KIT SCIENTIFIC REPORTS 7670

# Local Mesh Refinement in COM3D for Combustion Simulation

Ke Ren

KIT Scientific Publishing

Ke Ren

**Local Mesh Refinement in COM3D for Combustion Simulation**

Karlsruhe Institute of Technology
KIT SCIENTIFIC REPORTS 7670

# Local Mesh Refinement in COM3D for Combustion Simulation

by
Ke Ren

KIT Scientific Publishing

Report-Nr. KIT-SR 7670

Dissertation, Karlsruher Institut für Technologie
Fakultät für Maschinenbau, 2014
Referenten: Prof. Dr. Andreas G. Class, Prof. Dr. Xu Cheng

**Impressum**

Karlsruher Institut für Technologie (KIT)
KIT Scientific Publishing
Straße am Forum 2
D-76131 Karlsruhe

KIT Scientific Publishing is a registered trademark of Karlsruhe Institute of Technology. Reprint using the book cover is not allowed.

www.ksp.kit.edu



Print on Demand 2014

ISSN 1869-9669
ISBN 978-3-7315-0225-8
DOI: 10.5445/KSP/1000041147

# Local Mesh Refinement in COM3D for Combustion Simulation

Zur Erlangung des akademischen Grades

**Doktor der Ingenieurwissenschaften**

der Fakultät für Maschinenbau
Karlsruher Institut für Technologie (KIT)

genehmigte

**Dissertation**

von

M. Sc. Ke Ren

aus Jiangsu

Tag der mündlichen Prüfung: 24. Januar 2014
Hauptreferent: Prof. Dr. Andreas G. Class
Korreferent: Prof. Dr. Xu Cheng

## Acknowledgements

I would like to express my gratitude to all those who have helped me during the writing of this thesis.

I gratefully acknowledge the help of my supervisor Prof. Dr. Andreas G. Class and Prof. Dr. Xu Cheng. I do appreciate their patience, encouragement, and professional instructions during my thesis writing.

Also, I would like to thank Dr. Alexei Kotchourko, Dr. Alexander Lelyakin and Dr. Zhanjie Xu, who gave me considerable help in programming, combustion simulation and code verification. Without their consistent and illuminating instruction, this thesis could not have reached its present form.

Last but not the least, my gratitude also extends to my family who have been assisting, supporting and caring for me all of my life.

# Kurzfassung

Der Konflikt zwischen Auflösung und Aufwand von rechnergestützten Aufgaben ist ein lang bekanntes Problem, das dazu anregte, dass Werkzeuge entwickelt werden, die die Genauigkeit und Effizienz von Rechneranwendungen steigern. Dies gilt auch für das Verbrennungsprogramm COM3D. Dies beschränkt die Anwendbarkeit von COM3D bei großskalige industrierelevanten Problemstellungen. Speziell industrielle Aufgaben, die lokal begrenzte eine hohe Auflösung verlangen, wie eine Gaslekage an einem Auslass in einen großen Behälter oder eine Detonationswelle in einem großen Tunnel, führen dazu, dass eine hohe Auflösung für das gesamte Rechengebiet angewendet werden muss. Das führt wiederum zu einem sehr hohen Rechenaufwand. Zur Überwindung dieses Problems wird die fortschrittliche Technik des „patch-based“ lokale blockstrukturierten Gitterverfeinerung (LMR) in das Verbrennungsprogramm COM3D implementiert. „Patch-based“ lokale blockstrukturierten Gitterverfeinerung bedeutet die lokale Verdopplung der Gitterdicht und geht einer mit hängeden Knoten, Mehre „patch-Level“ erlauben höhere Verfeinerungsgrade

Gegenstand dieser Arbeit ist die Implementierung eines effizient arbeitenden Systems zur lokale Gitterverfeinerung (LMR) für die Löser der Euler- und Navier-Stokes-Gleichungen sowie für die Berechnung von Detonationen. Das Hauptaugenmerk liegt hierbei auf der zeitlichen Diskretisierung für Räumlich verfeinerte Bereiche des Rechengitters sowie auf der mathematischen Analyse des “patch-based LMR“ Ansatzes bezüglich der Genauigkeit von Ergebnissen. Außerdem wird eine neue verbesserte Methode zur „patch-based“ Gebietszerlegung entwickelt um parallelisierte Rechnungen mit lokale Gitterverfeinerung zu optimieren. Für eine ausgeglichene Lastverteilung auf die einzelnen parallel ablaufenden Teilrechnungen berücksichtigt diese Methode, gegenüber herkömmlichen Methoden zur „patch-based“ Gebietszerlegung zusätzliche Eigenschaft. Es handelt sich hierbei z. B. um die Komplexität des Rechengitters, um die Effizienzsteigerung der adaptiven lokale Gitterverfeinerungen und um die Optimierung des Datenaustausches innerhalb der COM3D-Datenstruktur.

Mehrere Simulationen, einschließlich der Detonationinitiierung, welche unter dem enormen Rechenaufwand leiden der mit einer gleichförmigen Rechengitterstruktur einhergeht, werden mit der hier entwickelten LMR-Technik berechnet und werden so die Vorteile der Methodik nachgewiesen. Die Anwendung der LMR in COM3D verspricht eine sehr viel höhere Leistungfähigkeit und erlaubt umfassende Studien zur Physik in laufenden großskaligen Projekten wie der systematisch Analyse des Wasserstoffrisikos bei schweren nuklearen Störfällen und der Aufarbeitung einer Validierungs-&Verifizierungsdatenbasis (V&V).

**Keywords:** lokale Gitterfeinerung, time discretization, ghost Region Berechnung, operator-splitting, hybrid Interpolation, Lastenverteilung

# Abstract

Conflict between resolution and total computational effort is a long term issue challenging scientists who are committed to develop computational tools with concerns of both accuracy and efficiency. Such conflict in the combustion code COM3D largely restrains the software application in large-scale industry relevant simulations. Especially for the industrial problems requiring fine resolutions locally, such as the gas leakage from small nozzle in large containment and detonation wave in large tunnel, the whole computational domain has to be cover with the fine resolution which results in a very high computational effort. In order to optimize the code and to alleviate the conflict, the advanced technique of patch-based block-structured local mesh refinement (LMR) is implemented in the code. The patch-based block-structured local mesh refinement means that locally the grid density is doubled resulting in hanging nodes, multiple patch levels allow higher refinement ratio.

The work is to implement the robust and effective LMR in the solution of Euler equations, the solution of Navier-Stokes equations and the simulation of detonation. Most effort of the implementation is devoted to the temporal refinement in LMR and the mathematical analyses for the patch-based LMR approach. In addition, an advanced patch-based domain decomposition method is developed to optimize the parallel computation with local mesh refinement. The novel method considers more properties than usual approaches: geometrical complexity in computational domain, efficiency improvement in adaptive LMR and optimization of data communications within the COM3D data structure.

Several simulations, including detonation initiation, suffering from the huge computational efforts accompanied by the uniform structured grids are calculated with the newly developed LMR technique, which manifests the advantages of the technique. By the implemented LMR in COM3D, much higher working efficiency and more comprehensive studying of physics are envisioned in ongoing large-scale projects such as systematically analysis for hydrogen hazard in nuclear sever accidents and review of validation and verification (V&V) experimental database.



**Keywords:** local mesh refinement, multiple time steps, ghost region calculation, operator-splitting, hybrid interpolation, load balancing

# Contents

# List of Figures

# List of Tables

## Symbols

| **Symbols in analyses** | |
|---|---|
| T | **Time** |
| $x, y, z$ | **Coordinate axis** |
| $\Delta x, \Delta y, \Delta z$ | **Grid scale in each direction** |
| $\Delta h$ | **Grid scale of the Cartesian grid** |
| $\Delta t$ | **Time scale** |
| $O(\Delta^2)$ | **Second order small terms** |
| $x, u$ | **General variables (scalar or vector)** |
| $f(u), f(x)$ | **Normal maps** |
| $f_x, f_u, f'$ | **First order derivative of the map** |
| $f_{xx}, f_{xxx}$ | **Second, third order derivatives of the map** |
| $u_n, x_n$ | **Results at the step n** |
| $u_i, x_i$ | **Results at the intermediate phase i** |

| Symbols in governing equations | |
|---|---|
| $\rho$ | **Density** |
| $u, v, w$ | **Velocity in X, Y, Z direction (in governing equations)** |
| $p$ | **Pressure** |
| $e$ | **Total energy** |
| $T$ | **Temperature** |
| $Y_i$ | **Density of gas component i** |
| $R$ | **Constant in the ideal status equation** |
| $C_v$ | **Constant- volume specific heat capacity** |
| $C_p$ | **Constant- pressure specific heat capacity** |
| $c$ | **Sound speed** |
| $M_i$ | **Molecular mass of gas specie** |
| $\mu$ | **Viscosity** |
| $\lambda$ | **Thermal conductivity** |
| $D_i$ | **Diffusion coefficient of gas components i** |
| $H_i$ | **Heat capacity of gas components i** |
| $k$ | **Chemical reaction rate** |
| $E_i$ | **Energy released by chemical reaction** |
| $K_{ch}$ | **Pre-exponential factor** |
| $E_{ch}$ | **Activation energy** |

# 1. Background

On March 11, 2011, the tsunami in Fukushima knocked out backup power systems that were needed to cool the reactors and caused core damage, hydrogen explosions, and radioactive releases. Radioactive contamination from the Fukushima plant forced the evacuation of communities up to 25 miles away, affecting up to 100,000 residents, many of whom remain indefinitely barred from their homes. Complete decommissioning and dismantlement of the plant is expected to take 40 years, and the total cost of the disaster was recently estimated by a Japanese government committee to exceed $75 billion. The impact of the Fukushima nuclear accident not only monomer present environmentally and economically, and its impact on nuclear safety is enormous.

Since the operation of the first nuclear power plant in Russia, the safety issue of the power plants has been concerned a lot by the public and the governments. Nuclear safety is indeed the basis for construction and operation of nuclear power plants, and dominates its development and innovation. Among the safety issues, the most influential one is severe accidents: Three Mile Island accident in 1979 taught people the possibility of sever accidents and the Chernobyl accident in 1986 had greatly strengthened the concerning for large range of emergency responds. Similarly, the Fukushima nuclear accident in 2011 had a great impact to the world nuclear industry.

Fukushima has prompted a reexamination of nuclear plant safety requirements around the world. Lots of countries have reduced their speed in construction of nuclear power plants and enhanced the safety level of the power plant. Taking the Chinese nuclear industry as an example, the nuclear power plant which has the core damage frequency (CDF) less than $10^{-4}$ per reactor-year is still acceptable before the Fukushima nuclear accident, but the requirement of CDF has been set to 10-5 and large release frequency (LRF) has been set to $10^{-6}$ after the accident.

The nuclear safety is related with the quality of design and equipment, arrangement of system, prevention and mitigation of severe accident and so on. Preventing the release of radioactivity is the most important task in the nuclear safety, and the key of preventing radioactive releases is to ensure the integrity of nuclear containment. During the sever accidents hydrogen is the most threatening to the integrity of nuclear containment, because both hydrogen explosion and containment direct heating (CDH) can cause huge pressure loads and then break the containment. Fukushima nuclear accident is the typical radioactive releases related with the hydrogen explosion, and the explosion has triggered more attention from the official side and the public. In the view of official nuclear authorities, the review of the hydrogen in nuclear safety must be strengthened and a high level for the evaluation of hydrogen hazard must be established. In the view of the public, unless concrete and persuasive evident about the safety of hydrogen is given, they will always be in the panic for radioactive release caused by hydrogen. Therefore, under the requirements from both government and public, more accurate and efficient codes for evaluation of nuclear safety, especially for the evaluation of hydrogen combustion damage, are required.

Normally, hydrogen safety analysis is accomplished by either lump parameter (LP) code or computational fluid dynamics (CFD) code. The former with the simplified conservation between

control volumes usually have a very fast simulation speed and good stability. The latter solves the strict conservation which is usually expressed by Navier-Stokes equations and has good ability in describing detailed phenomenon. In simulation of hydrogen distribution, the two types of codes play considerable equal importance in the nuclear industry. However, in the simulation of hydrogen combustion the situation is quite different. Because of lack of information about the local flow field and the turbulence field, and the effect of these two on the combustion rate, the lump parameter approaches have a limited capability to quantitatively predict detailed hydrogen combustion behavior. In contrast, the CFD approaches can achieve more precise local flow information so that the description of combustion is more accurate. With the more attention to hydrogen safety, the CDF based hydrogen safety codes have been largely used in nuclear safety analyses.

### 1.1. COM3D

COM3D [18] is a CFD based tool for the simulation of turbulent combustion and detonation in industry relevant problems. In the software, simulations are based on the solution of compressible Navier-Stokes equations. Several solvers and computational methods are coded to provide different accuracies in time and space. COM3D has very good performance in many industrial problems and has already been used in safety analyses for hydrogen-related industries and the severe accident analyses for nuclear power plants. Figure 1.1 shows the simulation of combustion in nuclear containment with COM3D as an example [21].

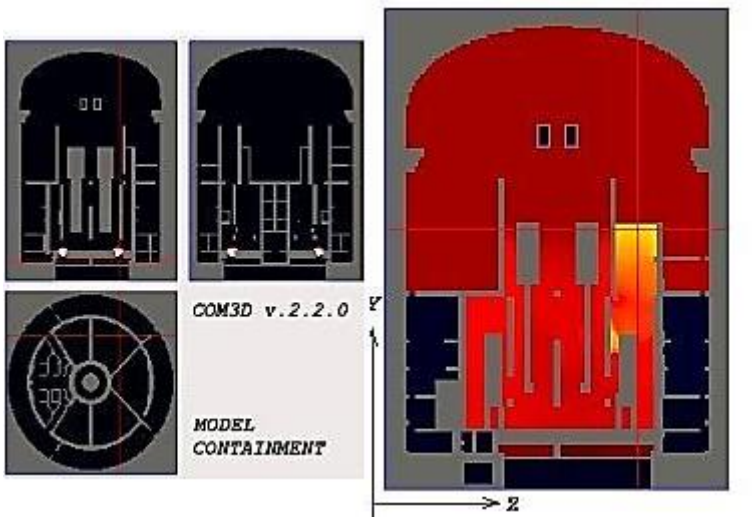



Figure 1.1 Hydrogen combustion simulation in nuclear containment with COM3D

After Fukushima nuclear accident, more and more nuclear safety regulators and nuclear power plant design institutes have the interest to use the code COM3D for the evaluation of hydrogen combustion damage. However, the current grids structure used in COM3D restricts the usage of the code in nuclear industrial problems. Since CFD based combustion simulation tools strongly depend on the gas dynamics, a proper balance between computational effort and resolution must be found. Thus, as a CFD based combustion simulation tool, COM3D often faces shortcoming from resolution and suffers a lot in practical usage.

## 1.2. Shortcoming in current COM3D

For the numerical simulation in CFD, the degree of spatial resolution and total computational effort are usually two important criteria for evaluation of the code performance. However, both items are mutually constraining factors. Normally, high-resolution means smaller-scale computational grid and smaller calculation time step, which will greatly increase the total computational effort of a numerical simulation. Therefore, the balance between the two factors is an important task.

Although parallel computation has been utilized to increase the calculation speed in COM3D, high resolution can only be achieved at the cost of a large computational effort. Moreover, since the computational grid in COM3D is currently uniform Cartesian style (equal distance in 3D), the conflict becomes even more severe in larger simulations. COM3D must use the same fine resolution in the whole computational domain to meet higher resolution requirement. Figure 1.3 shows such situation when finer resolution in required locally in the computational domain.

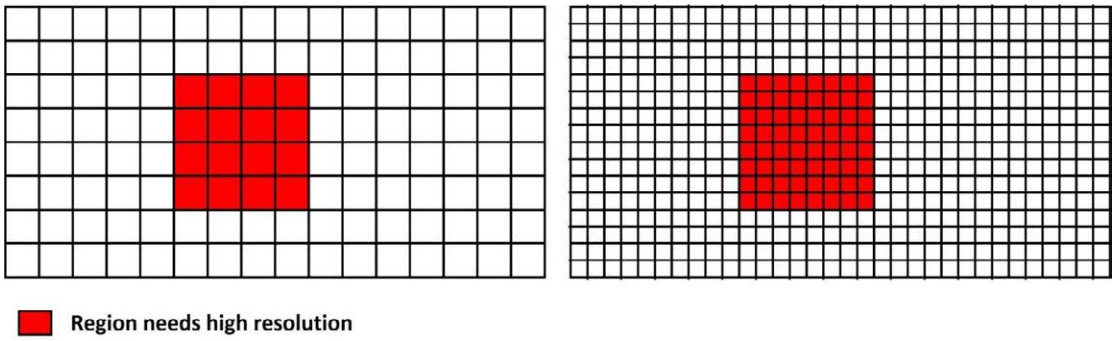



Figure 1.2 Full grids refinement in 2D

Increase of total computational effort will be extremely fast under the uniform Cartesian style of grids, the Table 1.1 shows the situation of total computational efforts with different refinement ratios in computational domain.

Table 1.1 Relation between refinement ratio and total computational efforts

| Refinement ratio | 2D problem | | 3D problem | |
|---|---|---|---|---|
| | Cell number (times) | Computational effort (times) | Cell number (times) | Computational effort (times) |
| 2 | 4 | 8 | 8 | 16 |
| 4 | 16 | 64 | 64 | 256 |
| 8 | 64 | 512 | 512 | 4096 |

Huge computational cost for high resolution may bring lots of restrictions to the code COM3D, such as the restriction in industrial design process, weak performance in expressing detailed physics and the obstacle for the code development.

### 1.2.1. Restriction in design process

Design of nuclear power plant is very complicated and requires a huge amount of analyses, calculation and comparison. Therefore, the design process is also a process of optimization. In the safety analysis, the code COM3D should plays a very important role for the nuclear containment.

However, it may be impossible to achieve extended application of the code to containment design and optimization as the huge computational cost brought by the current grids style.

. In the nuclear containment, the connection between compartments, choice of the hydrogen mitigation measures and positioning hydrogen mitigation facilities can all influence the behavior of hydrogen gas. The differences in geometrical structures may have quite different influences on hydrogen combustion: the narrow structure and obstacles may enhance the mixtures of hydrogen and oxygen; large space in containment can prevent the overlapping of pressure wave; proper mitigation equipment can largely increase the ignition temperature of the fuel and prevent the containment from destruction. For CFD code COM3D, the hydrogen safety analysis requires considerable fine resolution to construct those detailed and local geometric structures, and the resolution may result in quite large computational cost. Therefore, all the design options cannot be evaluated and the final design may not be the most optimized option.

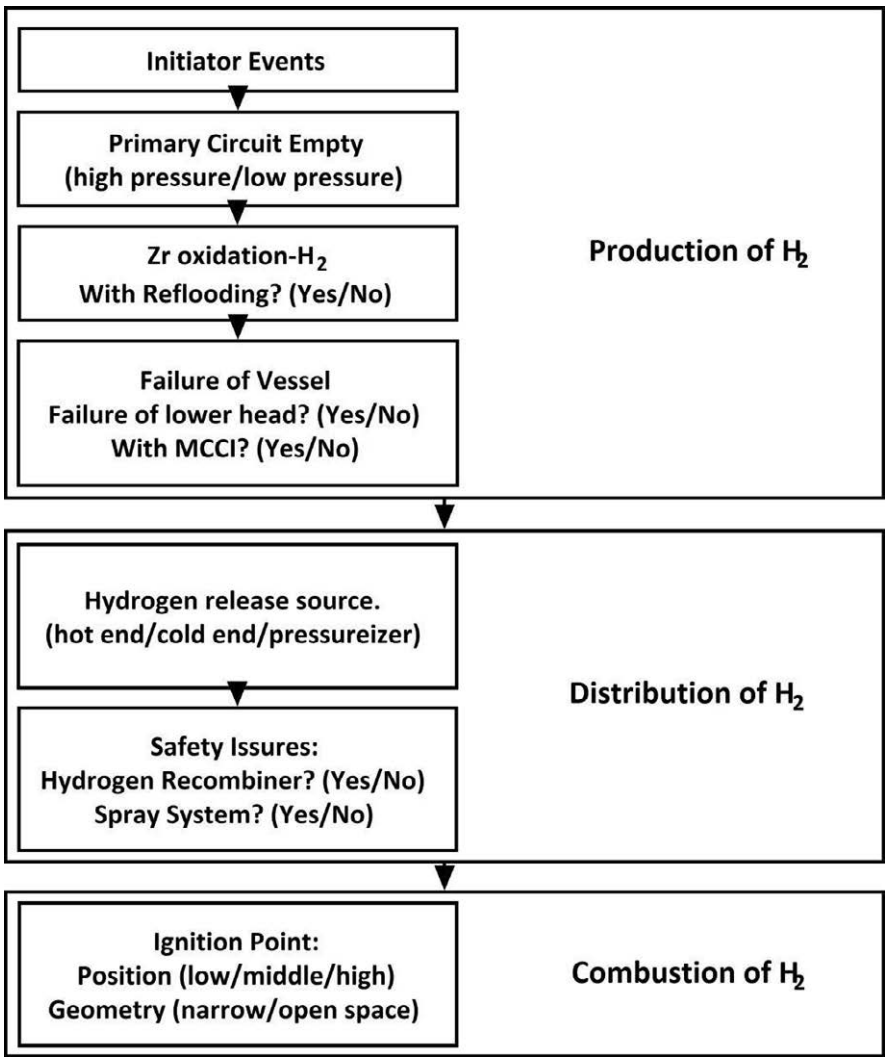



Figure 1.3 Dominate factors of hydrogen combustion scenarios

Even in the evaluation of hydrogen safety for exiting nuclear power plants, huge computational efforts brought by the current grids hinder the comprehensive safety simulations heavily. The Figure 1.3 shows the possible factors that may dominate hydrogen distribution and combustion

behaviors. Their combination can result in hundreds different scenarios. Comprehensive numerical studying of all the scenarios may requires several years. Support of engineering design can only be provided by a few generic scenarios even though it would be desirable to study more cases.

So, to meet the design requirement and optimize the design process of the nuclear power plants, calculation efficiency of the CFD code COM3D must be improved.

### 1.2.2. Restriction in detail description

Current grids structure in COM3D restricts the expression of necessary details in large industrial problems. Sometimes detailed description of small-scale boundary is necessary in large-scale industrial problems, such as the simulation of small LOCA in nuclear containment and some small scale hydrogen recombiners in hydrogen related industrial problems. Establishment of small-scale boundary usually requires high resolutions, but the huge computational cost brought by the resolution might be unacceptable, especially under the uniform Cartesian grids structure. Normally senior users build equivalent boundary conditions in the computational domain to envelope the real accident conditions, but such treatment sometimes may get the results far different from the reality.

The compromise between resolution and total computational efforts may lead to the loss of real physics in industrial relevant simulations. For CFD codes, especially for the CFD based combustion codes, resolution used in the computational domain is an important factor dominating the accuracy of the final result. In the simulation of detonation and diffusion with the code COM3D, resolution should be fine enough, otherwise the parameter as flame speed, temperature, pressure and distribution of gas spices might be far different from the reality. Figure 1.4 shows the importance of resolution in one-step Arrhenius detonation simulation, it is clear that under coarse resolution pressure of detonation wave cannot be accurately predicted. Commonly, to avoid losing important physical phenomenon in the simulation, analysis for resolution sensitivity must be done before the real simulation task. Unfortunately, most users rarely make the sensitive analysis for the current grid structure make the analysis quite expensive. In this situation, the real physics can be lost quite easily by non-professional code users and then even thread the safety of the nuclear containment.

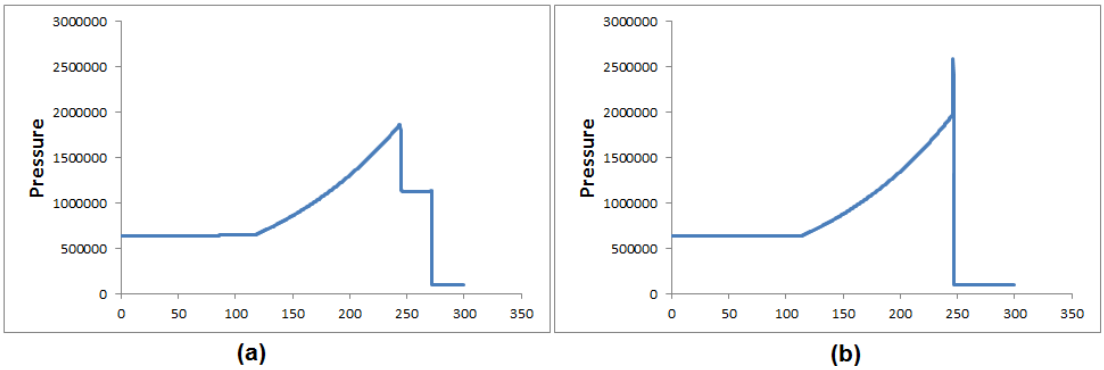



Figure 1.4 1D detonation simulation with resolution 0.1m (a) showing unphysical step structure and finer resolution 0.001m (b) showing proper physical behavior

Over all, current grid structure has severely hampered the description of detailed physics, the improvement of grid structure in code COM3D is quite necessary.

### 1.2.3. Restriction in code development

Besides the limitation in practical application, current grid structure also restricts the improvement of the code. As the development of numerical schemes, physical models and computer technology, the code COM3D should improve itself as well to keep its excellence in hydrogen combustion simulation. However, the uniform grids structure may impede the development of the code explicitly or implicitly.

More and more new physical or chemical models are developed to describe the local details, but current uniform grid structure in COM3D can hardly satisfy the application condition of those new models. Take the detonation chemical model in COM3D as an example. Although the chemical models have been improved a lot after decades of research in detonation wave, COM3D still uses the simple one-step Arrhenius model or parameter method to simulate the chemical behavior in detonation. Although new chemical models are more accurate and can give more detailed physics in simulation than the one-step chemical approach, huge computational cost with the uniform grid structure prevents the implementation of the new models in COM3D. Similarly, the implementations of new turbulent models and combustion models is also restricted by current grids structure. Unless the grid structure is improved, COM3D will lose its advantages in combustion simulation in future without the new models.

In addition, current grid structure may neglect the improvement of the code implicitly. In industrial simulations, the detailed numerical work often has to compromise on the control of total computation efforts. Senior code users always avoid using those advanced models in large-scale industrial problems, especially in the cases that only the uniform structured grid is available. In this situation, those models which require high resolutions in the calculation might be ignored by the users in industrial area. Currently, some of the physical models and numerical schemes in COM3D, such as the LES turbulent models, high accuracy RK algorithm and the one-step Arrhenius detonation model, are facing the embarrassment of neglect by the users. The embarrassment will be continued and even extended to some other new implemented models, unless the grid structure of COM3D has been improved.

For the existing models in COM3D, current grid structure also restricts their improvements. Parameter modification, verification and validation are quite necessary tasks in the code development, and huge amount of computations should be made for supporting the tasks. Such as the studying of detailed detonation behaviors, including the simulation of cellular structures, researches about the regeneration criteria and modification of pre-exponential factors, very high resolutions and repeated calculations are required. However, the huge computational efforts brought by the current grid structure limit the amount and quality of the computations, and then impedes the scientific studying and improvement of the models. Similarly, in the verification and validation (V&V) process, large amount of computational cost caused by the grids may also become one big difficulty for the testing of current models.

## 1.3. Overview of the thesis

After the discussion about the shortcoming of COM3D, there is no doubt that the current grid used in COM3D should be improved. In the thesis, the technique of local mesh refinement [4] [8] [9], in which finer resolution can be approached locally instead of the full computational domain, is introduced.

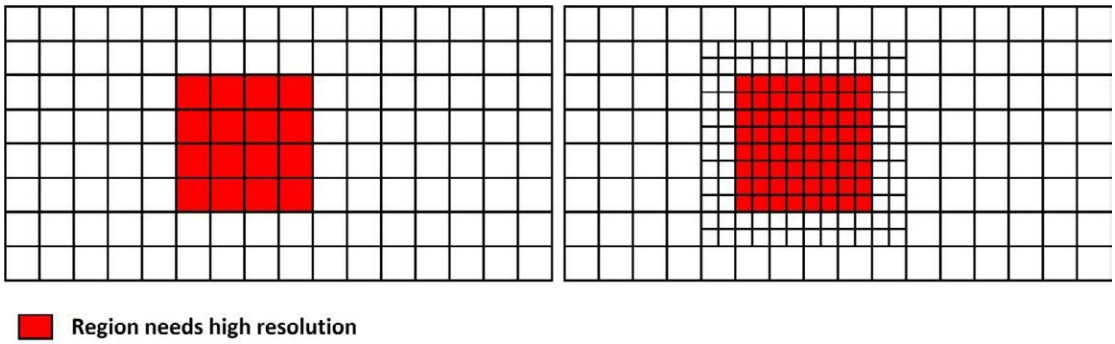



Figure 1.5 Local mesh refinement in 2D

Figure 1.4 shows one type of local mesh refinement to achieve finer resolution locally in the example which has been shown in Figure 1.2. In above figure, finer resolution just covers the region where smaller cell size is needed, so total cell number is much less than the former fully refined one. Consequently, the total computational efforts will be much less as well.

In this thesis, numerical technique of local mesh refinement (LMR) is implemented in COM3D. In following chapter, the basic algorithm of LMR and some of previous applications of LMR are described. In Chapter 3, implementation details for basic gas dynamics and chemical reaction are presented. This Chapter is devoted to the application of multiple time steps method in all numerical schemes and detonation simulation by means of time-operator splitting. In order to optimize the parallel computation with LMR in complex industrial problems, a new domain decomposition method is developed in Chapter 4. Finally, applications with the newly developed LMR methods are presented in Chapter 5. Chapter 6 concludes the main results of this thesis..

# 2. Review of combustion codes, LMR and load balancing

In the thesis the technique of local mesh refinement is introduced to the industrial hydrogen combustion code COM3D. Chapter 2 is focused on the review of combustion theory, overview of CFD based industrial combustion codes, the local mesh refinement technique and the decomposition algorithms for the computational domain with local mesh refinement.

## 2.1. Basics of physical phenomena

In general combustible gas mixture can support two combustion modes: deflagration and detonation according to the hydrodynamics of flame propagation. These two modes of combustion can be easily distinguished from one another, by the velocity, structure and mechanism of propagation of the reaction front.

### 2.1.1. Deflagration

"*A deflagration is the most common mode of flame propagation in accidental gas explosions. It is defined as an explosion where the combustion wave propagates at subsonic velocities relative to the unburnt gas immediately ahead of the flame (which itself is set in motion by the expanding combustion products.*" [10]. Deflagration is the combustion regime, which is due to heat diffusion – the direct transfer of heat from the burning gas to the fresh fuel, which is still unburned. Typical deflagrations propagate at speeds on the order of 1-100 m/s.

In deflagration the propagating velocity can span more than three orders of magnitude. The mechanism of flame propagation will be quite different in the different velocity regimes.

Laminar flame:

When the fuel cloud is ignited by a weak ignition source (i.e. a spark or a hot surface) the flame starts as a laminar flame. For a laminar flame, the basic mechanism of propagation is molecular diffusion of heat and mass [10]. This diffusion process of heat and mass into the unburnt gas is relatively slow and the laminar flame will propagate with a velocity of the order of 0.3-4m/s. The propagation velocity of the laminar flame depends on the type of fuel and the fuel concentration. Hydrogen has higher burning velocities due to fast chemical kinetics and high molecular diffusivity, the velocity of the flame is about 1-3m/s.

The major risk, which is considered in nuclear and industrial safety, is the risk for loss of containment integrity and failure of safety systems, due to the combustion of the hydrogen produced in the course of the accident. The risk is twofold: pressure loads of various magnitudes may occur, and thermal loads. Owing to the mild chemical reactivity of the laminar flame, pressure rise and temperature rise are quasi-static, and little attention has been paid on such type of deflagration in the industrial code COM3D.

.

Turbulent flame:

In most accidental explosions the laminar flame will accelerate and transit into a turbulent flame, since the flow field ahead of the flame front becomes turbulent [10]. The turbulence is caused by

the interaction of the flow field with process equipment, piping, structures etc. One of the mechanisms causing the increased burning rate in turbulent deflagrations is the wrinkling of the flame front by large turbulent eddies, which increases the flame surface. Figure 2.1 shows a wrinkled flame front in the left part, the $\delta$ in the figure denotes the thickness of the flame.

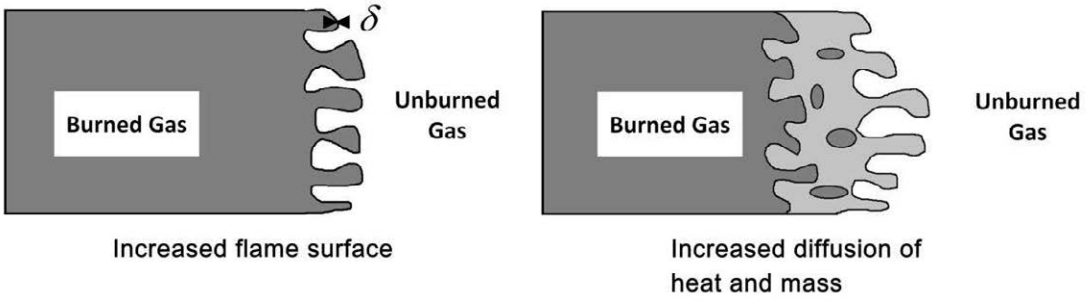



Figure 2.1 Wrinkled flame front & Flame brush

Differently from the laminar flame, the pressure built-up caused by the turbulent flame cannot be ignored. In its propagation path, a shock wave front is generated in front of the flame surface.

The effect of obstacles- or boundary-layers-induced turbulence can eventually transform the wrinkled flames into a turbulent flame brush [10]. The flame acceleration is enhanced by increase in the surface area of the laminar flamelets inside the flame brush. For sufficiently high levels of turbulence, the flamelet structure may be destroyed and then replaced by a distributed reaction zone structure. In the right part of Figure 2.1, the flame brush is shown. In this situation, the flame surface is increased, and the chemical reaction becomes more intensive. As a result, the pressure built-up caused by this more intensive combustion is higher and becomes critical issue in the hydrogen safety. With some obstacles located in its propagation path, the turbulent deflagration can be accelerated further and be sufficient to possibly lead to Deflagration to Detonation Transition (DDT), which must be avoided in the industrial design process.

The turbulent flame in an enclosure produces dynamic pressure with strong variation of time that, in some cases, may be high enough to threaten the integrity of the enclosure or its substructures. Most CFD codes put a lot of attention on the simulations of such kind of deflagration regime.

Indeed, the simulation of combustion is very complex in practice, with involvement of a large number of interactive sub-processes, including turbulent flows and chemical reactions. The mathematical description of a practical turbulent combustion process turns out to be highly non-linear and time- and space-dependent. Accurate modeling of turbulent combustion requires a proper consideration of all the important physics and chemistry, such as turbulence and chemical reactions, multiple length scales, flame-acoustic interactions.

The governing equations for the modeling of viscid flow can be expressed simply as the format in (2-1).

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \dfrac{\partial \rho u_i}{\partial x_i} = 0 \\ \dfrac{\partial \rho u_j}{\partial t} + \dfrac{\partial \rho u_j u_i}{\partial x_i} = -\dfrac{\partial p}{\partial x_j} + \dfrac{\partial \tau_{ij}}{\partial x_i} + \rho g_j \\ \dfrac{\partial \rho e}{\partial t} + \dfrac{\partial (\rho e + p) u_i}{\partial x_i} = \rho g_j u_j + \dfrac{\partial u_i \tau_{ij}}{\partial x_i} + \dfrac{\partial}{\partial x_i}(\lambda \dfrac{\partial T}{\partial x_i}) \\ \dfrac{\partial \rho Y_j}{\partial t} + \dfrac{\partial \rho Y_j u_i}{\partial x_i} = \dfrac{\partial}{\partial x_i}(D_j \rho \dfrac{\partial Y_j}{\partial x_i}) \\ e = \sum_{j=1}^{N} Y_j C_{V,j} T + \dfrac{1}{2} u_i u_i \end{cases} \quad (2\text{-}1)$$

In (2-1), $\rho$ is the density, $u_j$ is the velocity in direction j, $p$ is the pressure, $g_j$ is the body force in direction j, $e$ is the total energy, $\tau_{ij}$ is the viscous stress, $T$ is temperature and $Y_j$ is the mass fraction of a certain gas component. The factor $\lambda$ is the heat conductivity, $D_j$ is the gas diffusion factor and $C_V$ is the heat capacity. In the turbulent models, extra items such as turbulent viscosity, turbulent energy and turbulent dissipation will be considered in the viscosity and diffusion. The common turbulence models include Reynolds Averaged Navier-Stokes (RANS) models and Large Eddy Simulation (LES) models.

In general, two alternative methods have been developed for the simulation of chemical reaction in turbulent combustion. The first one is based on chemical reaction modeling in order to achieve closure of the mean conserved mass fraction equations, and the second one is based on the 'forest fire' model [42]. The first approach is based on conservation equations averaged with respect to the turbulent fluctuations of the flow variables, including eddy break-up models, eddy dissipation-concept models and laminar flamelet concept. For simulating various combustion regimes in a full scale industrial building, some combustion codes use the so-called 'forest fire' model which includes a hypothesis on the starting time for the burning and a global constant representing an effective burning rate or an effective turbulent burning velocity.

### 2.1.2. Detonation

"*Detonation is defined as a supersonic combustion wave (i.e. the detonation front propagates into unburnt gas at a velocity higher than the speed of sound in front of the wave)* [10]". Contrary to the deflagration discussed above, where the propagation through the combustible gas is due to the heat transfer by thermal diffusion, detonation is entirely different regime of propagation of combustion, involving shock waves. The shock wave traveling through the combustible mixture compresses and heats the gas behind the shock front. If the shock wave is sufficiently strong, then the rise in temperature may be sufficient to ignite combustion behind the shock front. Thus, detonation waves are shock waves which are sustained by the energy of the chemical reaction that is initiated by the shock compression and heating. In fuel-air mixtures at atmospheric pressure, the detonation velocity is typically 1500-2000m/s and the peak pressure is 15-20bar.

An actual detonation is a three-dimensional shock wave followed by a reaction zone. The leading shock consists of curved shock segments. At the detachment lines between these shock segments,

the shock wave interacts in a Mach Stem configuration. A simplified illustration of the actual detonation structure is given in part a) of Figure 2.2, in which the shock wave and the chemical reaction zone are attached together and propagate towards the unburned gas mixture. In experiments of detonation, the visible fish shell pattern or cellular structures (shown in part b) of Figure 2.2) caused by overlapping of detonation shock wave front are one key characteristic of the burning phenomenon. The size of the fish shell pattern generated by the triple point (Mach stem) of the shock wave is a measure of the reactivity of the mixture representing a length scale characterizing the overall chemical reaction in the wave. This length scale $\lambda$ is often referred to as the cell size or the cell width. The more reactive the mixture, the smaller the cell size is.

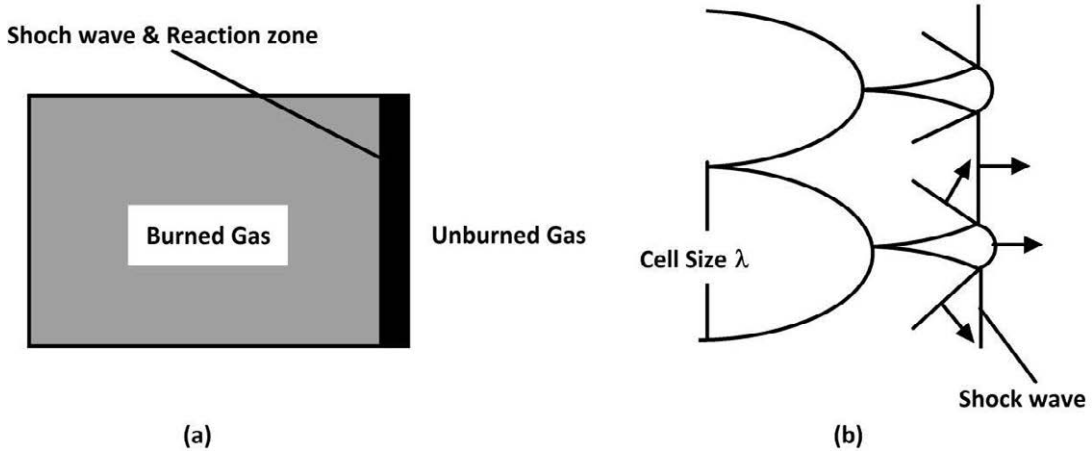



Figure 2.2 Actual structure of a detonation front & Detonation cells

The cell size is measured experimentally and there are some variations in the reported results. A variation of a factor of two is not uncommon. Figure 2.3 show the cell size of hydrogen-air detonation under different environments.

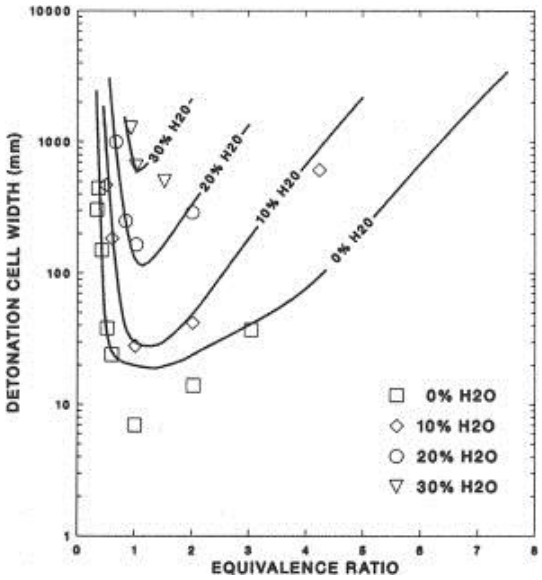



Figure 2.3 Detonation cell width under different initial conditions

The detonation cell size or width ties together the chemical reaction rate. The further a mixture is from stoichiometric (the less energetic the chemical reaction) the larger is the detonation cell size.

Due to the very short time scales of a detonation, diffusive processes do not have to be taken into account and only the Euler equations are solved. The Euler equation with the consideration of different gas components is given in (2-2).

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \dfrac{\partial \rho u_i}{\partial x_i} = 0 \\ \dfrac{\partial \rho u_j}{\partial t} + \dfrac{\partial \rho u_j u_i}{\partial x_i} = -\dfrac{\partial p}{\partial x_j} + \rho g_j \\ \dfrac{\partial \rho e}{\partial t} + \dfrac{\partial (\rho e + p) u_i}{\partial x_i} = \rho g_j u_j \\ \dfrac{\partial \rho Y_j}{\partial t} + \dfrac{\partial \rho Y_j u_i}{\partial x_i} = 0 \\ e = \sum_{j=1}^{N} Y_j C_{V,j} T + \dfrac{1}{2} u_i u_i \end{cases} \tag{2-2}$$

In the chemical reaction modeling, one-step Arrhenius reaction and Heaviside-function model are two common methods. The one-step Arrhenius reaction is one of the most popular models in the simulation of detonation, and can express the characteristics of detonation. The Heaviside-function model is more practical model for the simulation of detonation phenomenon in industry relevant problems with coarse grid calculation.

### 2.1.3. Combustion in large-scale industry relevant problems

Hydrogen production, distribution and combustion in post-accident containment are very complex and highly plant- and scenario-specific phenomena. Local high hydrogen concentrations can be reached in a short time, leading to combustible gas mixtures in the containment. Moreover, a long term pressure build-up may occur due to steam generation through decay heat and/or through the generation of non-condensable gas from the interaction of the molten core with the containment basement concrete. An understanding of hydrogen mixing, distribution and combustion is crucial for planning and implementing effective hydrogen management measures. A large number of CFD combustion codes are developed for studying the inner-containment combustion and guiding the design of containment structures.

Figure 2.4 shows the simulation of turbulent hydrogen-air-steam combustion in reactor containment under the consideration of the scenario ‘small break loss of coolant accident’ by COM3D. The grid used in this simulation is about two million and with the size $40\,cm \times 40\,cm \times 40\,cm$ [50]. Besides the simulation for the containment of PWR, CFD codes are also used to make analysis for the hydrogen combustion in BWR. In 2011, a hydrogen detonation simulation was done by COM3D for the Mark I type BWR in Fukushima.

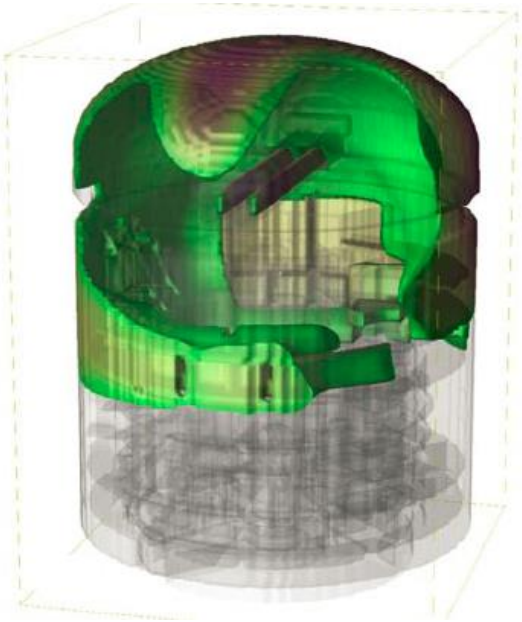


Figure 2.4 Distribution of hydrogen 0.5s after the ignition in EPR containment

Along with the increasing pressure on the environment, people pay more and more attention to the clean energy. As a green energy carrier, hydrogen becomes more and more important in the energy system. In recent years, many hydrogen facilities, such as hydrogen fuel cells and hydrogen engine, come into the sight of people. Attendant, the hydrogen safety in common civilian facilities becomes very important. Figure 2.5 shows the pressure wave resulted from detonation of 8g hydrogen in workshop cell [50].

