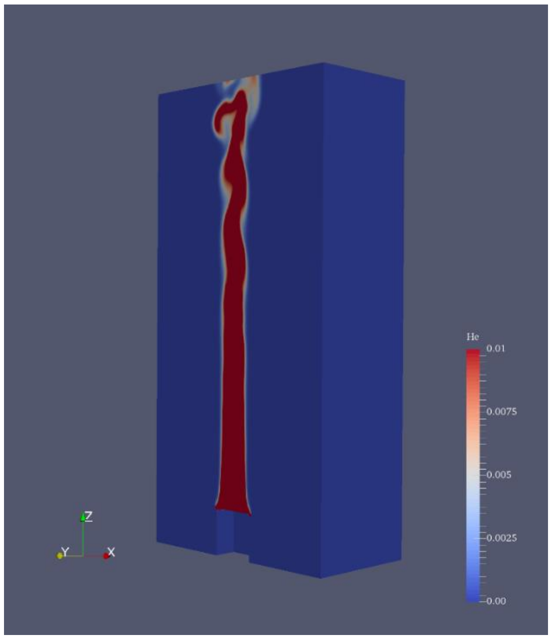


Fig. 4.3.1.3 limitedLinear <1.0> helium plume field at time = 10s

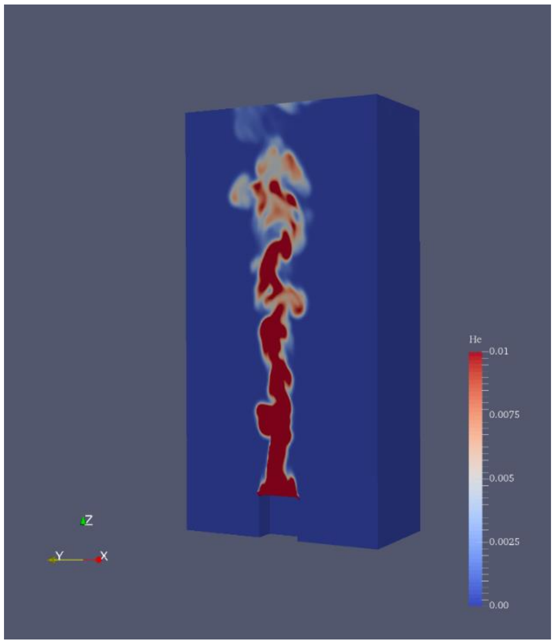


Fig. 4.3.1.4 limitedLinear <0.2> helium plume field at time = 10s

For the helium mass fraction value range, users can customize the outlook of it. Users can find a Mapping Data section on the right side of ParaVew user interface. Click the *Rescale to custom range* button, then users may define the maximum value and minimum value of helium mass fraction in *Set Range* section. Here, the maximum value is 0.01 and minimum value is 0.

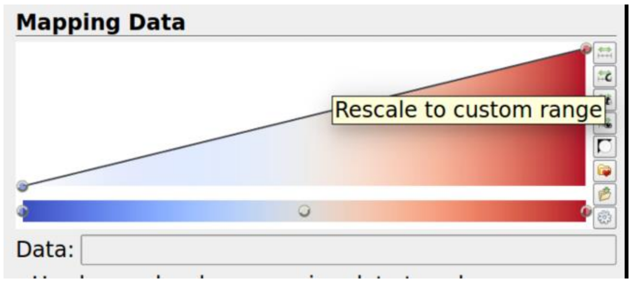
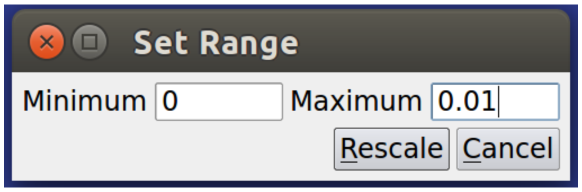


Fig. 4.3.1.5 *Mapping Data* Section in ParaView

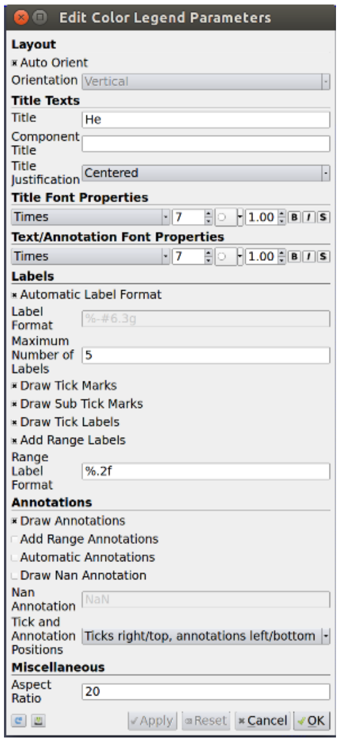


Set Range

Minimum 0 Maximum 0.01

Fig. 4.3.1.6 *Set Range* section in ParaView

In addition, users can customize the font type and size in the *Edit Color Legend Parameters* section. As the helium field value range shows, the range title is *He*, text content is from 0 to 0.01. Font type is Times and font size is 7 for both title and text. Users can also define the color title and text in this section. The color used in example is white.



Edit Color Legend Parameters

Layout

- [✓] Auto Orient
- Orientation: Vertical

Title Texts

- Title: He
- Component Title:
- Title Justification: Centered

Title Font Properties

- Times | 7 | 1.00 | B I S

Text/Annotation Font Properties

- Times | 7 | 1.00 | B I S

Labels

- [✓] Automatic Label Format
- Label Format: %-#6.3g
- Maximum Number of Labels: 5
- [✓] Draw Tick Marks
- [✓] Draw Sub Tick Marks
- [✓] Draw Tick Labels
- [✓] Add Range Labels
- Range Label Format: %.2f

Annotations

- [✓] Draw Annotations
- [ ] Add Range Annotations
- [ ] Automatic Annotations
- [ ] Draw Nan Annotation
- Nan Annotation: NaN
- Tick and Annotation Positions: Ticks right/top, annotations left/bottom

Miscellaneous

- Aspect Ratio: 20

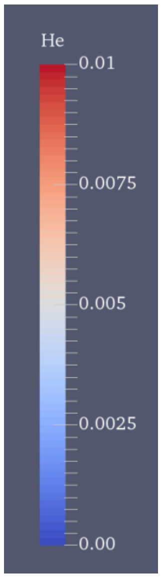


Fig. 4.3.1.7 Edit Color and Legend Parameters

Fig. 4.3.1.8 Helium Mass Fraction Range

Plots

Following plots show the instantaneous relative error at different height. To quantify the relative error, root mean square of the instantaneous relative at different heights are calculated.

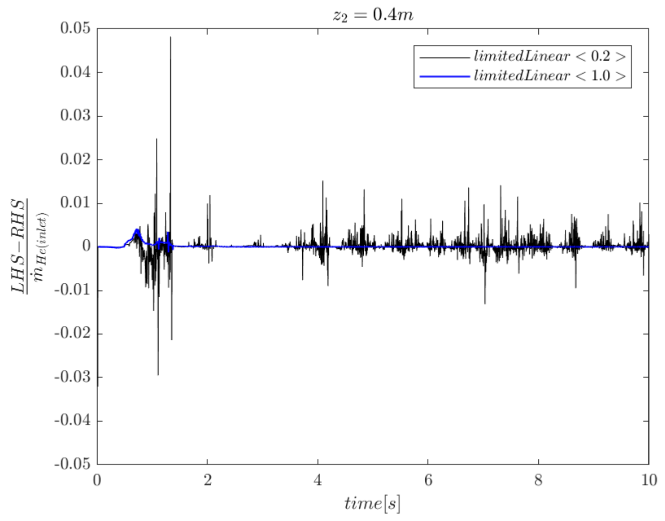


Fig. 4.3.2.1 Instantaneous relative error: limitedLinear <0.2> vs. limitedLinear <1.0> at z_2 = 0.4m

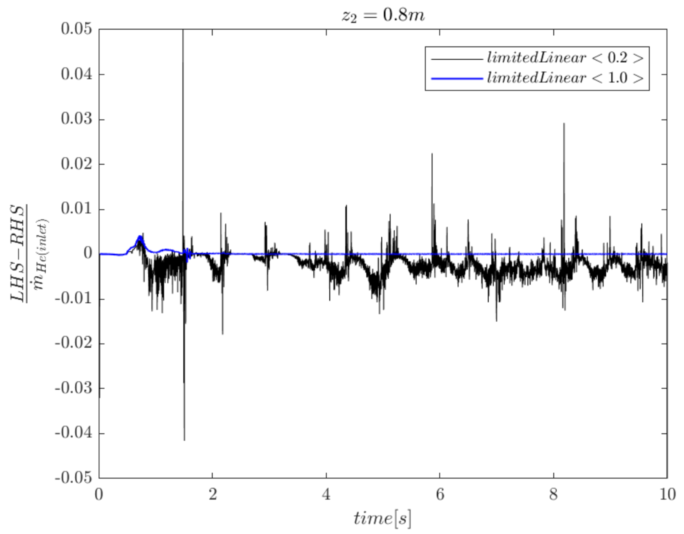


Fig. 4.3.2.2 Instantaneous relative error: limitedLinear <0.2> vs. limitedLinear <1.0> at z_2 = 0.8m

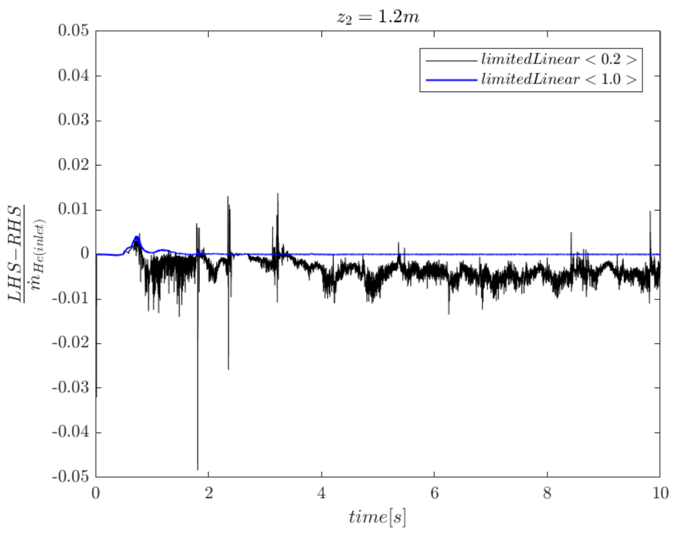


Fig. 4.3.2.3 Instantaneous relative error: limitedLinear <0.2> vs. limitedLinear <1.0> at z_2 = 1.2m

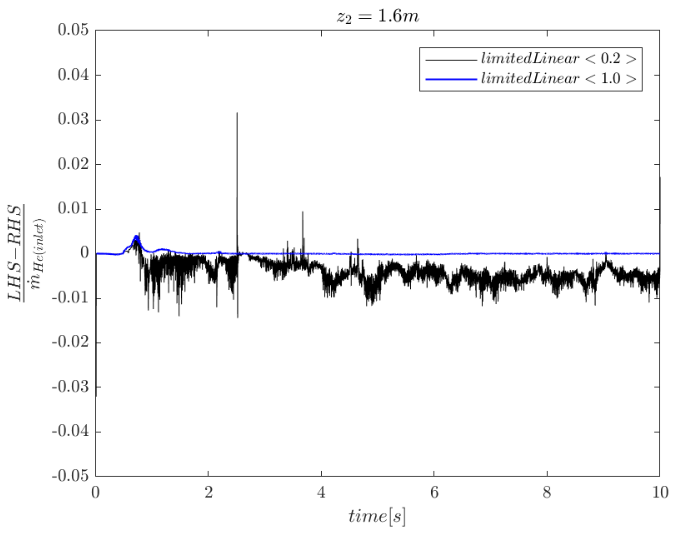


Fig. 4.3.2.4 Instantaneous relative error: limitedLinear <0.2> vs. limitedLinear <1.0> at z_2 = 1.6m