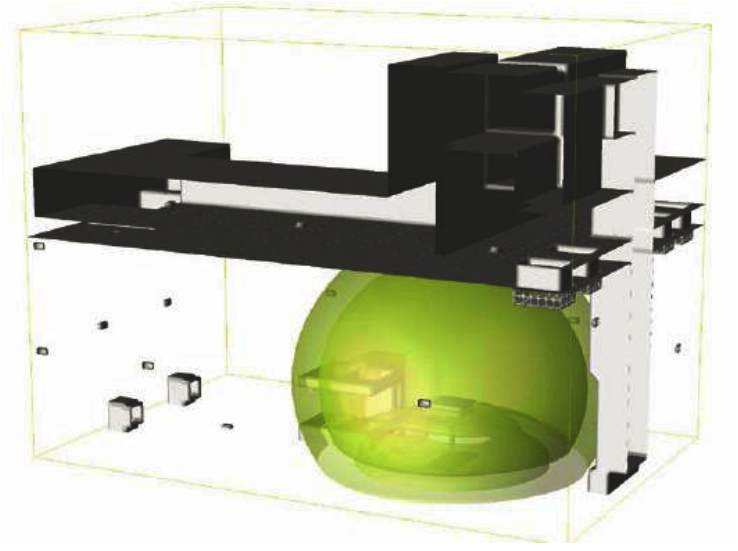


Figure 2.5 Pressure wave resulted from detonation in 160$m^3$ workshop

In the two combustion simulations above, the construction of computational domains considers the complex geometry in the containment and workshop. For the large-scale industrial problems, reliable physical can chemical models are only sufficient conditions to guarantee the credibility of the final results, a flexible grid structure is also very important to the final results [50].

## 2.2. Review of CFD combustion codes

Nowadays, CFD based combustion simulation has been a very popular methodology in combustion simulation [42]. In this section, several well-known CFD combustion codes are reviewed, including the simulation range, numerical methodology and grid structures.

### 2.2.1. COM3D

The turbulent fluid dynamics code COM3D was developed at the Institute for Nuclear and Energy Technologies (IKET) in the Karlsruhe Institute of Technology (KIT) with the aim to simulate combustion including explosion and detonation, transport, and mixing of hydrogen and other gases in nuclear reactor containments and other industrial facilities. The turbulent reactive flow code is intended to model development of large-scale combustion events in geometrically complex environment with multiple compartments and internal structures in a multi-block computational domain.

The software has been used in the simulation of combustion and explosion, together with the system of industrial risk mitigation of hydrogen and burnable gases in nuclear containment, auxiliary buildings and complex nonnuclear facilities. It has been applied for parametric study of the influence of geometrical design and of the accident scenario details on the potential risk resulting from pressure and thermal loads in the conditions of severe accident. The code was used to study combustion of the gaseous systems in wide range of the initial and boundary conditions for different combustion regimes ranging from slow deflagration to strong flame acceleration and detonation [8].

The COM3D code for can use either Reynolds Averaged Navier-Stokes (RANS) or Large Eddy Simulation (LES) method for the modeling of flow turbulence. Three turbulence models are available in COM3D in the frame of RANS approach [21]:

- standard k−ε model;
- renormalization group (RNG) k−ε model;
- shear stress transport (SST) model;
- three sub-grid for LES method:
- eddy viscosity – eddy diffusivity method;
- mixed eddy viscosity – eddy diffusivity method;
- dynamic eddy viscosity – eddy diffusivity method.

A set of combustion models is available in the code. It incorporates both classic models and the newest developments, provides necessary flexibility and robustness in the calculations. The following models are offered for the combustion [21]:

- EBU standard;
- EBU-SB;
- Hjertager model;
- Eddy-diffusivity;
- Eddy-diffusivity extended;
- KYLCOM;
- gradient (Zimont) mode.

The turbulent models and combustion models above can cover the simulations of turbulent combustion with a large range of propagation speed. For detonation simulation one of the two models can be applied: one-step Arrhenius reaction and Heaviside-function model.

The mesh used in COM3D code is rectangular Cartesian equidistant uniform multi-block simply connected domain with cell-centered variables. The uniform mesh style is more convenience in numerical work and can provide the possibility to utilize high order schemes in simulation. However, computational resources can hardly be concentrated under such kind of grid structure.

Within the HYCOM project (integral large scale experiments on hydrogen combustion for severe accident code validation) within the 5th Framework Program of the European Commission, a study has been conducted in order to investigate the accuracy of seven numerical codes that have been developed for combustion hazard assessments [42]. The code COM3D is one of them. The study includes two stages: (1) a validation stage against experimental combustion data and (2) a full scale combustion simulation in a real scale simplified modern PWR containment. The validation stage was instrumental in calibrating the combustion models against small scale and large scale experiments. It has been shown that for fast combustion regime all codes are capable of capturing the main features of hydrogen combustion, both qualitatively and quantitatively. More problems appear for slow deflagrations, but all the codes were able to capture the maximum overpressure. Overall, the validation stage was successful in achieving a quite good agreement between the simulations and the small and large scale experiments.

In practical applications, code COM3D is usually used for estimation of short period time combustion. In nuclear safety issues, such as the analysis of over pressure loads brought by hydrogen combustion in design phase and the reproduction of combustion numerically in damage evaluation after the real accident, COM3D plays a quite important role. In the non-nuclear area, such as the evaluation of hydrogen combustion hazard of hydrogen refueling station or the combustion damage brought by the leak of auto gas tank, the code COM3D has a very good performance as well, and its importance now is growing as the development of hydrogen technology.

Totally speaking, after 20 years' development and V&V, COM3D has already been developed as an experienced simulation tool for fast reacting flow. However, as mentioned in Chapter 1, the code meets its bottleneck after long time development. Improving the grid structure is quite important for code development in future. In the thesis, implementation of local mesh refinement has been focused on the detonation simulation and solution of Navier-Stokes equations.

### 2.2.2. ANSYS CFX

ANSYS CFX is a commercial Computational Fluid Dynamics (CFD) program, used to simulate fluid flow in a variety of applications [2]. It is a high-performance, general purpose fluid dynamics program that has been applied to solve wide-ranging fluid flow problems for over 20 years. In CFX, the simulation of combustion is combined by the simulation of gas dynamics and the simulation of chemical reaction.

The vast majority of industrial flows are turbulent, so ANSYS CFX software has always placed special emphasis on providing leading turbulence models to capture the effects of turbulence accurately and efficiently. For statistical turbulence models, ANSYS CFX provides numerous common two-equation models and Reynolds–stress models. However, particular focus is placed on the widely tested shear stress transport (SST) turbulence model, as it offers significant advantages for non-equilibrium turbulent boundary layer flows and heat transfer predictions. In addition, ANSYS CFX provides a number of scale-resolving turbulence models, such as large- and detached-eddy simulation (LES and DES) models.

In chemical part, ANSYS CFX provides a rich framework to model chemical reactions and combustion associated with fluid flow. The eddy-dissipation model (EDM) and finite-rate chemistry (FRC) models are provided in ANSYS CFX software for relatively fast and slow reactions, respectively, when compared to the mixing of reactants due to turbulent fluid flow. Simulations are not limited, however, to either extreme, as the two models can be combined, with the reaction rate being taken as the minimum of the two, both for single and multi-step reactions, from pre-defined or user-defined reactions.

The mesh flexibility in ANSYS CFX is its one big feature, which is lacking in COM3D. ANSYS CFX software provides complete mesh flexibility, including the ability to solve flow problems using unstructured meshes that can be generated about complex geometries with relative ease. Supported mesh types include triangular, quadrilateral, tetrahedral, hexahedral, pyramid, and prism (wedge). Figure 2.6 shows the computational domain constructed by unstructured grids in ANSYS CFX.

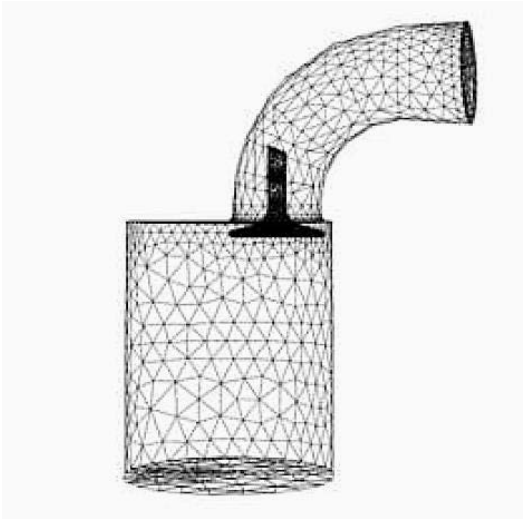


Figure 2.6 Grid structure in ANSYS CFX [2]

In the study of codes' application [8], the ANSYS CFX shows good capability in simulation of sub-sonic turbulent combustion. For the detonation simulation, the code does not pay much attention.

### 2.2.3. REACFLOW

REACFLOW is a CFD code developed at the Joint Research Centre of the European Union in Ispra, Italy. It is a parallel, finite-volume, unstructured mesh code, which may be used to model two or three dimensional geometries.

In the simulation of gas flow, the common compressible flow models such as Euler equations, Navier-Stokes equations and standard κ−ε model are available. In addition, to be better able to calculate slow-flow phenomena, REACFLOW contains a module for simulating incompressible, variable density flows.

In the simulation of chemical reaction, REACFLOW employs two methods for the calculation of the chemical source terms, the first is based on finite rate chemistry and the other is based on the eddy dissipation concept (eddy break-up model). The use of finite rate chemistry is more applicable when the influence of the turbulence on the chemical reactions is negligible. For flames that are turbulent, the eddy dissipation concept may be used.

One of the biggest features of REACFLOW is its grid structure. In studies of explosions the regions of interest are generally much smaller than the total flow domain. It is therefore advantageous to be able to concentrate the computational effort in these regions. REACFLOW has an adaptive grid capability, which allows regions of the grid to be refined or coarsened locally, depending on the local conditions. Such as the combustion simulation shown by Figure 2.7, with the help of the unstructured grid the code can concentrate the most working effort on reproduction of chemical reaction zone of the flame and the propagation of shock wave produced by the flame [90].

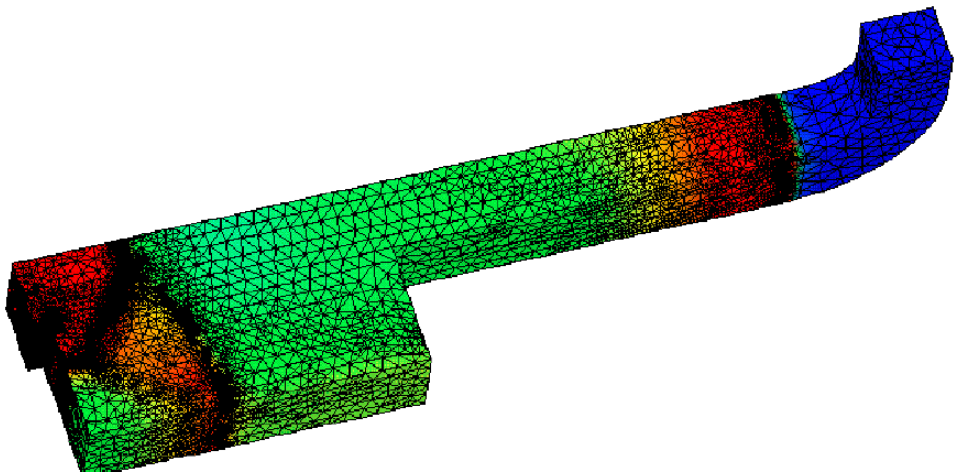


Figure 2.7 Computational domain for detonation simulation in Rut facility [90]

In the study of codes' application, it shows good capability in simulation of detonation. In simulation of deflagration, since the simple turbulent model and chemical reaction model used by REACFLOW lack the capability in closure the complicated physical and chemical processes, the code failed in may practical applications [53]. Comparing to the simulation of turbulent deflagration, REACFLOW is used more in detonation simulation.

### 2.2.4. TONUS

The TONUS code is the French hydrogen risk analysis code developed by CEA and IRSN over the last decade, to model hydrogen release, distribution and combustion in PWR reactor containment [52]. The code has both multi-compartment lumped-parameter and CFD formulations. The code is mainly used for the analysis of hydrogen safety in nuclear power plant.

In the simulation of gas dynamics, the Euler equations and Navier-Stokes equations are used. The turbulence models in the software are mixing length model and κ−ε model, the former is one algebraic turbulence model and the latter one is the standard closure two equations model. In large-scale industrial simulations, the two turbulence models are quite common.

In the simulation of chemical reaction, the TONUS provides several methods to cover a variety of combustion modes. In general, the chemical reaction in the governing equations is given as a source term, and the chemical reaction rate is used in the term to describe the consuming or producing ratio of different gas species. For laminar flame, the laminar 'Arrhenius' rate is used. For the turbulent deflagration, the 'Eddy Break-Up' rate is used. For detonation, the Arrhenius global reaction rate is used.

As a CFD combustion code emphasizing on the nuclear containment safety, TONUS uses the rectilinear grid and curvilinear grid to have good body fitting for the cylinder structured buildings. Both grid structures are compatible to the technique of local mesh refinement. In the part (a) of Figure 2.8 the computational domain of Rut facility covered by rectilinear grid is shown, and part (b) shows the use of curvilinear grid in construction of the computational domain in ENACCEF hydrogen combustion tests [10].

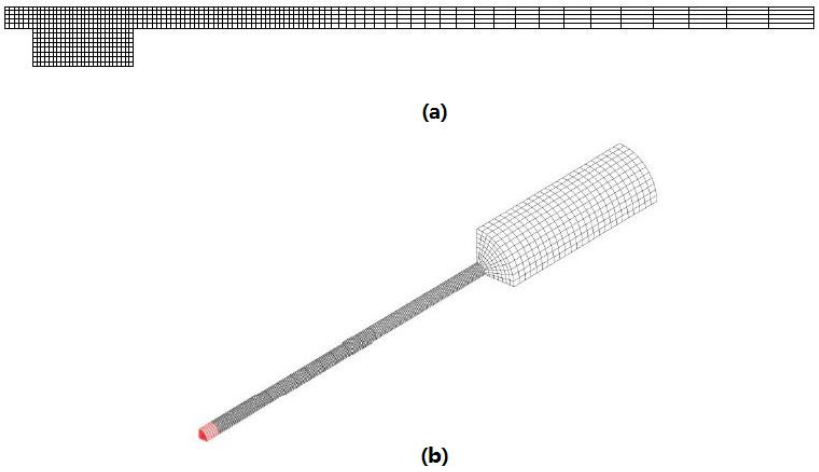



Figure 2.8 Grid structures available in TONUS [10]

In the study of codes' application, TONUS shows good capability in simulation of turbulent laminar flame and detonation. However, the simple turbulent models and simplified chemical models used by the code can hardly reproduce the complex chemical and gas flow motion and then largely restrict its application in turbulent deflagration simulations.

After the review of other CFD combustion codes, in the view of physical models, the code COM3D has the similar or better capability in combustion simulation. However, in the construction of computational domain, the mesh styles in other codes as more flexible. As illustrated in Chapter 1, the current grid structure limits the utility of the code in large-scale industrial problems. For example, the simulation of the scenario "Depressurize the reactor coolant system" or "small break loss of coolant accident" in PWR containment [91] may be very difficult under the current uniform Cartesian grid. If the 40cm cell size is used in the computational domain to save computational efforts, the coarse cell size can hardy simulate the exact structure, location and physical parameters of the gas nozzle. On the other hand, if an effetely small cell size is used to simulation the gas nozzle, very huge amount of computational efforts may be resulted from the uniform grid structure. The simulation of detonation in large-scale domain has the similar problem. Detonation simulation is cell size dependent, especially for the One-step Arrhenius detonation which requires several cells to present the chemical reaction zone. If the uniform grid is available only, simulations of decay and re-initiation of detonation wave in large tunnel usually requires very high resolution to catch the detailed physical phenomenon of detonation. There is no doubt that optimization of the grid structure has become one of the most urgent work.

## 2.3. Introduction of local mesh refinement

In order to make the numerical solution through finite volume method, the problem domain is decomposed into a serial of structured or unstructured cells which have homogeneous physical quantities in each cell. As illustrated in former chapter, the computational domain covered by smaller cells can express the gradients of quantities and physical phenomenon more detailed, but the total computational efforts may become unacceptable. In contrast, although the computational domain with larger cell sizes usually requires less computational resources, the use of large cell size may result in big numerical diffusion and even unreasonable solutions. Indeed, contradiction between the high resolution and computational saving is a long time problem that puzzles people who are committed to find efficient and accurate calculation method. The most common method used to solve the historical problem is the local mesh refinement (LMR) technique, in which different resolutions can be achieved in one computational domain, but the use of such technique should depend on detailed grid structure and calculation method used in the numerical work.

### 2.3.1. Local mesh refinement technique

#### 2.3.1.1. LMR for unstructured (irregular) grid

"An unstructured (or irregular) grid [64] is a tessellation of a part of the Euclidean plane or Euclidean space by simple shapes, such as triangles or tetrahedral, in an irregular pattern. Grids of this type may be used in finite element analysis when the input to be analyzed has an irregular shape. Unlike the structured grid, such as rectangular cell in Cartesian coordinate's space or element in cylinder coordination system, unstructured grids require a list of the connectivity which specifies the way a given set of vertices make up individual elements" (en.wikipedia.org).

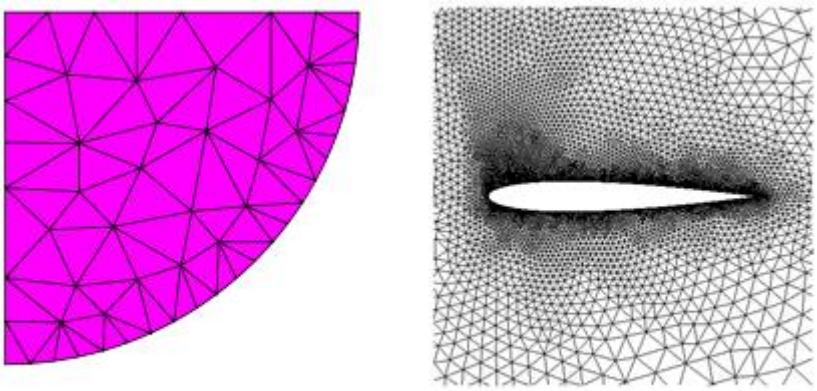



Figure 2.9 Unstructured grids example

In Figure 2.9, part (a) shows an example in which the unstructured grid is used to simulate 1/4 circular area, part (b) shows one more complicated application of the unstructured grid with local mesh refinement. As shown in the figure, unstructured grids have very good compatibility to the computational domains with complicated geometry structures, so for industrial application the unstructured grid can have excellent performances. In addition to the body fitting feature, since one edge (face) in computational domain can only be shared by at the most two cells even under the LMR, calculation of flux value of one cell is always the same no matter what resolution is used in this cell and no additional treatment is required to deal with the fluxes of the interface between two different resolutions.

However, computations with unstructured grids still have their own short comings. Firstly, no efficient mature grid generation technique of unstructured grid is available currently [70]. So, to the cases in which the area of fine resolutions need to be regenerated frequently as the change of detailed physical phenomenon, the adaptive local mesh refinement in unstructured grid calculation may cost very huge computational and communication efforts. Then, the discretization and solvers used in the calculation with unstructured grids usually cannot provide very high accuracy [13] [39], such grid structure limits the use of high accurate solvers and methods. Moreover, limited by the irregular cell sizes in unstructured grids, only the uniform time step method can be used in calculation.

#### 2.3.1.2. LMR for structured (regular) grid

"A structured (regular) grid is a tessellation of n-dimensional Euclidean space by congruent parallel topes. Grids of this type appear on graph paper and may be used in finite element analysis as well as finite volume methods and finite difference methods" (en.wikipedia.org). Since the derivatives of field variables can be conveniently expressed as finite differences, it is easier to use high order scheme under such grid style.

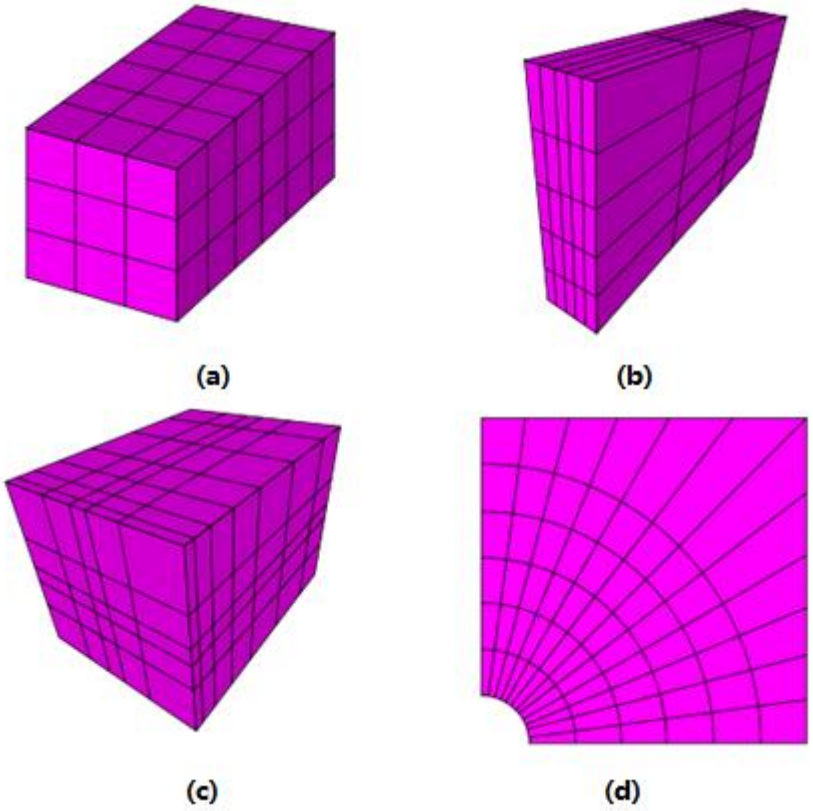



Figure 2.10 Structured grids example

In Figure 2.10, the common used structured grids Cartesian grid (a), regular grid (b), rectilinear grid (c) and curvilinear grid (d) are shown. The curvilinear grid shown in part (d) has a very special usage in cylinder or sphere computational domain, but the most commonly used grid structures are the former three. In the structured grids, the technique of local mesh refinement can be achieved by two methods.

In the rectilinear grid, similarly to the unstructured grid style, resolutions used in the computational domain can be used differently in different location but each edge can only be shared by at the most two cells. Thus, the local mesh refinement can be achieved under such grid structures quite easily and no additional treatment is required to deal with the differences in resolutions as well. Just as the two computational domains shown in Figure 2.11, discretization of the computational domain can be made compactly in certain regions in certain directions to realize the LMR.

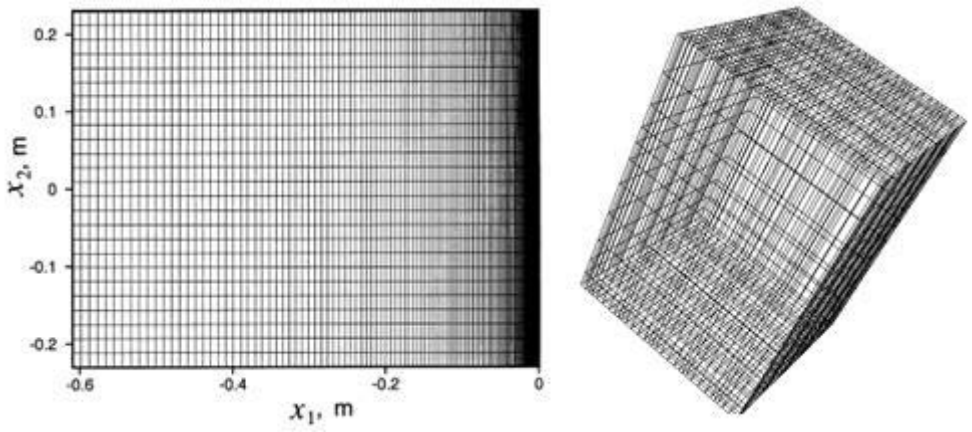



Figure 2.11 2D and 3D computational domains with rectilinear grids

However, owing to the limitation of rectilinear grids, the local mesh refinement may produce some fine regions which the high resolution is not necessary. Additionally, the same local refinement as in unstructured grids, computation with rectilinear grid requires uniform time step which is dominated by the cell with the least volume. Thus, compared to the unstructured grids, total computational efforts cannot be reduced efficiently with the local mesh refinement achieved in rectilinear grids. It is even worse that, in the simulations with adaptive LMR, the whole rectilinear grids structure has to be regenerated and results in very huge communication in data migration.

Another local mesh refinement for structured grid is achieved refining the cells in the same ratio, in which the aspect ratio of the cells can be kept [19] [20] [34] [37] [49] [94]. In a more detailed classification, such kind of local mesh refinement can be achieved by three different ways: the point-wised local mesh refinement, the patch-based local mesh refinement and the domain-based local mesh refinement. Figure 2.12 shows all the three different refinement modes to the same refinement request.

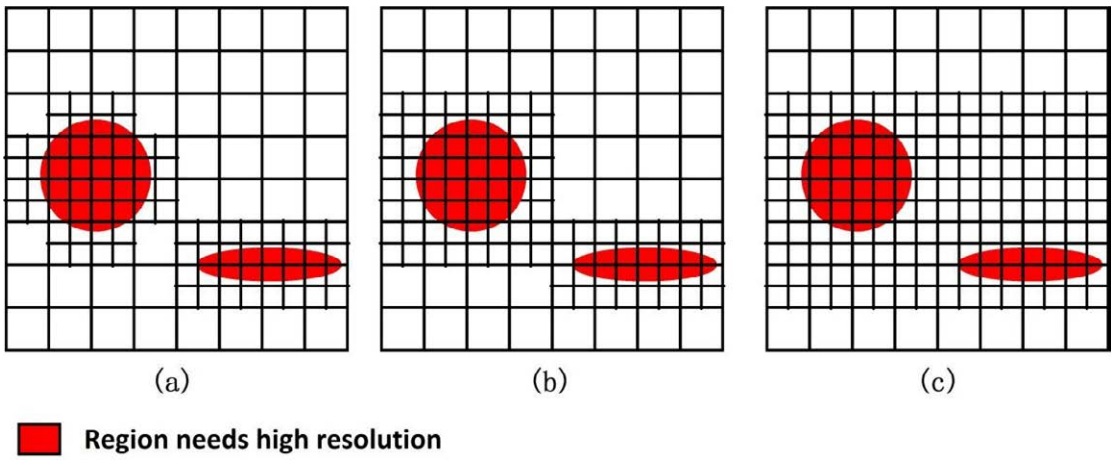



Figure 2.12 LMR under achieved by three different modes

In the above figure, the colored regions indicate the areas that need to be refined and three different structured local mesh refinement modes are used. In Figure 2.12, part (a) is the

point-wised local mesh refinement, only the cells which intersect with the colored regions are refined; part (b) shows how patch-based local mesh refinement works, the refinement is given as a series of patches, so some coarse cells which do not need to be refined may also be included in the refinement; the part (c) shows the domain-based local mesh refinement, in which the refinement is given as one big block-structured sub-domain, the minimal cover box of all the colored regions.

As shown in the figure, it is clear that the point-wised local mesh refinement makes the least refinement effort and the domain-based local mesh refinement needs the most refinement work, so point-wised approach requires less computational efforts than the domain-based approach. However, in view of practical implementation, domain-based local mesh refinement is the simplest and point-wised method is the most complicated in computation and communication. The patch-based local mesh refinement method is the hybrid of two other refinement methods. Each refinement patch in such refinement method has the similar numerical treatment to the patch with domain-based refinement. In the aspect of the total computational saving the patch-based method also has the character of the point-wised refinement method. Additionally, patch-based local mesh refinement can also be modified to emphasis more on the computational saving or computational simplification to optimize a specific simulation. Thus, as the refinement technique containing the advantages of both point-wised refinement and domain-based refinement, the patch-based local mesh refinement is the most acceptable structured refinement method.

The patch-based local mesh refinement is first introduced by Berger and Oliger in 1984 [6]. Differently from the local mesh refinement in unstructured grids or rectilinear grids, as shown in the left part of Figure 2.13, block-structured patch-based local mesh refinement may produce some un-matching cells around the fine-to-coarse interface (the edges separate the cells with different resolutions). So some additional work is required for such edges to maintain conservation. Moreover, shown by the right part of Figure 2.13, such refinement arrangement is different from the arrangements in unstructured grids and rectilinear grids. In the patch-based local mesh refinement, the finer resolution levels are given as block patches covering the coarse grid regions instead of embedding into the coarse grid. With such convenient refinement approach, arbitrary levels of refinement can be used in the simulation to reach ideal resolution locally. On each refinement level, under the assumption that enough boundary and initial conditions are provided, computation can be done individually.

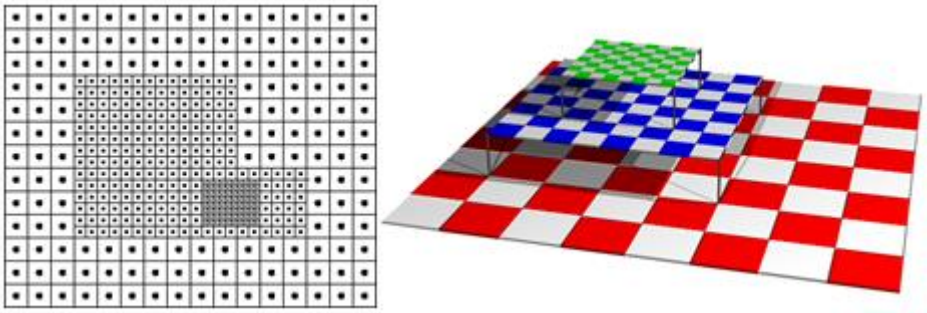


Figure 2.13 Patch-based local mesh refinement

2.3.1.3. Comparison of three LMR approaches

In section 2.3.1.1 and 2.3.1.2, three different local mesh refinement methods are introduced. Figure 2.14 shows the local mesh refinement cases refining the up-left corner of the computational domain with three different approaches.

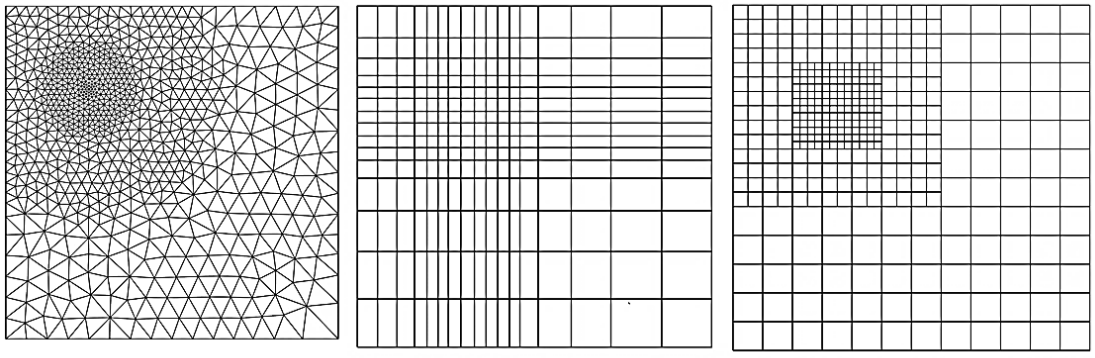



Figure 2.14 Local mesh refinement in with three different approaches

In the comparison of the three local mesh refinement approaches, the patch-based local mesh refinement is more flexible to use finer resolution adaptively. For the other two methods, if the refinement regions are modified according to the simulation needs, much more efforts should be required. In rectilinear grid work, the whole grid structure should be regenerated and all the old data must be transferred to the newly generated grids in data communication. In unstructured grid work, although the patch-based refinement is possible, some extra efforts should still be given to the construction of transition regions connecting different resolutions [59]. Additionally, in the grid regeneration, because of the possibility of non-matching cells between new and old grid structures, refinements in unstructured and rectilinear grids should also require very big efforts in the interpolation while transferring data from the old grid structure to the new. In contrast, in patch-based local mesh refinement, only the finer patches should be regenerated if the requirements of fine resolution changes and no transition region is needed at all. The data communication in the grid regeneration is also much simpler than the other two techniques.

Moreover, comparing to the other two refinement approaches, the patch-based local mesh refinement makes a simulation inherit the advantages of uniform grid computation [5] [35]. In the patch-based local mesh refinement, although different resolutions are used on different refinement levels, the grid structure on each refinement level is indeed still uniform. Thus, the patch-based local mesh refinement can optimize the solvers valid only for uniform grid structure. Meanwhile, unlike the other two refinement methods, the time steps used in patch-based refinement can also be different between different refinement levels [6] [60], with which the total computational efforts can be saved further.

However, the patch-based local mesh refinement has its own weakness. In the other two refinement approaches, no additional treatments should be made to deal with the differences in resolutions. In the patch-based local mesh refinement, cells which are adjacent to the coarse-to-fine interface have different resolutions and one edge may be shared by more than two

cells. So some additional treatments are needed to keep the basic conservation [7] [72] [86]. Additionally, in order to make calculation on finer levels successfully, some other data communications between different refinement levels should be made to provide boundary conditions in finer patches [7]. As a result, comparing to the other refinement methods, the usage of patch-based refinement requires more work in data communication between different refinement levels.

### 2.3.2. Local mesh refinement for COM3D

As illustrated in Chapter 1, contradiction between high resolution and total computational saving is still a very big problem to current COM3D. The new technique of local mesh refinement can solve basically the contradiction, although the parallelism can partially save computation time. In the detailed implementation of local mesh refinement in COM3D, selection of specific local mesh refinement method should be determined by the current numerical solvers, algorithms and grid structures. After all, the new technology is to optimize the original program rather than to build a set of new program.

Local mesh refinement for unstructured grid can express perfectly the complex geometry, especially in the industrial problems, and such a grid structure can also deal with the calculation with different resolutions easily. However, current COM3D is based on the uniform structured grid, implementation of unstructured grid local mesh refinement requires quite large modification in COM3D. For example, the data structures and detailed solvers need to be modified as the use of unstructured grids. It may make the situation even worse, some high order schemes even cannot be used after the implementation of unstructured grids. As a result, unstructured local mesh refinement is not fit for the optimization in COM3D.

Local mesh refinement in the rectilinear grid structure can also deal with the calculation with different resolution quite easily, moreover, such kind of grids contain the advantages of rectangular cells and the calculation is much simpler than that in unstructured grid. The implementation of rectilinear grids in COM3D still requires a big modification in data structure and some higher order schemes may be abandoned owing to different resolutions in the computational domain, although the rectilinear grid structure requires less modification than the unstructured grids does. Moreover, the local mesh refinement in rectilinear system is quite limited in computational saving while comparing to other two methods. Therefore, the local mesh refinement achieved by rectilinear grids is also not proper for the current COM3D.

Although the patch-based local mesh refinement method requires much more extra calculation and data communication between different refinement levels, this method indeed is the most suitable method for optimizing the current COM3D. Firstly, the refinement is accomplished by making finer levels as patches, so the grid structure on each individual refinement level is still in uniform style and the existing solvers can be used directly without any extra modification. Then, the patch-based local mesh refinement can save more computational efforts than the other two techniques for the function of the multiple time steps. Finally, as parallel computational software, the data structures and calculation in COM3D is also compatible to the extra data communication between different refinement levels required by the patch-based-local mesh refinement.

In all, after the discussion about the possibility to use the three local mesh refinement techniques in COM3D to balance the resolution and total computational efforts, the patch-based local mesh refinement is the best way to make the optimization of COM3D, because it requires the least modification in the software and can maintain the usage of all computational methods in the current code.

## 2.4. Block structured patch-based local mesh refinement

As the patch-based local mesh refinement is selected to optimize COM3D, some basic algorithms in the implementation of this technique in Cartesian grid are introduced in detail. In the sub-section 2.1, extra treatments to deal with the differences in resolutions have been highlighted as a weakness in the patch-based local mesh refinement. Thus, one of the most concentrated points in this section is to clarify the extra treatments in dealing with the differences in resolutions.

### 2.4.1. Inter-level data communication

#### 2.4.1.1. Coarse-to-fine interpolation

Boundary conditions are quite important in numerical simulation. In Cartesian grids, Dirichlet style boundary is the most frequently used and is usually stored in the non-existent areas called ghost cells. The ghost cells simplify the treatments in boundary conditions largely. Moreover, in parallel computation and computation with local mesh refinement, ghost cells play very important roles in the man-made parallel boundary or fine-to-coarse interface. Commonly in parallel computation, the ghost cells are filled by either the physical boundary data or the data of neighbors. Figure 2.15 shows the data communication in uniform grid parallel computation. The white cells shown in figure indicate the ghost cells and the gray cells are the major domain in parallelism.

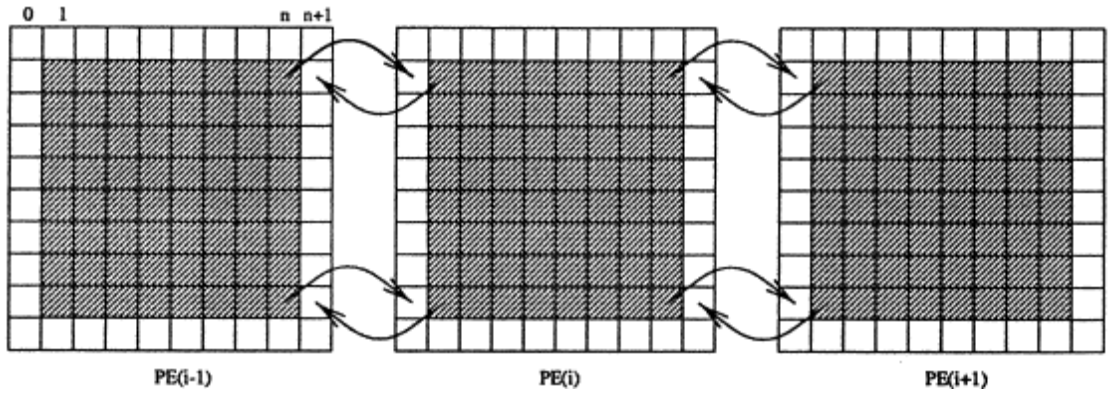



Figure 2.15 Data blocks and data communication in parallel computation

In the boundary updating of parallelism, simple data copying between two adjacent blocks is enough. However, in the boundary updating of the calculation with LMR, the situation may become more complicated.

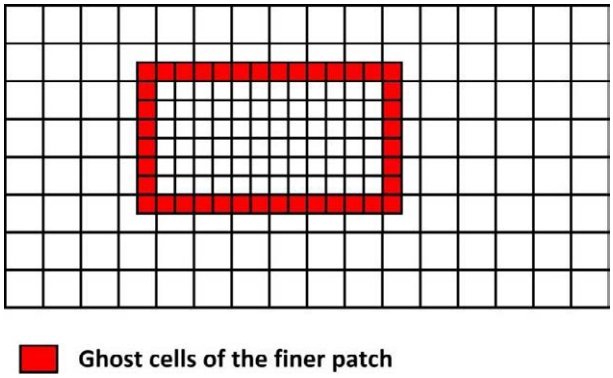


Figure 2.16 Ghost cells on finer levels in domain with patch-based LMR

In Figure 2.16, ghost cells on finer level which need special treatments are colored by gray. Differently from the ghost cells mentioned in general parallelism, the ghost cells here cannot be filled by the physical boundary values or direct data copying from other data blocks, the updating needs the help of the next coarser level [29]. Since the resolutions are different between the two levels, numerical interpolation is considered in boundary updating [33]. The data transfer to fill the ghost cells on finer level by interpolation is called coarse-to-fine interpolation.

Since some ghost values on finer level are interpolated, the efficiency and accuracy of the interpolation should be considered. According to the feature of patch-based LMR, besides the spatial interpolation, temporal interpolation should also be considered if the multiple time steps (refinement in time-scale) are used [6] [16]. When the patch-based local mesh refinement was first established, linear interpolation which can provide second order results was used in both time and space, because the used solver was only first order. Indeed, coarse-to-fine interpolation may introduce extra truncation errors and result in some unphysical reflection or oscillation [14], so higher order interpolation schemes are usually more attractive than the low accuracy schemes. In the work of [14], numerical reflection near the fine-to-coarse interface has been discussed in detail and the final conclusion shows that higher order interpolations have better ability in the reflection control than the low accuracy scheme.

In the simulation with high order schemes, truncation errors brought by the coarse-to-fine interpolation may reduce the accuracy in the final results, so choice of proper interpolation is quite important to maintain accuracy. To keep high order accuracy, third order parabolic interpolation and fifth order polynomial interpolation are used in the local mesh refinement in [61] [62] [63] and [4]. Figure 2.17 shows the details of the high order methods in 2D (part (a) is the third order parabolic interpolation, part (b) is the fifth order polynomial interpolation).

As shown in figure, high order accuracy interpolations are done by two steps. Firstly, the parabolic or five points interpolation is done on coarse level to get auxiliary data which are marked as the signal "●". Since high order interpolation method has been used in calculation of the auxiliary data, they have the third order or fifth order accuracy as well. Then, fine data marked as "X" and the auxiliary data are used in the second interpolation to get the ghost values. The final interpolated values "⊗" can also reach the third order or fifth order accuracy, because those auxiliary data and finer level data all have very high accuracy. In 3D cases, the second

interpolation phase actually is the same as the 2D case, but the auxiliary data used in the interpolation is achieved by more complicated 2D Taylor expansion [18].

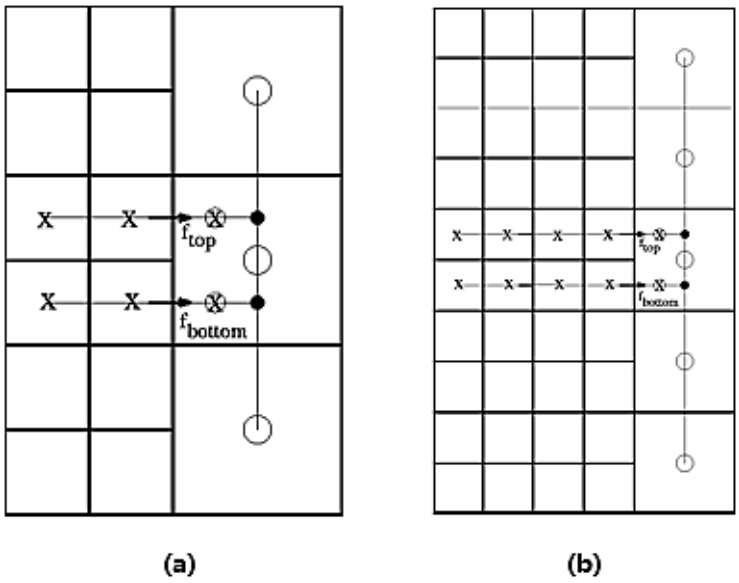



Figure 2.17 High order spatial interpolations

In coarse-to-fine interpolation, high order methods are more conductive to maintaining the accuracy of the numerical schemes, but excessive use of those methods should also be avoided.

$$u_{i+l,j+k} = u_{i,j} + \frac{\partial u_{i,j}}{\partial x}\Delta x + \frac{\partial u_{i,j}}{\partial y}\Delta y + \frac{1}{2}\left(\frac{\partial^2 u_{i,j}}{\partial x^2}\Delta x^2 + \frac{\partial^2 u_{i,j}}{\partial y^2}\Delta y^2 + 2\frac{\partial^2 u_{i,j}}{\partial y \partial x}\Delta x \Delta y\right) + O(\Delta h^3) \tag{2-3}$$

$$\begin{aligned} u_{i+l,j+k} = {} & u_{i,j} + \frac{\partial u_{i,j}}{\partial x}\Delta x + \frac{\partial u_{i,j}}{\partial y}\Delta y + \frac{1}{2}\left(\frac{\partial^2 u_{i,j}}{\partial x^2}\Delta x^2 + \frac{\partial^2 u_{i,j}}{\partial y^2}\Delta y^2 + 2\frac{\partial^2 u_{i,j}}{\partial y \partial x}\Delta x \Delta y\right) \\ & + \frac{1}{6}\left(\frac{\partial^3 u_{i,j}}{\partial x^3}\Delta x^3 + \frac{\partial^3 u_{i,j}}{\partial y^3}\Delta y^3 + 3\frac{\partial^3 u_{i,j}}{\partial y^2 \partial x}\Delta x \Delta y^2 + 3\frac{\partial^3 u_{i,j}}{\partial y \partial x^2}\Delta x^2 \Delta y\right) \\ & + \frac{1}{24}\left(\frac{\partial^4 u_{i,j}}{\partial x^4}\Delta x^4 + \frac{\partial^4 u_{i,j}}{\partial y^4}\Delta y^4 + 4\frac{\partial^4 u_{i,j}}{\partial y^3 \partial x}\Delta x \Delta y^3 + 4\frac{\partial^4 u_{i,j}}{\partial y \partial x^3}\Delta x^3 \Delta y + \right. \\ & \left. 6\frac{\partial^4 u_{i,j}}{\partial y^2 \partial x^2}\Delta x^2 \Delta y^2\right) + O(\Delta h^5) \end{aligned} \tag{2-4}$$

Formulas (2-3) and (2-4) show the general methods for achieving third and fifth order accuracy in 2D interpolations, it is clear that higher order interpolation usually requires more computational efforts. For those low order numerical schemes, the high order interpolations may waste lots of computational efforts but bring little improvements in accuracy. Therefore, detailed coarse-to-fine interpolation used in the optimization should depend on the solver in the numerical work.