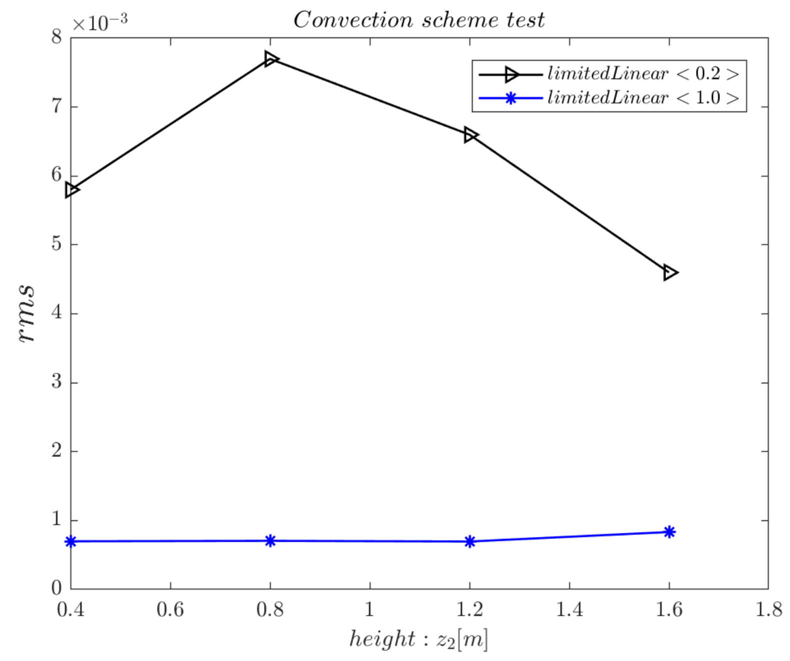


Fig. 4.3.2.5 root mean square of instantaneous relative error vs. heights for limitedLinear <0.2> and limitedLinear <1.0>

Discussion

As states before, this case is to check the effects of different convection schemes on mass (species) conservation. This case uses a smaller computational domain which allows higher resolution mesh (20×20 across burner) with acceptable computing (around 20 hours). Higher resolution grids allow better resolution of fluids behavior.

Comparing the two helium plume fields, the plume shape tells the difference between two convection schemes. At time = 10s, helium plume field with limitedLinear <1.0> has a more laminar flow, though it is a little turbulent near top of domain; helium plume field with limitedLinear <0.2> has a more turbulent flow through whole domain. Comparing the plots of relative error vs. time, curve of limitedLinear <1.0> have a smaller scale of relative error, while limitedLinear <0.2> have higher scale of error. In addition, curves of limitedLinear <1.0> have less fluctuations, while limitedLinear <0.2> have more fluctuations. To quantify the instantaneous relative error, root mean square of the relative error for different heights are calculated. From rms vs. height figure, the limitLinear convection scheme with coefficient equals 1 make mass more conserved mathematically.

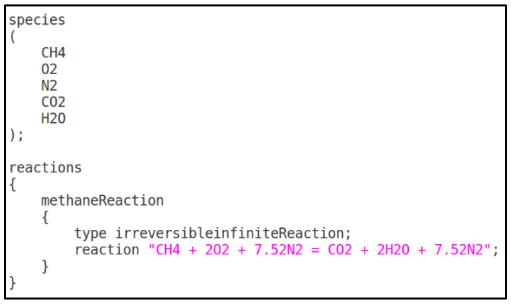
In conclusion, limitedLinear <1.0> works better in conservative properties. But it brings more numerical dissipation into simulation and does not capture the physics of flow and lose reality. The limitedLinear <0.2> brings less numerical dissipations in simulation, but more errors in conservative properties. However, it captures more turbulence and is more realistic. The limitedLinear <0.2> is believed to provide a better representation of turbulent fluctuations and is recommended for Large Eddy Simulation of turbulent flow/flame problems.

Next chapter will continue using convection scheme limitedLinear <0.2>, and talk about how number of outer loops in PIMPLE algorithm affect accuracy of species mass conservation statement for FireFOAM.

Chapter 5: Pool Fire Case

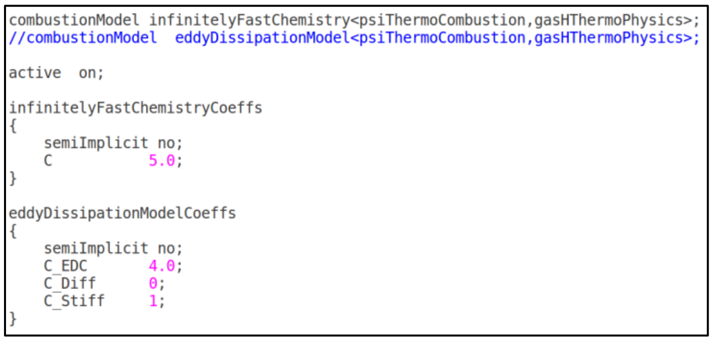
Case Description and Setup

This case keeps using same computational domain with three-dimensional helium plume case, which is 1m×1m×2m domain with a 0.2m×0.2m×0.2m burner. This case change fuel from helium to methane and add combustion in simulation. Users need to name a file called *CH4 to* replace the *He* file under 0 folder in case folder. Similarly, *CH4* file is used to define methane initial properties. To specify species associated with the combustion reaction, users need to use a file called *reactions* under *constant* directory, and use a file called *combustionProperties* to turn on combustion in simulation.



```
species
(
    CH4
    O2
    N2
    CO2
    H2O
);

reactions
{
    methaneReaction
    {
        type irreversibleinfiniteReaction;
        reaction "CH4 + 2O2 + 7.52N2 = CO2 + 2H2O + 7.52N2";
    }
}
```

Fig. 5.1.1 *reactions* file example



```
combustionModel infinitelyFastChemistry<psiThermoCombustion,gasHThermoPhysics>;
//combustionModel  eddyDissipationModel<psiThermoCombustion,gasHThermoPhysics>;

active  on;

infinitelyFastChemistryCoeffs
{
    semiImplicit no;
    C            5.0;
}

eddyDissipationModelCoeffs
{
    semiImplicit no;
    C_EDC        4.0;
    C_Diff       0;
    C_Stiff      1;
}
```

Fig. 5.1.2 *combustionProperties* file Example

For last two cases, diagnostic is tracking mass fraction of helium, which will not change through simulation because of no reactions occurring. In this fire case, methane will burn and generate products. It is not convenient to track methane. In this case, a new term is defined to assess species mass conservation.

$$Z' = 1 - \frac{Y_{N_2}}{Y_{N_2,air}} \qquad eqn. 5.1.1$$

In the above formula, $Y_{N_2,air}$ is a constant, which can be found in OpenFOAM library. The only variable is Y_{N_2} . Use Z' instead of Z is to distinguish from traditional mixture fraction definition Z.

Species conservation in terms of Y_{N_2} can be written as:

$$\frac{d}{dt}\left(\iiint_{CV} \rho Y_{N_2}\, dV\right) = -\oiint_{CS} \rho Y_{N_2}(\vec{u}-\vec{v})\cdot\vec{n}\, dS + \oiint_{CS} \rho D\left(\nabla Y_{N_2}\cdot\vec{n}\right) dS$$

eqn. 5.1.2

Ans specie conservation in terms of $Y_{N_2,air}$ can be written as:

$$\frac{d}{dt}\left(\iiint_{CV} \rho Y_{N_2,air}\, dV\right) = -\oiint_{CS} \rho Y_{N_2,air}(\vec{u}-\vec{v})\cdot\vec{n}\, dS$$

eqn. 5.1.3

$$Y_{N_2} = Y_{N_2,air}\ (1 - Z')$$

eqn. 5.1.4

Use eqn. 5.1.3 to replace Y_{N_2} in eqn. 5.1.2,

Unsteady term:

$$\frac{d}{dt}\left(\iiint_{CV}\rho Y_{N_2}\,dV\right)=\boxed{\frac{d}{dt}\left(\iiint_{CV}\rho Y_{N_2,air}\quad dV\right)}-Y_{N_2,air}\frac{d}{dt}\left(\iiint_{CV}\rho Z'\,dV\right)$$

eqn. 5.1.5

Convection term:

$$-\oiint_{CS}\rho Y_{N_2}\left(\overrightarrow{u_{N_2}}-\vec{v}\right)\cdot\vec{n}\,dS$$

$$=\boxed{-\oiint_{CS}\rho Y_{N_2,air}\,(\vec{u}-\vec{v})\cdot\vec{n}\,dS}+Y_{N_2,air}\oiint_{CS}\rho Z'(\vec{u}-\vec{v})\cdot\vec{n}\,dS$$

eqn. 5.1.6

Diffusion term:

$$\oiint_{CS}\rho D\left(\nabla Y_{N_2}\cdot\vec{n}\right)dS=-Y_{N_2,air}\oiint_{CS}\rho D(\nabla Z'\cdot\vec{n})dS$$

eqn. 5.1.7

Combining eqn. 5.1.2 to eqn. 5.1.7, species mass conservation equation in terms of Z'can be written as:

$$\frac{d}{dt}\left(\iiint_{CV}\rho Z'\,dV\right)=-\oiint_{CS}\rho Z'(\vec{u}-\vec{v})\cdot\vec{n}\,dS+\oiint_{CS}\rho D(\nabla Z'\cdot\vec{n})dS$$

eqn. 5.1.8

Above equation derivations explains the process of representing species mass conservation in terms of Z'. It is easy to observe that species mass conservation equation in terms of mixture fraction Z' has similar format of in terms of mass fraction ($Y_{N_2}\ or\ Y_{He}$). Thus, using Z' to replace Y_{He} in species mass conservation diagnostic, built in FireFOAM solver, can achieve the aim to tracking new defined mixture fraction Z'.

In helium plume case, higher velocity of air coflow is to avoid helium flowing out from domain though sides. This flame case is tracking new defined term mixture fraction Z' . If there are air entrainments from sides of domain, $Y_{N_2} = Y_{N_2,air}$, the term Z' will be zero, which will not affect results. Therefore, it is reasonable to only consider surfaces perpendicular to z-axis direction for right hand side of species mass conservation equation. To make sure above derivation process and using Z' to replace Y_{N_2} in diagnostic are correct, it is necessary to check if mixture fraction diagnostic can get same results as mass fraction diagnostic.

Mixture fraction diagnostic test

Before applying new defined mixture fraction diagnostic to flame case, it is necessary to use previous helium case to test the new mixture fraction diagnostic. If the new diagnostic works, it will get identical results for helium plume case with using previous mass fraction diagnostic. As mentioned before, flame case will use limitedLinear <0.2> convection scheme. Therefore, this test is conducted by comparing results of two diagnostics with using limitedLinear <0.2> convection scheme.