In addition, it is worth to notice that in Figure 2.17 the two high order interpolations are used to update only one layer of ghost cells. Similarly, in Chombo [18], both linear and parabolic interpolations are available, but only the linear interpolation method provides the function to update multiple layers of ghost cells [18]. For higher order interpolation methods cannot keep the conservation law, they usually lack the function of updating multiple layers of ghost cells. The third order interpolation results (since they have the same accuracy as the third order Taylor expression, they can also be expression as (2-5)) is,

$$u_{i+l,j+m,k+n} = u_{i,j,k} + \frac{\partial u_{i,j,k}}{\partial x}\Delta x + \frac{\partial u_{i,j,k}}{\partial y}\Delta y + \frac{\partial u_{i,j,k}}{\partial z}\Delta z + \frac{1}{2}\left(\frac{\partial^2 u_{i,j,k}}{\partial x^2}\Delta x^2 + \frac{\partial^2 u_{i,j,k}}{\partial y^2}\Delta y^2 + \frac{\partial^2 u_{i,j,k}}{\partial z^2}\Delta z^2 + 2\frac{\partial^2 u_{i,j,k}}{\partial y \partial x}\Delta x\Delta y + 2\frac{\partial^2 u_{i,j,k}}{\partial z \partial x}\Delta x\Delta z + 2\frac{\partial^2 u_{i,j,k}}{\partial y \partial z}\Delta z\Delta y\right) + O(\Delta h^3) \quad (2\text{-}5)$$

In formulas (2-5), the term of $u_{i,j,k}$ is coarse data, the $u_{i+l,j+m,k+n}$ is the interpolated fine data. In above expression, the average value of the interpolated results are $u_{i,j,k} + k\Delta u|_{i,j,k}\ \Delta h^2$ if 2 layers of ghost cells are filled by the third order interpolation in the case of refinement ratio 2, because the second order terms contain the square of the distance. Thus, extra values are introduced and the conservation cannot be kept on finer ghost region. In contrast, the linear interpolation is not influenced by the square of the distance, so the conservation can be kept. As a result, to those solvers requiring multiple layers of ghost cells, the use of high order interpolation method should be more concerned.

Besides the spatial interpolation, temporal interpolation methods may also be considered in the coarse-to-fine interpolation in certain cases. Similarly, temporal interpolation also introduces extra truncation errors. Thus, for higher temporal accuracy, high order interpolation is required in time. In the work of [16], parabolic interpolation method has been used in temporal interpolation to maintain the second order accuracy. Similarly to the spatial interpolation, higher orders temporal interpolations also cost more computational efforts. Moreover, higher order interpolations also require more data at different time steps, which may largely increase the magnitude of the total data storage. So, the use of high order scheme in temporal interpolation should depend on the specific solvers and algorithm used in computation as well.

As a result, the selection of the scheme in local mesh refinement should still depend on numerical schemes, although the use of high order interpolation can reduce the risk of accuracy loss.

#### 2.4.1.2. Fine-to-coarse data transfer

Besides the coarse-to-fine interpolation, fine-to-coarse data transfer is also important in the inter-level data communication [65]. In simulations with patch-based local mesh refinement, calculations on different refinement levels are given individually. In other words, the coarse regions covered by finer levels should be included in calculation as well [6] [60], because these regions provide necessary boundary data to the calculation and interpolation. Figure 2.18 shows the diffusion simulation with LMR, and the regions being refined are marked with thick lines.

Shown by the Figure 2.18 results of calculations with different resolutions are not the same, and the calculations approached by finer resolutions usually have better performances. In addition, those covered regions on coarse levels also dominate the coarse-to-fine interpolation. So, fine-to-coarse data transfer which transmits finer level data to coarse level to keep the unity of two adjacent levels is necessary.

Similarly to coarse-to-fine interpolation, fine-to-coarse data transfer should also be approached by some numerical treatments. However, differently from the coarse-to-fine interpolation, the data communication is done for the "real" physical domain. So, keeping the conservation of the physical quantities in fine-to-coarse data transfer is in high priority. In order to keep conservation, updating the covered regions by averaged finer data is the only choice.

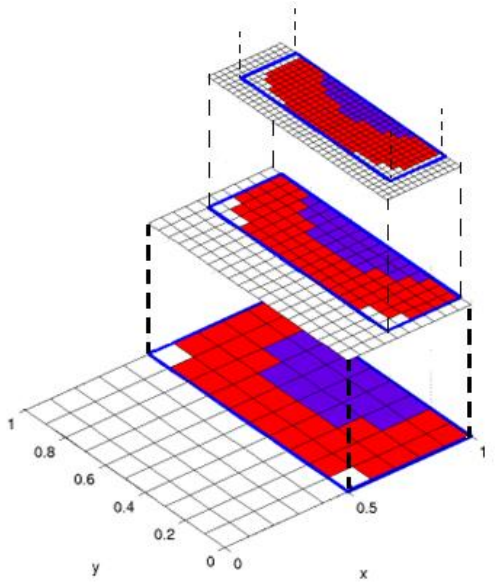



Figure 2.18 Finer covered region in the local mesh refinement

2.4.1.3. Flux correction in local mesh refinement

As illustrated in sub-section 2.3, one of the biggest differences between the patch-based local mesh refinement and the other two refinement approaches is the fine-to-coarse interface. In the patch-based local mesh refinement, some edges on the fine-to-coarse interface may be shared by $1+r^{d-1}$ cells (the r here means the refinement ratio between the two neighboring levels, d is the dimension). So, the fluxes used on each refinement level may have some difficulties in keeping conservation.

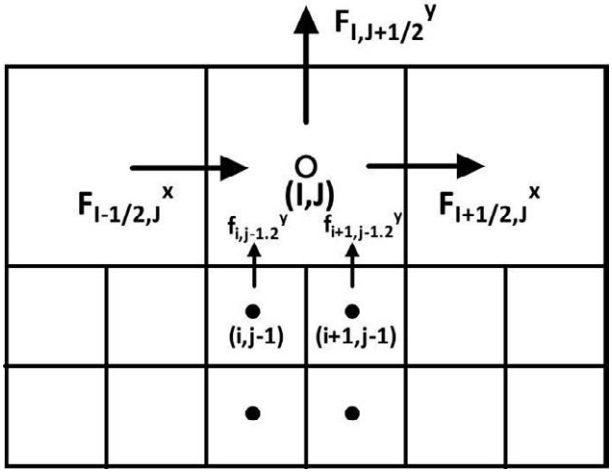



Figure 2.19 Fluxes on the fine-to-coarse interface

Just as the 2D case shown in Figure 2.19, the edges which are adjacent to the fine-to-coarse interface are shared by three cells. Since the calculation is given independently on each refinement level, it is quite possible that coarse flux $F_{I,J-1/2}{}^{y}$ is not equal to the average value of finer fluxes $f_{i,j-1/2}{}^{y}$ and $f_{i+1,j-1/2}{}^{y}$. In that case, the inflow is not equal to the outflow and the conservation of quantities on the fine-to-coarse interface is broken. As mentioned in section 2.4.1.2, keeping conservation in "real" physical domain is in very high priority and should be maintained in the implementation of LMR. In this situation, the treatment called as flux correction should be done in the cells which are adjacent to the fine-to-coarse interfaces.

In principle, either correcting the finer fluxes by the interpolated coarse flux values or substituting the coarse fluxes by the averaged finer flux values can meet the requirements of flux correction. However, because of the total computational efforts and better performance of the calculation with fine resolutions, substituting the coarse fluxes by the averaged finer flux values is more acceptable. In the cases that refinement in time is included in LMR, correcting the coarse fluxes with averaged finer fluxes is also more convenient.

### 2.4.2. Process of simulation with patch-based local mesh refinement

Besides the basic inter-level data communications illustrated above, the arrangement of the calculations on different refinement levels is also very important. Figure 2.20 roughly introduces the process of the forward Euler based calculation with three levels under refinement ratio two.

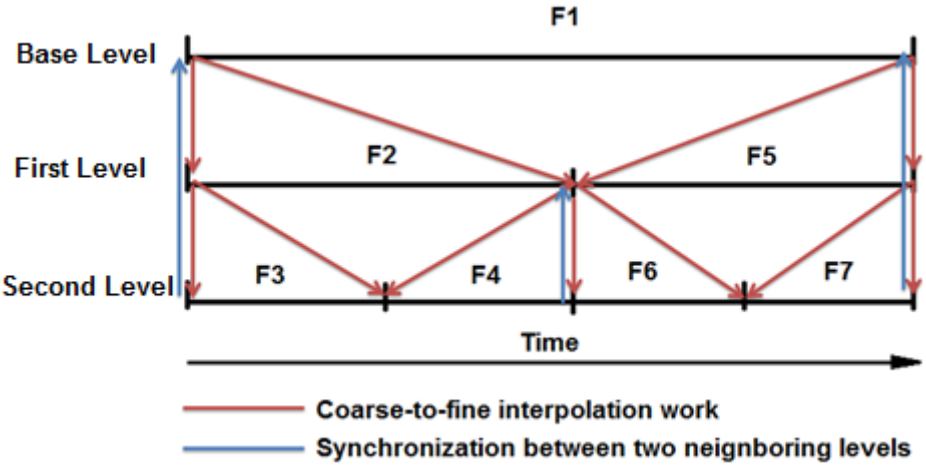



Figure 2.20 Process of Forward Euler scheme with LMR

As shown in Figure 2.20, refinement in time-scale is used in the calculation as well. In LMR with temporal refinement, the refinement ratio in time-scale should be depended on the detailed simulation. To the explicit simulation of inviscid flow [16] [61], the temporal refinement ratio is usually the same as the spatial refinement ratio. In the figure, the $Fi, i = 1, 2, 3, \cdots$ show the order of the calculation on each refinement level (in latter figures about calculation process, calculation order is indicated by the same way), red lines indicate the coarse-to-fine interpolation and the blue lines are the synchronization which means the fine-to-coarse data transfer and flux correction. From the point of view of programming, the process shown in Figure 2.20 can be given as the following format.

$LevelAdvance(l, t^l, \Delta t^l)$

$Boundary\ updating\ for\ u^l(t^l)$

$Initial\ update\ of\ u^l$ :

$u^l(t^l + \Delta t^l)_{i,j} = u^l(t^l)_{i,j} - \Delta t^l D^l(F^l)$

$Initialize\ /\ update\ flux\ register$ :

$if\ (l < l_{\max})\ \delta F^{l+1} = -F^l \cdot n_{CF}^{l+1}$

$if\ (l > 0)\ \delta F^l += \dfrac{1}{n_{ref}^{l-1}} < F^l \cdot n_{CF}^l >$

$Call\ to\ LevelAdvance\ for\ next\ finer\ level$ :

$if\ (l < l_{\max})$

$for(i = 0; i < n_{ref}^l; i++)$

$\Delta t^{l+1} = \dfrac{1}{n_{ref}^l} \Delta t^l$

$t^{l+1} = t^l + n\Delta t^{l+1}$

$LevelAdvance(l+1, t^{l+1}, \Delta t^{l+1})$

$Synchronize\ level\ l\ with\ level\ l+1$

As shown in the above expression, calculation should be given from the coarsest level to the finest level. After one step of standard calculation (without considering the interpolation and

synchronization between levels) on the coarse level, computation should be shifted to the next finer level. Then, depending on the existence of finer refinement level, computation should be shifted again or made further until to the same physical time as the next coarser level. When the finer level has reached the same physical time as the next coarser one, synchronization between the two neighboring levels is done. The above implementation of LMR includes all the features of patch-based local mesh refinement, such as temporal refinement, inter-level data communication and flux correction.

However, the above process can only illustrate the implementation of LMR in forward Euler scheme. In some more complicated algorithms, such as the alternating direction method (fraction step method) and Runge-Kutta methods (second order, third order and fourth order), the implementations of the patch-based local mesh refinement have been limited in previous works.

Table 2.1 Implementation of LMR in several calculation methods

| Calculation method | | Temporal refinement | High interpolation | Keep the accuracy |
|---|---|---|---|---|
| Forwards Euler [6] | | Yes | No | Yes |
| Alternating direction [56] | | No | No | Yes |
| Runge-Kutta | $2^{nd}$ order [16] | Yes | Yes | Yes |
| | $3^{rd}$ order [55] | No | Yes (space only) | No |
| | $4^{th}$ order [55] | No | Yes (space only) | No |

In Table 2.1, the previous implementations of LMR in 5 different algorithms are given. The table focuses on the items which can influence the accuracy and computational efforts of the simulation, such as the couplings of the computations between levels and the inter-level data communications. Temporal refinement is one big advantage of the patch-based local mesh refinement, but this refinement may reduce the accuracy in time-scale. Therefore, only half implementations shown in the table include the temporal refinement, and the uniform time step is used in the implementations in other high order algorithms [56] [55] to maintain the high temporal accuracy. High order interpolation used in coarse-to-fine interpolation is another important item for keeping the accuracy and controlling numerical reflection [16] [55] in LMR. The high order interpolations are used in the high order schemes to maintain accuracy as well. As the function of using high order solvers and algorithms is one of COM3D's most important features, the experiences of the implementations of LMR shown in Table 2.1 are quite valuable to the optimization of the code.

For some 1D high accuracy solvers, in order to maintain the good accuracy, alternating direction (AD) scheme [23] [75] has to be used in the solution of high dimension problems. In [56], the patch-based local mesh refinement with uniform time step has been implemented in front tracking solver. Differently from the usual explicit solvers, the front tracking solver does not desire the temporal refinement in LMR, because the numerical stability of the solver is not limited by time step. In the implementation of LMR in AD based front tracking solver, comparing to the implementation in forward Euler scheme, one big difference is the shifting of the calculation on each level is changed from steps to directions. In other words, the calculations are shifted between levels after finishing one direction's calculation in [56]. As shown by the table, implementation of

LMR in the AD based front tracking keeps the accuracy. In COM3D, the TVNI solver [38] is one important numerical method which also requires the alternating direction scheme in the high dimension's application. However, the TVNI solver requires very strict CFL comdition (normally less than 0.4) for stability. The refinement in time-scale may be necessary in the optimization in AD scheme. The implementation of local mesh refinement with temporal refinement in alternating direction scheme is given in detail in Chapter 3.

Besides the alternating direction approach, second and fourth order Runge-Kutta (RK) schemes are two other important high order algorithms used in COM3D. RK scheme is the multiple-step calculation scheme, the existence of the intermediate phases brings lots of inter-level data communication and makes the implementation of local mesh refinement diversified. The reference [16] gives a mature implementation of LMR in second order RK scheme. By consideration of the implementation of LMR in other calculation methods, implementation of LMR in second order RK is different in COM3D, but the accuracy of the algorithm can be maintained. The fourth order RK scheme requires more intermediate phases and achieves fourth order temporal accuracy, the implementation of LMR is also more complicated. As shown in Table 2.1, the uniform time step is used to keep the high accuracy in time. However, the simplified inter-level data communication for the intermediate phases may reduce the accuracy. Actually, RK4 with LMR attaching uniform time step and high order interpolation can achieve the same accuracy as the uniform grid case [85]. Lacking the understanding of each intermediate phase in RK4 results in the loss of accuracy in [55] (detailed analysis is given in appendix). In COM3D, the implementation of LMR in RK4 includes the use of multiple time steps, although the temporal accuracy may have some loss. In the next chapter, detailed temporal truncation error analyses and the reason why the loss in temporal accuracy is acceptable are given.

### 2.4.3. Detonation simulation with patch-based local mesh refinement

As one main function of COM3D is the simulation for combustion, the LMR should be considered in intensive chemical reaction. Currently, this thesis only focuses on the simulation of detonation. Owing to high resolution required by detonation in chemical reaction zone, local mesh refinement is quite meaningful to such simulation [51].

In COM3D, modeling of detonation is accomplished by the one step Arrhenius method coupled with the Euler equations [47]. However, differently from the Euler equations (PDE), the Arrhenius equation is ODE style and the numerical solution of such ODE problem is point-wised. The differences in numerical solution may bring some problems in the implementation of LMR.

The process of the commonly used forward Euler based detonation simulation with LMR is shown in Figure 2.21 [25] [26].

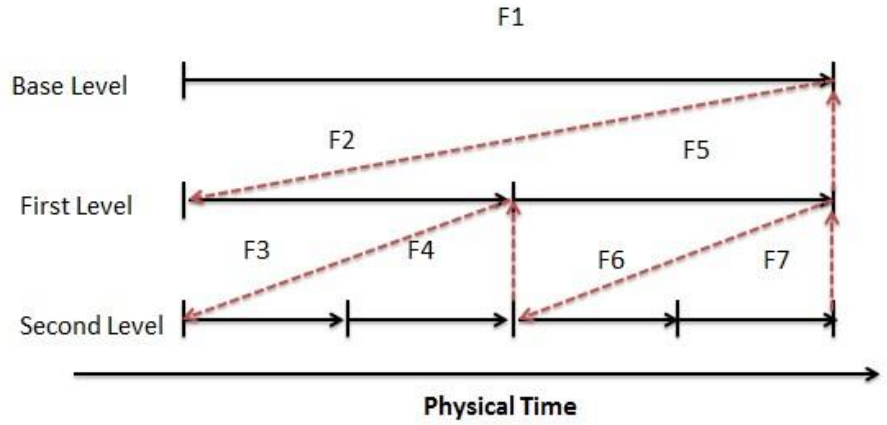



Figure 2.21 Process of forward Euler based detonation simulation with LMR [25]

AS shown in Figure 2.21, similarly to the process of forwards Euler based solution descript in section 2.4.2, the calculation are shifted to the other levels after the standard calculation (solution of Euler equations and the solution of chemical model) on one refinement level has been finished. Indeed, the point-wised chemical reaction may bring some problems to the simulation descript by Figure 2.21.

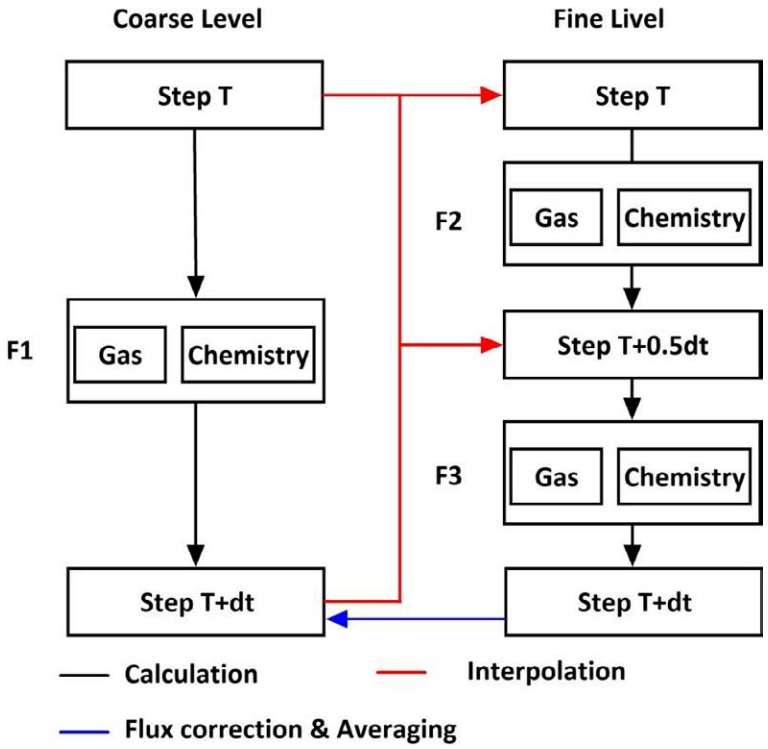



Figure 2.22 Detail process of detonation simulation in two neighbor levels

Figure 2.22 shows the calculations of two neighboring levels in detonation simulation in [27] [28] [31]. Calculations of both chemistry and gas dynamics are based on the former step's result, so the two calculation parts are arranged parallel in the figure. In one big calculation step, the detailed process is below.

(1) Firstly, the calculation on coarse level should be made. Since the calculation of chemical reaction is point-wised, the chemical calculation at this step has no contribution to the fluxes on coarse level.
(2) Then, owing to the refinement ratio between the two neighbors is two, two steps of simulation should be done at this big step on finer level. Similarly to coarse level fluxes, fluxes of the first small step on finer level are not influenced by chemical reaction.
(3) In the second step of calculation on the fine level, although the chemical reaction at this step does not influence the fluxes either, influences of the chemical work in the first small step are included in the fluxes. Thus, the total fluxes stored in fine flux register contain the influences of one small step of chemical reaction.
(4) When the two neighboring levels have reached the physical time T+dt, synchronization is done between the two levels. In the flux correction phase, since no chemical reaction is contained in the coarse flux register and one small step of chemical reaction is included in fine flux register, some extra chemical reaction is introduced to the coarse cells which are adjacent to the fine-to-coarse interface. On the coarse level, cells which are adjacent to the fine-to-coarse interface may contain more chemical reaction. Such numerical increasing of chemical reaction on coarse level is called redundant chemical reaction.

As shown by above analysis, it is clear that the directly used temporal refinement may result in redundant chemical reaction on coarse level in detonation simulation. As the coarse cells can dominate the coarse-to-fine interpolation, calculation on finer levels is also influenced. However, in [22-25], the influence of the redundant chemical reaction can be neglected. In the work of [22-25] [31] [32], patch-based local mesh refinement is used adaptively in the detonation simulation, the intensive reaction zones are always covered by fine resolution. In contrast, the areas around the fine-to-coarse interface in [22-25] are the completely burned or unburned area. Actually, the error is relatively small, although the chemical reaction in those coarse cells has been enlarged.

Unlike the application cases in [22-25], COM3D should deal with a variety of industrial problems, in which local mesh refinement approaches may be different in different situations. In the case that local mesh refinement is used statically, the influence of redundant chemical reaction may be quite significant when the detonation wave passes through the fine-to-coarse interface. In order to avoid the redundant chemical reaction, some other methods should be introduced to the detonation simulation with LMR.

### 2.5. Load balancing for patch-based local mesh refinement

In Chapter 1, it has already been mentioned that one feature of the software package COM3D is the parallelism. After the introduction of the patch-based local mesh refinement, attention should also be given to the parallel computation with patch-based local mesh refinement.

### 2.5.1. Parallelism and speedup

"Parallel computation is a form of computation in which many calculations are carried out simultaneously, operator on the principle that large problems can often be divided into smaller ones, which are then solved concurrently ("in parallel")" (en.wikipedia.org). With the technique of parallel computation, calculation speed can be increased by one or two orders of magnitude, although the total computational efforts cannot be decreased by a little bit. The most commonly used parallel mode in CFD is the data based parallelism, which is also the parallelism used in COM3D.

In data based parallelism, the whole computational domain is decomposed into a series of un-overlapped sub-domains [74]. Each processor takes one or several sub-domains to make the computation. Computational efforts in each processor have been largely reduced and the total computation speed is increased significantly. Normally, the speedup of the parallel computation (2-6) is used to evaluate the performance of parallel computation.

$$Speedup = \frac{T_{serial}}{T_{parallel}} \tag{2-6}$$

The speedup is the serial computing time to parallel computing time ratio, which describes how many times of computing time reduction can be gained through the parallelism. In principle, the more processors are used in the parallel computation the higher speedup can be achieved. However, the relation between the two is not linear. In order to evaluate the efficiency of the parallel computation, another concept parallel efficiency (2-7) is used.

$$Efficiency = \frac{T_{serial}}{N \cdot T_{parallel}} \tag{2-7}$$

The parallel efficiency illustrates the ratio of the CPU time in pure calculation to the whole parallel computation. In other words, computational resources are not only paid on the CFD calculation, some other works may also cost a considerable ratio of resources. Besides the computation in each sub-domain, data communication between different sub-domains also cost a big ratio of time. The total time cost $T_{parallel}$ can be expressed as the equation (2-8).

$$T_{parallel} = T_{computation} + T_{communication} \tag{2-8}$$

Besides a clear understanding of the time cost in parallel computation, a systematic knowledge of the parallel process is also important in making optimal domain decomposition. The detailed process of one step of calculation based on uniform grids in COM3D (calculation with LMR may have more or less the same process) is shown in Figure 2.23.

In the figure, besides the computation and communication phases mentioned in former sentence, two extra barrier phases are also made in the step. Due to a variety of reasons, processors may not finish the calculation or communication simultaneously. So, the barrier phase is used to synchronize the processors. With the barriers, calculations or communications are stopped until all the processors in the parallelism have reached this barrier. Therefore, both $T_{computation}$ and $T_{communication}$ are dominated by the processor which contain the most computation or

communication. In this situation, the task of the load balancing is to achieve balanced distribution of computation and communication.

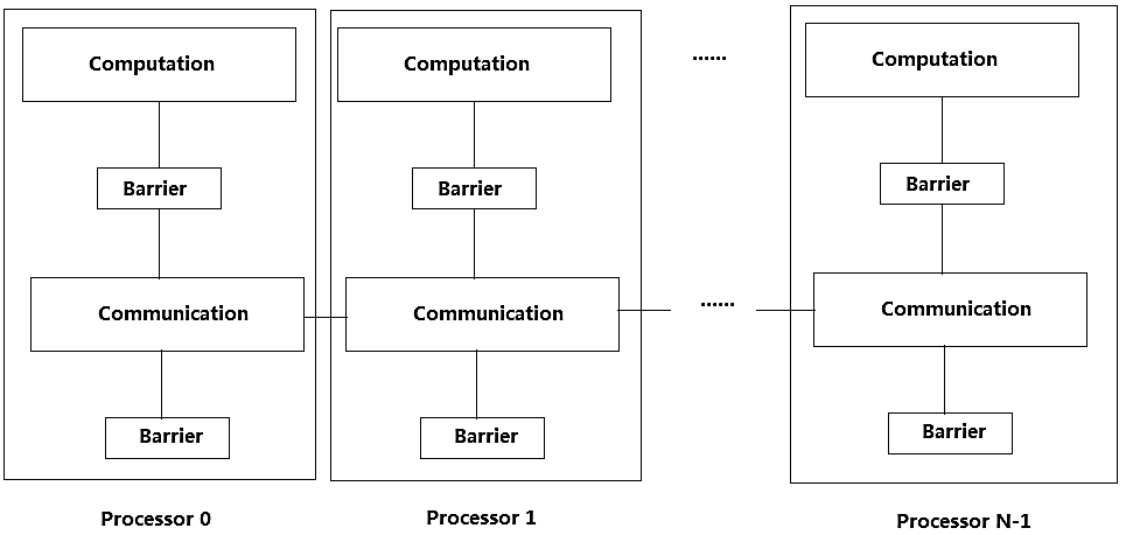



Figure 2.23 One step of parallel computation with N processors

In normal CFD parallel codes, the load balancing methods are committed to decompose the whole computational domain into a series of sub-domains which contain even volume and optimized surface-to-volume ratio. The geometry based domain decomposition which cost the least time and coding efforts is one of the most acceptable load balancing approaches for the structured grids.

Differently from normal CFD codes, COM3D commits to the simulation in large-scale industrial problems. In the industrial or practical application cases, such as the hydrogen combustion simulation in the EPR nuclear containment, detonation simulation in the rut facility and the flame simulation in the long tunnel, computational domains may have lots of solid structures or obstacles which make the simulation more complicated. In this situation, the normal simple geometry based domain decomposition can hardly achieve balanced distribution of computation and communication. So, a more complicated domain decomposition which considers the influences of solid structures is needed in COM3D.

### 2.5.2. Load balancing for patch-based local mesh refinement

As the introduction of patch-based local mesh refinement technique, domain decomposition may become more complicated than the uniform grids cases. Normally, the load balancing for the simulation with patch-based local mesh refinement can be achieved by patch-based decomposition, domain-based decomposition and the hybrid decomposition.

The patch-based approach [43] [71] [44]:

For the patch-based approach, the most straightforward method is to divide each patch or level into p blocks separately, where p is the number of processors, and distribute one block to each processor. Another method is to use a bin-packing or greedy algorithm to distribute the patches. For the partitioning to be effective, large patches may have to be divided. Regardless of the specific method, the partitioner can work either patch-by-patch or level-by-level.

In theory, the patch-based approach results in perfect load balance. In practice, some load imbalance can be expected due to sub-optimal patch aspect ratios, integer divisions and constraints on the patch size. Partitioning can be performed incrementally, as only patches created or altered since the previous time step need to be considered for re-partitioning. However, patch-based algorithms often result in high communication volumes and communication bottlenecks. The communication volume, especially inter-level communication, is generally increased when a patch is subdivided to create a lower load imbalance. Communication serialization bottlenecks can occur when overlaid patches are assigned to different processors. A coarser patch is typically assigned to fewer processors than a finer patch. A processor owning coarser patches will generally need to communicate with many processors having finer and overlaid patches, creating communication bottlenecks. Figure 2.24 shows a Patch-based decomposition for 1D case, $G_i^N$ is used to indicate the patch i on the refinement level N, and the Pj is used to show the process assignment. Shown by the figure, each patch is decomposed and distributed equally to each processor.

However, owing to the development of infiniband communication, the problems of the inter-level data communication has already become less and less. For the calculation based on patch-based local mesh refinement, the patch-based domain decomposition is the method which can best optimize the time cost in the pure computation phase. For the computation dominant simulation, the patch-based decomposition is the most suitable one.

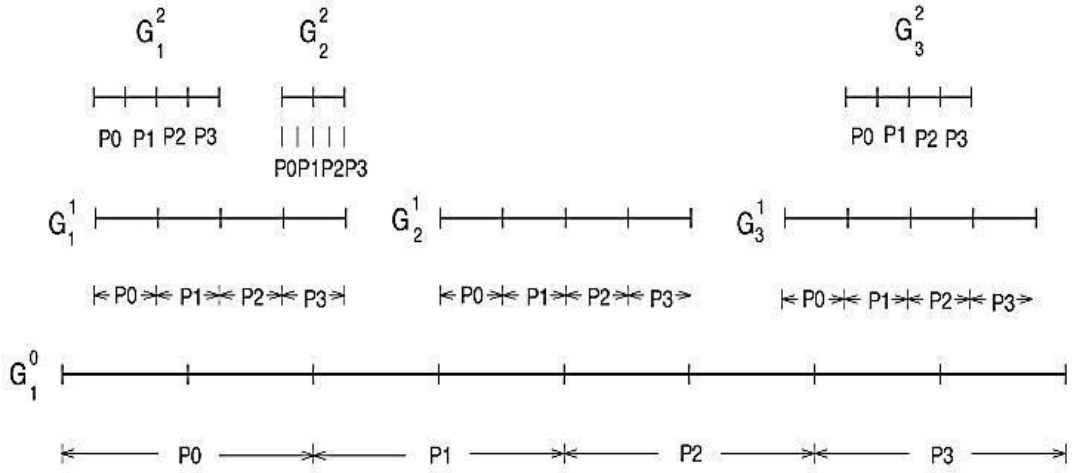



Figure 2.24 Patch-based decomposition for 1D case

The domain-based approach [78] [79]:

For domain-based algorithms, only the base grid is partitioned. Initially, the workloads of the refined patches are projected down onto the base grid, reducing the problem to partitioning a single grid with heterogeneous workload. The minimum patch size is determined by the size of the computational stencil on the base grid. As the base grid stencil corresponds to many grid points on highly refined patches, the workload of a minimum sized block can be large.

As overlaid grid blocks always reside on the same processor, inter-level communication is eliminated. A complete re-partition might be necessary when the grid hierarchy is modified. Because the computational stencil and base grid resolution impose restrictions on subdivisions of higher level patches, the load imbalance is often high for complex or deep grid hierarchies. Furthermore, synchronization bottlenecks are common as the division of a refinement level generally results in parts with widely differing workloads. Another problem with domain-based algorithms is "bad cut": many and small blocks with bad aspect ratios. These blocks occur when patches are cut in bad places, assigning only a tiny fraction of a patch to one processor while the majority of it resides on another processor.

As illustrated by the above sentences, the domain-based decomposition sometimes can be very unreliable, because it is very hard for the partitioner to consider the working balance on all the refinement levels. For the simulations in which the distribution of the finer patches is uniform, such as the simulation of the fluid in the water channel in nuclear fuel assembly, or the data communication between different processors is very slow, domain-based load balancing can show good performances. Figure 2.25 shows the 1D LMR case decomposed by domain-based decomposition.

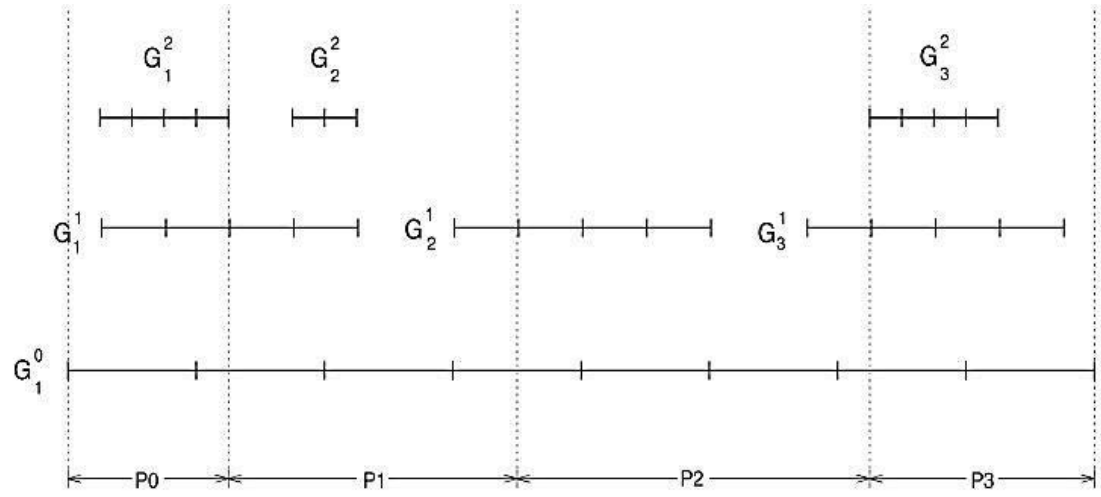



Figure 2.25 Domain-based decomposition for 1D case

A hybrid approach [77] [81]:
Both patch-based and domain-based algorithms perform well under suitable conditions, especially for simple and shallow grid hierarchies. Unfortunately, their shortcomings often make their performance unacceptable for deep and complex hierarchies. As a remedy, a hybrid approach can be used. By combining strategies from both the domain-based and the patch-based approach, it is possible to design a partitioner that performs well under a wider range of conditions.

To illustrate the hybrid approach, we describe the partitioning framework that is used as a basis for the Meta-Partitioner — Nature+Fable. Key concepts in Nature+Fable are separation of refined and unrefined areas of the grid and clustering of refinement levels. Separation of unrefined and refined

areas enables different partitioning approaches to be applied to structurally different parts of the grid hierarchy. Refinement levels are clustered into bi-levels. A bi-level consists of all patches from two adjacent levels — patches from refinement level k and the next finer level k + 1. If the coarser level is much larger than the finer level, the non-overlaid area of the coarser level can be removed from a bi-level. Each bi-level is partitioned with domain-based methods. Patch-based methods are used for all parts of the grid that are not included in bi-levels.

The hybrid partitioning algorithms arising from Nature+Fable can achieve a lower load imbalance than domain-based algorithms, since patches from at most two refinement levels are partitioned together. Because inter-level communication only exists between the bi-levels, communication volumes are generally smaller for the hybrid algorithms than for patch-based algorithms. Figure 2.26 shows the 1D LMR case decomposed by the hybrids approach, in the decomposition the based level is decomposed independently from the other two levels, and the second and the first level are decomposed together by the domain-based approach.

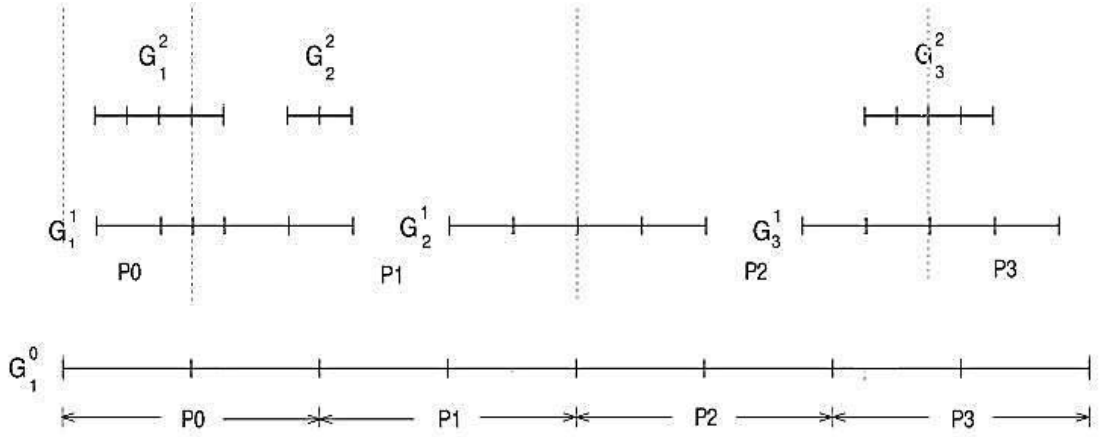



Figure 2.26 Hybrids decomposition for 1D case

The three approaches introduced above have their own advantages and disadvantages. For different simulations the three partitioners may perform quite differently, but none of them can always be the best. As mentioned in [80] and [81], the selection of suitable domain decomposition method should be depended on the detailed computational domain, local mesh refinement arrangement, inter-processor communication speed and coding in the LMR tools. In some domain decomposition tools, all three methods are included to deal with different computational domains and local mesh refinements [45] [46]. However, by considering the data communication mode in the local mesh refinement and the cluster used in computation, only the patch-based domain decomposition is selected for COM3D. The partitioner used in the parallel computation with the patch-based local mesh refinement in COM3D is given in Chapter 4.

## 2.6. Library for the implementation of local mesh refinement

As the local mesh refinement for COM3D has already been focused on the patch-based approach, the library which can provide both the function of parallelism and basic patch-based LMR tools is required. The libraries such as SAMRAI [73], Chombo [17] and AMROC [1] can meet the requirements of optimization in COM3D.

The library KeLP V1.4 [48] is used currently in uniform grid version COM3D to provide the parallel tools and block structured grids. In order to reduce the coding difficulties, the selection of library should be inclined to the library which has the similar data structure and communication mode to KeLP. The library Chombo has been used to make the implementation of local mesh refinement. In this thesis, all the coding is done based on the Chombo V3.0 which is developed by the Lawrence Berkeley Laboratory.

Chombo can only provide a basic structure of local mesh refinement for LMR in COM3D. Some inter-level data communications may need proper modification to make them fit for the calculation in COM3D. Especially for the coarse-to-fine interpolation, existence of solid structures in the computational domain and different requirements of the solvers may bring lots of problems to the current interpolations in Chombo. In the next chapter, the inter-level data communications will be considered.

In total, for a better understanding of the work and detailed technique approaches in the thesis, technical approaches and implementation area are given in the table 2.2.

Table 2.2 Summary of techniques used for implementation of LMR

| | |
|---|---|
| Local mesh refinement | Patch-based block-structured local mesh refinement |
| Implementation area in COM3D | Solution of Navier-stokes equations, simulation of hydrogen detonation |
| Parallelism | Multi-processors parallelism based on OpenMPI |
| Domain decomposition | Optimized decomposition algorithm based on patch-based partition |
| C++ library | Chombo V3.0 |

# 3. Implementation of LMR in COM3D

After the review of block-structured patch-based local mesh refinement, detailed implementation of this technique in COM3D is established. In this chapter, the implementations in solution of Euler equations, solution of NS equations and the detonation simulation are given.

## 3.1. Local mesh refinement in the solution of Euler equations

Euler equations are the governing equations of the inviscid flow. In COM3D, since the solution of NS equations and the simulation of combustion are all based on the solution of Euler equations, the implementation of local mesh refinement in the solution of Euler equations plays a very important role in the whole optimization.

The governing equations used in COM3D are shown by (3-1). In the formulas, u, v, w are the velocity in X, Y and Z directions, ρ is the local density, e is the total energy, p is the pressure value and the $Y_i$ indicates density of different gas components (the usual gas components are$H_2$, $O_2$, $N_2$and $H_2O$). The Euler equations used in COM3D describe the basic convection effects of mass, momentum, energy and gas components.

$$\begin{cases} \frac{\partial \rho}{\partial t}+\frac{\partial \rho u}{\partial x}+\frac{\partial \rho v}{\partial y}+\frac{\partial \rho w}{\partial z}=0 \\ \frac{\partial \rho u}{\partial t}+\frac{\partial \rho u^2}{\partial x}+\frac{\partial \rho uv}{\partial y}+\frac{\partial \rho uw}{\partial z}=-\frac{\partial p}{\partial x} \\ \frac{\partial \rho v}{\partial t}+\frac{\partial \rho uv}{\partial x}+\frac{\partial \rho v^2}{\partial y}+\frac{\partial \rho vw}{\partial z}=-\frac{\partial p}{\partial y} \\ \frac{\partial \rho w}{\partial t}+\frac{\partial \rho uw}{\partial x}+\frac{\partial \rho vw}{\partial y}+\frac{\partial \rho w^2}{\partial z}=-\frac{\partial p}{\partial z} \\ \frac{\partial \rho e}{\partial t}+\frac{\partial u(\rho e+p)}{\partial x}+\frac{\partial v(\rho e+p)}{\partial y}+\frac{\partial w(\rho e+p)}{\partial z}=0 \\ \frac{\partial Y_i}{\partial t}+\frac{\partial Y_i u}{\partial x}+\frac{\partial Y_i v}{\partial y}+\frac{\partial Y_i w}{\partial z}=0 \\ p=(\sum \frac{Y_i}{M_i})\frac{R}{C_V}(\rho e-\frac{1}{2}(\rho u^2+\rho v^2+\rho w^2)) \end{cases} \tag{3-1}$$

Before the detailed implementation of the technique, preparation of the implementation, such as the schemes in inter-level data communications and the temporal refinement, is introduced firstly.

### 3.1.1. Preparing for LMR in solution of Euler equations

As mentioned in Chapter 2, high order method can largely reduce the risk of accuracy loss in the implementation of LMR [63] [76], the selection of inter-level data communication should be based on the detailed solvers and algorithms used in COM3D. The coarse-to-fine interpolation, the fine-to-coarse data transfer and the flux correction are discussed in detail in this section.

#### 3.1.1.1. Coarse-to-fine interpolation

Since the truncation errors brought by the interpolation have a direct impact on the accuracy, the order of schemes used in data communication should be decided by the consideration of the solvers used in simulation. For example, high order interpolations are used to maintain the accuracy of the high order calculation schemes in references [4] and [16]. For the discussion of the inter-level data communication, solvers contained in COM3D are summarized firstly.

Table 3.1 Solvers in COM3D

| The solver | The ghost cells needed in calculation | Order of the scheme |
|---|---|---|
| Up Wind [22] | 1 | 1 |
| TVNI [38] | 2 | 2 |
| van Leer [78-80] | 2 | 2 |
| AUSM [57] | 1 | 1 |
| AUSM+ [58] | 1 | 1 |
| AUSM+ up [59] | 2 | 2 |

In Table 3.1, all the explicit solvers in COM3D used for the calculation of convection are listed. Half of the solvers require only 1 ghost cell in all directions, and the others higher order solvers require 2 ghost cells in each direction to make calculation. In current version of COM3D, all the patches in the parallel computation have two layers of ghost cells [51]. In the thesis, the first order Up Wind scheme, TVNI scheme and van Leer scheme are implemented with LMR, so two layers of ghost cells are necessary.

In the computation with local mesh refinement, all the patches on each refinement level should also contain at least the same number of ghost cells for continuing using the solvers in Table 3.1. As a result, the coarse-to-fine interpolation in COM3D should update multiple layers of ghost cells. Figure 3.1 shows the two layers of ghost cells in finer patches, when the refinement ratio two is used between the two neighboring levels.