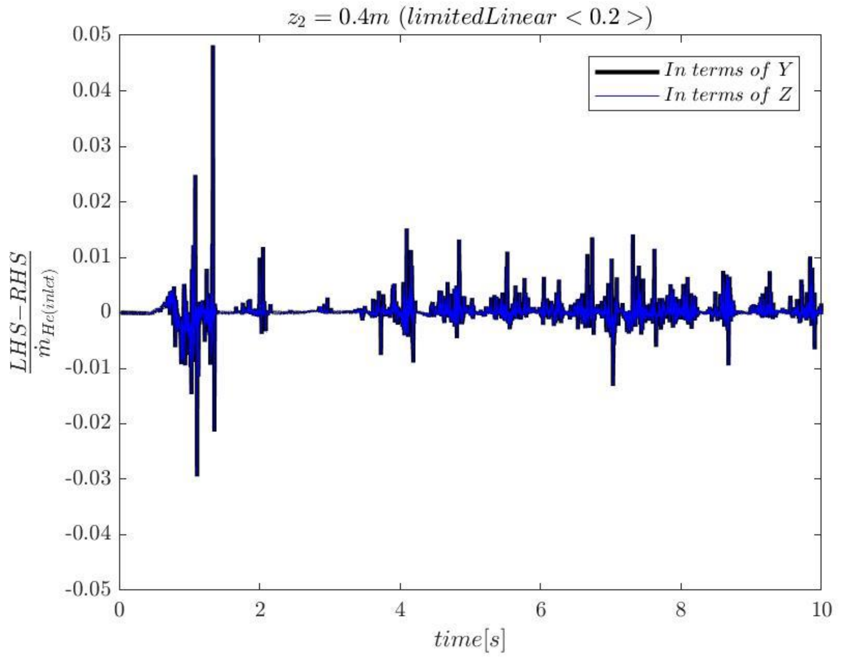


Fig. 5.2.1 Comparison between two diagnostics at z_2 = 0.4m

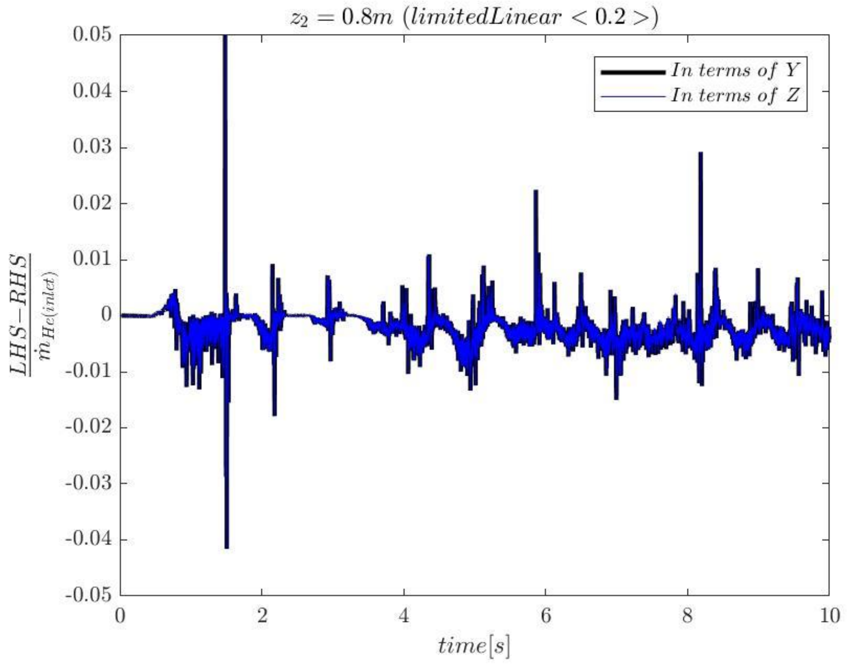


Fig. 5.2.2 Comparison between two diagnostics at z_2 = 0.8m

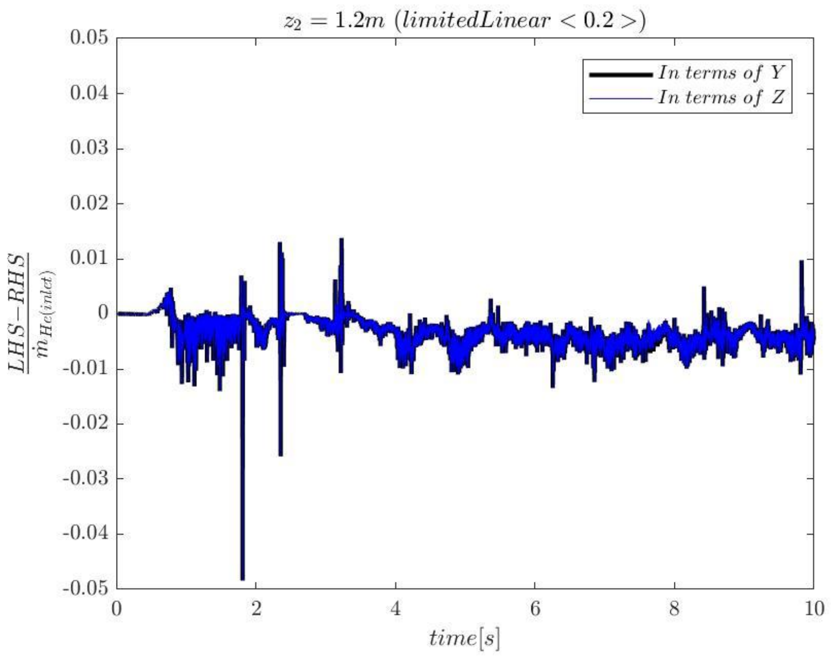


Fig. 5.2.3 Comparison between two diagnostics at z_2 = 1.2m

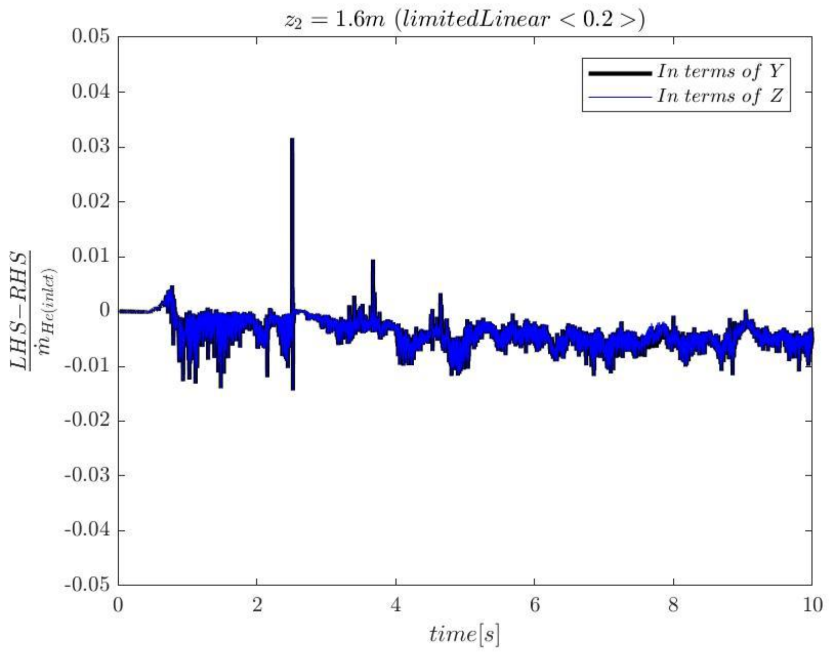


Fig. 5.2.4 Comparison between two diagnostics at z_2 = 1.6m

To better compare results with two diagnostics, the results are plotted in same figure. As above figures show, results of two diagnostics are identical with each other. It provides an evidence that the derivation process and using Z' to replace Y_{N_2} in previous mass fraction diagnostic are correct.

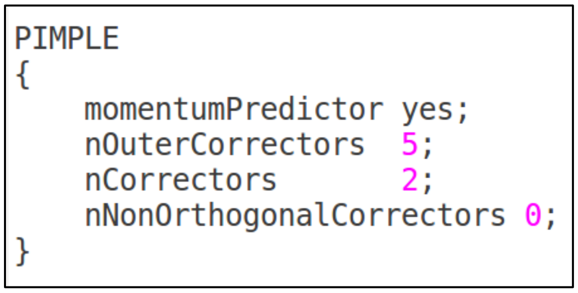
<u>*Number of outer loops*</u>

The number of outer correctors (*nOuterCorrectors*) in PIMPLE algorithm provides coupling in sequence of successive solution steps for conservation in mass, momentum and energy. This coupling is required to ensure the accuracy of conservation statements. Users can specify *nOuterCorrectors* in *fvSolution* file under *system* directory. The aim of this case is to assess the effects of different number of outer loops on accuracy of species mass conservation of FireFOAM solver. Six tests with different number of outer correctors are performed. The number of outer loops for each test is as below.

Test Number	(1)	(2)	(3)	(4)	(5)
nOuterCorrectors	n=2	n=3	n=4	n=5	n=10

Table. 5.3.1 Test Number for Flame Case

Hints: The time scheme: backward with variable time step and convection scheme: limtedLinear <0.2> are applied in these tests.



```
PIMPLE
{
    momentumPredictor yes;
    nOuterCorrectors  5;
    nCorrectors       2;
    nNonOrthogonalCorrectors 0;
}
```

Fig. 5.3.1 nOuterCorrectors specification in *fvSolution* file

<u>*Visualization of Fire Plume Methods*</u>

In three-dimensional helium plume case, visualization of helium plume used *Clip* method, which can be found in the tool bar of ParaView interface. This method also applies for visualization of fire plume. Users should pay attention to the field selected for visualization. For helium case, it is mass fraction of helium; For fire plume case, is can be temperature. Users may follow the instructions discussed in helium plume case to edit temperature range. This case selects a temperature range from 300K to 2500K. To make plume's temperature gradient more realistic, this case selects *2hot* to represent the buoyant fire plume. The *2hot is a* color gradient selection in *Choose Preset* section, which can be found on the right side of *Mapping Data*.

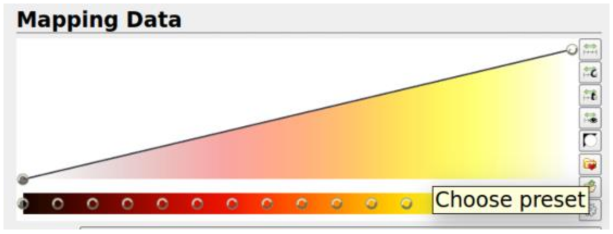
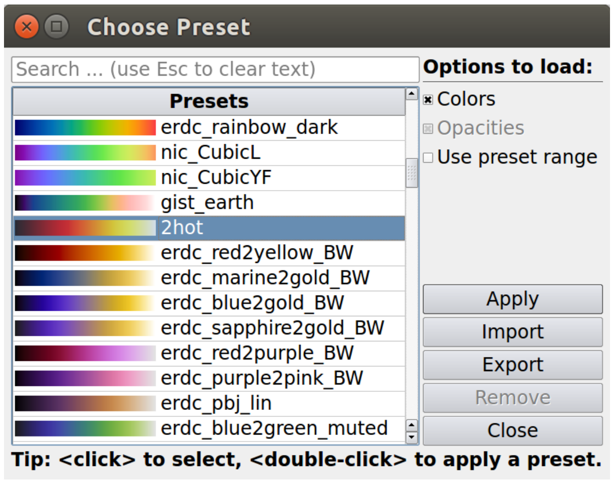


Fig. 5.4.1 *Mapping Data* Section in ParaView


Choose Preset

Search ... (use Esc to clear text)

Options to load:

[✓ Colors]

Opacities

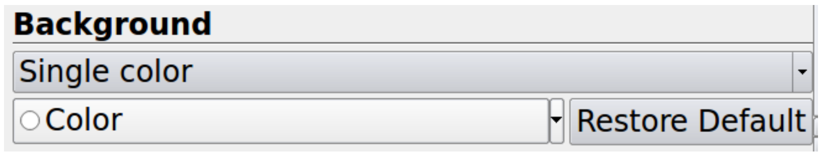
Use preset range

Presets
erdc_rainbow_dark
nic_CubicL
nic_CubicYF
gist_earth
2hot
erdc_red2yellow_BW
erdc_marine2gold_BW
erdc_blue2gold_BW
erdc_sapphire2gold_BW
erdc_red2purple_BW
erdc_purple2pink_BW
erdc_pbj_lin
erdc_blue2green_muted

Tip: <click> to select, <double-click> to apply a preset.