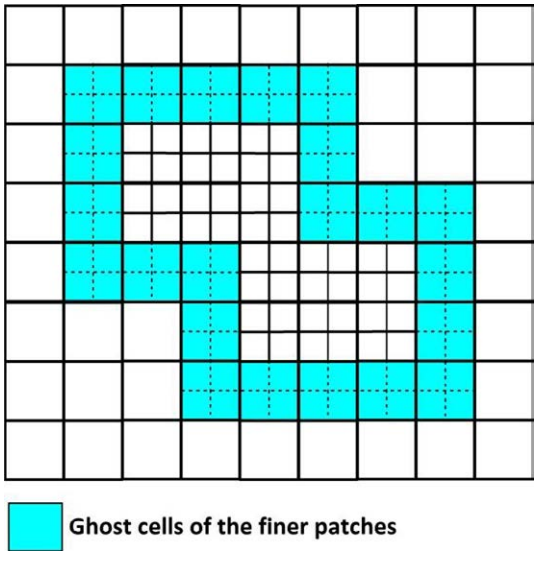



Figure 3.1 Ghost region updating under refinement ratio two

As shown in Figure 3.1, the ghost cells on finer level take the full volume of the finer level covered coarse cells, and the conversation of the quantities has to be kept in the interpolation. As mentioned in Chapter 2, high order interpolations can hardly satisfy the conservation in the boundary updating for multiple layers of ghost cells. The constant or linear interpolation which can keep the conservation is suitable.

In the constant interpolation, by considering the $0^{th}$ order Taylor series (3-2)

$$V_F = V_C + \sum_{i=0}^{D-1} f_{x_i}'(\theta_i)\Delta x_i \tag{3-2}$$

, the accuracy achieved is first order in space. On other hand, by considering the second order Taylor series (3-3)

$$V_F = V_C + \sum_{i=0}^{D-1} f_{x_i}'(x_C)\Delta x_i + \sum_{i=0}^{D-1} \frac{f_{x_i}''(\theta_i)}{2}\Delta x_i^2 \tag{3-3}$$

, the linear interpolation can provide second order accuracy in space. After consulting Table 3.1, in the calculation of convection part, linear interpolation has more advantages in keeping accuracy.

Besides keeping the conservation of all the physical quantities in the coarse-to-fine interpolation, controlling the unphysical oscillations in interpolation is also very important. In the linear interpolation, the first order derivatives can be approached by three different schemes: the central scheme, the backward scheme and the forward scheme. Those three different schemes have quite different performances in oscillation controlling. Figure 3.2 and Figure 3.3 show the performances of the three schemes when the shock wave passes through the fine-to-coarse interface.

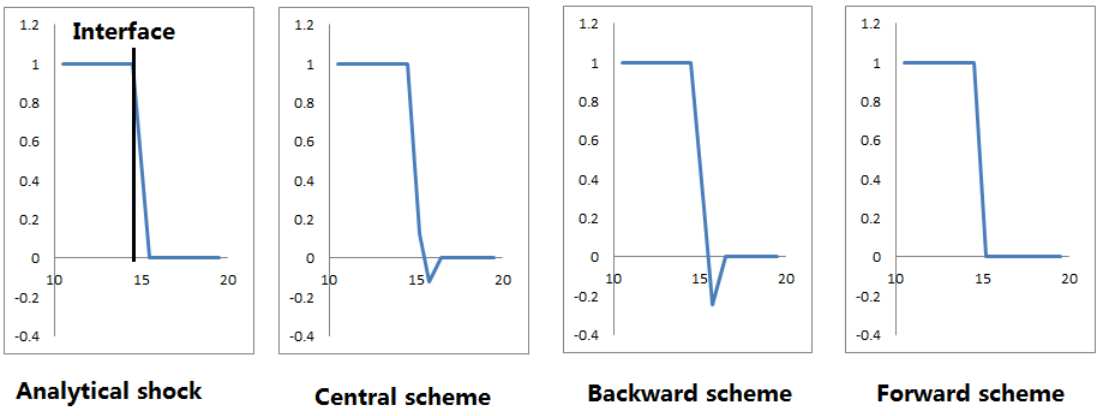



Figure 3.2 Linear interpolation for shock wave in Case 1

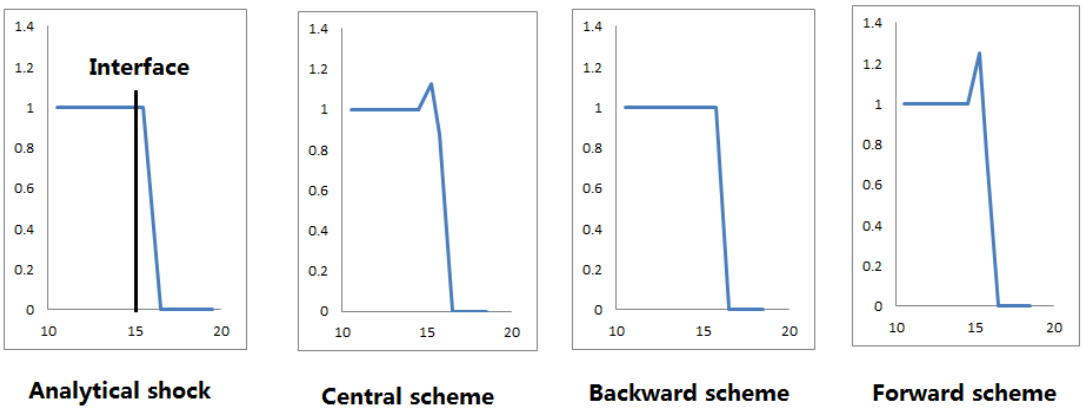



Figure 3.3 Linear interpolation for shock wave in Case 2

Figure 3.2 shows the shock wave which has just reached the fine-to-coarse interface and Figure 3.3 shows the situation when the shock has just passed through the fine-to-coarse interface. For convenience, the former one is called case 1 and the latter one is called case 2. In the two cases, the three schemes give different performances: the central scheme can result in unphysical oscillations in both two cases; the backward scheme can give very good performance in case 2 but even bigger oscillation in case 1; in contrast, the forward scheme performs oppositely as the backward scheme. Table 3.2 gives the summary of all the three schemes, including the format of the scheme, the accuracy can be achieved by the scheme and if unphysical oscillation can be generated by the scheme.

Table 3.2 Comparison of all three schemes in linear interpolation

| Derivative mode | Form | Accuracy | Oscillation in case 1 | Oscillation in case 2 |
|---|---|---|---|---|
| Central scheme | $\frac{\partial f}{\partial x}=\frac{f(x_{i+1})-f(x_{i-1})}{2\Delta x}$ | Second order | Yes | Yes |
| Backward scheme | $\frac{\partial f}{\partial x}=\frac{f(x_{i})-f(x_{i-1})}{\Delta x}$ | First order | Yes | No |
| Forward scheme | $\frac{\partial f}{\partial x}=\frac{f(x_{i+1})-f(x_{i})}{\Delta x}$ | First order | No | Yes |

As shown in the table, the central scheme has the best accuracy among all three derivative modes, but it is also the one which can produce unphysical oscillation in both cases. In contrast, the backward scheme and forward scheme can provide only first order accuracy but have better performances in controlling oscillations. In order to avoid unphysical oscillation and have higher accuracy in normal interpolation, the method used in coarse-to-fine interpolation in COM3D is as (3-4).

$$V_F = V_C + \sum_{i=0}^{D-1} f_{x_i}'(x_C)\Delta x_i$$

$$\begin{cases} f_{x_i}' = \begin{cases} 0 & if\ (f(x_i)-f(x_{i-1}))(f(x_{i+1})-f(x_i)) \le 0 \\ \min(|\dfrac{f(x_{i+1})-f(x_{i-1})}{2\Delta x}|, |\dfrac{f(x_i)-f(x_{i-1})}{\Delta x}|, |\dfrac{f(x_{i+1})-f(x_i)}{\Delta x}|) sign(f(x_{i+1})-f(x_{i-1})) & else \end{cases} \\ f_{x_i}' = \dfrac{f(x_i)-f(x_{i-1})}{\Delta x} \qquad if\ x_{i+1} \notin \Omega_C\ or\ x_{i+1} \in \Omega_{solid} \\ f_{x_i}' = \dfrac{f(x_{i+1})-f(x_i)}{\Delta x} \qquad if\ x_{i-1} \notin \Omega_C\ or\ x_{i-1} \in \Omega_{solid} \end{cases}$$

(3-4)

$\Omega_C$ is the coarse domain and $\Omega_{solid}$ is solid in the computational domain. Shown by the formulas, all three schemes are available in the interpolation to provide the most optimal result. Besides considering the boundary of the computational domain on the coarse level as Chombo does [18], the coarse-to-fine interpolation in COM3D concerns the influence of the complex geometry structures in the computational domain. The linear interpolation method for the coarse-to-fine communication in COM3D is shown as Figure 3.4.

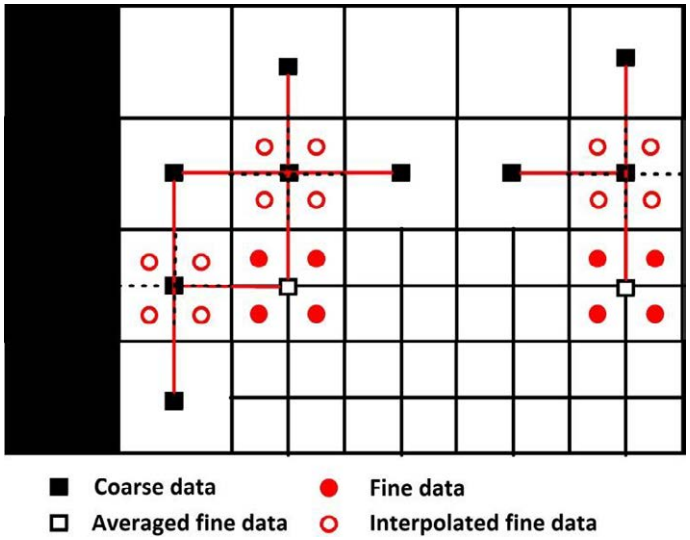



Figure 3.4 Coarse-to-fine interpolation in COM3D with LMR

Additionally, in the patch-based local mesh refinement, temporal interpolation should also be concerned when the refinement in time is included. In the coarse-to-fine interpolation, by considering the most frequently used algorithms (which can produce first and second order accuracy in time-scale) in COM3D, both the linear and parabolic interpolations are used in time-scale to meet the requirements of different algorithms.

It is worth to note that the above second order spatial interpolation just satisfies the requirements of the solvers shown in the Table 3.1, which are all for the calculation of convection. In other words, the spatial linear interpolation can only satisfy the implementation of local mesh refinement in the solution of Euler equations in COM3D. In the implementation of LMR in

solution of NS equations, some special treatments are introduced to the interpolation for maintaining an acceptable accuracy.

3.1.1.2. Data transfer between levels

Coarse-to-fine interpolation discussed in former section is only responsible for boundary updating on fine levels. Besides the boundary updating, some other data updating for the entire finer patches or some regions on the coarse level are also needed.

The coarse-to-fine data transfer is the coarse-to-fine direction data updating in LMR. This type of data communication provides the initial conditions on finer levels and plays a very important role in calculation with adaptive local mesh refinement. Currently, the constant initialization and linear interpolation are available in the LMR in COM3D. If the gradients of the quantities do not have to be maintained in the computational domain, the constant initialization is used to save computational efforts. Otherwise, the linear interpolation is used to maintain the gradients on the finer level.

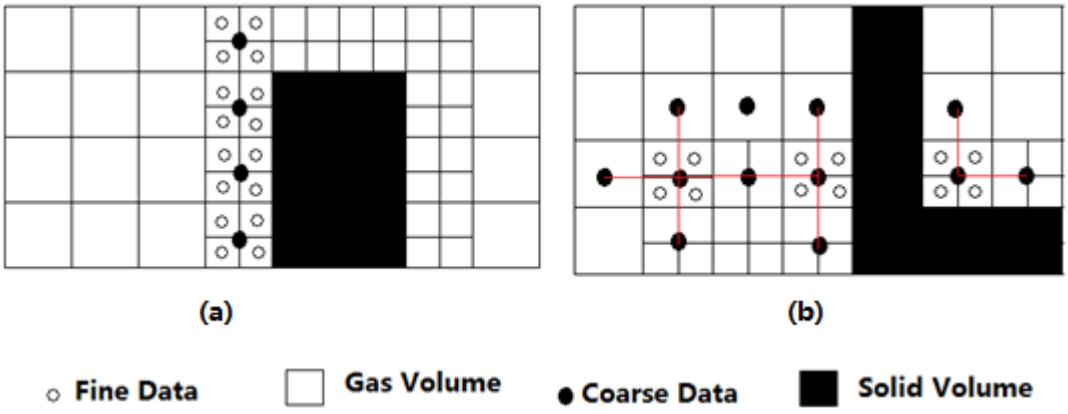



Figure 3.5 Coarse-to-fine data transfer in COM3D

As shown in Figure 3.5, the constant initialization in part (a) indeed is point-wised, the linear interpolation in part (b) has the similar process to the linear coarse-to-fine interpolation.

In the fine-to-coarse data transfer which is in charge of the fine-to-coarse direction data updating, as mentioned in Chapter 2, direct averaging is used in such kind of data communication to maintain the conservation. Since the numerical treatments in the fine-to-coarse data transfer is simple and point-wised, no additional description is given to this data communication.

3.1.1.3. Flux correction

Similarly to the other LMR cases, flux correction has to be considered in the local mesh refinement in COM3D to keep the conservation on fine-to-coarse interface. Since the refinement in time-scale is included in the LMR, the flux correction in COM3D should concern both the temporal averaging and the spatial averaging. The details of flux correction has been given in Chapter 2 already, no further description is necessary here.

#### 3.1.1.4. Multiple time steps

Before the detailed implementation of local mesh refinement in Euler equations, discussion about the importance of temporal refinement in local mesh refinement is made. As mentioned in Chapter 2, one big advantage of patch-based local mesh refinement is the time-scale refinement. Different time steps can be used on different refinement levels under the temporal refinement, such refinement is also called multiple time steps method. For the calculation on coarse level does not need to be done the same frequency as the finer levels, calculation with multiple time steps is more efficient than the calculation based on uniform time step. In Table 3.3, the computation saving by using multiple time steps in LMR with refinement ratio 2 under different finer covered ratio (the area ratio of the region refined on the coarser level) are given.

Table 3.3 Comparison between the multi-time steps and uniform time step

| Finer covered ratio in local mesh refinement | Total computation cost with multiple time steps | Total computation cost wit uniform time steps | Work saving |
|---|---|---|---|
| 1% | 1.16 | 2.16 | 86.2% |
| 5% | 1.8 | 2.8 | 55.6% |
| 10% | 2.6 | 3.6 | 38.5% |
| 15% | 3.4 | 4.4 | 29.4% |
| 20% | 4.2 | 5.2 | 23.8% |
| 25% | 5 | 6 | 20% |
| 30% | 5.8 | 6.8 | 17.2% |

Computation saving of multiple time steps is quite obvious, especially in the cases that the refined regions only cover very small ratio on the base level. It have to be admitted that the benefits of multiple time steps method become less as the increasing of the finer covered ratio. In some implementations, the multiple time steps method is abandoned when the method cannot gain big improvement in computational efficiency and even reduce the accuracy [55]. However, the situation in COM3D is different, the LMR is used in a very small region in large-scale industrial problems mostly. In this situation, the multiple time steps method can increase the computational efficiency significantly. Because of the reasons above, the multiple time step method has to be taken into account in the implementations of LMR.

### 3.1.2. Implementation of LMR

In the solution of Euler equations, the final accuracy actually depends on both the solver and the algorithm. For example, the forwards Euler based van Leer calculation can show first order accuracy in time and global second order accuracy in space, but the calculation approached by the second order RK scheme can have global second order accuracy in both time and space. So, the implementations of LMR should consider the features of each algorithm in COM3D.

#### 3.1.2.1. Forward Euler scheme

The forward Euler (FE) scheme can provide first order accuracy in time, which is the simplest but also the most saving one among the algorithms in COM3D. For large-scale industrial problems, such scheme is frequently used because of its good saving in computational efforts. The (3-5) shows the simplified expression of FE scheme.

$$\frac{\partial U}{\partial t}+\sum_{x_i}\frac{\partial F_i(U)}{\partial x_i}=k(U)\Rightarrow\frac{\partial U}{\partial t}=f(U)$$
$$U_{n+1}=U_n+\Delta t\times f(U_n) \tag{3-5}$$

The implementations of the patch-based local mesh refinement in FE scheme are quite mature [5-6], the optimization in FE based calculation in COM3D can follow the same way. Figure 3.6 shows the process of the forward Euler scheme with LMR in COM3D, the refinement ratio between the two adjacent levels is two.

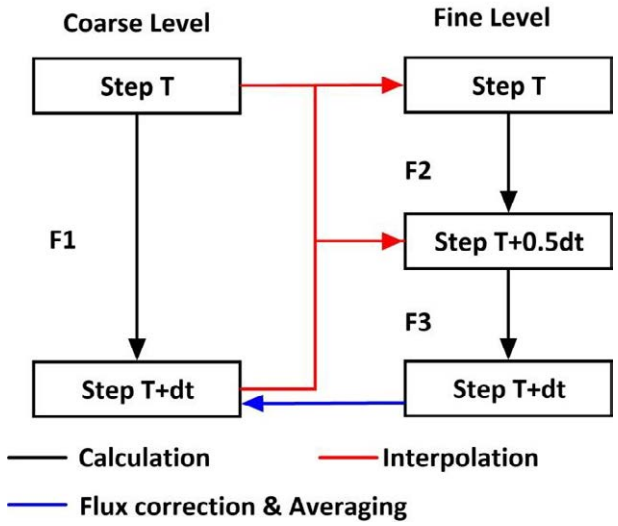



Figure 3.6 FE based solution of Euler equations with LMR

The process of Figure 3.6 is as below.

(1) Calculation is begun with the coarsest level. After one step of forwards explicit calculation, the coarse level can reach the time step T+dt.
(2) Then, one step of calculation on next finer level should be made, and the step T+0.5dt can be achieved.
(3) In the next step, since the first level is the finest refinement level, calculation on this level should be made further. After the boundary updating accomplished by both temporal and spatial interpolations, the fine level can go on the calculation as (2) and reach the step T+dt.
(4) Finally, when the two neighboring levels have all reached the step T+dt, it is the moment to do synchronization. The flux correction and fine-to-coarse data transfer are done on coarse level, and then the coarse-to-fine interpolation is made to update the ghost cells on finer level.

In the view of accuracy, both temporal and spatial second order linear interpolation can satisfy this first order algorithm with the solvers which can provide at the most global second order accuracy in space. The implementation of local mesh refinement in the forward Euler equations is quite mature, detailed truncation error analyses can be avoided here.

3.1.2.2. Alternating direction scheme

Besides the simple first order FE scheme, COM3D also contains second order algorithms. The solver TVNI is a second order 1D method, and the alternating direction (AD) scheme is used to keep its accuracy in high dimension problems. The process of the AD based solution is shown as (3-6).

$$\begin{array}{ll} U^A = U^n + \frac{1}{2}\Delta t \times F_X^n & U^{n+1} = U^D + \frac{1}{2}\Delta t \times F_X^D \\ \downarrow & \uparrow \\ U^B = U^A + \frac{1}{2}\Delta t \times F_Y^A & U^D = U^C + \frac{1}{2}\Delta t \times F_Y^C \\ \downarrow & \uparrow \\ U^{n+\frac{1}{2}} = U^B + \frac{1}{2}\Delta t \times F_Z^B \quad \rightarrow & U^C = U^{n+\frac{1}{2}} + \frac{1}{2}\Delta t \times F_Z^{n+\frac{1}{2}} \end{array} \tag{3-6}$$

In the formulas, the symbol $F_i^k$ means the derivative term in the direction i by using the data on step k. As shown by (3-6), calculation in AD based solution is given direction by direction. The X direction's convection terms are calculated firstly by using half time step 0.5Δt, and the intermediate phase A is reached. Then, based on the results in phase A, calculation is made with the same time step in Y direction to get the data in phase B. After that, the convection terms in direction Z are calculated to achieve the half time intermediate step n+0.5Δt. In order to keep the second order accuracy in time (detailed truncation error analysis for the alternating direction scheme is given in Appendix A), calculations with the opposite order are done in the rest half step of calculation. Finally, after all the six steps' calculations, the result in time step T+Δt can be achieved.

Under the uniform structured grids, implementation of AD based TVNI is quite simple. However, when the local mesh refinement with multiple time steps is implemented, there is one big problem in the boundary updating.

In the calculation with local mesh refinement or parallelism, each data patch contains ghost regions for storing necessary boundary conditions. Normally, these regions only perform as the boundary providers, no calculation should be done. Thus, the coarse-to-fine interpolation should be done after every direction's calculation in AD based simulation in LMR. However, to the local mesh refinement with the refinement in time-scale, boundary updating for the finer levels' ghost cells in the intermediate phases A, B, C, D may have some problems. These 4 phases are man-made phases for better CFL condition and accuracy, no real physical time corresponds to them. If the traditional process mentioned in former section [6] [7] is implemented directly, no data on the next coarser level can be used to make the interpolation. Some other methods rather than the traditional ways should be used in the AD based calculation.

In order to solve the problem of the ghost cells updating, some calculation should be done in the ghost region instead of interpolation. In the AD scheme, one direction's calculation only needs ghost cells in that direction. In addition, ghost regions in other directions have enough boundary conditions in the certain direction. So, those ghost regions can be included in the calculation of

certain direction and reach the same intermediate phases as the internal area does. Such numerical treatment is called ghost region calculation. The ghost region calculation in the 2D case is shown in Figure 3.7.

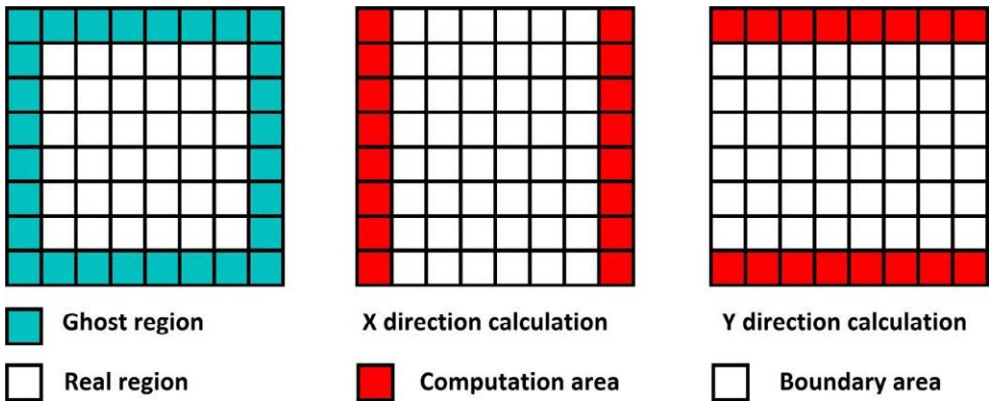



Figure 3.7 Ghost region calculation in 2D AD based calculation

As shown in Figure 3.7, in the calculation of X direction, the ghost cells in this direction (identified by color in the figure) are needed. Since both the internal region and the Y direction's ghost cells have enough boundary conditions in X direction, the two regions can all be included in numerical works and reach the phase A. In the Y direction's calculation, the ghost cells in Y direction are needed and the boundary data have already been in intermediate phase A, the calculation in this direction can be done directly without any inter-level interpolation. After the calculation in Y direction, in these 2D cases, the data reach the half time step intermediate phase $n + 1/2$, in which the coarse-to-fine interpolation work is already possible. In the next half time step's calculation, the ghost region calculation can be done in the same way and assist the data reaching the step $n + 1$. The process of the AD based calculation with ghost region calculation in 3D case actually is the same as the 2D case, expect for one more direction of calculation. So, with the ghost region calculation, the local mesh refinement with temporal refinement is possible for AD based calculation.

Moreover, in the solution of NS equations, totally 4 layers of ghost cells should be given to each patch when the molecular transportation terms is considered (detailed information is given in NS section). Since no calculation for molecular transportation is needed in the pure Euler problem, the AD based calculation with full scale local mesh refinement (both refinement in space and time) can be approached as Figure 3.8.

Detailed process of the calculation shown in figure is as below.

(1) For one big step (coarse time step) of computation, calculation should begin with the base level. One full step of alternating direction based calculation is done on the coarse level and the level can reach time T+dt.
(2) After the calculation on base level, similarly to the calculation in FE scheme, one step of calculation on next finer level should be made. Since double ghost regions are located in each patch, one step of AD based calculation can be made directly without coarse-to-fine interpolation.

(3) As no finer level exists, one more step of calculation should be made on the fine level. Similarly to the implementation in FE scheme, interpolation is done firstly to update the ghost cells on this level (here according to the accuracy of TNVI solver, both linear and parabolic temporal interpolations are acceaptable), then one further step of calculation can be made the same as (2) on this level.

(4) As soon as the second step of calculation on fine level has been finished, synchronization is done between the two neighboring levels.

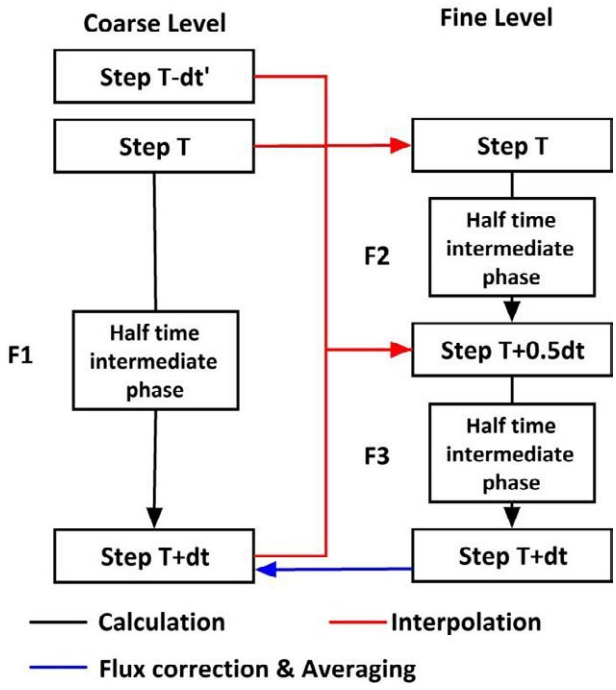



Figure 3.8 AD based solution of Euler equations with LMR

Supported by the mathematical analyses in Appendix B, the implementation of local mesh refinement in AD scheme maintains the same global accuracy as the uniform grid case. It is worth noting again that the process shown in the Figure 3.8 is only for the case that the pure Euler problem is solved. If molecular transportation is considered, inter-level data communication for the half time intermediate phase is necessary.

3.1.2.3. Runge-Kutta scheme

The alternating direction scheme can provide second order accuracy in time for the 1D second order solvers only. For the solvers with first order temporal accuracy, some other algorithms are needed to make them get second order or even higher order accuracy in time-scale. In COM3D, second order Runge-Kutta and fourth order Runge-Kutta are used to get higher temporal accuracy. Formulas (3-7) and (3-8) show the simplified expressions of RK2 and RK4 scheme.

RK2:

$$\frac{\partial x}{\partial t} = f(x)$$
$$k_1 = f(x_n), \quad k_2 = f(x_n + \Delta t \times k_1) \tag{3-7}$$
$$x_{n+1} = x_n + \frac{1}{2}\Delta t(k_1 + k_2)$$

RK4:

$$\frac{\partial x}{\partial t} = f(x)$$
$$k_1 = f(x_n), \quad k_2 = f(x_n + 0.5 \times \Delta t \times k_1)$$
$$k_3 = f(x_n + 0.5 \times \Delta t \times k_2), \quad k_4 = f(x_n + \Delta t \times k_3) \tag{3-8}$$
$$x_{n+1} = x_n + \frac{1}{6}\Delta t(k_1 + 2k_2 + 2k_3 + k_4)$$

The Runge-Kutta scheme is used frequently in the numerical solution of ODE. For the solution of PDE, the scheme is used to increase the accuracy in time. Detailed mathematical analyses for the above two schemes are given in Appendix C. In the analyses, it is worth to note that the intermediate phases $x_n + \Delta t \times f(x_n)/2$ , $x_n + \Delta t \times f(x_1)/2$ and $x_n + \Delta t \times f(x_2)$ in RK4 scheme have the third order accuracy in time at the most, but the higher order terms in each phase are very important in reaching higher order accuracy.

.

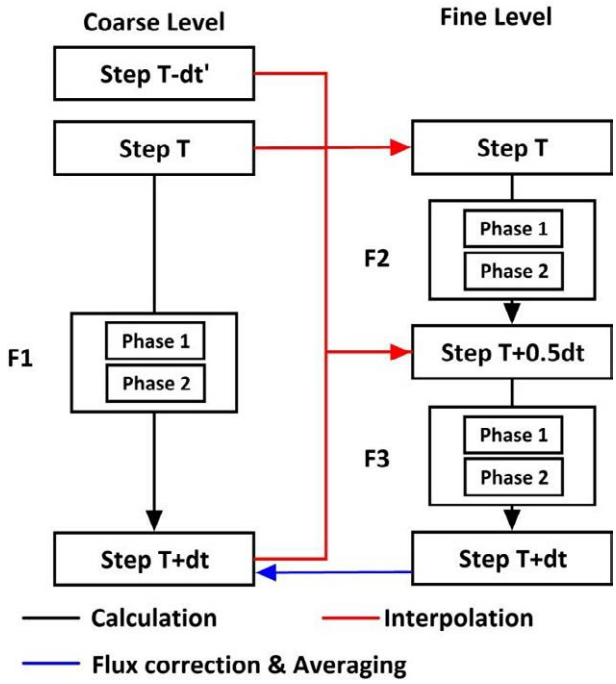



Figure 3.9 RK2 with LMR in the solution of Euler equations

In general, the implementation of local mesh refinement in RK2 is quite similar to the AD case, the four layers of ghost cells help avoiding the inter-level data communication for the intermediated phase of the scheme as well. The only difference is the third order temporal

interpolation in the inter-level data communication, which is for maintaining the high order temporal accuracy. The process of RK2 under the LMR is shown in Figure 3.9

Comparing to the implementation in second order RK scheme, the implementation of LMR in RK4 is more complicated. In common full scale local mesh refinements, the linear or parabolic temporal interpolations are used to make the coarse-to-fine interpolation in time. However, to the RK4 scheme, the two temporal interpolations cannot satisfy this fifth order algorithm. In the implementations of patch-based local mesh refinement for high temporal accuracy algorithms, such as RK3 and RK4, the multiple time steps method is usually not considered [55] [76] [85].

Unlike the other implementations, local mesh refinement with multiple time steps can be made in RK4 in COM3D. Currently, the solvers available in the software can provide second order accuracy in space at the most. Under the explicit ways, spatial and temporal step should satisfy the CFL condition (3-9).

$$CFL = \frac{\Delta t \cdot (c + |u|)}{\Delta x} \leq 0.2 \tag{3-9}$$

No matter how good accuracy can be achieved in time-scale, the total accuracy is second order in current COM3D. In other words, maintaining the second order temporal accumulated accuracy is enough. Similarly to the implementation in RK2, the third order parabolic temporal interpolation is used in the implementation of LMR in RK4. With the double layers of ghost cells and ghost region calculation, one coarse-to-fine interpolation is necessary between intermediate phases 2 and 3.

Besides, how to do the flux correction may be a problem in the implementation of LMR. In some uniform time step LMR, the flux correction is done once at the end and no other flux corrections are needed for the intermediate phases. In the implementation of LMR in COM3D, the flux correction in RK4 is done in the same way.

Figure 3.10 shows the detailed process of the RK4 based calculation with LMR. The total process shown in the Figure 3.10 is as below.

(1) One step of RK4 based calculation with the time step dt is begun with the coarsest level.
(2) In the next step, calculation on the finer level is done. One coarse-to-fine interpolation is done to update the boundaries between the calculations of phase 2 and phase 3. After the RK4 based calculation, fine level can reach the time T+0.5dt.
(3) Before the further step of calculation on finer level, coarse-to-fine interpolation is made to fill the ghost regions. Then, the calculation on the finer level is done the same as (2).
(4) When the two refinement levels reach the same time, synchronization is made between them.

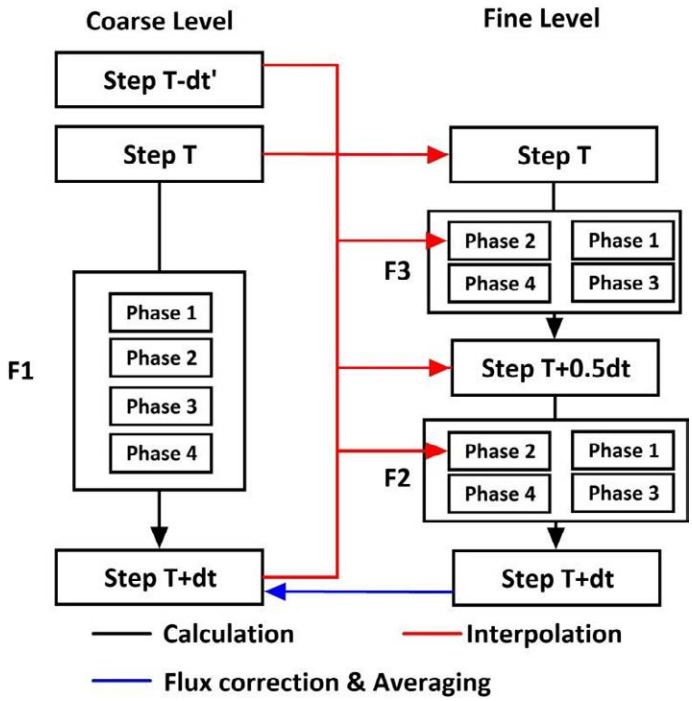



Figure 3.10 RK4 with full-scale LMR in solution of Euler equations

With the above process, the original fourth order scheme is reduced to second order in some cells (Appendix D). If the fourth order accuracy has to be maintained, implementation of LMR in RK4 can be done as Figure 3.11.

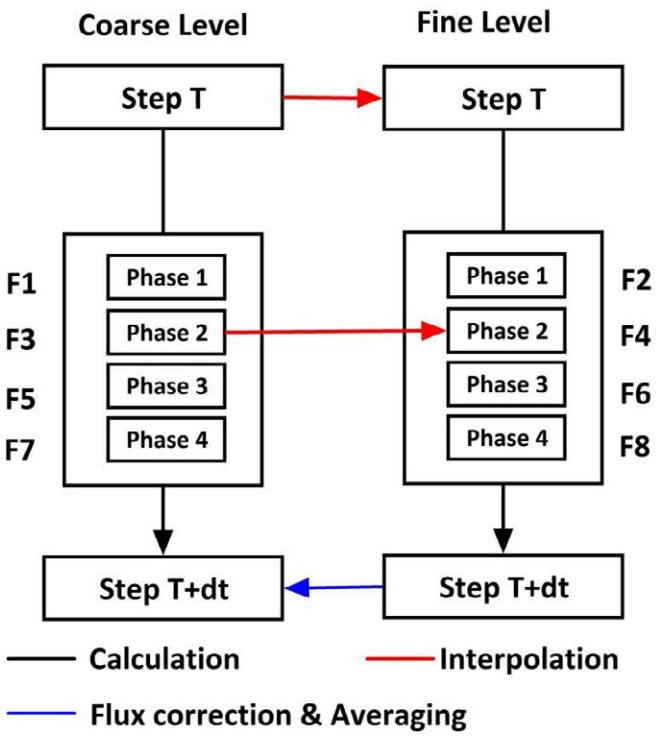



Figure 3.11 Uniform step LMR for RK4

In uniform step LMR, shifting of the calculation is by phases instead of by steps. The inter-level data communication should be done between two intermediate phases. In the uniform step calculation, no temporal truncation errors are produced by the interpolation, the RK4 scheme can maintain its high temporal accuracy.

Detailed analyses for the truncation error brought by LMR are given in Appendix D. The analyses show that the RK2 can maintain the same accuracy as the uniform grid calculations, but the RK4 scheme is reduced to a second order scheme around the fine-to-coarse interface.

### 3.1.3. Testing cases

The technique of patch-based LMR has been implemented in the solution of Euler equations, some testing cases will be made to show the validation. The validation of the implementation in solvers is proved by the simulation of sod shock tube, and the validation of the implementations of LMR in algorithms is proved by the solution of high order Riemann problem.

#### 3.1.3.1. The sod shock tube

The sod shock tube, which is a very classical testing case in CFD, is used to test the implementation of local mesh refinement. The domain and initial statement of this 1D problem are shown by Figure 3.12 and Table 3.4.

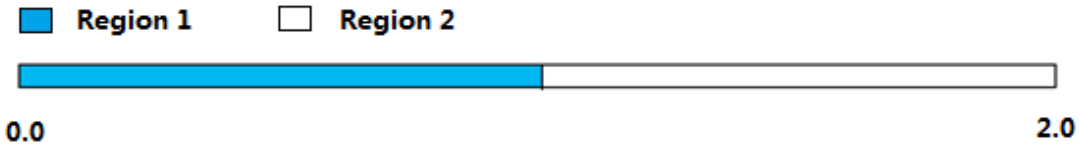



Figure 3.12 Computational domain of sod shock tube

Table 3.4 Initial condition & Grids of the sod shock tube

| **Grid Information** | |
|---|---|
| **Computational Domain:** $(0,0,0)\times(199,1,1)$<br>**Cell Size: 0.01m** | |
| **Initial Condition** | |
| **Region 1** | **Region 2** |
| **Domain:** $(0,0,0)\times(99,1,1)$<br>**Pressure:1Bar**<br>**Temperature:300K**<br>**Velocity: 0, 0, 0**<br>**Gas Spices (mole ratio):**<br>$H_2:O_2:N_2:H_2O=2:1:3.76:0$ | **Domain:** $(100,0,0)\times(199,1,1)$<br>**Pressure:0.1Bar**<br>**Temperature:274K**<br>**Velocity: 0, 0, 0**<br>**Gas Spices (mole ratio):**<br>$H_2:O_2:N_2:H_2O=2:1:3.76:0$ |

In the simulation, a shock wave is generated and propagates from the middle to the right. The pressure in analytical solution at T=0.01s is shown in Figure 3.13.

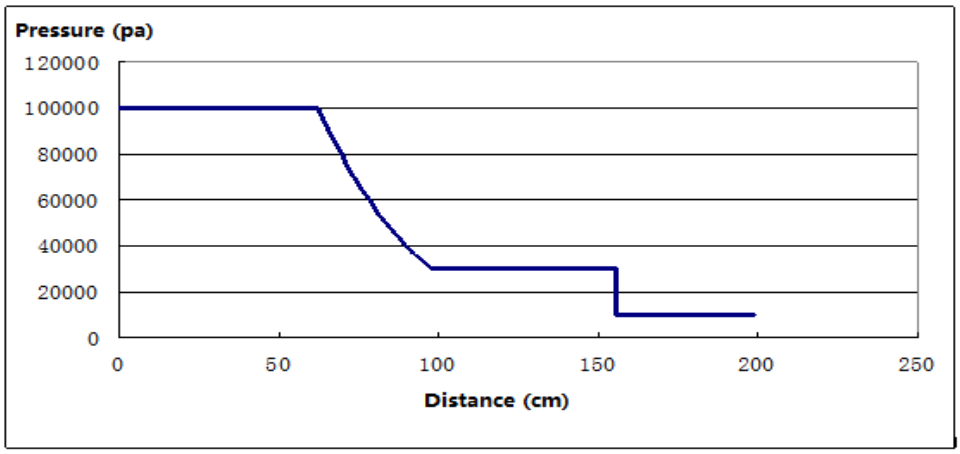



Figure 3.13 Analytical solution of the sod shock tube at T=0.01s

In the numerical solution, since the use of homogeneous discrete, final results are usually influenced by the numerical diffusion. Besides the simulation with the 1 cm grids, another simulation using two times finer grids is made simultaneously to control the unphysical diffusion and to show the advantages of high resolution. The results of the two simulations and the analytical solution at the time 0.01s are shown in Figure 3.14.

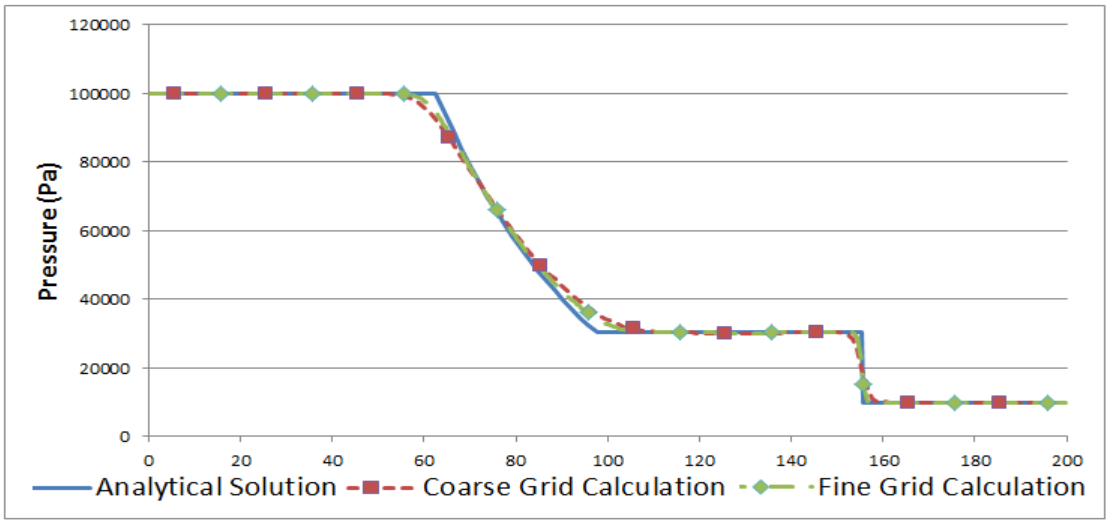



Figure 3.14 Comparison between the calculations and analytical solution

Shown by the above comparison, simulation with finer grids has better performance in shock front. The local mesh refinement can be made locally and statically from 2/5 to 4/5 of whole tube length where the shock passes to optimize the performance. The comparison between results of local mesh refinement and fine grid calculation is shown in Figure 3.15.

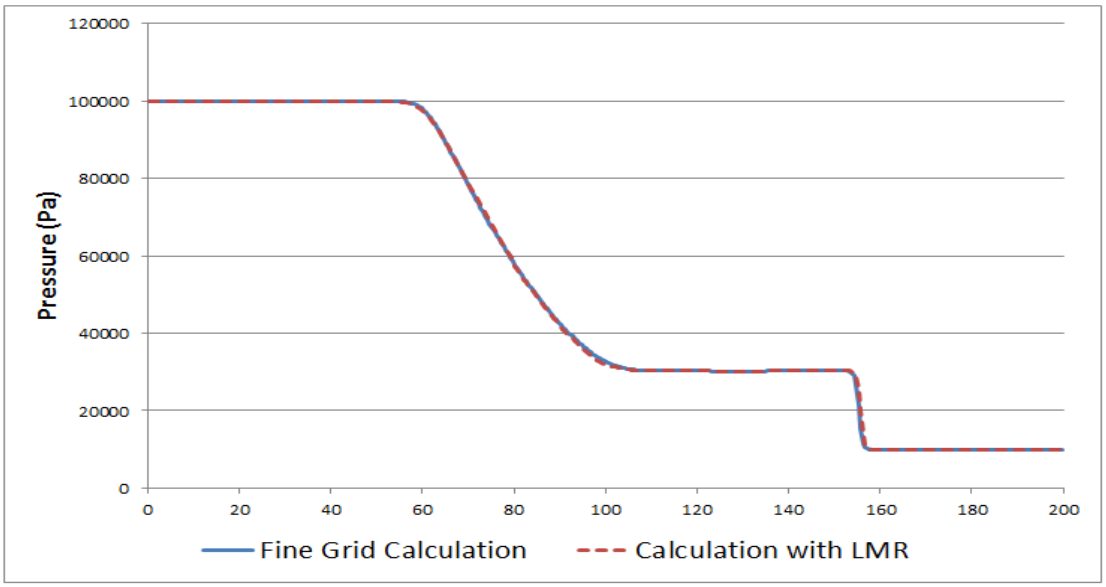



Figure 3.15 Comparison between local mesh refinement and full refinement

The comparison shows the local mesh refinement can give almost the same performance in the shock simulation as the full grid refinement. The simulation in coarse part in the local mesh refinement case also shows better result than the coarse grid solution.

The propagation of shock in 1D case is quite simple, the local mesh refinement technique can be used adaptively in a simple way to show its advantages in time saving. Different refinement covered ratios are used for the adaptive attempting. The following Figure 3.16 shows the total time cost of full grids refinement case and LMR with the covered ratio 12.5%, 10% and 5%:

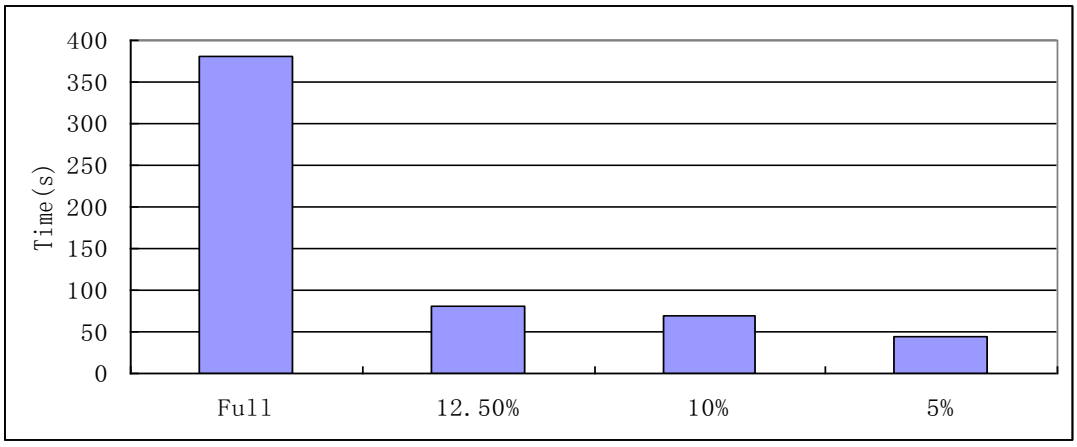



Figure 3.16 Total time cost of the full refinement case and local mesh refinement cases

As shown in Figure 3.16, the local mesh refinement can save large ratio of computation time. Comparison between the results of coarse grids calculation and calculation with adaptive local mesh refinement (ALMR) shows that ALMR optimizes the performance in shock simulation.

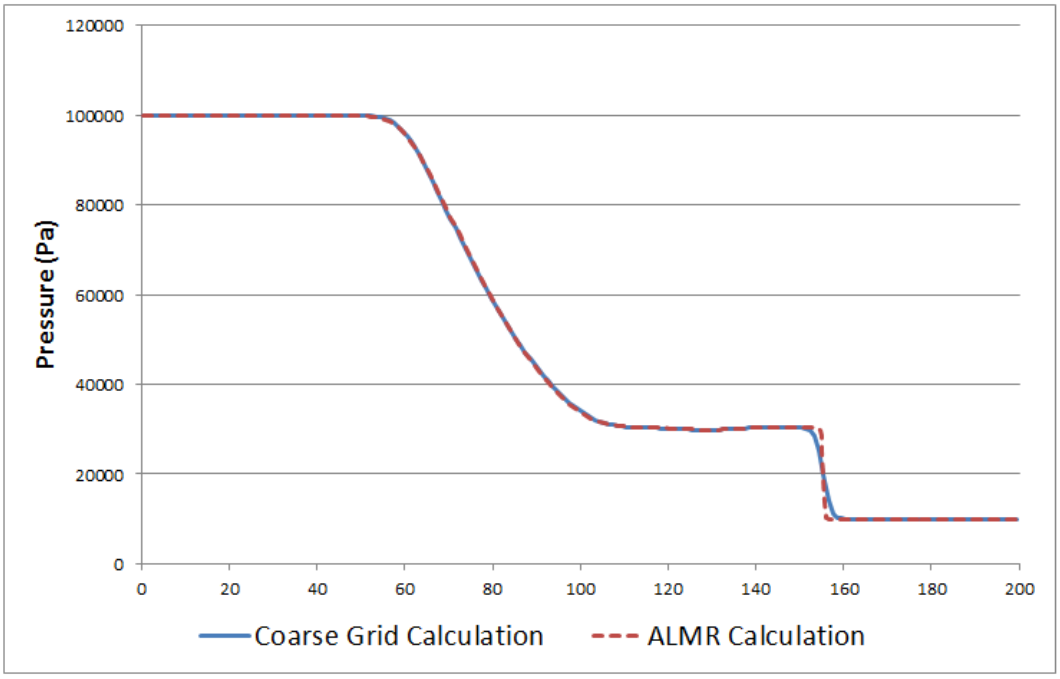



Figure 3.17 Comparison between the simulation with adaptive LMR and coarse grid simulation

3.1.3.2. The 2D Riemann problem

The sod shock tube can verify the implementation of local mesh refinement in solvers. The validation of the implementation in algorithms should be carried out by high dimension testing cases. The extended 1D Riemann problem (the sod shock tube) is used. Detailed information about the local mesh refinement and initial conditions in the 2D testing case is given by Figure 3.18.

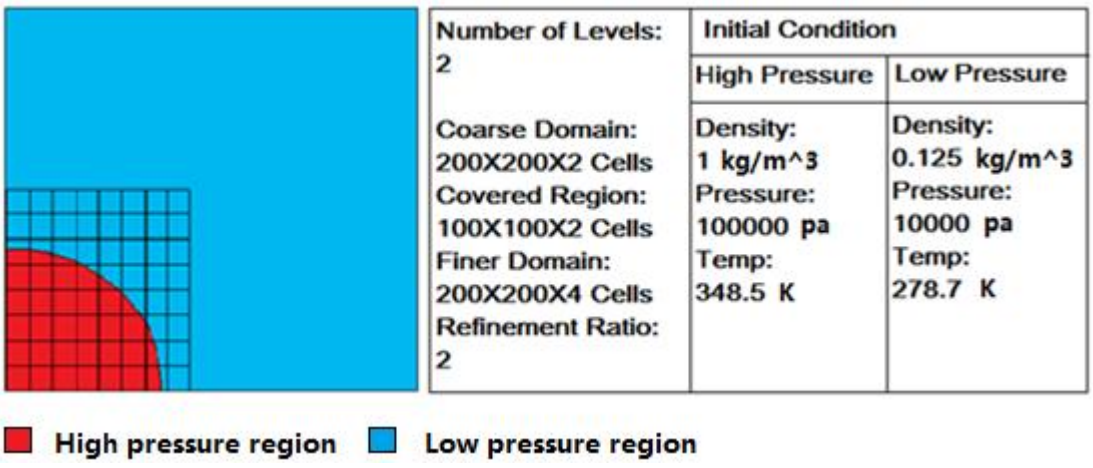



Figure 3.18 Initial condition and grids of the extended Riemann Problem

In the calculation with local mesh refinement, numerical reflection or oscillation may be produced around the fine-to-coarse interface [3] [14], especially in the cases that big gradient passes through the fine-to-coarse interface. As shown in Figure 3.19, the high pressure area is covered in the finer region at the very beginning, and the shock wave front will pass through the fine-to-coarser

interface at some time. This 2D testing case is used to study whether the big numerical reflection or oscillation can be produced by current LMR treatment or not. In the simulation, both AD based TVNI method and RK4 based van Leer method are used. The AD based TVNI simulation is shown in the following figures, in which the fine-to-coarse interfaces are identified by black solid lines. In Figure 3.19, distribution of pressure is described at the moment when the shock wave in the horizontal direction has just passed through the interface. Figure 3.20 shows the pressure values at the time when the whole shock wave has passed through the interface.

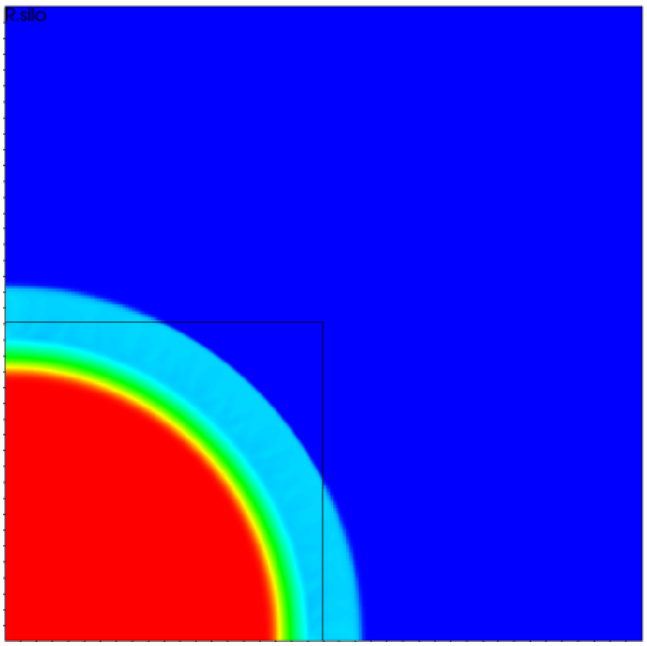



Figure 3.19 Pressure of the extended 2 D Riemann problem under local refinement at T=0.0045s

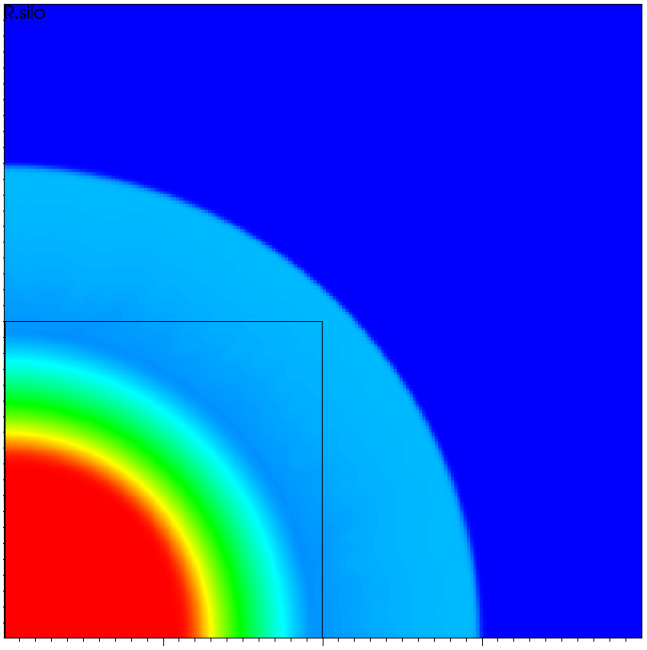



Figure 3.20 Pressure of the extended 2 D Riemann problem under local refinement at T=0.01s

In the above two figures, no big visible reflection or oscillation can be found around the fine-to-coarse interface. To verify the results further, pressure values of the simulation with LMR are compared to the pressure values of the simulation with full grids refinement. Following Figure 3.21 and Figure 3.22 show the comparison of the pressure values in horizontal direction.

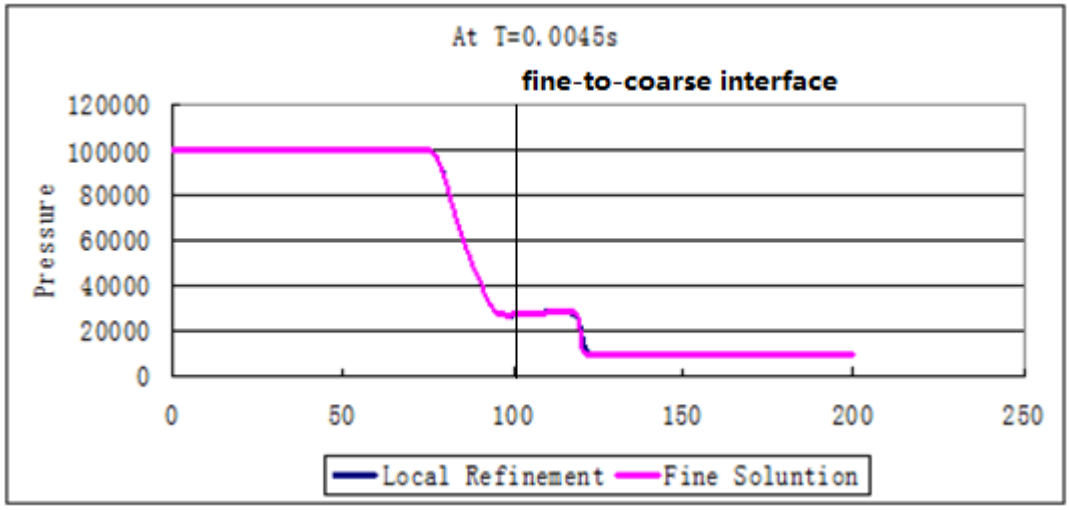



Figure 3.21 Comparison between the local mesh refinement and full refinement in the pressure at T=0.0045s.

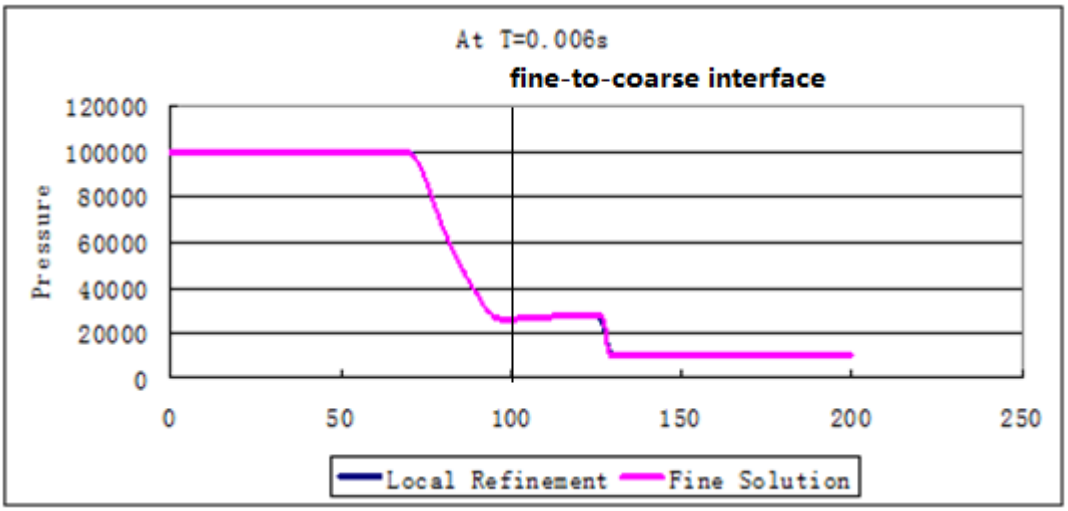



Figure 3.22 Comparison between the local mesh refinement and full refinement in the pressure at T=0.006s.

In the figures, both two solutions show almost the same results in the region around the fine-to-coarse interface. The differences between the two curves are in the expression of the shock, which is caused by the differences in resolutions. So, the implementation of LMR in the solution of Euler equations may not cause big numerical reflection or oscillations around fine-to-coarse interface. The simulation made by the RK4 based van Leer method can also give the similar result, but they are not repeated here.

In all, the testing for solvers and algorithms has shown rationality of the LMR in solution of Euler equations. In next steps, optimization for NS equations and detonation can be made based on the LMR work in Euler.

### 3.2. Local mesh refinement in the solution of NS equations

In this section, the implementation of LMR in the solution of standard NS equations is done, the implementation is very important to the optimization of all the RANS models. The NS equations used in COM3D are as following:

$$
\begin{cases}
\dfrac{\partial \rho}{\partial t}+\dfrac{\partial \rho u}{\partial x}+\dfrac{\partial \rho v}{\partial y}+\dfrac{\partial \rho w}{\partial z}=0 \\
\dfrac{\partial \rho u}{\partial t}+\dfrac{\partial \rho uu}{\partial x}+\dfrac{\partial \rho uv}{\partial y}+\dfrac{\partial \rho uw}{\partial z}=-\dfrac{\partial p}{\partial x}+\dfrac{\partial \tau_{xx}}{\partial x}+\dfrac{\partial \tau_{xy}}{\partial y}+\dfrac{\partial \tau_{xz}}{\partial z} \\
\dfrac{\partial \rho v}{\partial t}+\dfrac{\partial \rho vu}{\partial x}+\dfrac{\partial \rho vv}{\partial y}+\dfrac{\partial \rho vw}{\partial z}=-\dfrac{\partial p}{\partial y}+\dfrac{\partial \tau_{yx}}{\partial x}+\dfrac{\partial \tau_{yy}}{\partial y}+\dfrac{\partial \tau_{yz}}{\partial z} \\
\dfrac{\partial \rho w}{\partial t}+\dfrac{\partial \rho wu}{\partial x}+\dfrac{\partial \rho wv}{\partial y}+\dfrac{\partial \rho ww}{\partial z}=-\dfrac{\partial p}{\partial z}+\dfrac{\partial \tau_{zx}}{\partial x}+\dfrac{\partial \tau_{zy}}{\partial y}+\dfrac{\partial \tau_{zz}}{\partial z} \\
\dfrac{\partial \rho e}{\partial t}+\dfrac{\partial \left(\rho eu+pu\right)}{\partial x}+\dfrac{\partial \left(\rho ev+pv\right)}{\partial y}+\dfrac{\partial \left(\rho ew+pw\right)}{\partial z}=\dfrac{\partial}{\partial x}\left(\lambda \dfrac{\partial T}{\partial x}\right)+ \\
\dfrac{\partial}{\partial y}\left(\lambda \dfrac{\partial T}{\partial y}\right)+\dfrac{\partial}{\partial z}\left(\lambda \dfrac{\partial T}{\partial z}\right)+\sum H_i \sum \dfrac{\partial}{\partial x_j}\rho D_i \dfrac{\partial Y_i}{\partial x_j}+\dfrac{\partial}{\partial x}\left(u\tau_{xx}+v\tau_{yx}+w\tau_{zx}\right)+ \\
\dfrac{\partial}{\partial y}\left(u\tau_{xy}+v\tau_{yy}+w\tau_{zy}\right)+\dfrac{\partial}{\partial z}\left(u\tau_{xz}+v\tau_{yz}+w\tau_{zz}\right) \\
\dfrac{\partial Y_i}{\partial t}+\dfrac{\partial Y_i u}{\partial x}+\dfrac{\partial Y_i v}{\partial y}+\dfrac{\partial Y_i w}{\partial z}=\dfrac{\partial}{\partial x}\rho D_i \dfrac{\partial Y_i/\rho}{\partial x}+\dfrac{\partial}{\partial y}\rho D_i \dfrac{\partial Y_i/\rho}{\partial y}+\dfrac{\partial}{\partial z}\rho D_i \dfrac{\partial Y_i/\rho}{\partial z}
\end{cases}
\tag{3-10}
$$

$$
\tau_{ij}=2\mu S_{ij},\quad S_{ij}=\frac{1}{2}\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)-\frac{1}{3}\delta_{ij}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}+\frac{\partial w}{\partial z}\right)
$$

μ—mix viscosity of the fluid, λ—mix heat transfer coefficient, $D_i$—mix gas diffusion coefficient of $Y_i$

In the detailed numerical work in COM3D, solution of the NS equations is based on the solution of Euler equations and the solution of molecular transportation (3-11).

$$V_{M_x} = \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z}$$
$$V_{M_y} = \frac{\partial \tau_{yx}}{\partial x} + \frac{\partial \tau_{yy}}{\partial y} + \frac{\partial \tau_{yz}}{\partial z}$$
$$V_{M_z} = \frac{\partial \tau_{zx}}{\partial x} + \frac{\partial \tau_{zy}}{\partial y} + \frac{\partial \tau_{zz}}{\partial z}$$
$$T_e = \frac{\partial}{\partial x}\left(\lambda \frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(\lambda \frac{\partial T}{\partial y}\right) + \frac{\partial}{\partial z}\left(\lambda \frac{\partial T}{\partial z}\right)$$
$$D_e = \sum H_i \sum \frac{\partial}{\partial x_j} \rho D_i \frac{\partial Y_i}{\partial x_j}$$
$$V_e = \frac{\partial}{\partial x}\left(u\tau_{xx} + v\tau_{yx} + w\tau_{zx}\right) + \frac{\partial}{\partial y}\left(u\tau_{xy} + v\tau_{yy} + w\tau_{zy}\right) + \frac{\partial}{\partial z}\left(u\tau_{xz} + v\tau_{yz} + w\tau_{zz}\right)$$
$$D_{Y_i} = \frac{\partial}{\partial x} \rho D_i \frac{\partial Y_i/\rho}{\partial x} + \frac{\partial}{\partial y} \rho D_i \frac{\partial Y_i/\rho}{\partial y} + \frac{\partial}{\partial z} \rho D_i \frac{\partial Y_i/\rho}{\partial z}$$

(3-11)

$V_{M_i}$— Viscosity term of the momentum in the direction i
$T_e$— Heat transfer term in the energy equation
$V_e$— Energy changes made by the viscosity of the fluid
$D_e$— Energy changes made by the diffusion of gas species
$D_{Y_i}$— Diffusion term of the gas species in the specie i

In sub-section 3.1, it has already been proved that the linear spatial interpolation can satisfy the requirements of the convection solvers. Differently from the convection term, the molecular transportation terms are second order derivatives which may require higher order interpolation for keeping the accuracy. Before the discussion about detailed implementation of local mesh refinement in NS equations, some treatments for maintaining the accuracy are introduced.

### 3.2.1. Preparing for LMR in solution of NS equations

#### 3.2.1.1. Parabolic and linear hybrid interpolation

In the coarse-to-fine interpolation, if the interpolated results have second order accuracy in space and time, the problem shown by (3-12) may occur (under the explicit method, the stability of the molecular transportation requires $\mu \times \Delta t / \Delta x^2 \le 0.4$).

$$\begin{cases} \tilde{u}_{n-1} = u_{n-1} + O(\Delta t^2) + O(\Delta x^2) \\ \tilde{u}_n = u_n \\ \tilde{u}_{n+1} = u_{n+1} \end{cases} \tag{3-12}$$

$$\frac{\partial^2 u}{\partial x^2} = \frac{\tilde{u}_{n+1} + \tilde{u}_{n-1} - 2\tilde{u}_n}{\Delta x^2} = \frac{u_{n+1} + u_{n-1} - 2u_n}{\Delta x^2} + O(1) + O(\Delta t)$$

It is clear that the second order interpolated boundary can result in the similar truncation error in the solution of molecular transportation, some higher order interpolations are required to avoid such big errors. However, as mentioned in section 3.1, the simple high order interpolations may result in un-conservation in the ghost regions in COM3D. In order to avoid the truncation errors produced by the linear interpolation and to keep the conservation of those physical quantities, the extra correction terms should be used in the coarse-to-fine interpolation.

In the calculation of molecular transportation, the forward Euler scheme and the second order RK scheme are used. Essentially, the calculation of molecular transportation is based on the one step forward Euler scheme which requires one layer of ghost cells. In order to unify the expressions of both convection solution and solution of molecular transportation, the forward Euler scheme used in the solution of molecular transportation can be turned into the format (3-13).

$$u_{n+1} = u_n + \Delta t \times \sum_{i=1}^{d} (F_R^i(u) - F_L^i(u)) / \Delta x_i \tag{3-13}$$

In (3-13), the explicit solution of molecular transportation is expressed as the differences of the flux values. Differently from the fluxes in convection, the fluxes here are first order derivatives. In order to maintain the accuracy of the flux values in Euler part, the interpolation in inter-level communication should be at the least third order in accuracy. The third order parabolic interpolation in time has already been established in Euler part of work, but the third order spatial interpolation is unavailable.

Actually, the high order spatial interpolation is not required in every ghost cells. In section 3.1, the mathematical analyses show that the influences of second order spatial errors can be reduced to $O(\Delta t \cdot \Delta x) \sim O(\Delta x^3)$ after the calculation for convection. However, in the calculation of molecular transportation, truncation error in time and space only keep the same order after the calculation. In the convection part of calculation, the original third order accuracy points can still keep the same order of accuracy when its neighbors are second order in accuracy; but in the calculation of molecular transportation, the original third order accuracy points are reduced to second order if some neighbors are second order in accuracy.

In current implementation of LMR, several things should be noticed. Firstly, the calculation of molecular transportation requires one layer of ghost cells in the forward Euler approach, and the calculation with second order RK scheme requires 2 layers of ghost cells. Then, after no more than two steps of forward Euler or one step of RK2 based calculation, the ghost regions on finer levels have to be updated. Thirdly, the final accuracy is considered in the internal region, the results in ghost regions are not important. Therefore, for achieving at least first order accuracy, the four layers of ghost cells should be updated by different interpolations. The inner two or one layers of ghost cells should be updated by the third order interpolation in space, and the others should be filled with second order method. In order to keep the conservation, extra values produced by the third order interpolation are given to those second order ghost cells (those extra values are second order terms and cannot reduce the accuracy).

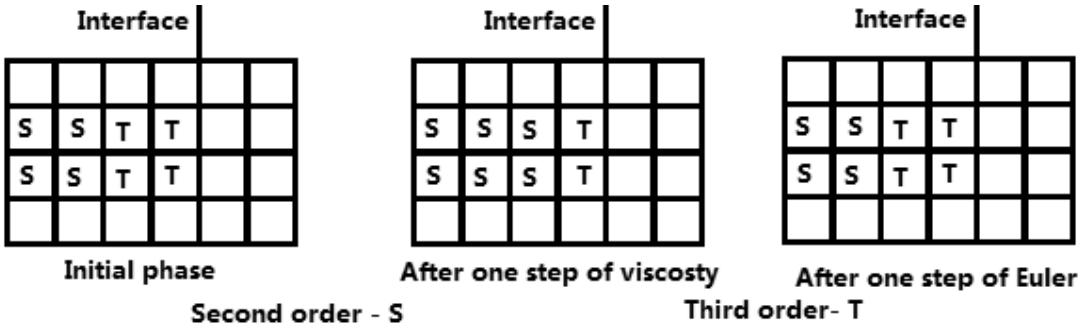



Figure 3.23 Accuracy of the ghost cells after the calculation

If the parabolic and linear hybrid interpolation is used to update the boundary, the internal regions always have the third order accuracy boundary in the calculation of molecular transportation. In the forward Euler scheme, one inner layer of ghost cells updated by parabolic interpolation can satisfy the requirement in accuracy. In the AD based calculation, the numerical work for molecular transportation is made through the RK2 scheme which requires two layers of ghost cells. In the simulation for viscid flow, ghost cells in finer patches must be updated every half step of calculation, and the two inner layers of third order accuracy ghost cells are enough for maintaining the accuracy. In the RK scheme, ghost cells should be updated after every two steps of calculations. In the calculation of the molecular transportation part in each intermediate phase, boundary conditions of the non-ghost regions always have third order accuracy. As a result, the parabolic and linear hybrid interpolation can avoid the big errors in calculation of molecular transportation and maintain the conservation.

Figure 3.24 shows the parabolic and linear hybrid interpolation under the refinement ratio two and refinement ratio four. In principle, for higher order scheme, at least the refinement ratio four should be used to have two layers of third order ghost cells. However, to the calculation with forward Euler scheme, only one layer of third order ghost cells are enough. So, in the solution of NS equations, at least refinement ratio four is required by the high order schemes, but the FE based calculation does not have such limitation.

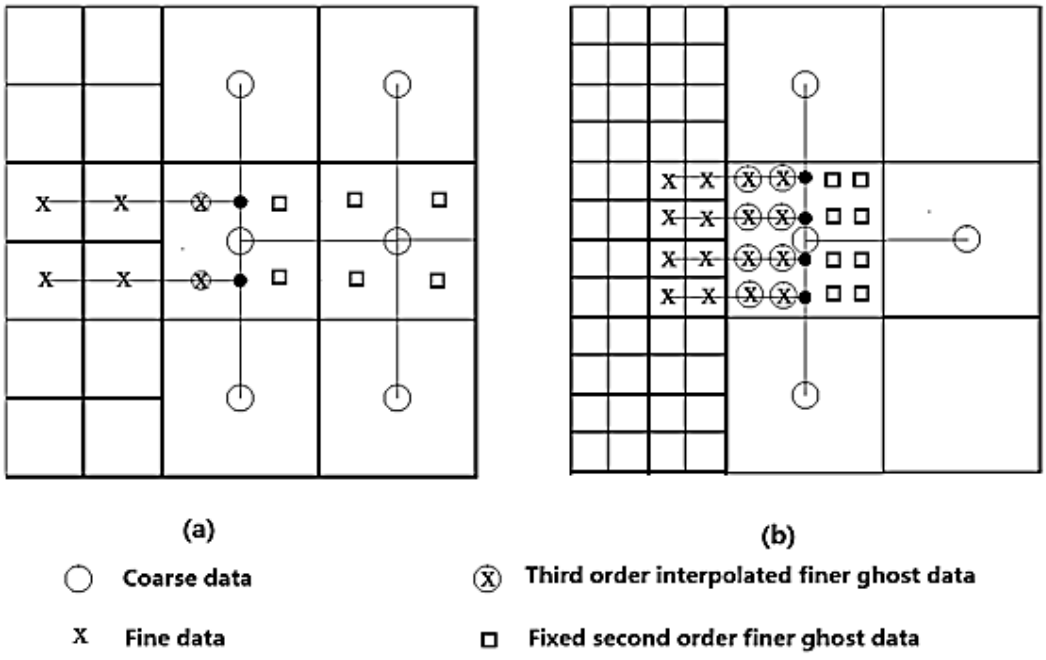



Figure 3.24 Parabolic and linear hybrid interpolation

3.2.1.2. Flux limiter on coarse level

Similarly, on the coarse level, the second order spatial errors produced by the fine-to-coarse data transfer can also bring big truncation errors in the calculation of molecular transportation. Especially to those schemes containing several intermediate phases, the big errors can be spread to

a large region on coarse level. In order to control such big truncation error, the special flux limiters should be used.

Actually, truncation error can be avoided by removing the second order term (3-14).

$$k\times(\frac{\partial^2 u}{\partial x^2}\Delta x^2+\frac{\partial^2 u}{\partial y^2}\Delta y^2+\frac{\partial^2 u}{\partial z^2}\Delta z^2)=k\times\Delta h^2\Delta u \tag{3-14}$$

The constant $k$ in (3-14) depends on the detailed refinement ratio used between the two adjacent refinement levels. Figure 3.25 shows the method in the calculation of flux limiters.

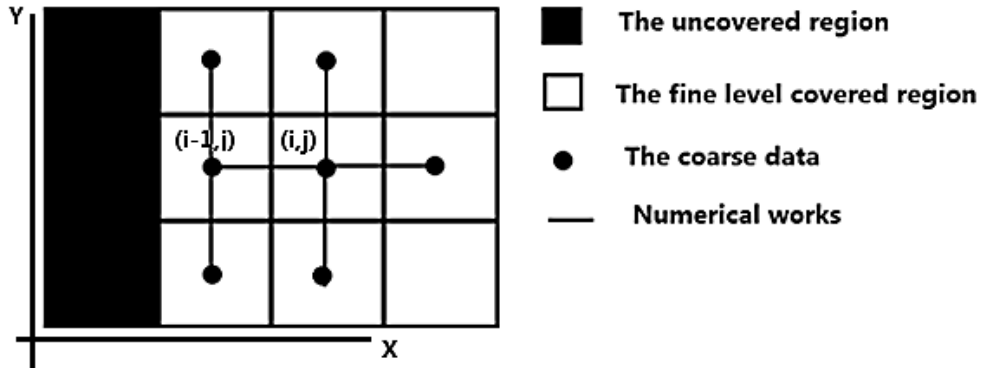



Figure 3.25 Calculation of flux limiter

In the calculation of the flux limiter, the data in the covered coarse regions can only be used. In the calculation of second order derivatives in the directions without the interface (as the Y direction in Figure 3.25) or far from the interface (as the X direction of point (i , j)), central differences can be used to get the value.

$$\begin{aligned}\frac{\partial^2 u}{\partial x^2}|_i&=\frac{u_{i+1}+k\times\Delta x^2\Delta u|_{i+1}+u_{i-1}+k\times\Delta x^2\Delta u|_{i-1}-2u_i-2k\times\Delta x^2\Delta u|_i}{\Delta x^2}\\&=\frac{u_{i+1}+u_{i-1}-2u_i}{\Delta x^2}+O(\Delta x^2)\end{aligned} \tag{3-15}$$

In the calculation of the second order derivatives in the directions with the interfaces (as the X direction of point (i-1, j)), the one side difference is used. Formulas in (3-16) is the right hand side calculation method, the left hand side is similar.

$$\begin{aligned}k_1&=\frac{u_{i+1}+k\times\Delta x^2\Delta u|_{i+1}-u_i-k\times\Delta x^2\Delta u|_i}{\Delta x}=\frac{\partial u}{\partial x}|_i+\frac{\Delta x}{2}\frac{\partial^2 u}{\partial x^2}|_i+O(\Delta x^2)\\k_2&=\frac{u_{i+2}+k\times\Delta x^2\Delta u|_{i+2}-u_i-k\times\Delta x^2\Delta u|_i}{2\Delta x}=\frac{\partial u}{\partial x}|_i+\Delta x\frac{\partial^2 u}{\partial x^2}|_i+O(\Delta x^2)\\\frac{2(k_2-k_1)}{\Delta x}&=\frac{\partial^2 u}{\partial x^2}|_i+O(\Delta x)\end{aligned} \tag{3-16}$$

As a result, the third order spatial accuracy can be maintained at least in the calculation of the flux limiter. In the calculation of the flux of molecular transportation on fine-to-coarse interface, when no flux limiter is used, the truncation error will be like (3-17).

$$\frac{u_{i+1}-u_{i-1}+k\times\Delta x^2\Delta u\,|_{i-1}}{\Delta x}=\frac{\partial u}{\partial x}\,|_{i-\frac{1}{2}}+O(\Delta x) \tag{3-17}$$

When the flux limiter is used, the truncation errors can be reduced as shown in (3-18).

$$\frac{u_{i+1}-u_i+k\times\Delta x^2\Delta u\,|_i-k\times\Delta x^2\Delta u\,|_i-O(\Delta x^3)}{\Delta x}=\frac{\partial u}{\partial x}\,|_{i+\frac{1}{2}}+O(\Delta x^2) \tag{3-18}$$

For the calculation of the first order derivation in the cells center, when both covered and uncovered coarse cells are related, the flux limiter should be used to control the big errors as well.

$$\frac{u_{i+1}-u_{i-1}+k\times\Delta x^2\Delta u\,|_{i-1}-k\times\Delta x^2\Delta u\,|_{i-1}-O(\Delta x^3)}{2\Delta x}=\frac{\partial u}{\partial x}\,|_i+O(\Delta x^2) \tag{3-19}$$

In the other flux calculations or cell center derivative calculations, the truncation errors are shown as the equations in (3-20).

$$\begin{aligned}
F_{i+\frac{1}{2}}^{covered}&=\frac{u_{i+1}+k\times\Delta x^2\Delta u\,|_{i+1}-u_i-k\times\Delta x^2\Delta u\,|_i}{\Delta x}=\frac{\partial u}{\partial x}\,|_{i+\frac{1}{2}}+O(\Delta x^2)\\
F_i^{covered}&=\frac{u_{i+1}+k\times\Delta x^2\Delta u\,|_{i+1}-u_{i-1}-k\times\Delta x^2\Delta u\,|_{i-1}}{2\Delta x}=\frac{\partial u}{\partial x}\,|_i+O(\Delta x^2)\\
F_{i+\frac{1}{2}}^{uncovered}&=\frac{u_{i+1}-u_i}{\Delta x}=\frac{\partial u}{\partial x}\,|_{i+\frac{1}{2}}+O(\Delta x^2)\\
F_i^{uncovered}&=\frac{u_{i+1}-u_{i-1}}{2\Delta x}=\frac{\partial u}{\partial x}\,|_i+O(\Delta x^2)
\end{aligned} \tag{3-20}$$

Therefore, all the flux values or center values can keep second order accuracy in space. In the calculation of molecular transportation, the big errors can be avoided. In the calculation with multiple intermediate steps, since the truncation errors produced by the calculation are equivalent to third order spatial errors, the dominant spatial truncation errors in the covered coarse regions are the same as the initial values of the big step. In this situation, the flux limiters are always the same in one whole step of calculation. Benefiting from the special LMR data storage, in which the results in one or two former steps are saved, calculations for the flux limiters become quite easy.

#### 3.2.1.3. Stability in the solution of NS equations

Before going to the detailed implementation of LMR in NS equations, it is necessary to discuss the stability of the numerical method. In COM3D, since the solution of NS equations is given by the combination of Euler part results and molecular transportation results, the stability of both two numerical methods should be considered. The stability of the Euler part of work follows the requirement of CFL condition (3-21).

$$CFL=\frac{\Delta t\times(c+|u|_{\max})}{\Delta x}<1.0 \tag{3-21}$$

Theoretically, the numerical method can keep stable when the CFL number is less than 1.0. Practically, the CFL number are usually required to be less than 0.5 or 0.4 (in the van Leer solver, the CFL number should even be less than 0.3) to keep reasonable physical properties. In the calculation of molecular transportation, the stability is controlled by the RED condition (3-22).

$$RED = Max\{\frac{D_i \times \Delta t}{\Delta x^2}, \frac{\mu \times \Delta t}{\Delta x^2}, \frac{\lambda \times \Delta t}{\Delta x^2}\} \leq 0.5 \tag{3-22}$$

The RED number is required to be less than 0.4 to get a better simulation results. In the numerical solution of NS equations, both the CFL condition and the RED condition should be satisfied. However, the situations of stability are different from different conditions. For example, in the convection dominant simulations, the CFL decides the time step used for the calculation; but in the molecular transportation dominant simulations, the time step used in the calculation is decided by RED number. Either CFL or RED which one determines the selection of time step, the calculation process is not influenced in the uniform grid simulations, but the calculation process is influenced in the full scale LMR simulations. In convection dominant calculation, the temporal refinement ratio is the same as the spatial refinement ratio; in the molecular transportation dominant calculation, the refinement ratio in time is square of the spatial refinement ratio; it is possible that different levels are dominated by different conditions (CFL or RED). Since the LMR is implemented in the standard NS equations and most of the numerical works are convection dominant, the stability of the simulation in this thesis is controlled by a new criterion JUD (3-23).

$$\begin{aligned} JUD &= Max(c+|u|_{\max}, \frac{D_i}{\Delta x}, \frac{\mu}{\Delta x}, \frac{\lambda}{\Delta x}) \\ \frac{\Delta t \times JUD}{\Delta x} &\leq STN \end{aligned} \tag{3-23}$$

The JUD number is the combination of both the CFL condition and the RED condition, the number STN is given by users (usually less than 0.3). With the JUD number, there is no doubt that both the convection part and the molecular transportation part can keep stable in numerical works. Meantime, in the LMR implementation, the new criterion makes the temporal refinement ratio the same as spatial refinement ratio. The JUD number can show very good performances in the convection dominant simulations, but results in more computational efforts in molecular transportation dominant simulations.

After the discussion of inter-level data communications and stability, the detailed implementation of local mesh refinement in calculation of molecular transportation can be given. Similarly to the implementations in the solutions of Euler equations, the implementations of LMR are given by forward Euler, AD scheme and RK scheme separately in the next sections.

### 3.2.2. Implementation of LMR

#### 3.2.2.1. Forward Euler based solution of NS equations

In the forward Euler based simulations, the first order scheme is enough for the calculation of molecular transportation. The details of one step of FE are shown in Figure 3.26.

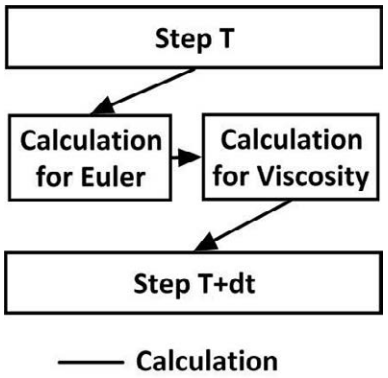



Figure 3.26 One step of FE based solution of NS equations

The calculation of molecular transportation is based on the results of the Euler part of calculation. Similarly to the AD based calculation case, the ghost data for the calculation of molecular transportation cannot be updated by interpolation when the multiple time steps are used in the local mesh refinement. With the 4 layers of ghost cells and the ghost region calculation, updating of boundary conditions for the calculation of molecular transportation can be solved. The process of FE based solution of NS equations with LMR is shown in Figure 3.27, which is quite the same as the case in section 3.1.2.

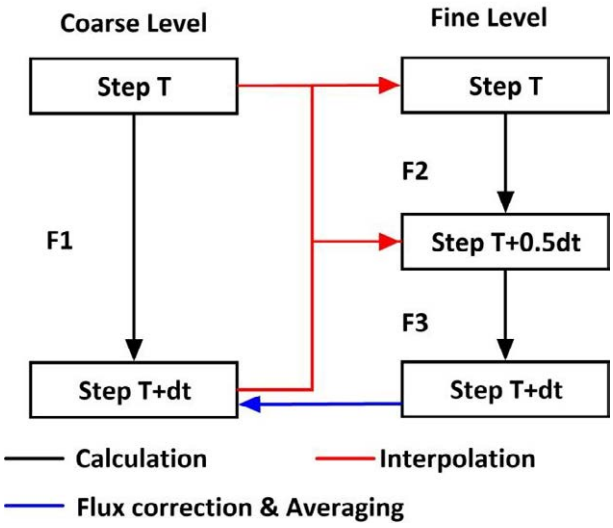



Figure 3.27 The FE based solution of NS equations with LMR

3.2.2.2. AD based solution of NS equations

In the calculation for convection part, the AD based TVNI approach can provide second order accuracy in time and space in general. In the solution of NS equations, in order to maintain the high accuracy, high order algorithm is used in the calculation of molecular transportation. The calculation process of the AD based solution of NS equations with uniform grids is shown in the Figure 3.28.

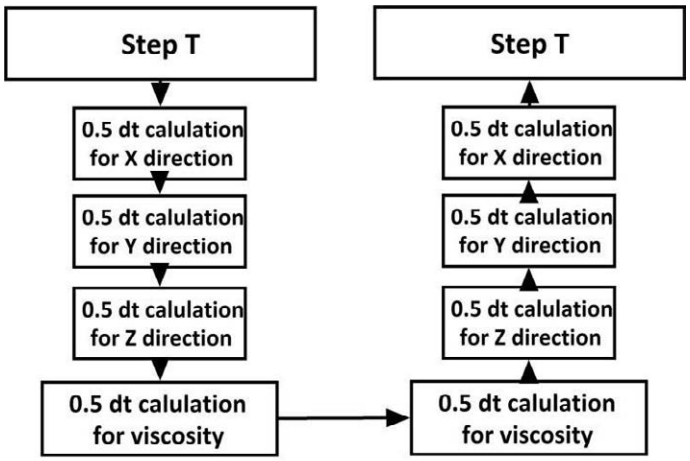



Figure 3.28 One step of AD based solution for NS equations

In order to maintain second order accuracy, calculation of the molecular transportation is done by the second order RK scheme. The detailed analyses for the method shown in Figure 3.28 are given in following parts, for convenience, the 2D case (3-24) is used.

$$\frac{\partial u}{\partial t} = X(u) + Y(u) + V(u) \tag{3-24}$$

The $X(u)$ is the convection term in X direction, $Y(u)$ is the convection term in Y direction and the $V(u)$ is the molecular transportation term. In Figure 3.29, calculation process of the solution for 3D NS case is given as $u_{n+1} = L_x^{\Delta t/2} L_y^{\Delta t/2} L_z^{\Delta t/2} L_V^{\Delta t/2} L_V^{\Delta t/2} L_z^{\Delta t/2} L_y^{\Delta t/2} L_x^{\Delta t/2} u_n$ (the analyses are based on the LW solver, then the proof for the TVNI solver is trivial), so the 2D NS case can be solved as the form $u_{n+1} = L_x^{\Delta t/2} L_y^{\Delta t/2} L_V^{\Delta t/2} L_V^{\Delta t/2} L_y^{\Delta t/2} L_x^{\Delta t/2} u_n$. The 2D case is analyzed in (3-25) to show such calculation method can provide second order accuracy, and the 3D case can be proved by the same way.

$$\begin{aligned}
u_{n+1} &= L_x^{\Delta t/2} L_y^{\Delta t/2} L_V^{\Delta t/2} L_V^{\Delta t/2} L_y^{\Delta t/2} L_x^{\Delta t/2} u_n \\
&= L_x^{\Delta t/2} L_y^{\Delta t/2} L_V^{\Delta t/2} L_V^{\Delta t/2} (u_n + \frac{\Delta t}{2}(X+Y) + \frac{\Delta t^2}{4} Y_u X + \frac{\Delta t^2}{8} Y_u Y + \frac{\Delta t^2}{8} X_u X + O(\Delta t^3)) \\
&= L_x^{\Delta t/2} L_y^{\Delta t/2} (u_n + \frac{\Delta t}{2}(X+Y+2V) + \frac{\Delta t^2}{2} V_u (X+Y+V) + \frac{\Delta t^2}{8}(2Y_u X + Y_u Y + X_u X) + O(\Delta t^3)) \\
&= u_n + \frac{\Delta t}{2}(X+Y+2V) + \frac{\Delta t^2}{2} V_u (X+Y+V) + \frac{\Delta t^2}{4} Y_u X + \frac{\Delta t^2}{8} Y_u Y + \frac{\Delta t^2}{8} X_u X + O(\Delta t^3) + \\
&\quad \frac{\Delta t}{2}(X+Y) + \frac{\Delta t^2}{4}(X_u X + X_u Y + Y_u X + Y_u Y + 2X_u V + 2Y_u V) + \frac{\Delta t^2}{8}(2X_u Y + Y_u Y + X_u X) + O(\Delta t^3) \\
&= u_n + \Delta t(X+Y+V) + \frac{\Delta t^2}{2}[V_u (X+Y+V) + X_u (X+Y+V) + Y_u (X+Y+V)] + O(\Delta t^3)
\end{aligned} \tag{3-25}$$

In the AD based simulation of viscid flow, since the introduction of molecular transportation, more boundary values are required for half time step calculation. In the calculation of molecular transportation, the second order RK scheme is as (3-26).

$$x_{n+1} = x_n + 0.5k_1 + 0.5k_2$$
$$k_1 = hf\left(x_n\right) \tag{3-26}$$
$$k_2 = hf\left(x_n + k_1\right)$$

The two steps of FE based calculations require 2 layers of ghost data. For the half time step of calculation with AD scheme, the calculations on finer refinement levels require four layers of ghost cells. In the calculation with LMR, in order to provide enough boundary conditions, the coarse-to-fine data communication is necessary in the half time intermediate step. Otherwise, 8 layers of ghost cells are required by each patch on finer levels. For saving the computational efforts and memory storage, inter-level communication for the intermediate phased is used in AD based method.