Fig. 5.4.2 *Choose Preset* Section in ParaView

Besides, to better visualize the fire plume, this case selects color of write as a background. Users may go to Properties section and edit the background color for their cases.


Background

Single color

Color

Fig. 5.4.3 *Background* section in ParaView

The plume visualization will be presented in results section.

Results:

This case adds combustion into simulation, which makes the simulation more expensive. This case shrinks the duration of flame case simulation from 10 seconds to 3 seconds. The darker side of temperature range is colder, while the brighter side is hotter.

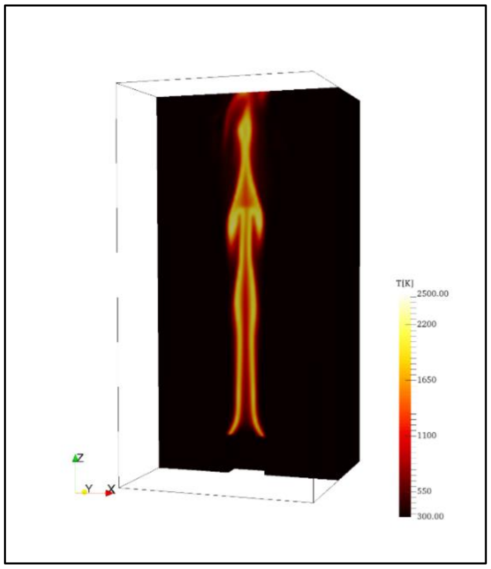


Fig. 5.5.1 Fire Plume at time = 2.5s

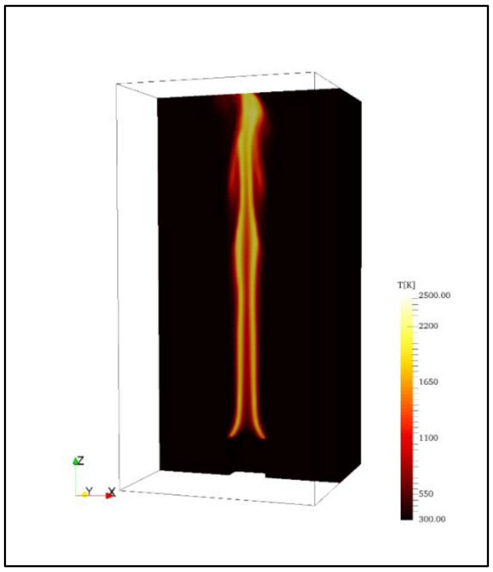


Fig. 5.5.2 Fire Plume at time = 3s

Figs. 5.4.1 and 5.4.2 look unsteady laminar rather than turbulent. That is likely due to the co-flow

To represent the results quantitively, this figure shows the correlation between *nOuterCorrectors* and the root mean square of instantaneous relative error. The x-axis is the value of *nOuterCorrectors*; The y-axis is the root mean square of instantaneous relative error. Four curves represent results at different elevation.

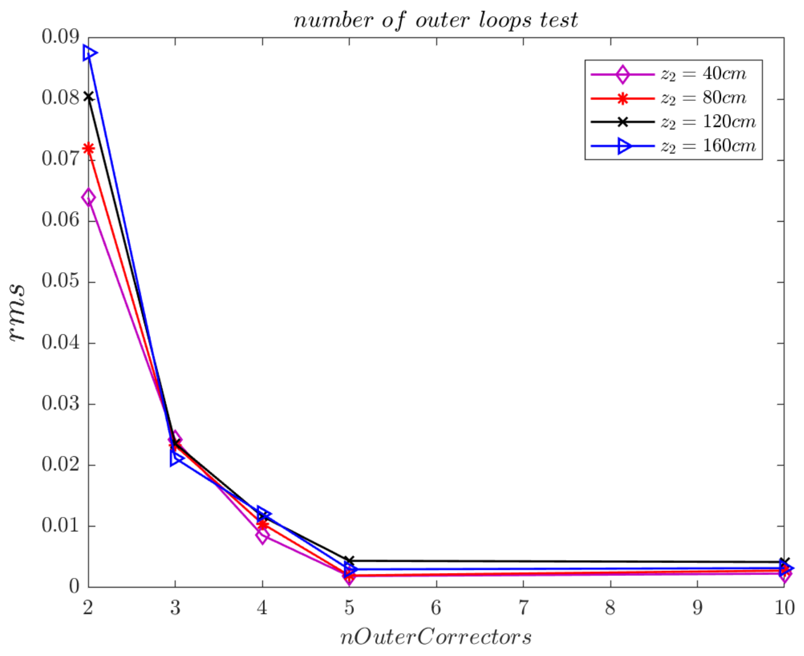


Fig. 5.5.3 root mean square (rms) of instantaneous relative error vs. nOuterCorrectors

Discussion and Conclusion

Fire involves consumption of reactants and generation of products. Previous mass fraction diagnostic will not apply to a fire case. This case develops a mixture fraction diagnostic. Helium plume case is performed to check if the mixture fraction diagnostic meets requirements of species mass conservation assessment. As Fig.5.2.1 to Fig. 5.2.4 shows, the curves of mixture fraction diagnostic and mass fraction diagnostic match with each other. These two diagnostics will generate identical results. This provides an evidence that mixture fraction diagnostic can be applied into fire plume case.

As adding combustion into simulation, the simulation becomes more computationally expensive. This fire plume case shrinks duration of simulation from 10 seconds (helium plume case) to 3 seconds. As Fig. 5.4.2 shows, fire plume arrives at top of computational domain. Figs. 5.4.1 and 5.4.2 look unsteady laminar rather than turbulent, it is likely due to air co-flow. However, it does not affect the assessment of species mass conservation.