In some papers, in order to save the computational efforts, the AD scheme for the solution of NS equations can be approached as $u_{n+1} = L_x^{\Delta t/2} L_y^{\Delta t/2} L_V^{\Delta t} L_y^{\Delta t/2} L_x^{\Delta t/2} u_n$. Indeed, such calculation process can achieve the same accuracy as the method shown in Figure 3.29. However, by considering the boundary requirement on finer level in the solution of NS equations, coupling format shown by Figure 3.29 must be used. Otherwise, the data may reach a manmade intermediate phase, which makes interpolation impossible. The calculation on finer level with the AD based method is shown as the Figure 3.29.

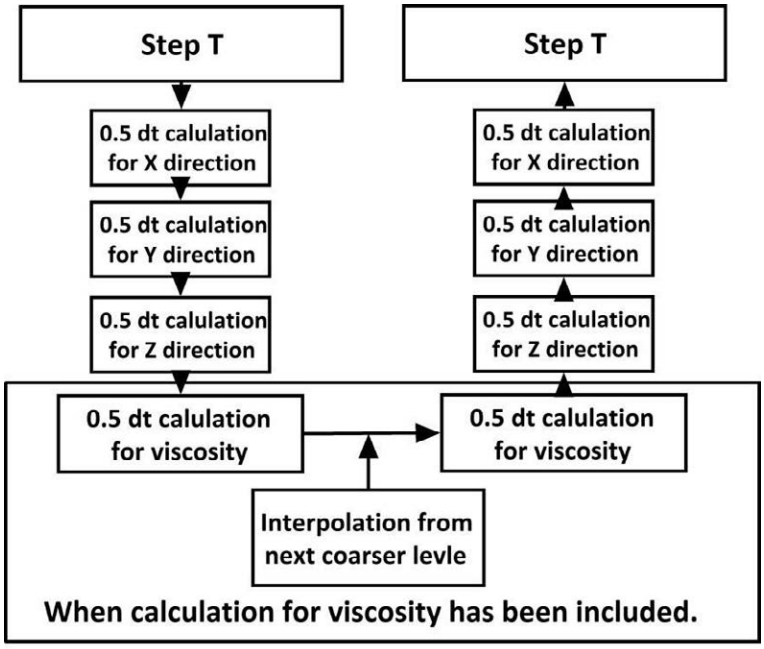



Figure 3.29 One step of AD based solution of NS equations on finer level

It is worth to note that the interpolation for intermediate phase has some differences to the normal interpolation. In the coarse-to-fine interpolation in the solution of NS equations, parabolic scheme is used in time scale to reach third order accuracy. However, the normal parabolic interpolation work may fail to maintain the second order accuracy fluxes in calculation of molecular transportation. The result of the half time intermediate phases is as (3-27).

$$u_n+\Delta t/2\cdot(X(u_n)+Y(u_n)+Z(u_n)+V(u_n))+(\Delta t/2)^2/2\cdot(2V'X+ 2V'Y+2V'Z+V'V+X'X+2Y'X+2Z'X+2Z'Y+Z'Z+Y'Y)|_{u_n}+O(\Delta t^3) \quad (3\text{-}27)$$

In contrast, the normal third order interpolated data is as (3-28).

$$u_n+\Delta t/2\cdot(X(u_n)+Y(u_n)+Z(u_n)+V(u_n))+(\Delta t/2)^2/2\cdot(V'X+X'V+V'Y+Y'V+ V'Z+Z'V+V'V+X'X+Y'X+X'Y+Z'X+X'Z+Z'Y+Y'Z+Z'Z+Y'Y)|_{u_n}+O(\Delta t^3) \quad (3\text{-}28)$$

In the second half step calculation for molecular transportation, since the differences in second order terms between the ghost data and interval data, the accuracy of flux values may be reduced. In order to maintain the second order accuracy, interpolated data should have the same form. From the expression of the calculation results of $L_x^{\Delta t/2}L_y^{\Delta t/2}L_z^{\Delta t/2}L_V^{\Delta t/2}u_n$ and $L_V^{\Delta t/2}L_z^{\Delta t/2}L_y^{\Delta t/2}L_x^{\Delta t/2}u_n$, it is not hard to get,

$$Fix=(L_V^{\Delta t/2}L_z^{\Delta t/2}L_y^{\Delta t/2}L_x^{\Delta t/2}u_n-L_x^{\Delta t/2}L_y^{\Delta t/2}L_z^{\Delta t/2}L_V^{\Delta t/2}u_n)/2= (\Delta t/2)^2/2\cdot[(X'V-V'X)+(Y'V-V'Y)+(Z'V-V'Z)+ (X'Y-Y'X)+(X'Z-Z'X)+(Y'Z-Z'Y)]|_{u_n}+O(\Delta t^3) \quad (3\text{-}29)$$

Supported by the above Fix term, some correction can be done for the intermediate interpolation to make the final interpolated results have the same form as the internal region. In the arbitrary small step k under the temporal refinement ratio N, the intermediate phases on finer level have the form (3-30).

$$\begin{aligned}
Second &= (V'X+X'V+V'Y+Y'V+V'Z+Z'V+V'V+X'X+ Y'X+X'Y+Z'X+X'Z+Z'Y+Y'Z+Z'Z+Y'Y)|_{u_n}\\
&u_{n+k}+\Delta t_N/2\cdot(X(u_{n+k})+Y(u_{n+k})+Z(u_{n+k})+V(u_{n+k}))+(\Delta t_N/2)^2/2\cdot(2V'X+ 2V'Y+2V'Z+V'V+X'X+2Y'X+2Z'X+2Z'Y+Z'Z+Y'Y)|_{u_{n+k}}+O(\Delta t^3)\\
&=u_n+k\Delta t_N\cdot(X(u_{n+k})+Y(u_{n+k})+Z(u_{n+k})+V(u_{n+k}))+(k\Delta t_N)^2/2\times Seond+\\
&\quad \Delta t_N/2\cdot(X(u_n)+Y(u_n)+Z(u_n)+V(u_n))+k\Delta t_N{}^2/2\times Seond+\\
&\quad (\Delta t_N/2)^2/2\cdot(2V'X+2V'Y+2V'Z+V'V+X'X+2Y'X+2Z'X+2Z'Y+\\
&\quad Z'Z+Y'Y)|_{u_{n+k}}+O(\Delta t^3)\\
&=u_n+(k+1/2)\Delta t_N\cdot(X(u_{n+k})+Y(u_{n+k})+Z(u_{n+k})+V(u_{n+k}))+[(k\Delta t_N)^2+k\Delta t_N{}^2]/2\\
&\quad \times Seond+(\Delta t_N/2)^2/2\cdot(2V'X+2V'Y+2V'Z+V'V+X'X+2Y'X+2Z'X+\\
&\quad 2Z'Y+Z'Z+Y'Y)|_{u_{n+k}}+O(\Delta t^3)
\end{aligned} \quad (3\text{-}30)$$

The normal third order interpolation has the form:

$$u_n + (k+1/2)\Delta t_N \cdot (X(u_{n+k}) + Y(u_{n+k}) + Z(u_{n+k}) + V(u_{n+k})) + (k\Delta t_N + 1/2 \cdot \Delta t_N)^2 / 2$$
$$\times Seond + O(\Delta t^3)$$
$$= u_n + (k+1/2)\Delta t_N \cdot (X(u_{n+k}) + Y(u_{n+k}) + Z(u_{n+k}) + V(u_{n+k})) + [(k\Delta t_N)^2 + k\Delta t_N{}^2]/2 \quad (3\text{-}31)$$
$$\times Seond + (\Delta t_N / 2)^2 / 2 \cdot (V'X + X'V + V'Y + Y'V + V'Z + Z'V + V'V + X'X +$$
$$Y'X + X'Y + Z'X + X'Z + Z'Y + Y'Z + Z'Z + Y'Y)|_{u_n} + O(\Delta t^3)$$

It is clear that using $Fix / N^2$ can make the interpolated data have the same form. Therefore, the accuracy can be maintained.

The process of AD based solution of NS equations with LMR is shown as Figure 3.30.

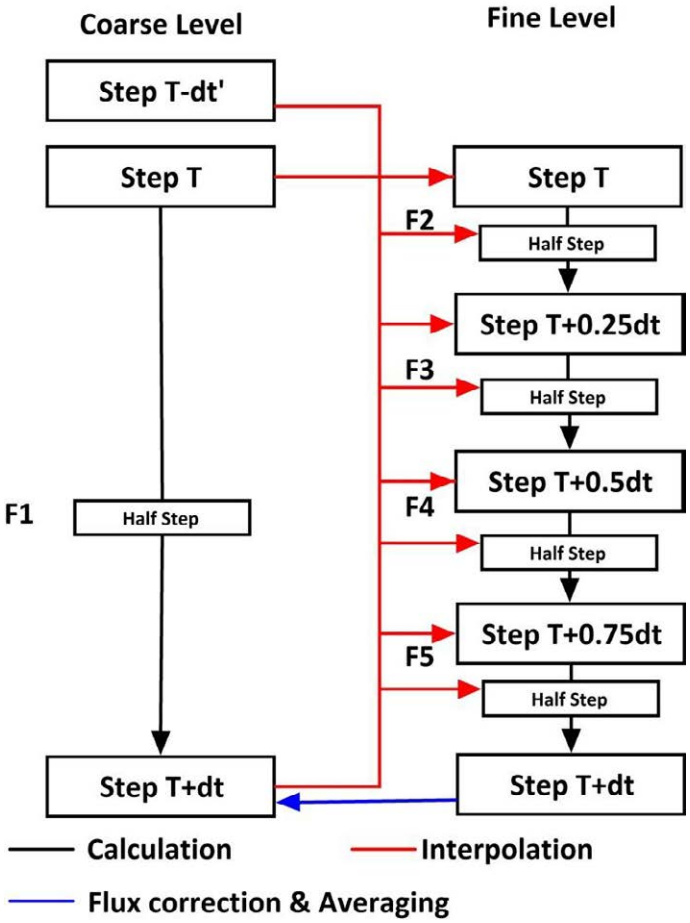



Figure 3.30 AD based solution of NS equations with LMR

The above figure is the computation with 2 refinement levels. As the AD based calculation is a high order scheme, refinement ratio four is used between the two adjacent levels. Comparing to the inter-level data communications used in the solution of Euler equations, the data communication is more complicated because of the big amount of ghost data required by molecular transportation.

The truncation errors brought by the LMR are analyzed in Appendix E, and the results show that the high order accuracy can be maintained in the solution of Navier-Stokes equations.

3.2.2.3. RK based solution of NS equations

As illustrated in the solution of Euler equations, one big step of RK calculation is accomplished by the calculations of several man-made intermediate steps. In each intermediate step's calculation, the calculation of convection and molecular transportation are given paralleled, which is shown as Figure 3.31.

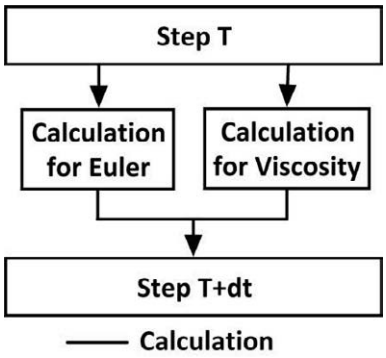



Figure 3.31 One step of calculation for intermediate phase in RK scheme

Since both the calculation of Euler part and molecular transportation part are all based on the results of former step or phase, requirements of the ghost data on finer levels are the same as the pure Euler case. The processes of RK based solutions with LMR are the same as the process in section 3.1.2, except the more complicated spatial interpolation and coarse level's flux limiter.

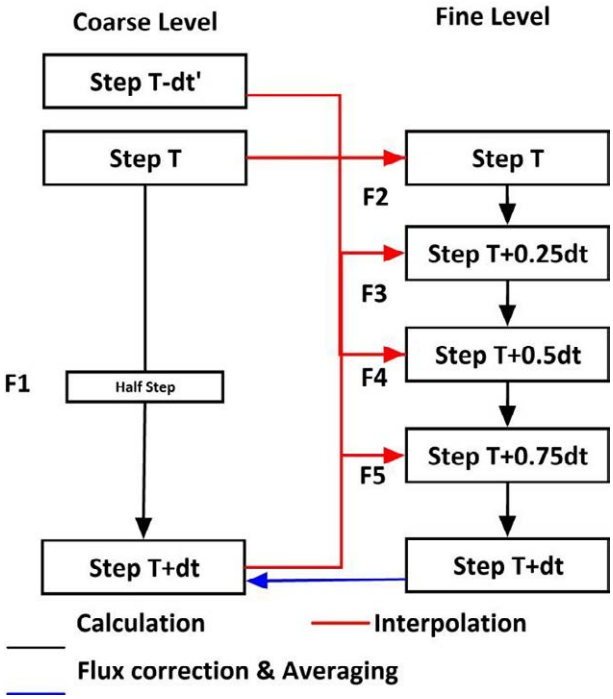



Figure 3.32 RK2 based solution of NS equations with LMR

Figure 3.32 is the cases when two refinement levels are used. Since the RK2 scheme is the high order scheme, refinement ratio between the base level and the first level is four. The process of RK4 based NS solution with local mesh refinement is also the same as the process described in section 3.1.2.

### 3.2.3. Testing case

Differently from the Euler equations, NS equations contain the influence of viscosity, diffusion and heat transfer. In this situation, the influence of molecular transportation terms containing the first order derivatives should be considered in the flux correction. The stability of those derivative terms in the flux should be taken into account in the numerical work. In the user's guide of Chombo [18], the explicit flux correction for the molecular transportation is marked as unstable in some calculations.

Theoretically, the stability can be guaranteed if both CFL and RED criteria have been satisfied under the explicit solution. In the implementation of LMR in COM3D, the calculation satisfied JUD criteria. However, it is necessary to test if the explicit flux correction is unstable in the simulation of viscid flow.

A 2D viscid flow with un-slipping boundary is used to make the testing. The computational domain is shown by Figure 3.33.

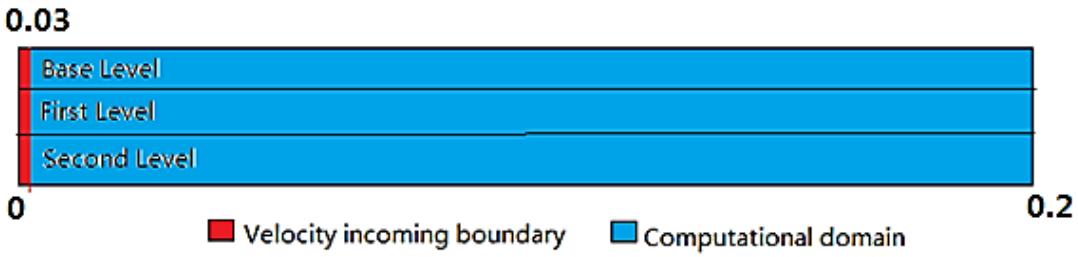



Figure 3.33 Computational domain of the 2D simulation of viscid flow

The detailed information of the computational domain and the boundary condition are shown in Table 3.5.

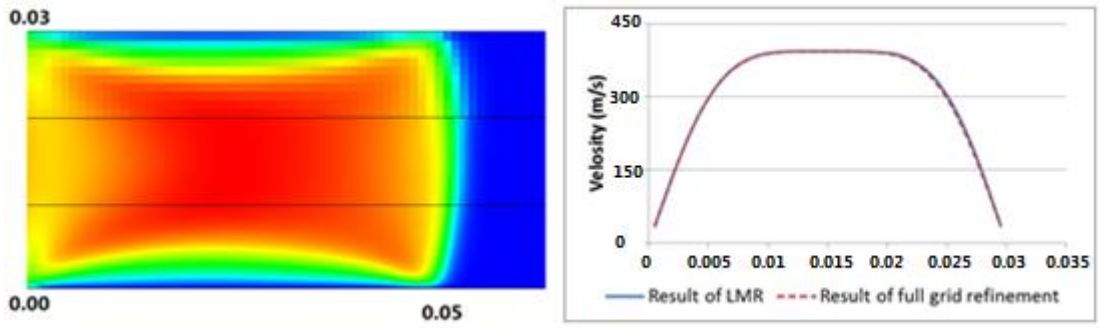
Table 3.5 Grid Structure and Initial Condition of the 2D viscid flow problem

| Grid Structure | | |
|---|---|---|
| Base Level | First Level | Second Level |
| Computational domain:<br>(0, 0, 0)X(199, 29, 1) | Computational domain:<br>(0, 0, 0, )X(799, 119, 7)<br>Refinement ratio: 4 | Computational domain:<br>(0, 0, 0)X(3199, 479, 31)<br>Refinement ratio: 4 |
| Boundary Condition and Initial Condition | | |
| Incoming Flow | Computational Domain | |
| Velocity (m/s): | Velocity (m/s): | |

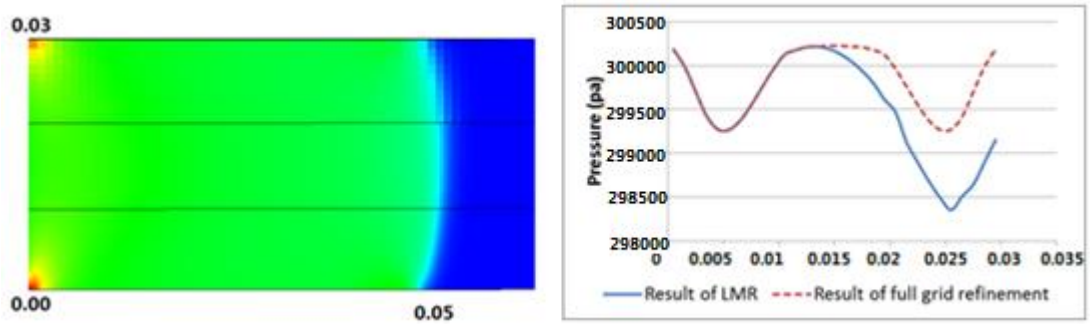
| | |
|---|---|
| $(V_x, V_y, V_z) = (321.39, 0, 0)$<br>Pressure (Pa): 324670<br>Temperature (K): 434.8<br>Gas Components:<br>$H_2 : O_2 : N_2 : H_2O = 2 : 1 : 3.76 : 0$ | $(V_x, V_y, V_z) = (0, 0, 0)$<br>Pressure (Pa): 101300<br>Temperature (K): 298<br>Gas Components:<br>$H_2 : O_2 : N_2 : H_2O = 2 : 1 : 3.76 : 0$ |

In this testing case, due to the un-slipping boundary and the effect of viscosity, velocity in X direction has gradient in Y direction. As shown in Figure 3.34, local mesh refinement is made statically in the computational domain and the fine-to-coarse interfaces are located parallel in Y direction, the velocity in X direction around the interface can show whether the explicit flux correction for viscosity is unstable. In order to get the results faster, the viscosity factor in this simulation is increased by 10000 times.

In this calculation, the refinement ratio four is used between every two adjacent levels. With such refinement ratios, it is possible to use all the solvers and calculation methods in COM3D to make the simulation. The methods used in the simulation include the FE based Up-wind scheme, AD based TVNI solver and RK4 based van Leer solver. Following discussion is focused on the calculation result of the second order TVNI solver. Physical quantities at the step 580 have been shown in Figure 3.34.




(a)




(b)

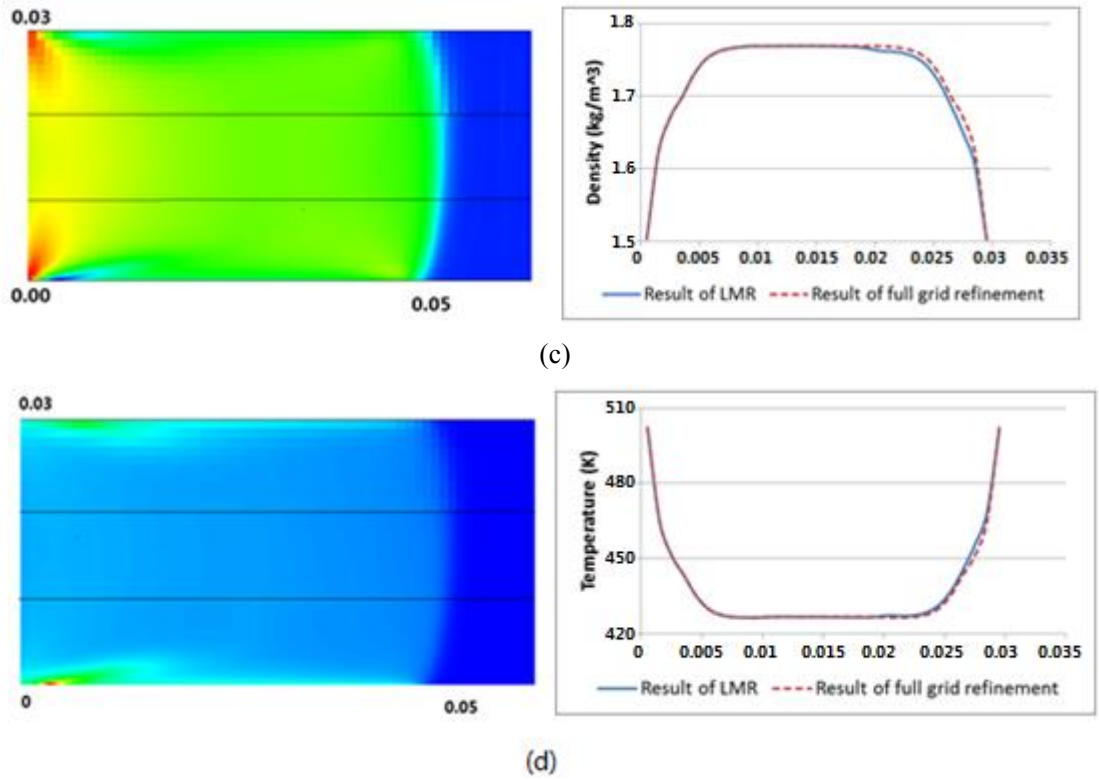



Figure 3.34 Results of the 2D viscid flow simulation

In Figures 3.34, coarse-to-fine interfaces between different refinement levels are marked by black lines, and the quantities such as velocity in X direction (a), pressure (b), density (c) and temperature (d) are shown. In the expression of each quantity, the left part is its distribution in the computational domain and the right part is the comparison of the results at the position X=0.025 between LMR and full grid refinement.

In the distribution of velocity in X direction, no obvious unphysical oscillation can be detected around the fine-to-coarse interfaces. As shown on the right side, result of the simulation with local mesh refinement has almost the same curve as simulation with full grid refinement. The more important is no oscillation around the positions 0.01 and 0.02 where the coarse-to-fine interfaces are located. The velocity in X direction can best describe whether explicit flux correction can result in a divergence results or unphysical oscillation, especially for the cases in which viscosity has been increased by 10000 times to generate large gradient of the velocity and amplify the possible instability. The velocity data show that the current explicit flux correction can keep stable and maintain the physical properties.

Besides the distribution of velocity, in the distributions of pressure, density and temperature, no obvious instability can be found around the interfaces either. In the comparisons of the pressure and density between the LMR case and full grids refinement case, big differences are founded. Actually, these two different results show the absolute errors only, the relative error of those two physical quantities are all less than 0.2%. In the second order alternating direction based TVNI calculation, the implementation of local mesh refinement and the explicit treatments of the flux correction are reasonable.

The calculations of FE based Up-wind scheme and RK 4 based van Leer method can also show the similar results to the TVNI based calculation, no further descriptions are made here.

### 3.3. Local mesh refinement for detonation simulation

Software COM3D is the CFD tool for the simulation of turbulent combustion and detonation. After the discussion of implementations of LMR in two basic gas dynamics, the LMR in combustion simulation should be established. In the thesis, the implementation of LMR in detonation simulation is given only.

In the hydrogen detonation simulation, the modeling method is the one step Arrhenius approach. The governing equations of one step Arrhenius detonation simulation is shown as (3-32) [51]. In (3-32), $\sum E_i \alpha_i kY$ is the energy released by chemical reaction, $\alpha_i kY_i$ is the consuming of the gas component $Y_i$, the coefficient $k$ is the reaction factor gotten by the Arrhenius law and $\alpha_i$ is the constant. The two constants $E_{ch}$ and $K_{ch}$ used in the model are 8368 and $1.0\times10^7$.

$$\begin{cases} \dfrac{\partial \rho}{\partial t}+\dfrac{\partial \rho u}{\partial x}+\dfrac{\partial \rho v}{\partial y}+\dfrac{\partial \rho w}{\partial z}=0 \\ \dfrac{\partial \rho u}{\partial t}+\dfrac{\partial \rho u^2}{\partial x}+\dfrac{\partial \rho uv}{\partial y}+\dfrac{\partial \rho uw}{\partial z}=-\dfrac{\partial p}{\partial x}, \dfrac{\partial \rho v}{\partial t}+\dfrac{\partial \rho uv}{\partial x}+\dfrac{\partial \rho v^2}{\partial y}+\dfrac{\partial \rho vw}{\partial z}=-\dfrac{\partial p}{\partial y} \\ \dfrac{\partial \rho w}{\partial t}+\dfrac{\partial \rho uw}{\partial x}+\dfrac{\partial \rho vw}{\partial y}+\dfrac{\partial \rho w^2}{\partial z}=-\dfrac{\partial p}{\partial z} \\ \dfrac{\partial \rho e}{\partial t}+\dfrac{\partial u(\rho e+p)}{\partial x}+\dfrac{\partial v(\rho e+p)}{\partial y}+\dfrac{\partial w(\rho e+p)}{\partial z}=\sum E_i \alpha_i kY_i \\ \dfrac{\partial Y_i}{\partial t}+\dfrac{\partial Y_i u}{\partial x}+\dfrac{\partial Y_i v}{\partial y}+\dfrac{\partial Y_i w}{\partial z}=\alpha_i kY_i \\ p=(\sum \dfrac{Y_i}{M_i})\dfrac{R}{C_V}(\rho e-\dfrac{1}{2}(\rho u^2+\rho v^2+\rho w^2)) \\ k=K_{ch}e^{-\frac{E_{ch}}{T}} \end{cases} \tag{3-32}$$

The simulation of detonation is based on the solution of Euler equations, so the detonation model (3-32) can be expressed as (3-33).

$$\frac{\partial U}{\partial t}=G(U)+C(U) \tag{3-33}$$

In (3-33), $G(U)$ means the Euler equations and the $C(U)$ is the chemical reaction shown by (3-34).

$$C(u)=\begin{cases} \dfrac{\partial \rho e}{\partial t}=\sum E_i \alpha_i kY_i \\ \dfrac{\partial Y_i}{\partial t}=\alpha_i kY_i \\ k=K_{ch}e^{-\frac{E_{ch}}{T}} \end{cases} \tag{3-34}$$

Normally, the time-operator splitting method, in which the calculation of gas dynamics and chemical reaction has been decoupled and calculated separately, is used in simulation of detonation. Due to some historical reasons, the time-operator splitting method in COM3D is a little different from the usual case [25] [26]. In the calculation in COM3D, the calculation of chemical reaction is based on the solution of hydrodynamics.

(1) In one step of calculation, gas dynamic should be calculated separately by the data in time step n firstly (shown as (3-35)). The pure gas result at the step n+1 can be achieved.

$$U_{n+1,G} = U_n + \Delta t \times G(U_n) \tag{3-35}$$

(2) Latterly, calculation for the chemical reaction is based on the pure gas data $U_{n+1,G}$. Temperature of pure gas result is used to calculate the reaction factor. Then, gas consumption and the energy increases are added to the pure gas dynamic result to get the step n+1.

$$U_{n+1} = U_{n+1G} + \Delta t \times C(U_{n+1G}) \tag{3-36}$$

Although the calculation process of detonation simulation in COM3D is different from the usual case, simulations still suffer from the redundant chemical reaction. In order to keep the temporal refinement in detonation simulation, a new method should be developed to couple the gas dynamics and chemical reaction. Or the uniform time step should be used in the implementation.

### 3.3.1. Redundant chemical reaction

The mathematical analyses for the redundant chemical reaction are given in this section. For the normal forward Euler based detonation simulation with refinement ratio two, the flux values of the coarse and fine levels are shown by (3-37).

$$\begin{aligned} F_C &= g(u_n) \\ F_F^1 &= g(u_n) \\ F_F^2 &= g(u_{n+1/2}) = g(u_n + \frac{\Delta t}{2} \times G(u_n) + \frac{\Delta t}{2} \times C(u_n)) \\ &= g(u_n) + \frac{\Delta t}{2} \times g_u G + \frac{\Delta t}{2} \times g_u C + \mathrm{O}(\Delta t^2) \end{aligned} \tag{3-37}$$

After the refluxing in the synchronization, the redundant chemical reaction introduced to the next coarse level is shown as the formula (3-38).

$$\frac{\Delta t}{4\Delta x} \Delta t \times g_u C = \frac{\Delta t}{4\Delta x} \Delta t \times uC \tag{3-38}$$

In the mathematical view, the redundant chemical reaction is a second order term. However, this second order term may be transmitted into a first order term in some special situations.

$$CFL = \frac{\Delta t \times (c + |u|_{\max})}{\Delta x} \sim 0.2 \tag{3-39}$$

By considering (3-39), when the shock wave front does not reach the interface or the interface has some distance behind the shock wave, the velocity $u$ is much smaller than the $c+|u|_{\max}$ and the redundant chemical reaction is quite limited. For the cases that shock wave just reaches the interface, the $u$ will be quite close to $|u|_{\max}$ (which is much bigger than the sound speed $c$ in detonation), and the redundant chemical reaction is the first order term (3-40).

$$\frac{\Delta t}{4\Delta x}\Delta t \times g_u C = \frac{u\Delta t}{4\Delta x}\Delta t \times C \sim 0.025\Delta t \times C \tag{3-40}$$

In normal cognitive, the 0.025 times more chemical reaction seems very small. However, differently from the calculation of convection, the calculation of chemical reaction is determined by the chemical reactivity factor

$$k = K_{ch} e^{-\frac{E_{ch}}{T}}$$

which shows exponentially growth as the increasing of the temperature. The numerical increases of 0.025 times chemical reaction may be significant to the calculation in coarse cells, when the shock wave passes through the interface form the fine side to the coarse side.

### 3.3.2. Further operator splitting method for detonation

One method to solve the redundant chemical reaction is the further operator splitting treatment, in which the calculations for chemical reaction are done before or after the calculations for gas dynamics in one base level's time step Δt. In this situation, the calculations of gas dynamics can make contribution to fluxes only, and no redundant chemical reaction can be generated by the flux correction. Since the new method splits the calculation operator in several time steps' calculation, which is different for the original operator splitting method [93], it is named with further operator splitting method.

First of all, it is necessary to test if the further decoupled gas dynamics and chemical reaction can maintain the numerical and physical property of detonation waves. For the point-wised calculation, the analyses for the further operator splitting can be made under the uniform grid calculation, and the rationality of the method in LMR is trivial.

Supported by the analyses in Appendix F, the total truncation error is still second order in time. However, differently from the normal calculation, the calculation method with further operator splitting treatment has the truncation errors $O((k\times\Delta t)^2)$ instead of $O(\Delta t^2)$. Actually, the temporal truncation error is increased by several times but kept in the same order.

Then, several testing cases shown in Table 3.6 are made to test if the further operator splitting treatment can maintain the physical properties of detonation in numerical simulations. As shown, different coupling modes are used in the four testing cases. In calculations of the gas dynamics, all the works are done by the solver van Leer with the CFL=0.2. In the calculation of chemistry, the detailed numerical works are given differently in each case. For longer time operator splitting such as case 2 and 3, the calculation is given by several small steps to prevent the possible instability. To the shorter time operator splitting such as case 1, the calculation is done by one single step.

Table 3.6 Testing cases for further operator splitting treatment

| | Number of gas steps between two chemical calculation | The time step for calculate the reaction | Number of steps in each chemical reaction. |
|---|---|---|---|
| Normal case | 1 | $dt$ | 1 |
| Test case 1 | 2 | $\sum_{i=0}^{1} dt_i$ | 1 |
| Test case 2 | 4 | $(\sum_{i=0}^{3} dt_i)/2$ | 2 |
| Test case 3 | 8 | $(\sum_{i=0}^{7} dt_i)/4$ | 4 |

Indeed, in the numerical work of ODE, condition for stability is very lax and does not depend on the cell size, the special treatments for cases 2 and 3 are for caution only. The comparisons of density and pressure are shown in flowing figures.

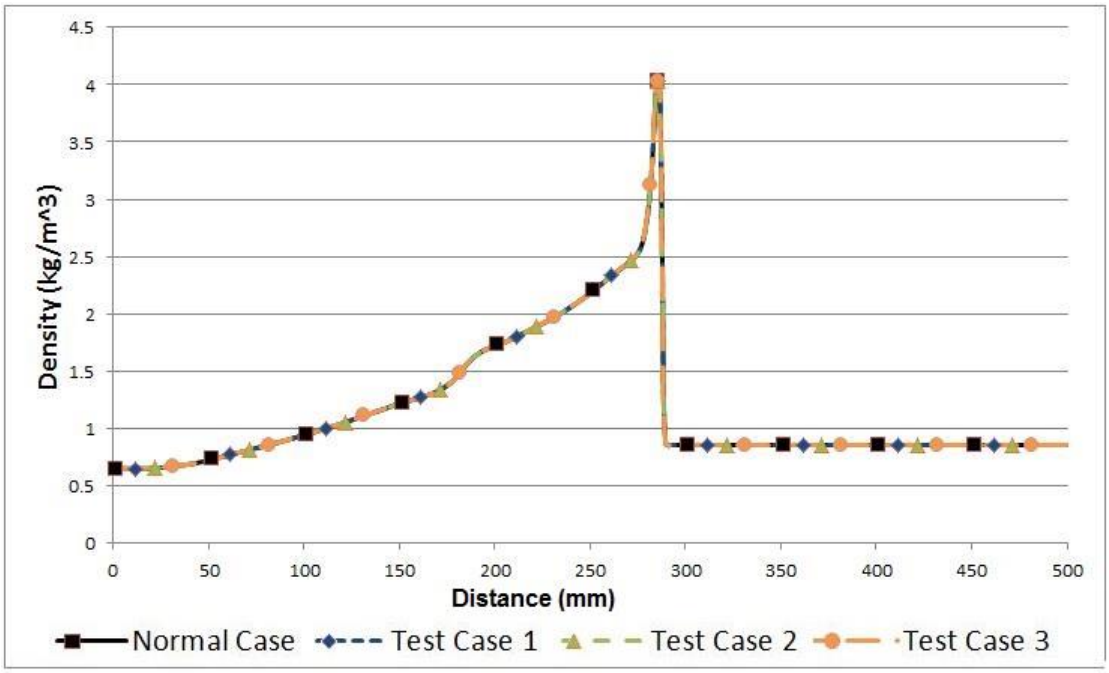



Figure 3.35 Density curves of the four cases at step 3000

Figure 3.35 is the comparison of the density among all the testing cases and normal computation method in COM3D at step 3000. In the comparison, four curves almost completely overlap to each other and no obvious differneces can be found among them.The Figure 3.36 is the comparison of pressure among all the cases.

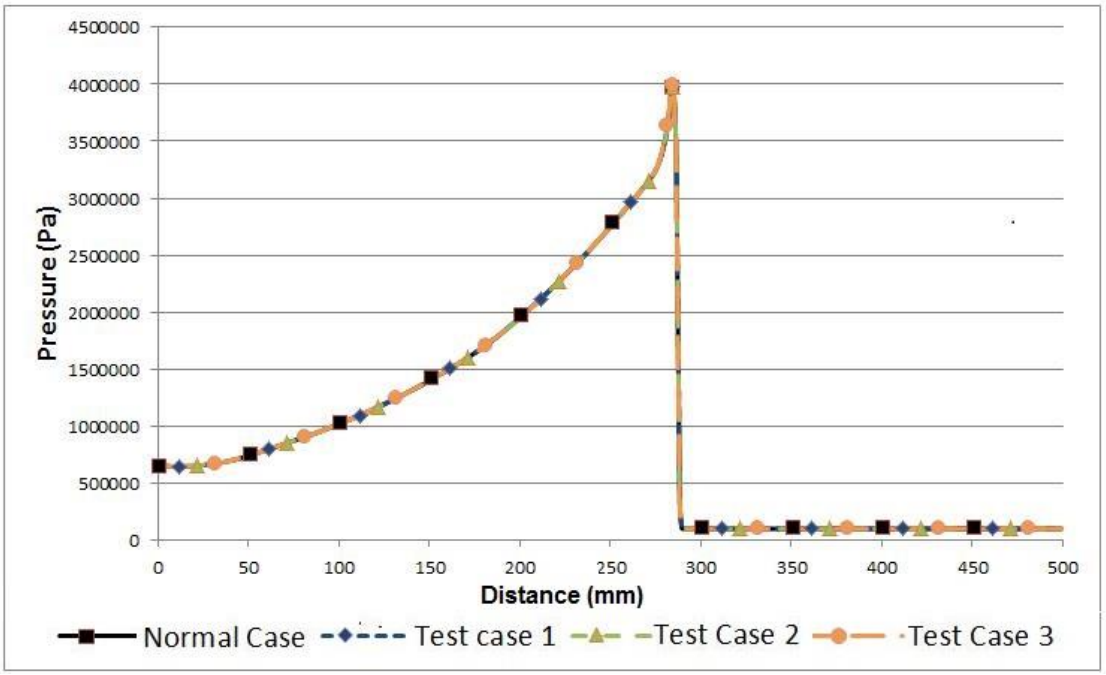



Figure 3.36 Pressure curves of the four cases at step 3000

In Figure 3.36, no big differences can be found either. However, above two comparisons are the results of short time calculation. In order to prove the reasonableness of the further operator splitting, longer time results should be compared also. The following Figure 3.37 is the comparison of density at step 15000.

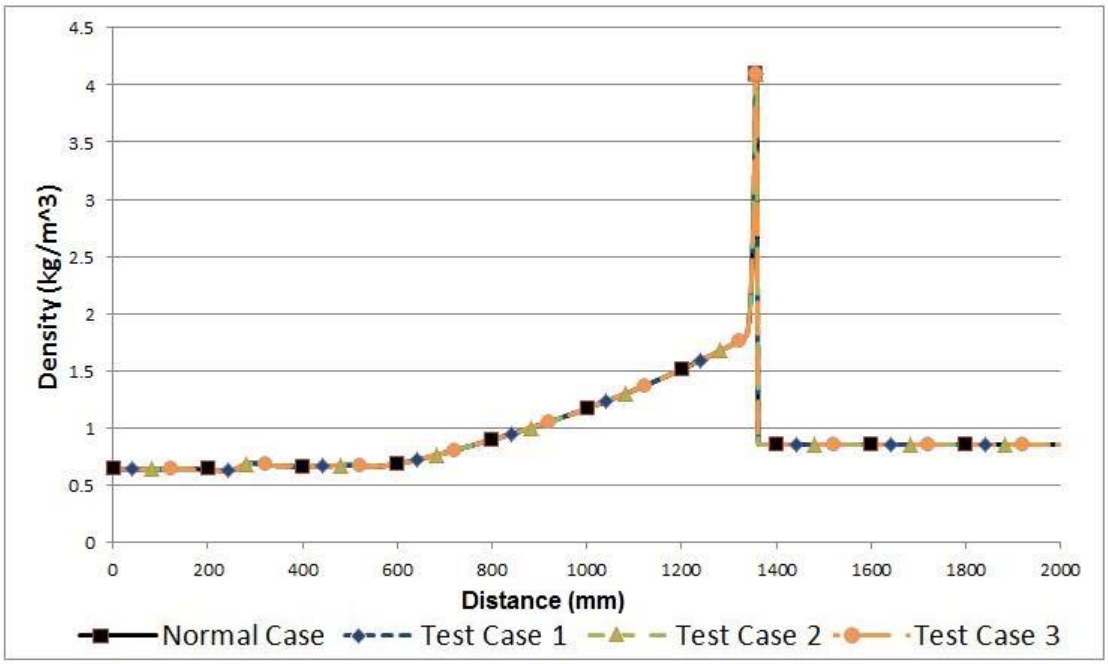



Figure 3.37 Density curves of the four cases at step 15000

In the comparison of density shown by Figure 3.37, quite similarly to the comparison in step 3000, all testing cases have almost the same result as the normal calculation in step 15000.

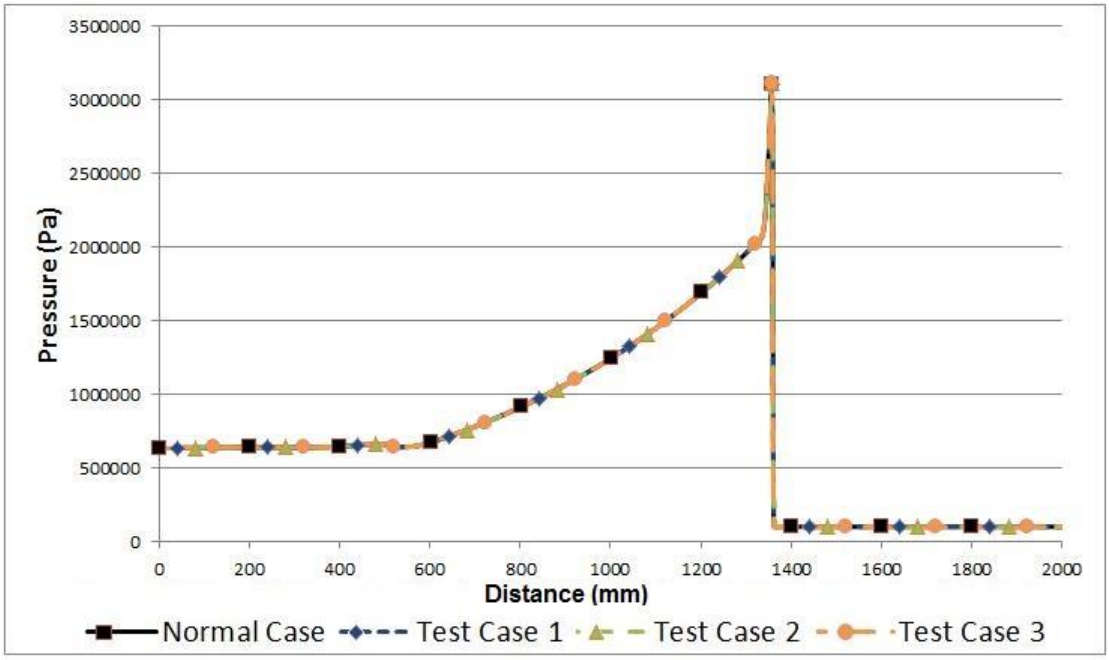



Figure 3.38 Pressure curves of the four cases at step 15000

The curves of the pressure shown in Figure 3.38 also support the reasonableness of the further operator splitting treatment. Besides, the comparisons in other steps have been made as well. Similarly, no big difference can be found among the normal calculations and three testing cases either. So, in detonation simulations, in the view of propagation speed and peak values, no big unphysical phenomenon can be found in calculations approached by further operator splitting.

In high dimension detonation simulations, the appearance of the cellular structures is very important [15] [36]. Meantime, the size of the cellular structure is another important criterion used to judge whether the simulation can accurately reflect the true physical property of the detonation. 2D detonation is used to make similar testing.

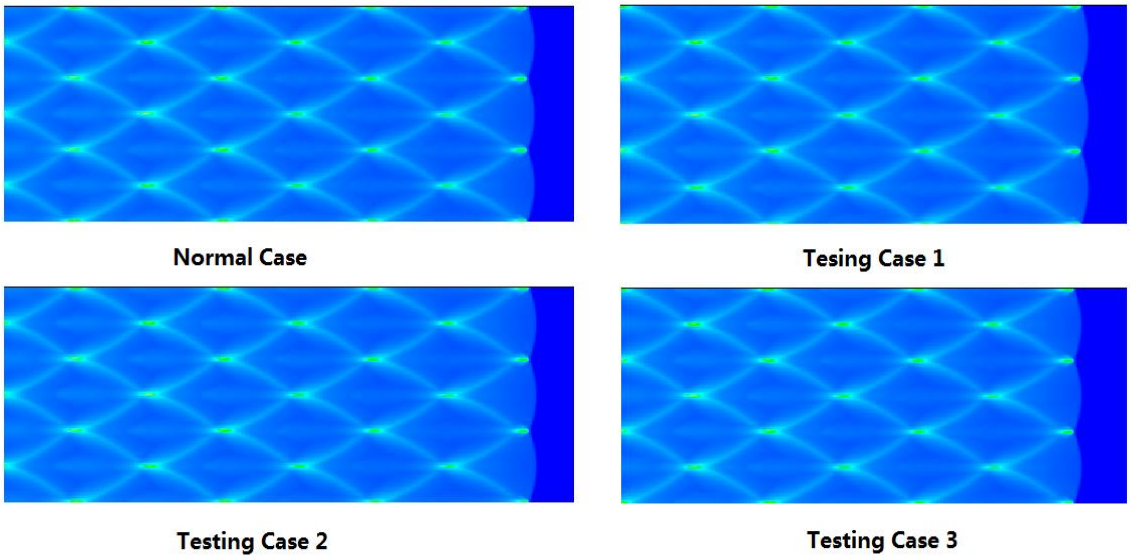



Figure 3.39 Simulation of detonation cells in all the testing cases

Figure 3.39 shows the detonation cells gotten by all four modes of calculations. All calculations succeed in the simulations of the appearance of detonation cells. Moreover, the four calculations can get the same number of detonation cells with the same width. The further operator splitting method shows good performance in simulation of cellular structures.

Actually, there are still some differences among the calculations in the 1D and 2D testing cases. Owing to the differences in coupling, detailed time steps may not be exactly equal to each other. The four simulations may have about 0.01% differences in comparisons of density and pressure (they are not shown in the 1D and 2D detonation cases). Such differences can be ignored in large-scale industrial simulations.

Supported by the above comparisons, the first order detonation simulation with further operator splitting can be done as shown in Figure 3.40. In this figure, the refinement ratio two is used and the multiple time steps method is maintained in the gas dynamics. The calculation about gas dynamics is done quite the same as the former sections (shown as the figure, the forward Euler scheme with temporal refinement is used in calculation of gas dynamics). Differently from the synchronization in former gas calculations, fine-to-coarse data transfer and coarse-to-fine interpolation are not necessary at the end of the pure gas work. The calculations for chemical reaction are done with coarse level time step (If it is necessary, calculation for chemistry can also be achieved by several small steps of chemical work). The fine-to-coarse data transfer and coarse-to-fine interpolation are done after the chemical calculation.

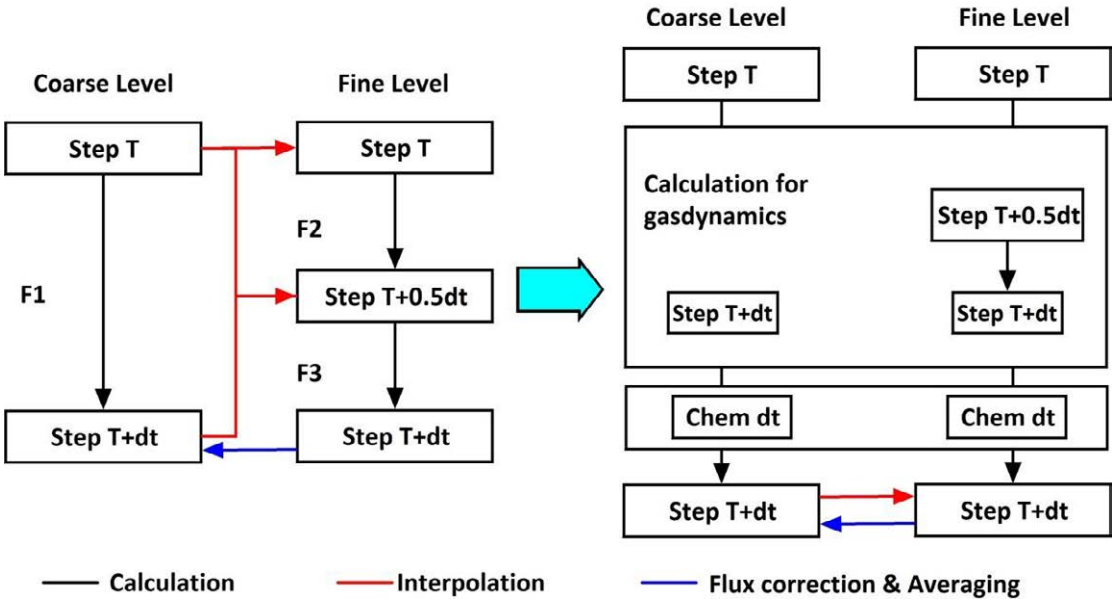



Figure 3.40 First order detonation simulation with further operator splitting

Besides the above first order further operator splitting detonation simulation, second order further operator splitting method is also developed for some calculations which require higher accuracy. In the past, since the computational efforts required by the industrial problems are usually very big,

only the first order scheme is available. With the technique of local mesh refinement, total cells number has been largely reduced, and higher order calculation method is possible now. Currently, the AD based TVNI scheme is used in calculation of gas dynamics. The second order further operator splitting method is shown in Figure 3.41.

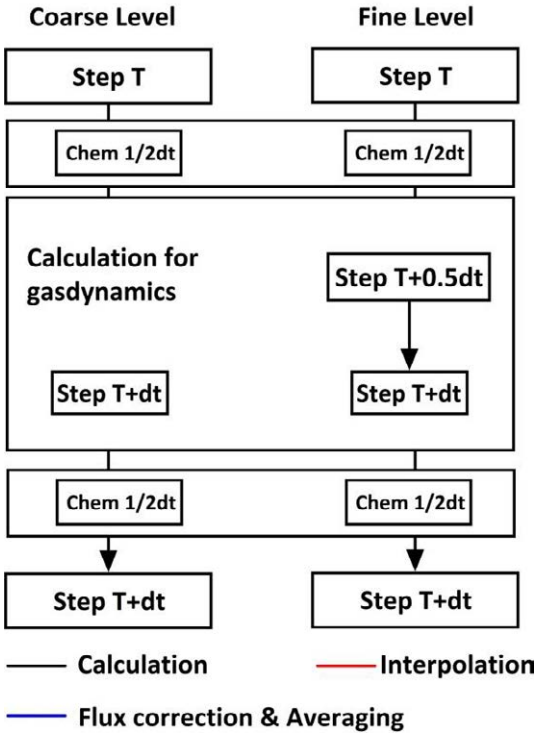



Figure 3.41 Second order detonation simulation with further operator splitting

As shown in Figure 3.41, the second order further operator splitting is similar to the AD based method for the solution of NS equations. Calculation of chemistry is given by second order RK scheme to achieve high order accuracy in time. In one big step of calculation, the chemical work is decoupled into two parts and arranged in a symmetrical way. The analyses in Appendix F show that the calculation can provide third order accuracy for single step of calculation.

However, the implementation of further operator splitting scheme should take care of the limitations in time step. In order to avoid some unphysical results, the consuming of the fuels in one time step should be restricted by the amount remained in the cells. Therefore, for some large-scale industrial problems, the further operator splitting scheme might be implemented partially in some finer levels and the other coarser levels should use the time step on finer levels (the biggest time span used on those fine levels implemented with further operator splitting).

### 3.3.3. Testing case

In section 3.3.2, the stability of the further operator splitting method has been proved. Moreover, in the view of the propagation speed, peak pressure value and the global distribution of the data, the results achieved by further operator splitting method have given almost the same as the usual coupling mode in the uniform grid calculation. It is the turn to test if there are some big differences between the local mesh refinement and full grid refinement. The 1D detonation

simulation is done as the testing case. The computational domain of this 1D problem is shown as following Figure 3.42.


Figure 3.42 Computational domain of 1D detonation

The initial condition of this 1D problem and the grid structure are shown in the Table 3.7

Table 3.7 Initial condition and grid structure of the 1D detonation

| **Grid Information** | |
|---|---|
| **Computational Domain:** $(0,0,0)\times(499,1,1)$<br>**Cell Size: 0.001m** | |
| **Initial Condition** | |
| **Region 1** | **Region 2** |
| **Domain:** $(0,0,0)\times(19,1,1)$<br>**Pressure:100Bar**<br>**Temperature:3000K**<br>**Velocity: 0, 0, 0**<br>**Gas Spices (mole ratio):**<br>$H_2:O_2:N_2:H_2O=2:1:3.76:0$ | **Domain:** $(20,0,0)\times(599,1,1)$<br>**Pressure:1Bar**<br>**Temperature:298K**<br>**Velocity: 0, 0, 0**<br>**Gas Spices (mole ratio):**<br>$H_2:O_2:N_2:H_2O=2:1:3.76:0$ |

In this 1D case, similarly to the former sod shock tube case, local mesh refinement is used adaptively for tracking the most intensive chemical reaction zone. One finer level is used here, the refinement ratio between the base level and the finer level is four. In order to show validation, calculation with full grid refinement is made as well to make comparison.

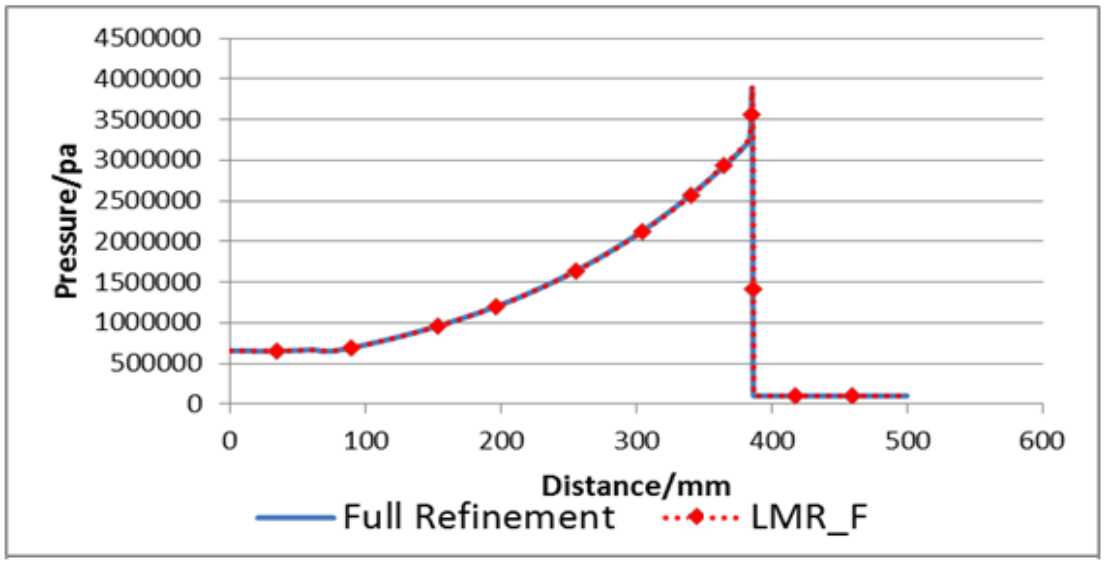



Figure 3.43 Comparison of pressure at T=0.00015s

Figure 3.43 shows the comparison of the pressure between the LMR with further operator splitting and full grid refinement at the time 0.00015s. In calculation with LMR, finer level only covers a small area around wave front (the region where the peak pressure value is located). In global view, two simulations have almost the same curve shape and peak values. Since the tinny difference cannot be expressed clearly in the global description, the comparison between the two calculations in the peak value region has been made and shown in Figure 3.44.

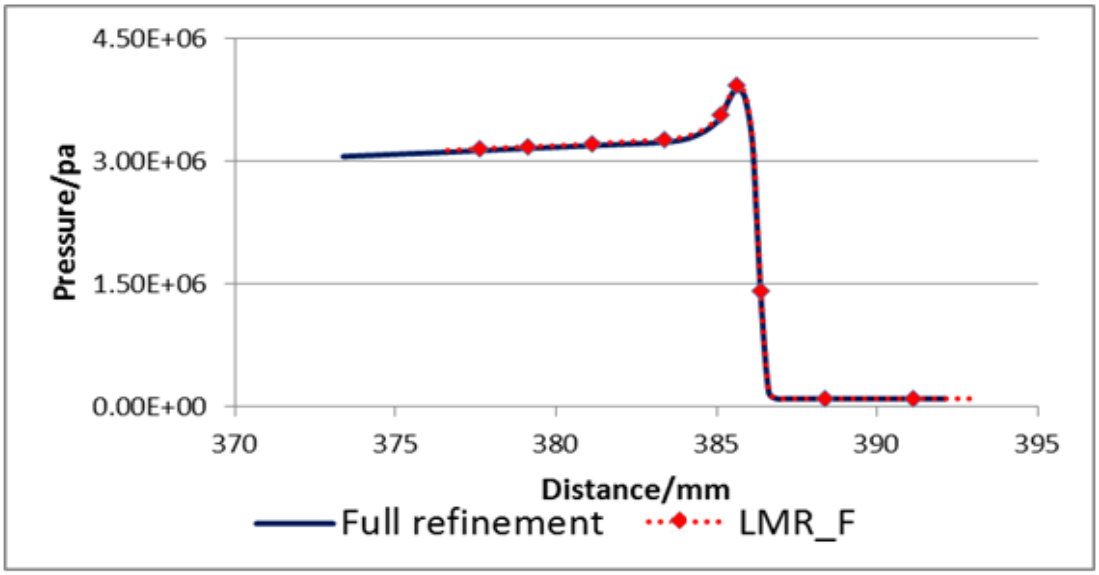



Figure 3.44 Comparison around the peak at T=0.00015s

In the above more detailed comparison, it is not difficult to find that the two simulations have the same propagation speed. The pressure values in the above figure show that the calculation with local mesh refinement can have almost the same performance as the calculation with full grid refinement. Besides the results at the time T=0.00015s, the global comparison of the pressure at the time T=0.00019s is shown in Figure 3.45.

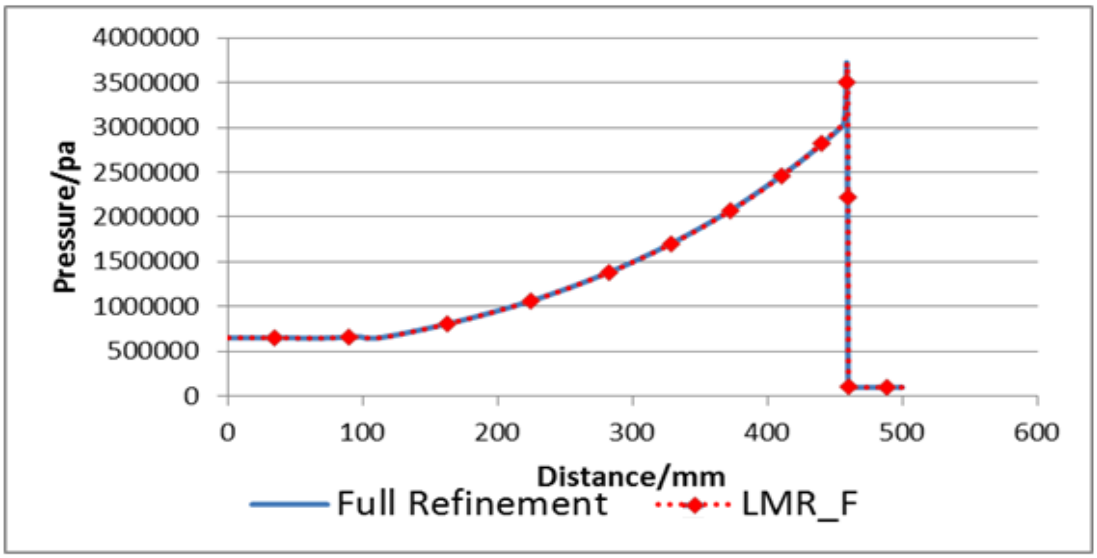



Figure 3.45 Comparison of pressure at T=0.00019s

Similarly to the comparison at the time 0.00015s, the results at the 0.00019s also give very good compatibility to each other. In the time T=0.00019s, the detailed comparison around the peak region is made as well.

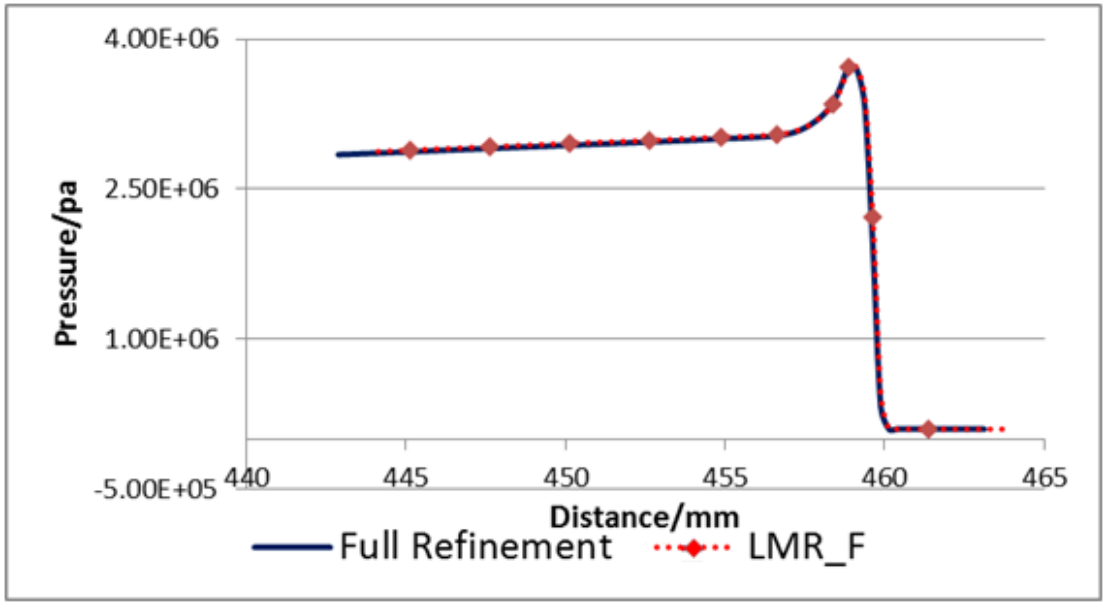



Figure 3.46 Comparison around the peak at T=0.00019s

In the comparison shown in Figure 3.46, calculation with local mesh refinement gives very good performance once more.

As a result, in this 1D detonation simulation, the accuracy of the calculation with LMR has been proved. Meantime, supported by the above calculation results, the validation of the propagation speed and the peak pressure value in the calculation with LMR can be guaranteed in other detonation simulations. Latterly, in the practical application chapter, the local mesh refinement is used in high dimension simulations to test the performances in other aspects in the detonation simulation.

# 4. Load Balancing

In Chapter 3 the technique of local mesh refinement is introduced to the industrial hydrogen combustion code COM3D to improve its efficiency in simulation of detonation and laminar flow. In this chapter, to maintain the high efficiency brought by the new technique under the parallelism, proper domain decomposition algorithm is considered. Although lots of mature and efficient domain decomposition algorithms were developed since 2003 [77], none of them can satisfy the computational domain in COM3D. Differently from the computational domain of common scientific computation, domains for the industrial problems usually contain a considerable ratio of complex geometry structures and may require special consideration. In the following parts, the specific treatments required by the geometry and communication mode in COM3D are discussed firstly, and then a new domain decomposition algorithm is established to optimize the computation with COM3D by consulting the characteristic of the code.

## 4.1. Load balancing for the patch-based LMR in COM3D

In Chapter 2, three kinds of domain decompositions have been introduced. Since the three methods have different emphasis, the selection of domain decomposition should depend on the calculation methods and communication modes in LMR of COM3D.

The most popular domain decompositions are patch-based domain decomposition and hybrid decomposition. As introduced in Chapter 2, the patch-based decomposition pays attention on the data distribution on each level only. The parallelism with patch-based domain decomposition can gain a very good speed up on each level. However, this method lacks the attention on the optimization of inter-level data communication, and may result in very big communication amount between levels. The hybrid domain decomposition contains the advantages of both patch-based decomposition and domain-based decomposition, so the optimizations of both data distribution on each level and inter-level data communication are concerned. In the load distribution, this method cannot make the partition as good as the patch-based approach; and in the optimization of inter-level data communication, the hybrid approach cannot save the inter-level communication as the domain based decomposition.

Although the patch-based domain decomposition is provided in Chombo, the function of load balancing in the code only takes in charge of the distribution of the sub-domains instead of decomposition and distribution [18]. The new domain decomposition should be developed for COM3D with local mesh refinement, especially the patch-based approach will be used.

The patch-based method may result in very huge amount of inter-level data communication, but such shortage is not very serious in the communication. In the three types of inter-level data communication in Chombo, the data communication approached by MPI has been optimized. The process of coarse-to-fine interpolation is shown in the Figure 4.1.

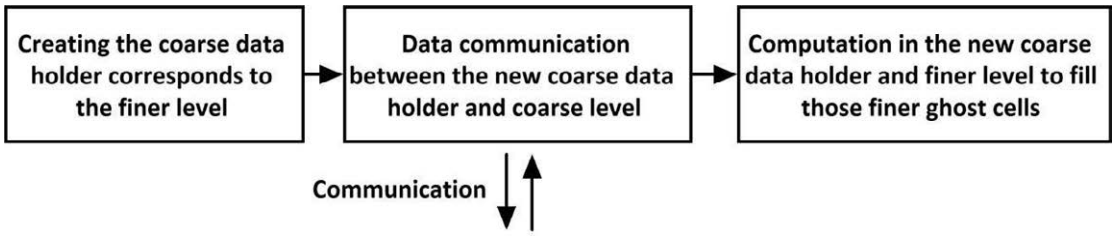



Figure 4.1 Process of coarse-to-fine interpolation

As shown in Figure 4.1, the code creates a new data holder by the coarsened finer grid, and the MPI-based data communication is done between the coarse level data and new temporary coarse data. The communication work is done on the coarse level, in which the total communication amount has been largely reduced. In fine-to-coarse data transfer work, the detailed process is shown as the figure 4.2.

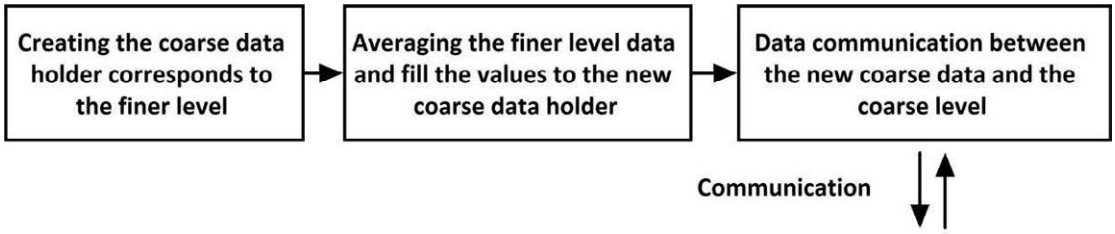



Figure 4.2 Process of fine-to-coarse data transfer

Quite similarly to the coarse-to-fine interpolation, temporary data holder is created in the fine-to-coarse data transfer process. The averaged finer data are transferred to the coarse level through the coarse-to-coarse communication. The process of flux correction is more complicated, containing the flux register updating and the refluxing. Detailed process is shown in Figure 4.3.

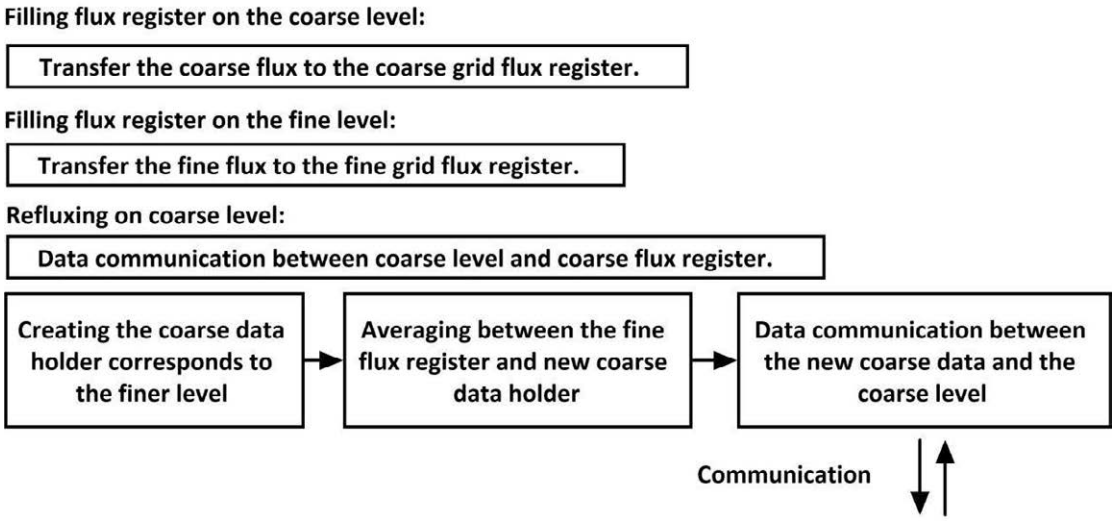



Figure 4.3 Process of flux correction

To make the flux correction, 2 sets of flux registers are created on both coarse and fine levels. Those flux registers have the same distribution as the data grids, so the register updating does not

require the MPI-based communication. In the refluxing, similarly to the other two communications, the averaging should be done between the finer level and the temporary coarse data holder, and the MPI-based data communication is done between the coarse level and the temporary coarse data holder only.

As mentioned in Chapter 2, the selection of decomposition should also depend on the computer used in the parallel computation. The clusters used for the parallel computation are usually infiniband, in which the speed of data transfer can be quite high. Therefore, in the computation with local mesh refinement, the distribution of computational loads is more important than the optimization of inter-level data communication.

In the calculations with domain-based decomposition or hybrid decomposition, some of the inter-level data communications can be done by data copying inside one processer instead of the MPI-based communication between processors. However, the computation, such as interpolation and averaging, still takes a large ratio in the communication process. So, the MPI-based data communication does not take a very big ratio in the whole inter-level data communication, and the advantage of the hybrids decomposition is quite limited.

The patch-based approach has been selected in the development of load balancing. Domain decomposition for the uniform grid case can be established, and then the development of load balancing for the computational domain with LMR will be based on the uniform grids decomposition.

## 4.2. Load balancing for uniform grid

Firstly, the new load balancing algorithm is developed for the uniform grid computational domain. In order to get a better distribution of total computational loads in COM3D, the new decomposition should concern the influences of the solid structures.

### 4.2.1. Design of Load Balancing in COM3D under the uniform grid

In COM3D, the geometries are marked as a set of solid cells in the domain, and sometimes those solid cells store very important boundary information. However, in the detailed computation, only a small ratio of boundary conditions are stored in those solids cells (most boundary conditions are store in the zone part instead), and most of the solid volumes are not included in the calculation. In order to improve the parallel computation and optimize the memory storage, some unnecessary solid areas should be removed.

Some loops are done in the computational domain to get the distribution of the gas cells. The gas distributions are used to find junctions in the computational domain. The junction here is defined as the face where the gas cells number on both sides are quite different, as the example shown in Figure 4.4.

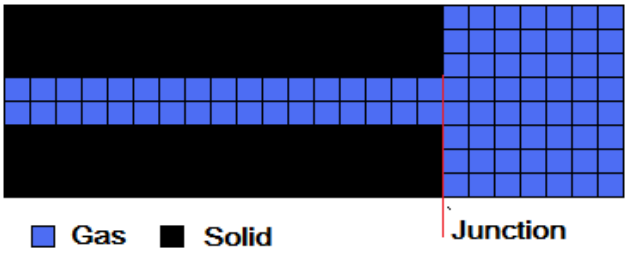



Figure 4.4 One computation domain with junction

The junction in Figure 4.4 is the connection of two different gas structures. On the left of the junction, most of area is covered by solid and a small ratio of the volume is gas. In contrast, the part on right side is full of gas cells. If decomposition is done to the computational domain directly without considering the junction, some sub-domains may contain large ratios of solid cells, which may waste lots of computation time and memory storage.

By the splitting at the junction and proper shrinking of two sub-domains, the unnecessary solid cells can be removed, and then the computational domain can be decomposed into two pure gas blocks. Consequently, large amount of memory and computational efforts can be saved. The simple geometry based decomposition can make further splitting in the two blocks perfectly, which means that lots of working efforts can be saved. The definition of junctions is very important to both the domain decomposition and parallel computation.

In the new decomposition method, the potential junctions are found by the distribution functions. After traversing the cells in computational domain, the program can get the gas distribution functions $D[d]\,(d=0,1,2)$ which indicate the gas distribution in each direction, solid distribution function $SD[d]\,(d=0,1,2)$ which indicate the solid distribution in each direction, right gas continuous distribution functions $RD[d]\,(d=0,1,2)$ which conclude the distribution of the gas cells which do not contain the solid surface on the right hand side in each direction and right solid continuous distribution functions $RSD[d]\,(d=0,1,2)$ which conclude the distribution of the solid cells whose neighbor cell on the right side is still solid in each direction. The judgment of junctions is begun from the longest edge of the computational domain to the shortest. In the judgment, comparisons are done among the four distribution functions to find if there are big differences in some position. When the difference between the two distribution functions of gas or solid at some position is big enough, the mark of junction is made. The solid cell distribution functions are used to find the expansion type of junction and the gas distribution functions are used to fine the contraction type of junctions. It is worth to note that the right continuous distribution functions are for preventing the situation shown in the Figure 4.5. In the figure, since the distribution of gas cells is uniform along the longest edge, the junction will be missed if the distributions of gas and solid cells are the only references for the junction judgment. In contrast, with the help of the right continuous distribution in which the right side status of each cell is detected, such junction can be discovered.

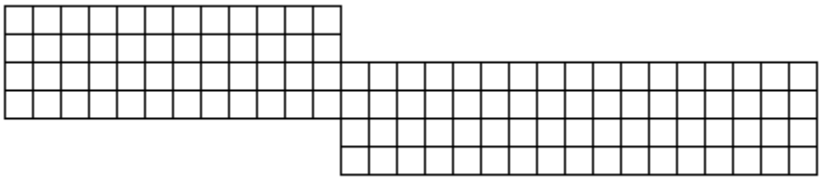


Figure 4.5 One special junction

In one round of junction judgment, several junctions might be located, but the splitting is done only at one of them. In order to optimize the data communication, splitting is done at the junction which is the nearest to the gravity center of the domain in this direction. After the splitting, the shrinking of the two sub-domains is made. In case of the existence of other junctions, the junction judgment and splitting should be done again to the sub-domains if they are still big enough (the total computational efforts contained by this sub-domain is still very big). However, overmuch work of junction judgments and splitting may cause bad data communication and huge time cost in traversing, no more than 3 times of judging and splitting are done at the moment. In the case that no proper junction has been found, the sub-domain is sent to the next step's splitting. The above junction judging and splitting is called as step 1, the process is shown in Figure 4.6.

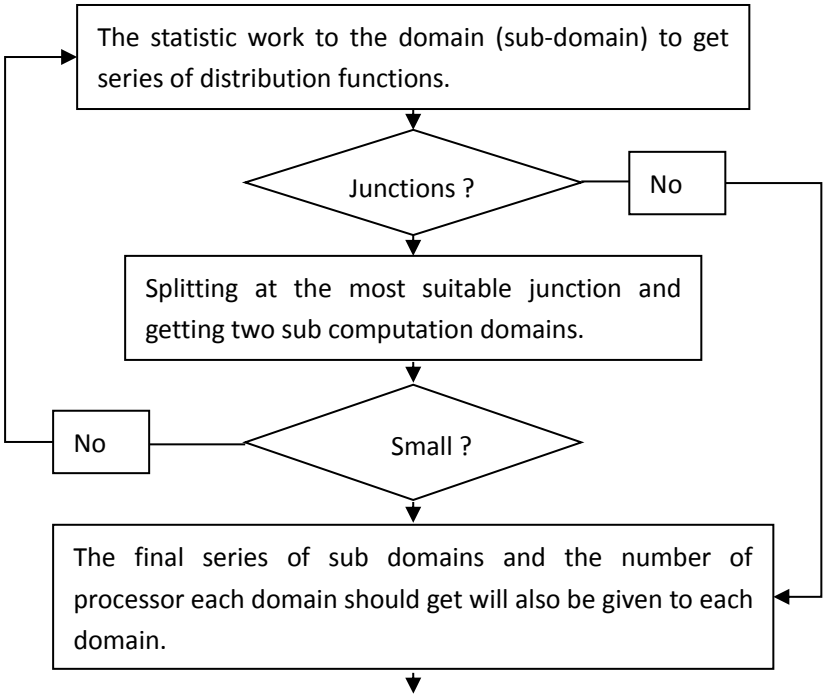



Figure 4.6 Process of step 1

In the step 2, the sub-domains are decomposed further. When the sub-domain has an extremely large aspect ratio, it can be split along the longest edge directly. The sub-domains achieved by the 1D final decomposition do not need any further decomposition. If the sub-domain is not suitable for the 1D final decomposition, splitting work is made along the longest side with the 4-Mod, 16-Mod or 64-Mod splitting methods to get several data blocks and one remain block.

Under the 4-Mod splitting mode, the sub-domain with N processors (means the sub-domain contains the working load for N processors) is split along the longest side, a series of ([N/4]) data blocks with four processors and one remain block with $M \equiv N(\mathrm{mod}\,4)$ processors are gotten. In the other two splitting mode, the integers 16 and 64 are used to make the module. In order to achieve optimal surface-to-volume ratio, which evaluates the communication-to-computation ratio, different splitting modes are used in different geometry structures. The cubic region has the best surface-to-volume ratio among the rectangular blocks with the same volume, all the three splitting modes mentioned above are devoted to make each final data block near to the cubic size. The 3 splitting modes are used to simplify the decomposition in next step, in which three kinds of data blocks need to be cut. Comparing to the splitting in bisection, the splitting method used here requires less traversing and saves time.

In the cutting phase, there are two available methods to make the decomposition more efficient. In the case that the fraction of the gas is very high, simple geometry method is used. In the cases that ratio of the gas cells is not big enough, the traversing is necessary in the decomposition. The process of step 2 is shown in Figure 4.7.

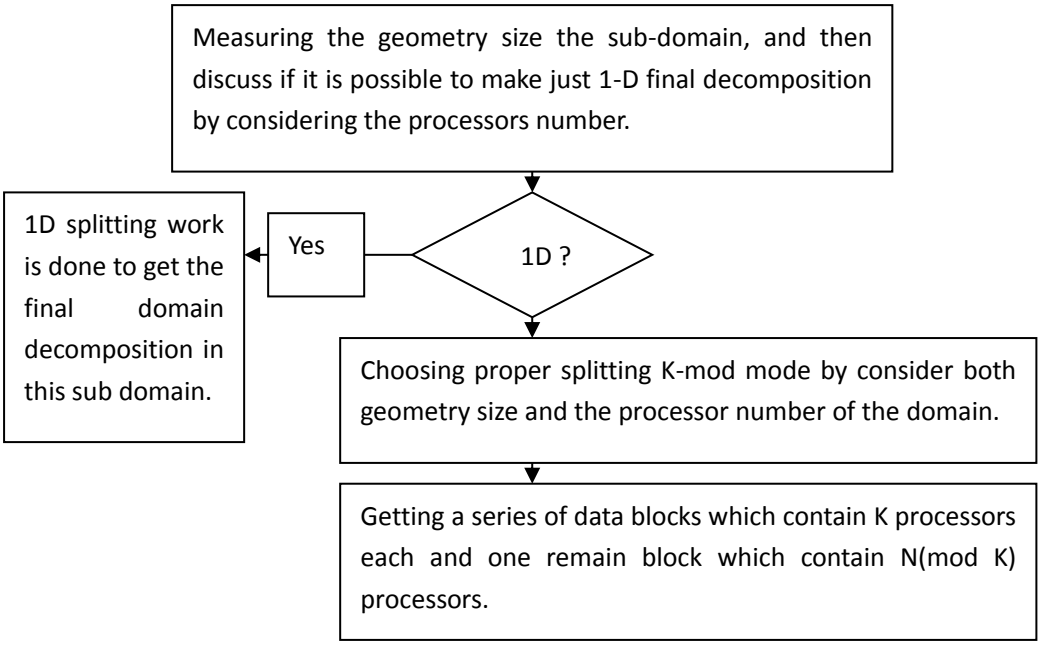



Figure 4.7 Process of step 2

After the further decomposition in the step 2, the final decompositions for data blocks and remain blocks are done in step 3. In this part of work, two kinds of blocks are decomposed differently, the one used to make the final decomposition in the data blocks is called step 3.1 and the one for the remain blocks is called step 3.2.

In the step 3.1, each data block should be cut into 4, 16 or 64 pieces. Similarly to the process in step 2, the fraction of gas cells in the block is verified for selecting efficient cutting method. Three cutting method are optional. First one is the simple geometry based decomposition, corresponding

to the cases that the gas fraction is very high in the data block. The splitting in geometry based decomposition considers the geometry size of the block and keeps a good surface-to-volume ratio. Second option is the hybrid decomposition for the cases in which the gas cells are with a very big ratio but the influence of the solid cells cannot be ignored. In this type of decomposition, the splitting is dominated by both geometry size of the block and the gas distribution in the block. The third option is the bisection for the data block with large solid fraction. In the splitting, the block should be traversed by several times until the decomposition is finished. The loops (the times of traverses required for statistics of the gas cells in the domain) required by the three splitting methods are quite different, the comparison of them is shown in Table 4.1.

Table 4.1 Loops for statistics of gas cells in three splitting methods

| The option | Corresponding cases | Loops at N=4 | Loops at N=16 | Loops at N=64 |
|---|---|---|---|---|
| Geometry based method | The gas fraction f(g)>b in the block | 0 | 0 | 0 |
| Hybrid | b>=f(g)>=a | 1-2 | 1-3 | 2-3 |
| Bisection | F(g)<a | 2 | 4 | 6 |

As shown in Table 4.1, the geometry based decomposition costs the least time in splitting, since it does not contain any traversing. The Hybrid decomposition is in the second position, it still needs to traverse the cells several times to achieve an optimal partition. The bisection decomposition is in the last position, in order to get optimal result, all the cells should be traversed by $\log_2 N$ times. With these three methods, the decomposition of each block is always done as efficient as possible, and the costs in the domain decomposition can be controlled under an acceptable level. For the calculation with adaptive local mesh refinement, in which the decomposition on finer level should be made frequently, the efficiency gained by the above decomposition is very important. Detailed process of this step is shown in Figure 4.8.

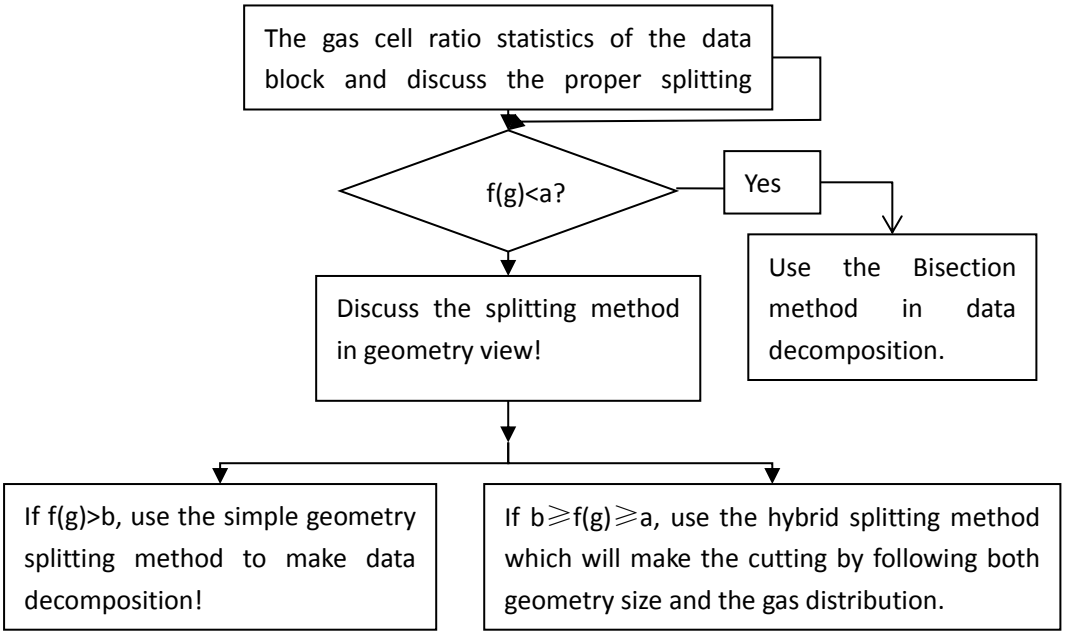



Figure 4.8 Process of step 3.1

The step 3.2 is for the decomposition of remain blocks. In this part, the 4 or 16 module decomposition is used here for further splitting until no further decomposition can be made. Then, the normal data blocks produced by the decomposition are sent to step 3.1 to make the final decomposition and the last remained block is decomposed along its longest edge. The detailed process of this step is shown in Figure 4.9.

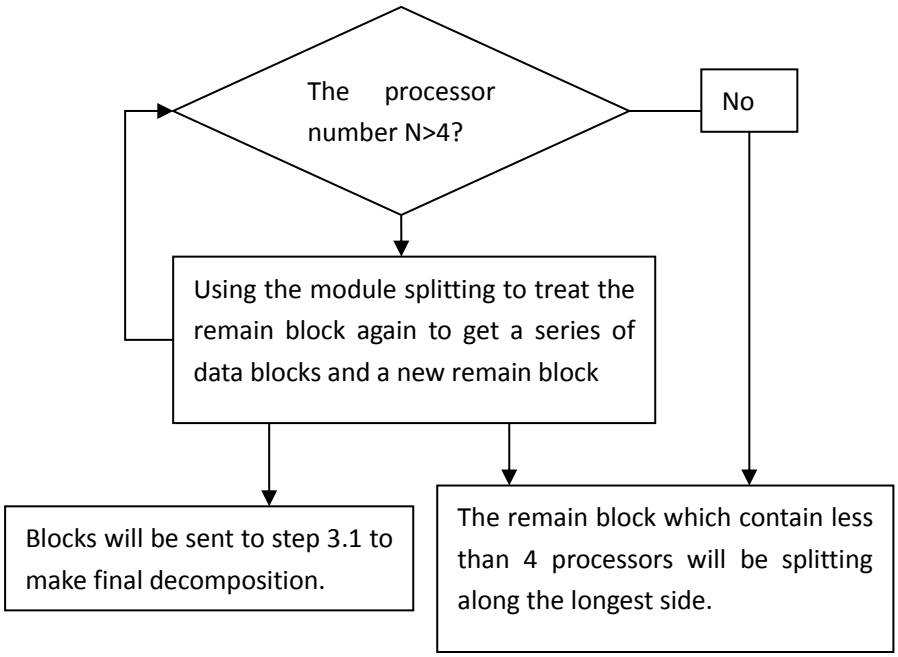



Figure 4.9 Process of step 3.2

According to the method above with several different steps of decompositions, a series of sub-domains with the near same computational loads and an optimal data communication can be established.

4.2.2. Splitting uniform grids domains with new load balancing

Some testing cases are made in this section to verify the load balancing. Before testing, the unbalancing ratio (UBR) used to evaluate the performance of domain decomposition is introduced.

Unbalancing ratio: $\kappa = \frac{N_{\max}}{N_{aver}}$

$N_{\max}$ : the maximum computational load contained by processor after domain decomposition
$N_{aver}$ : the ideal computational load

With this new concept, the performances of different domain decompositions can be compared. The one which has smaller unbalancing ratio usually gains higher Speedup in the parallel computation.

The first testing case is the decomposition for EPR containment. The geometry information about the containment is shown in Table 4.2. Distribution of gas cells in EPR containment is symmetric in some directions, this testing case can also be regarded as the computation domain with symmetric gas distribution.

Table 4.2 Grid information of EPR containment

| EPR Containment | |
|---|---|
| Domain Size | 118X155X118 |
| Total Cells Number | 2,158,220 |
| Total Gas Cells Number | 1,152,402 |
| The Gas Volume Fraction | 53.4% |

In the testing, 8 different numbers of processors are used to test if the new domain decomposition can keep on making very good partition in a large range of processor numbers. The Table 4.3 shows the comparisons of the partitions between the new load balancing and old geometry based domain decomposition.

Table 4.3 Comparison between two decompositions in EPR

| Number of processors | Average (gas cells) | Decomposition by LB | | Decomposition by geometry | |
|---|---|---|---|---|---|
| | | Gas cells | UBR | Gas cells | UBR |
| 16 | 72025 | 73333 | 1.018 | 92630 | 1.286 |
| 32 | 36012 | 37308 | 1.036 | 46712 | 1.297 |
| 48 | 24008 | 25128 | 1.047 | 30432 | 1.268 |
| 64 | 18006 | 19237 | 1.068 | 33032 | 1.834 |
| 80 | 14405 | 15378 | 1.068 | 26220 | 1.820 |
| 96 | 12004 | 12599 | 1.050 | 21449 | 1.787 |
| 112 | 10289 | 11117 | 1.080 | 19634 | 1.908 |
| 128 | 9003 | 9772 | 1.085 | 17100 | 1.899 |

As shown by the table, geometry based decomposition gives acceptable partition when the processor number is less than 60. As the increasing of processors number, the unbalancing ratio (UBR) of the geometry based decomposition increases. The new load balancing method gives much better result at any number of processors.

So, in the industrial simulations which contain symmetrical or regular gas distribution, the new load balancing method can give very good performance, especially in the cases that large numbers of processors are used.

After the testing with symmetric gas distribution, the testing with irregular gas distribution should be made as well. The Rut facility is used here and information about the computational domain is shown in Table 4.4.