As Fig. 5.4.3 indicates, with the increasement of number of outer loops, the root mean square (rms) of instantaneous relative error will decrease for every height. Besides, the figure also indicates that rms will converge as number of outer loops increases for every height. When n = 2, the error is all above 5% at every elevation, which is not acceptable for simulation accuracy needs. When n increases to 3, there is a big drop of rms value. Using *nOuterCorrectors* = 3 is the value used in previous studies by the University of Maryland fire modeling group. This result provides a verification of using *nOuterCorrectors* = 3 in combustion simulation. From n =4 to n

= 5, there is still a small decrease of rms; From n= 5 to n = 10, the rms' change is really small which is not easy to observe in this y-axis scale. As the values of nOuterCorrectors go to infinity, the value of residual errors will converge. This figure indicates the root mean square of instantaneous relative error will converge around n = 5 for this case.

In conclusion, for this pool fire case, when *nOuterCorrectors* increases, the root mean square of relative error decreases and will converge at around n = 5. Besides, the cost of simulation will become more expensive with a larger *nOuterCorrectors*. Users may select a *nOuterCorrectors* value that meet their accuracy needs and under their computing budget.

Chapter 6: Conclusion and Future Work

Conclusion

Multiple cases are designed to assess species mass conservation of FireFOAM. In two-dimensional helium plume case, it assessed the time scheme that affects left hand side of species mass conservation equation; In three-dimensional helium plume case, it assessed the convection scheme that affects right hand side of species mass conservation equation. In this flame case, it is to assess the number of outer loops (nOuterCorrectors) in PIMPLE algorithm effects on species mass conservation.

In helium plume 2-D case, backward time scheme has achieved higher accuracy than Euler time scheme. Euler time scheme is first order scheme, while backward is second order scheme. The result is as expected. In helium plume 3-D case, limitedLinear <0.2> brings less numerical dissipations into simulation but provides a better representation of turbulent fluctuations. In pool fire case, as value of *nOuterCorrectors* increase, the root mean square (rms) of instantaneous relative error decrease. And the result shows that rms will converge at around *nOuterCorrectors* equals 5.

Future Work

There are other time schemes available in OpenFOAM library, i.e. *Crank-Nicolson* time scheme, more tests may be done in the future to assess the effects of other time schemes on species mass conservation of FireFOAM. This project has studied limitedLinear <0.2> and limitedLinear <1.0> performance on turbulent flow and species mass conservation statement. There are also other coefficients that can be chosen for limitedLinear <coefficient> convection scheme. More studies may be done to assess the other coefficients' effects on species mass conservation. Also, studies can extend to check other convection schemes' effects on the accuracy of species mass conservation statement. In pool fire case, the fire plume looks more like an unsteady laminar flow. As stated before, it is likely due to air co-flow. Future studies for this case may remove air co-flow and simulate a more turbulent fire problem. Also, we have studied *nOuterCorrectors* equals 2, 3,4,5 and 10. Future studies may assess more values of *nOuterCorrectors* effects on species mass conservation statement of FireFOAM.

Bibliography

[1] NASA. *Overview of CFD Overview of CFD Verification and Validation.* Retrieved from: https://www.grc.nasa.gov/www/wind/valid/tutorial/overview.html

[2] Ding, Y. M., Wang, C. J., & Lu, S. X. (2014). *Large eddy simulation of fire spread.* Procedia engineering, 71, 537-543.

[3] Wang, Y., Chatterjee, P., & de Ris, J. L. (2011). *Large eddy simulation of fire plumes.* Proceedings of the Combustion Institute, 33(2), 2473-2480.

[4] Maragkos, G., Rauwoens, P., & Merci, B. (2012). *Application of FDS and FireFOAM in large eddy simulations of a turbulent buoyant helium plume*. Combustion Science and Technology, 184(7-8), 1108-1120.

[5] Verma, S. (2019). *A Large Eddy Simulation Study of the Effects of Wind and Slope on the Structure of a Turbulent Line Burner* (Doctoral dissertation). University of Maryland, College Park, MD

[6] Vilfayeau, S. (2015). *Large eddy simulation of fire extinction phenomena* (Doctoral dissertation). University of Maryland, College Park, MD

[7] Li, H. (2017). *Development of a Suite of Verification Tests for the CFD Fire Model FireFOAM (Master dissertation).* University of Maryland, College Park, MD

[8] Córdoba, S. V. *Assessment of the Fire Model FireFOAM and Development of a User's Guide. (Master Thesis).* University of Maryland, College Park, MD

[9] OpenFOAMWiki. *OpenFOAM guide/ The PIMPLE algorithm in OpenFOAM*. Retrieved from: https://openfoamwiki.net/index.php/OpenFOAM_guide/The_PIMPLE_algorithm_in_OpenFOAM

[10] Wikipedia contributors. (2019, April 6). OpenFOAM. In *Wikipedia, The Free Encyclopedia*. Retrieved 20:19, April 12, 2019, from https://en.wikipedia.org/w/index.php?title=OpenFOAM&oldid=891261049

[11] The OpenFOAM Foundation. *OpenFOAM User Guide Version 6*. Chapter 1 Introduction. July 10th ,2018.

[12] ParaView. Retrieved from: https://www.paraview.org/

[13] The OpenFOAM Foundation. *OpenFOAM User Guide Version 6*. Chapter 4 Introduction. July 10th ,2018.

[14] OpenCFD Ltd. *OpenFOAM: User Guide.* Time Schemes. Retrieved from: https://www.openfoam.com/documentation/guides/latest/doc/guide-schemes-time.html

[15] Moukalled, F., Mangani, L., & Darwish, M. (2016). The finite volume method in computational fluid dynamics. *An advanced introduction with OpenFoam® and Matlab®. Nueva York: Springer. Recuperado de http://www. gidropraktikum. narod. ru/Moukalled-et-al-FVM-OpenFOAM-Matlab. pdf*.

[16] OpenFOAM. Retrieved from: https://openfoam.org/

[17] FM Global. *Open Sources Fire Modeling*. Retrieved from: https://www.fmglobal.com/research-and-resources/research-and-testing/theoretical-computational-and-experimental-research/open-source-fire-modeling

[18] Holzmann, T. (2016). *Mathematics, numerics, derivations and OpenFOAM®*. Loeben, Germany: Holzmann CFD.