Table 4.4 Grid structure of the Rut Facility

| Rut Facility | |
|---|---|
| Domain Size | 98X94X416 |
| Total Cells Number | 3,832,192 |
| Total Gas Cells Number | 924,722 |
| The Gas Volume Fraction | 24.1% |

Unlike the EPR Containment, the Rut facility is not symmetric in any direction and the gas volume fraction is much smaller. The equal computational load distribution is more important in such kind of problems. The comparison between the load balancing and geometry based decomposition is shown in Table 4.5.

Table 4.5 Comparison between two decompositions in Rut Facility

| Number of processors | Average (gas cells) | Decomposition by LB | | Decomposition by geometry | |
|---|---|---|---|---|---|
| | | Gas cells | UBR | Gas cells | UBR |
| 16 | 57795 | 60203 | 1.042 | 162606 | 2.813 |
| 32 | 28897 | 30362 | 1.051 | 83114 | 2.876 |
| 48 | 19265 | 20589 | 1.070 | 54476 | 2.827 |
| 64 | 14448 | 15610 | 1.080 | 42206 | 2.921 |
| 80 | 11559 | 12512 | 1.082 | 35930 | 3.108 |
| 96 | 9632 | 10296 | 1.070 | 28770 | 2.987 |
| 112 | 8256 | 9157 | 1.109 | 35995 | 4.360 |
| 128 | 7224 | 7832 | 1.084 | 21536 | 2.981 |

As shown in Table 4.5, it is clear that the unbalancing ratio of geometry based decomposition is much bigger than the load balancing method developed in this thesis. Sometimes, the computation with the geometry based method may cost 3 times in time more than the calculation with absolute average decomposition. The new load balancing method gives much better result at any number of processors in the simulation with irregular gas distribution as well.

In the view of unbalancing ratio of domain decomposition, the new load balancing shows very good performance. In the former pages, it has been mentioned that a reasonable surface-to-volume ratio should be kept in each final block to optimize the data communication. In the testing cases above, the optimization in this aspect is achieved by two parts. Firstly, only one block of data is given to each processor for the optimization of the communication, although one processor can get

multiple blocks of data to achieve better UBR in the computation. Since the surface-to-volume ratio may be reduced largely with the existence of the multiple data blocks, the efficiency of the parallelism is worried. In the step 1, some big solid occupied areas have already been removed, the benefits of multiple blocks in one processor become limited in the contribution of UBR. Compared to multiple blocks distribution, the single block data is more proper to the current simulation. Another optimization of data communication in the new method is the limitation of the block size in splitting. The geometry structure of the rectangular region is measured before each splitting to make each sub-domain close to cubic size in final partition. In the final partitions of both EPR containment and Rut facility, the aspect ratio of each data block is less than 2 or even 1.5.

### 4.3. Patch-based decomposition for COM3D with LMR

In the section 4.2, a new domain decomposition method has been established for the uniform grid case. Since the patch-based decomposition is decided to be used in the software, the uniform grids decomposition can be used to make the decomposition on the base level and guide the decomposition on finer levels.

#### 4.3.1. Differences between the base level and finer level

In the computational domain with local mesh refinement, the base level is the same as the uniform grid case, and the finer level may be composed by a list of block-structured patches. The domain decomposition for the base level is the same as the uniform grid case, but more work may be done in the decomposition on finer levels.

On the fine levels, for the patches in which the cell number is less than average, domain decomposition is unnecessary. For the patches which need to be decomposed further, the necessity of the junction judgment should be discussed. To the big patch block with more than $N$ (the detailed value should be decided by the computer and industrial simulation) processors, the junction judgment is necessary. To the patches with less number of processors, the discussion of junctions may be a waste of time. The decomposition in the step 2 on finer level may produce small remain block in which the number of gas cells is less than average. In principle, partition with small remain blocks usually has a better distribution of computational loads, and the data communication may also be increased a lot. For an optimal communications, the production of small remain block is allowed for the cases that more than $K$ (here, this number should also be decided by the computer used and detail industrial problem) patches are on finer level. For the cases that the existence of remain block is not allowed, the processor number is rounded in decomposition. The discussion on the finer level is shown in Figure 4.12:

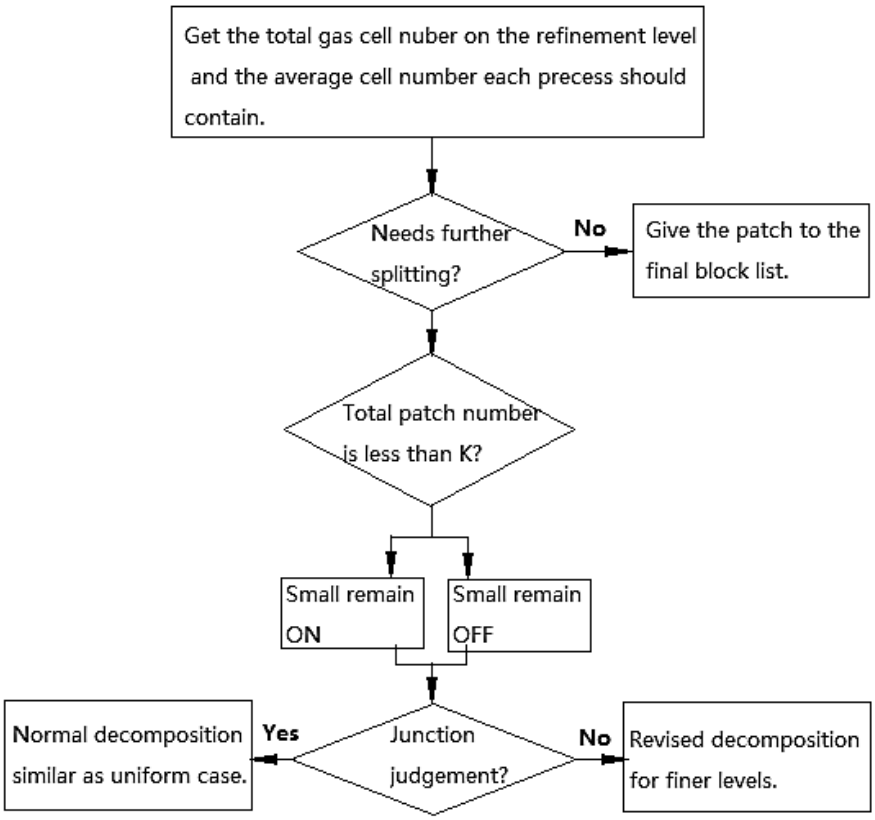



Figure 4.10 Discussion of splitting mode for each patch on finer level

4.3.2. Decomposition on finer levels

The decomposition for the patches which need the junction judgment is shown in the figure 4.13.

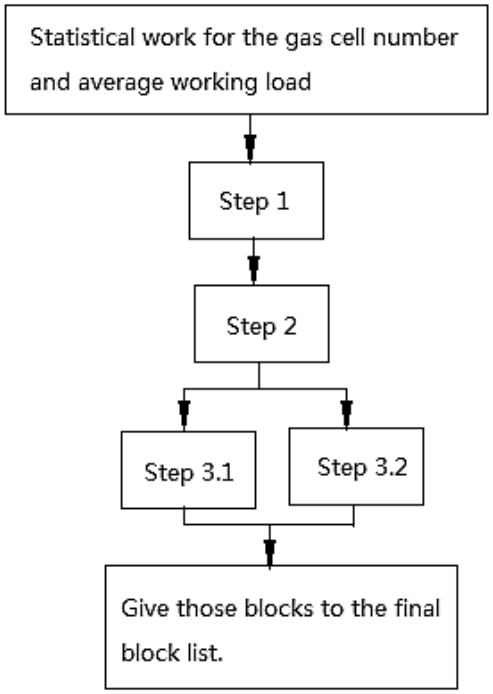



Figure 4.11 Decomposition process for the big patches

The step 2 and step 3 shown in the above may produce small patches which do not contain enough gas cells. For the decomposition which does not need the junction judgment, the process is simpler and shown in Figure 4.14.

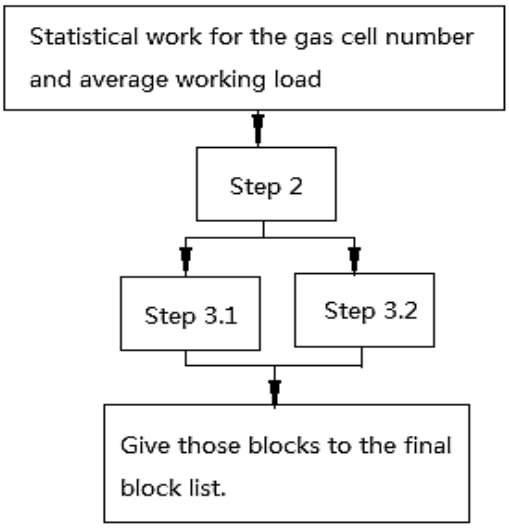



Figure 4.12 Decomposition process for the patches without junction judgment

At the end, all the decomposed blocks should be sent to the final block list to make further arrangement. In many literatures, it has been referred that an optimal block arrangement in the final distribution can optimize the inter-level data communication [82]. Fortunately, the optimal block arrangement [67] [78] [79] has been included in the load balancing in Chombo, this code can be used for the final block distribution for COM3D.

In summary, in the decomposition for the domain with LMR, the base level should be decomposed by the method used in uniform case, and the decompositions on finer levels are done by the methods shown in the Figure 4.13 and Figure 4.14. At the last, the block distribution is made by using the load balancing code in Chombo. The patch-based domain decomposition for the COM3D with LMR is established.

### 4.3.3. Speedup gained by load balancing in detonation simulation with LMR

In order to show the performance of the new domain decomposition, the 2D detonation simulations with LMR are made under different numbers of processors. The time cost, speedup and parallel efficiency of the simulations with different processors are shown in Table 4.6.

Table 4.6 Speedup gained by the new decomposition method in Detonation simulation

| Number of processors | Time cost (hours) | Speedup | Efficiency |
|---|---|---|---|
| 1 | 43.5 | 1 | 1 |
| 2 | 22 | 1.98 | 0.98 |
| 4 | 11.4 | 3.81 | 0.95 |
| 8 | 6 | 7.25 | 0.91 |
| 16 | 3.3 | 13.18 | 0.82 |
| 32 | 1.8 | 24.17 | 0.76 |
| 64 | 1.2 | 36.25 | 0.57 |

Similarly to the common parallel computation, the speedup of the parallel computation increases as the increasing of the number of processors used in the calculation, but the efficiency of the parallel computation decreases as the increasing of the processors.

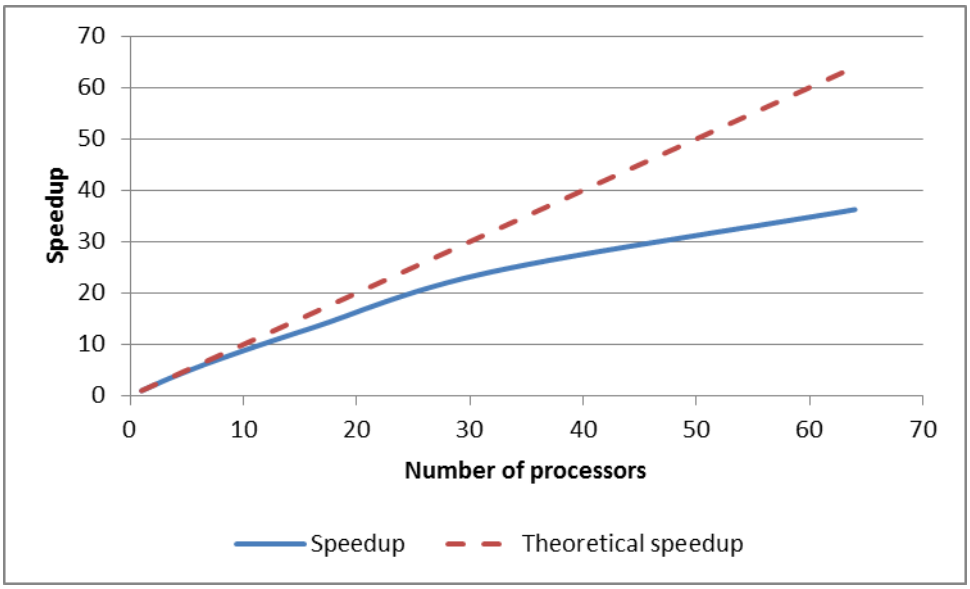



Figure 4.13 Speed up gained by the new decomposition

To show the speedup gained by the new domain decomposition in a more clearly way, the comparison between the real speedup and theoretical speedup is shown in the Figure 4.15. The patch-based decomposition for the computational domain with local mesh refinement in COM3D manages to optimize the parallel computation.

# 5. Application of LMR

With the technique of local mesh refinement and the optimal domain decomposition, some simulations suffering a lot from the huge computational efforts under the uniform grid can be calculated quite fast now. In this chapter, several simple and classical simulations are made instead of complex industrial problem, which is under the consideration of verification.

## 5.1. Code verification with LMR

In this section, the 2D advection and diffusion of nitrogen into oxygen is simulated to show the advantages of LMR in simulation of laminar flow. The results of simulation with local mesh refinement are compared with the analytical solution, so the comparison can also verify the implementation of the LMR technique. In this 2D problem, the analytical solution is gotten from the 2D problem (5-1) [91]. In (5-1), $u$ stands for the mass fraction of nitrogen gas, $V$ indicates the advection velocity and the parameter $D$ is the diffusion factor. In this 2D diffusion case, $y_0 = 0.02$ and $y_1 = 0.03$.

$$\begin{aligned} &\frac{\partial u}{\partial t} + V\frac{\partial u}{\partial x} - D\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = 0 \\ &u(x, y, 0) = 0 \\ &u(0, y, t) = \begin{cases} 1, if\ \ y \in [y_0, y_1] \\ 0, otherwise \end{cases} \end{aligned} \tag{5-1}$$

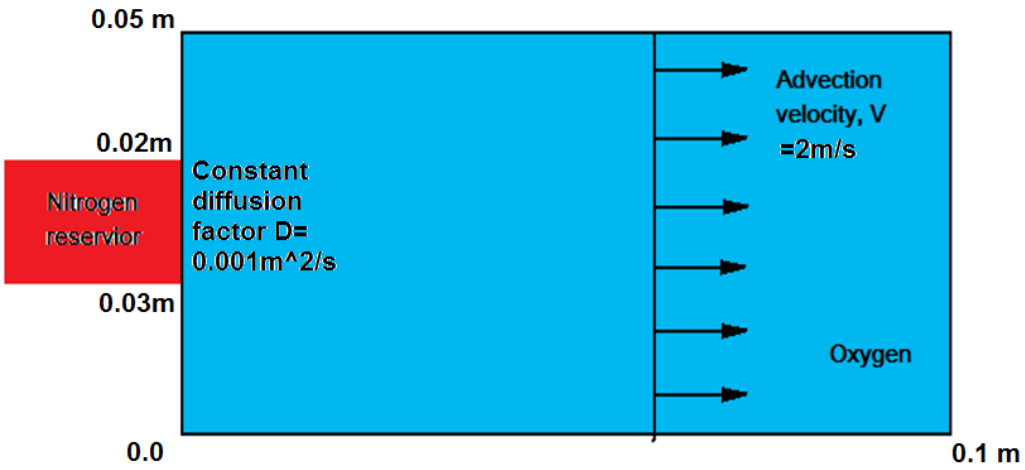



Figure 5.1 2D advection diffusion problem of nitrogen into oxygen

In this 2D problem (shown by Figure 5.1), the computational domain is filled with oxygen initially, and a constant advection in X direction is installed in the domain. In the left bound of this 2D domain, the nitrogen sorce is setted at ($y_0$, $y_1$), and the same concetion velocity is given as well at the boundary. The detailed grid, boundary and initial conditions of this 2D diffusion problem are shown in Table 5.1.

Table 5.1 Grid information and the initial conditions of 2D diffusion

| Grid Information | |
|---|---|
| Nitrogen Reservior | Computational domain |
| Region: (-2, 20, 0)X(-1, 29, 1)<br>Resolution: 0.001m | Region: (0, 0, 0)X(99, 49, 1)<br>Resolution: 0.001m |
| Initial Condition: | |
| Nitrogen Reservior | Computational domain |
| Velocity(m/s): (2, 0, 0)<br>Pressure (Pa): 101300<br>Temperature (K): 298<br>Gas Components:<br>$H_2 : O_2 : N_2 : H_2O = 0:0:1:0$ | Velocity(m/s): (2, 0, 0)<br>Pressure (Pa): 101300<br>Temperature (K): 298<br>Gas Components:<br>$H_2 : O_2 : N_2 : H_2O = 0:1:0:0$ |

Velocity 2m/s and the diffusion factor 0.001 $m^2 / s$ are fixed in the problem, therefore analytical solution of this problem can be achieved by the Green's function method [91] easily:

$$u(x,y,t) = \int_0^t \frac{1}{\sqrt{64\pi D\tau^3}} \cdot \{(x - V\tau)\exp[-\frac{(x-V\tau)^2}{4D\tau}] + (x+V\tau)\exp[\frac{Vx}{D} - \frac{(x+V\tau)^2}{4D\tau}]\} \cdot [erfc(\frac{y-y_1}{\sqrt{4D\tau}}) - erfc(\frac{y-y_0}{\sqrt{4D\tau}})]d\tau \tag{5-2}$$

In the simulation of diffusion, final results suffer a lot from numierical diffusion introduced by the discrete differences and sometimes the unphysical diffusion can even be several times bigger than the natural diffusion. High order numerical schemes and small cell sizes can be used to reduce the impact of numerical diffusion. In this section, the van Leer solver is used to make the simulation in convection and central difference is used in calculation of molecular transportation, the numerical diffusion can only be affected by the cell sizes.

### 5.1.1. Numerical diffusion caused by cell size

Before the simulation of 2D diffusion, 1D diffusion is calculated to show the influences of numerical difffusion under different resolutions. The computational domain of this 1D diffusion and the initial conditions are shown in Figure 5.2 and Table 5.2.


Figure 5.2 1D diffusion

Table 5.2 Initial conditions of 1D diffusion

| Initial Conditions | |
|---|---|
| Region 1 | Region 2 |
| Velocity(m/s): (0, 0, 0)<br>Pressure (Pa): 101300<br>Temperature (K): 298<br>Gas Components:<br>$H_2 : O_2 : N_2 : H_2O = 1:0:0:0$ | Velocity(m/s): (0, 0, 0)<br>Pressure (Pa): 101300<br>Temperature (K): 298<br>Gas Components:<br>$H_2 : O_2 : N_2 : H_2O = 1:1:3.76:0$ |

In the 1D diffusion simulation, three different resolutions are used. The coarsest resolution is 0.001m and the other two resolutions are made by refinement ratio two and four. Similarly to the 2D advection diffusion, diffusion factor is the constant 0.001. In Figure 5.3, mass fractions of hygrogen gas at physical time T=0.00049s are given. It is clear that diffusions of hydrogen gas are different under different resolutions, and the one with coarser resolution has bigger numerical diffusion. This figure also shows that the numerical diffusion can be controled by finer resolutions.

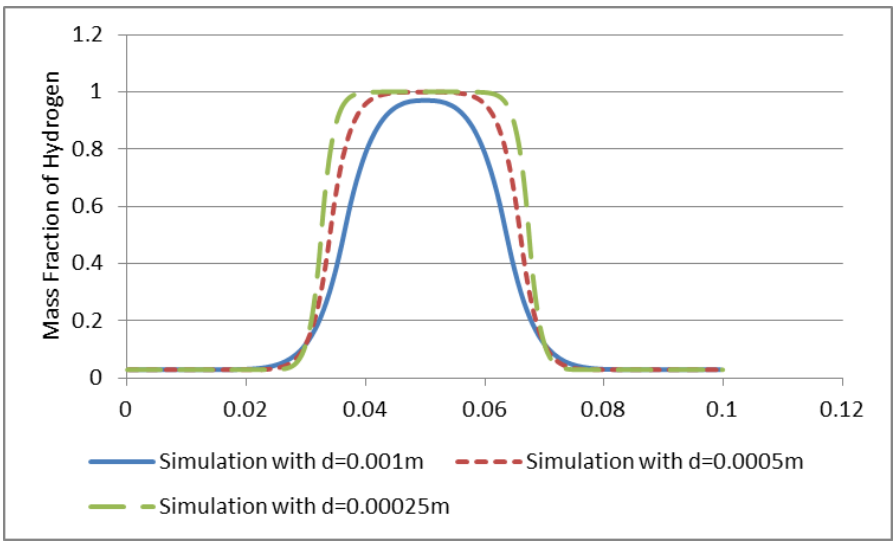



Figure 5.3 Mass fraction of hydrogen gas at the T=0.00049s

### 5.1.2. 2D diffusion simulation with LMR

In the 1D simulations, numerical diffusions brought by different resolutions show the importance of high resolution. In 2D simulations, the influence of the numerical diffusion is more serious than the 1D case. Uniform grids clauclations with three different resolutions for the simulation of 2D advection diffusion are made. Grid information of all three simulation cases are shown in Table 5.3.

Table 5.3 Grid information of uniform calculation

| Case | Computational domain | Cell size |
|---|---|---|
| Case 1 | (0, 0, 0)X(99, 49, 1) | 1mm |
| Case 2 | (0, 0, 0)X(199, 99, 1) | 0.5mm |
| Case 3 | (0, 0, 0)X(399, 199, 1) | 0.25mm |

Calculations with three different resolutions are made long enough and reach the physical time 0.0108s. Figure 5.4 shows the simulation result of the case 1.

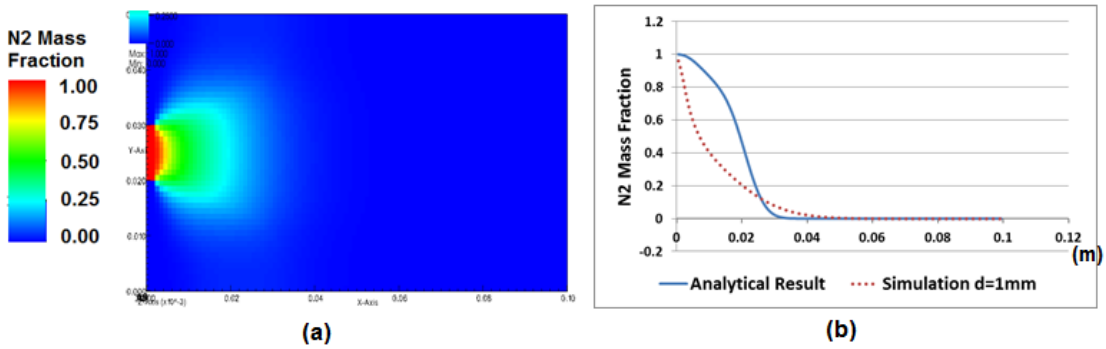



Figure 5.4 Results of simulation with resolution d=1mm

In Figure 5.4, mass fraction of the nitrogen gas is shown in part (a) and the comparison of nitrogen gas mass fraction in central line between the numerical work with resolution 1mm and the analytical solution (which is shown by solid line in the figure) is shown in part (b). The total diffusion with resolution 1mm is much bigger than the natural diffusion.

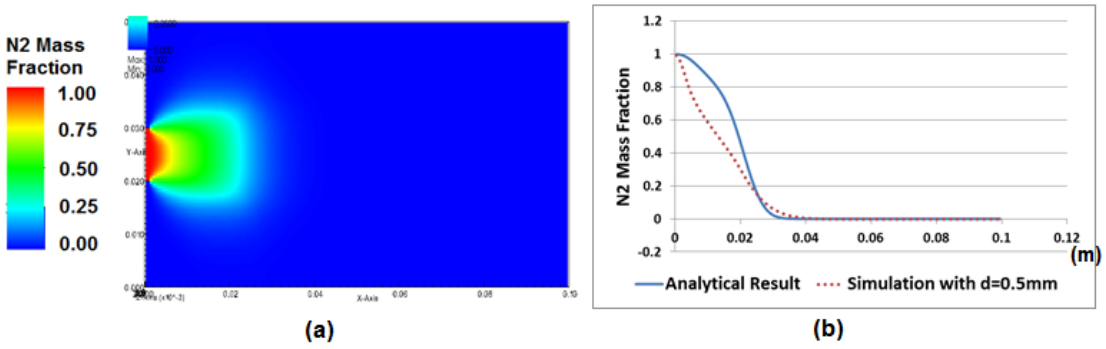



Figure 5.5 Results of simulation with resolution d=0.5mm

Figure 5.5 shows the result of the 0.5mm resolution simulation. In part (b), comparison between numerical work with resolution 0.5mm and analytical solution still shows big differences. However, Compared with Case 1, the result has already been largely improved.

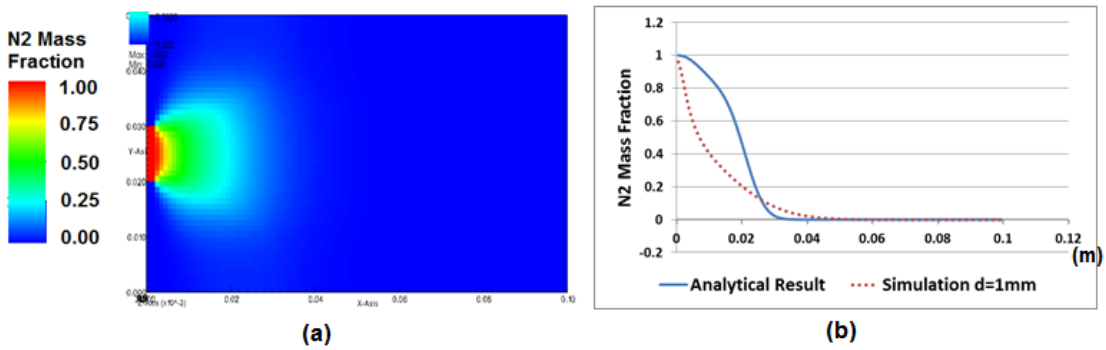



Figure 5.6 Results of simulation with resolution d=0.25mm

Figure 5.6 shows the result of simulation with resolution 0.25mm. In the comparison of mass fraction at position y=0.125mm, this uniform grid simulation shows very good result in most parts expect the area near nitrogen source. Comparing to the numerical diffusions in former two coarser grid simulations, numerical diffusion in Case 3 has been reduced a lot.

According to the performances of simulations with different resolutions, it is quite possible that better simulation results can be achieved when finer resolution such as 0.125mm is used in the calculation. However, that may result in quite huge computational efforts.

Table 5.4 Total computational cost of the three uniform grid simulations

| Computational case | Total computational cost |
|---|---|
| Case 1 | 12 hours for 30000 steps with 2 CPU |
| Case 2 | 100 hours for 60000 steps with 2 CPUs |
| Case 3 | 210 hours for 120000 steps with 8 CPUs |

As shown in Table 5.4, the computational cost is increased by 8 times when the refinement ratio two is used. Almost 3 months are required if the resolution 0.125m is used in this 2D simulation.

In order to reduce total computational efforts, technique of adaptive local mesh refinement is used. As shown in the Case 3, the resolution 0.25mm shows bad performances in controlling the numerical diffusion near the nitrogen reservoir. In the simulation with ALMR, two finer levels are used. Besides the first level with the refinement ratio four, the second refinement level with refinement ratio two is used near the reservoir to get better result. In order to save time, the forward Euler scheme is used, and the refinement ratio two is possible in LMR. The re-gridding phase is as below.

(1) At the end of each big step of calculation, the distribution of nitrogen gas in the covered coarser regions is detected. When the mass fraction near the bond of those regions is big enough, finer grid should be extended.
(2) For convenience, both two finer levels are extended by 10% in the grids regeneration. The extension of finer grids is done in the X direction because of the advection effect in X direction.
(3) After the regeneration of the first and second refinement levels, finer level data holders are re-defined and re-initialized. Both linear coarse-to-fine data transfer and data copying from auxiliary data holder are used in the re-initialization. Then, the auxiliary data holders are regenerated and reinitialized.
(4) Finally, all the inter-level data communications are re-built.

Initial grid information is shown as following table and figure.

Table 5.5 Initial grid information of 2D diffusion with LMR

| Grid information | | |
|---|---|---|
| The base level | The first level | The second level |
| Computational domain:<br>(0, 0, 0)X(99, 49, 1)<br>Resolution:<br>1mm | Computational domain:<br>(0, 46, 0)X(39, 139, 7)<br>Resolution:<br>0.25mm | Computational domain:<br>(0, 160, 0)X(39, 239, 15)<br>Resolution:<br>0.125mm |

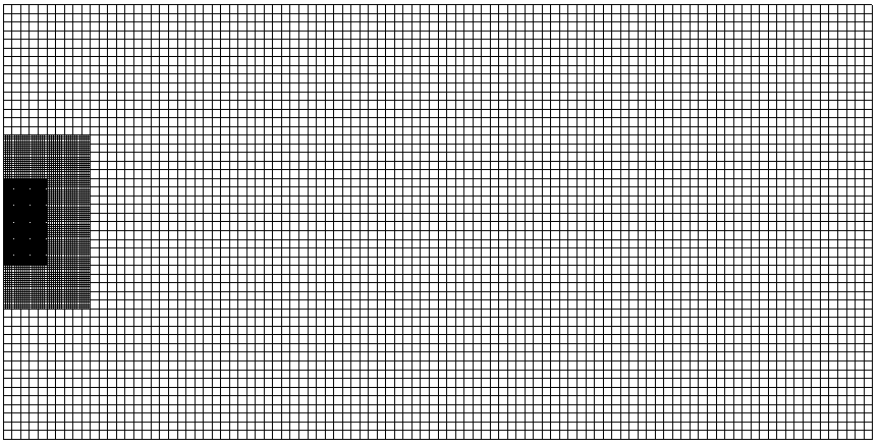


Figure 5.7 Initial grid of the simulation of 2D diffusion

Similarly to the uniform grid calculation, simulation with adaptive local mesh refinement (ALMR) is calculated long enough. Grid structures have been changed several times as the diffusion of nitrogen. The results at step 10000, 20000 and 30000 are given to show the performance of simulation with ALMR.

The result of simulation at step 10000 corresponding to physical time T=0.0036s is shown in Figure 5.8, mass fraction of nitrogen gas is shown in part (a) and the grid structure is shown in part (b).

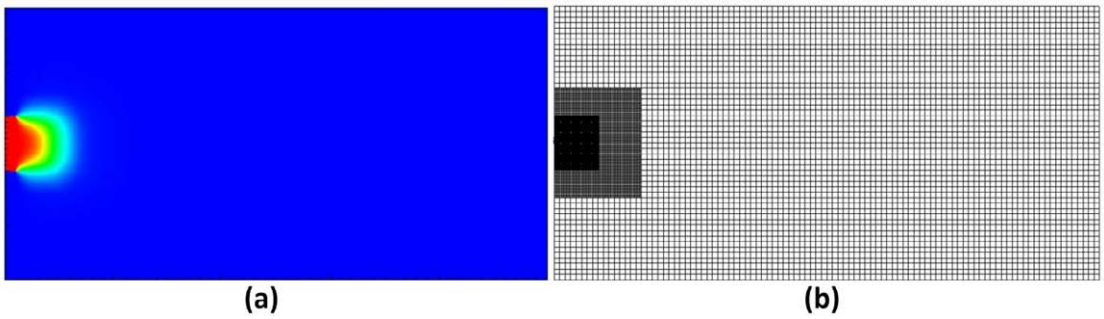



Figure 5.8 Nitrogen mass fraction (a) and grids (b) of the simulation with ALMR at T=0.0036s

As shown in the figure, the areas where mass fraction of nitrogen is bigger than 0.05 are covered by the first refinement level. For compensating the bad performance of resolution 0.25mm near nitrogen source, the second refinement level is extended to cover all the regions where mass fraction of nitrogen is bigger than 0.8. Following figure shows the comparison between simulation result and analytical solution at the position y=0.5mm.

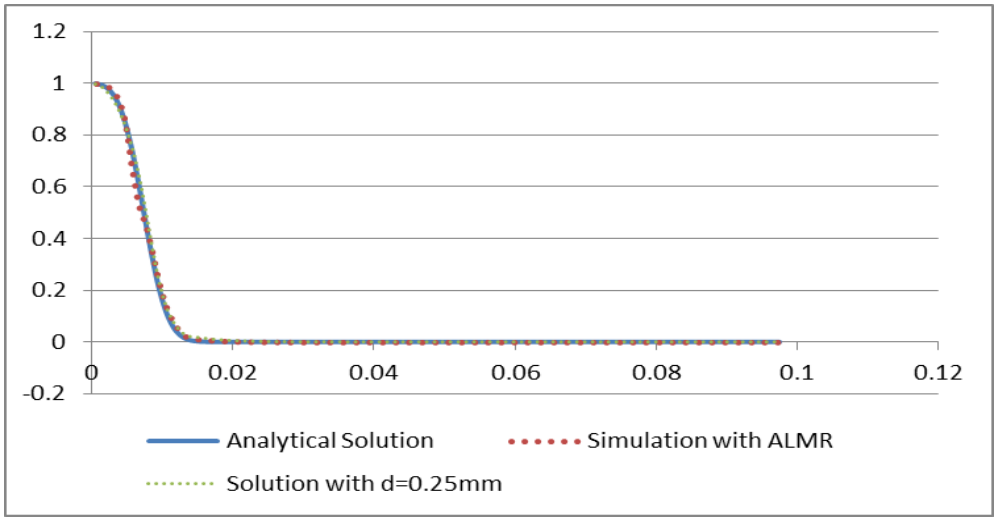



Figure 5.9 Comparison of central line nitrogen mass fractions at T=0.0036s

Shown by the figure, results of the simulation with ALMR is quite close to the analytical solution at the time T=0.0036s. The uniform grid calculation with resolution 0.25mm shown in the figure can also give a very close result at this moment. Thus, benefit of the second refinement level is not very obvious at step 10000.

The nitrogen gas mass fraction and the grid structure of the simulation with ALMR at step 20000 are shown in Figure 5.10. This moment corresponds to the physical time T=0.00723s.

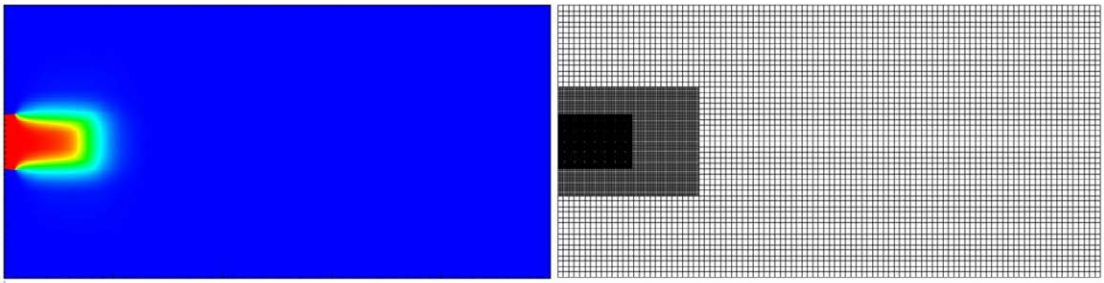



Figure 5.10 Nitrogen mass fraction (a) and grids (b) of the simulation with ALMR at T=0.0072s

In Figure 5.11, mass fractions of nitrogen at y=0.5mm of the ALMR work, uniform calculation with resolution 0.25mm and analytical solution are shown. Result of the simulation with ALMR is still quite close to the analytical solution. In contrast, the result of simulation with resolution 0.25mm has shown bad performance near the source of nitrogen. So, the second refinement level with the resolution 0.125mm has shown its superiority.

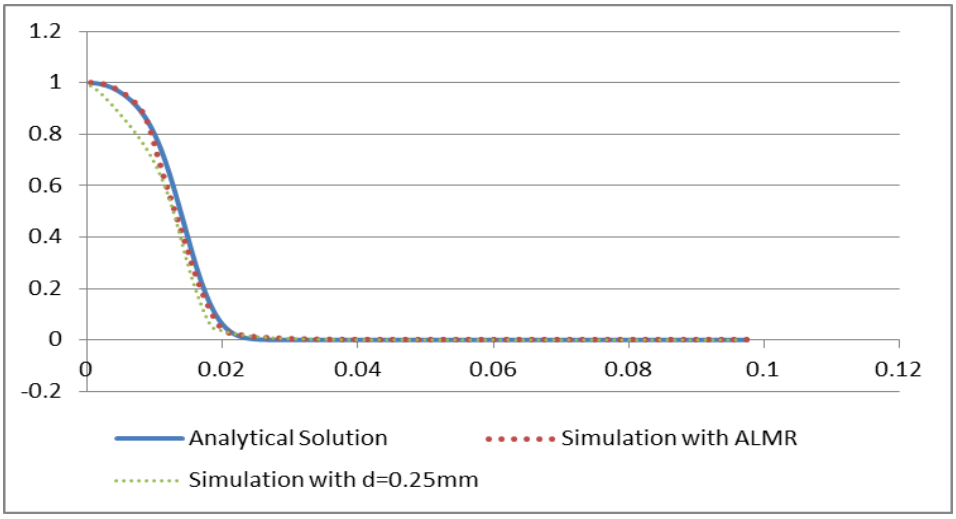



Figure 5.11 Comparison of central line nitrogen mass fractions at T=0.0072s

The distribution of nitrogen gas and the grid structure at step 30000 are shown in Figure 5.12, this moment corresponds to the physical time T=0.0108s.

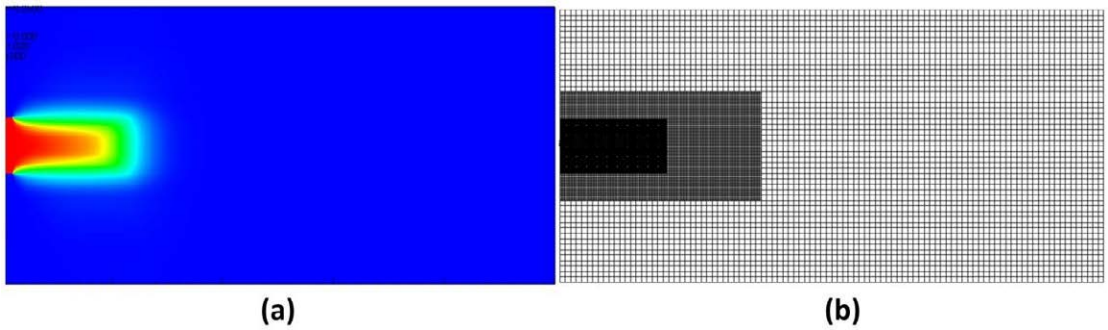



Figure 5.12 Nitrogen mass fraction (a) and grids (b) of the simulation with ALMR at T=0.0108s

The Figure 5.13 shows the mass fraction of nitrogen gas at the position y=0.5mm of ALMR work, uniform grid simulation with resolution 0.25mm and analytical solution.

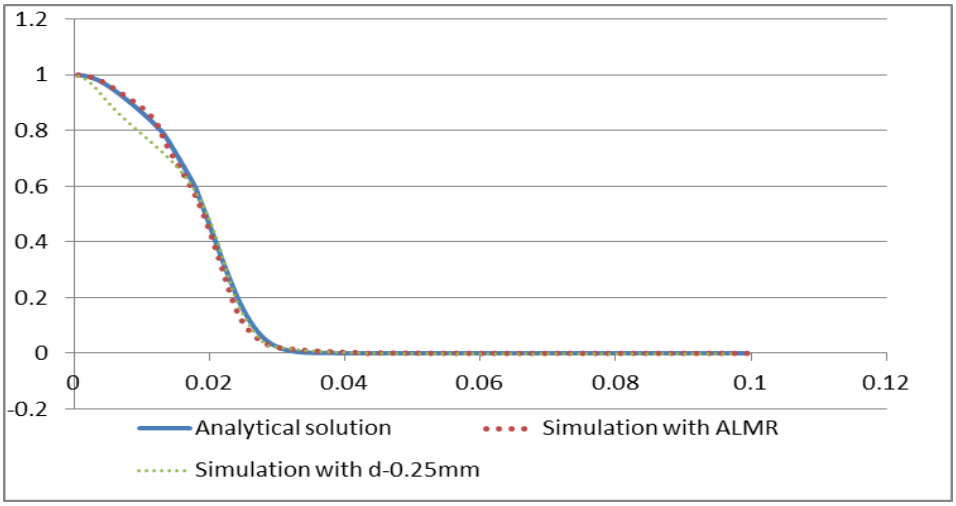



Figure 5.13 Comparison of central line nitrogen mass fractions at T=0.0108s

In the comparison, result of ALMR keeps on making very good performance, but the simulation with resolution 0.25mm fails in the area near the nitrogen source. The local mesh refinement technique successfully controls the big numerical diffusion near the source part and gives a very good simulation result.

In this advection diffusion simulation, besides the accurate expression of physics, the computation has also been saved a lot. Table 5.6 shows the comparison of computational costs between the simulation with ALMR and the simulation with full grids refinement.

Table 5.6 Comparison of total computational costs of 2D diffusion simulations

| Highest Resolution | Simulation with local mesh refinement | Simulation with full grids refinement |
|---|---|---|
| 0.25mm | one finer level, resolution 0.25mm, 220 hours (9 days) for 30000 steps with 2 CPUs | 210 hours (9 days) for 120000 steps with 8 CPUs |
| 0.125mm | First level, resolution 0.25mm<br>Second level, resolution 0.125mm<br>340 hours (14 days) for 30000 steps with 8 CPUs | 80 days for 240000 steps with 8 CPUs (estimation) |

Besides the cost of ALMR with 2 finer levels, time cost with one finer level is shown in the table as well. In the simulation, the finer region with the resolution 0.25mm covers the areas where mass fraction of nitrogen gas is bigger than 0.05 and the simulation result is quite the same as the uniform grid calculation. As shown in the table, under the same highest resolution, simulations with ALMR only take about 20% total computational efforts of the uniform ones.

In longer time calculation, the simulation with adaptive local mesh refinement can still show very good performance. Figure 5.14 shows the simulation result at step 37000 which corresponds to T=0.0130s. Part (a) is the distribution of nitrogen gas, part (b) is the nitrogen distribution at position y=0.5mm of the simulation with ALMR and analytical solution. As shown in the comparison in part (b), result of the simulation with ALMR is very close to the analytical solution. Comparisons in Y direction between the analytical solution and simulation with ALMR are shown in (c) and (d). In part (c), as the finest resolution at position X=9.5mm is 0.125mm, two curves fit quite well to each other. In part (d), since the highest resolution here is at position X=19.5mm is 0.25mm, simulation does not fit the analytical result perfectly.

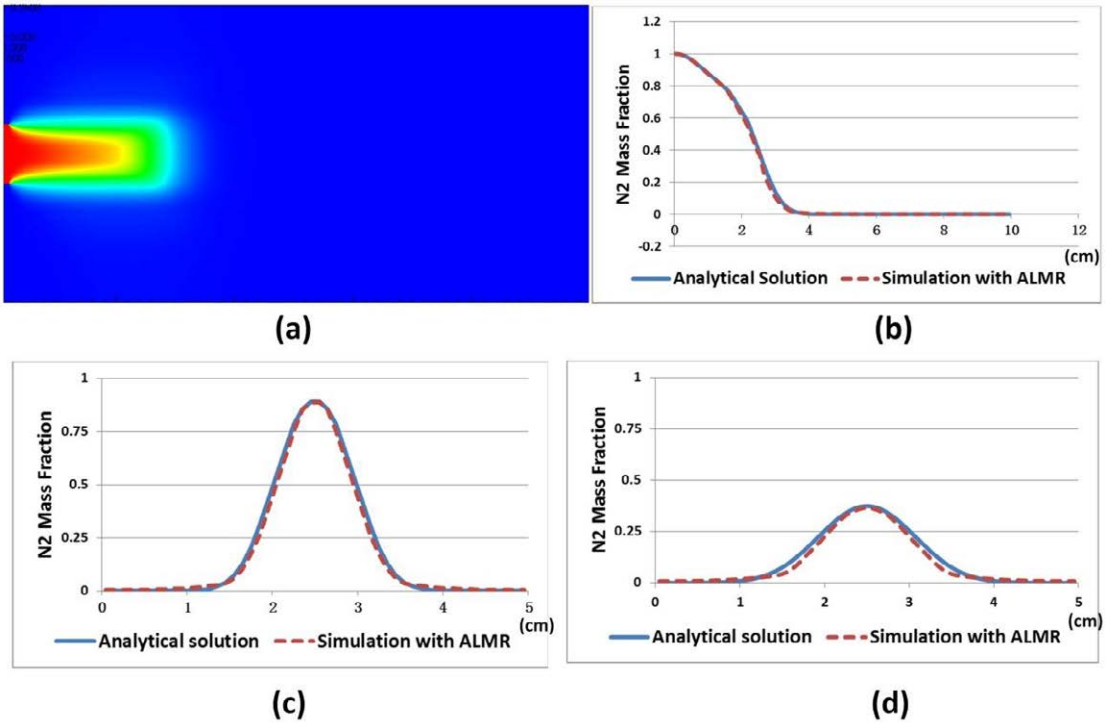



Figure 5.14 Results of simulation with ALMR at T=0.013s

The simulation is stopped at step 50000, which corresponds to physical time T=0.0180s.

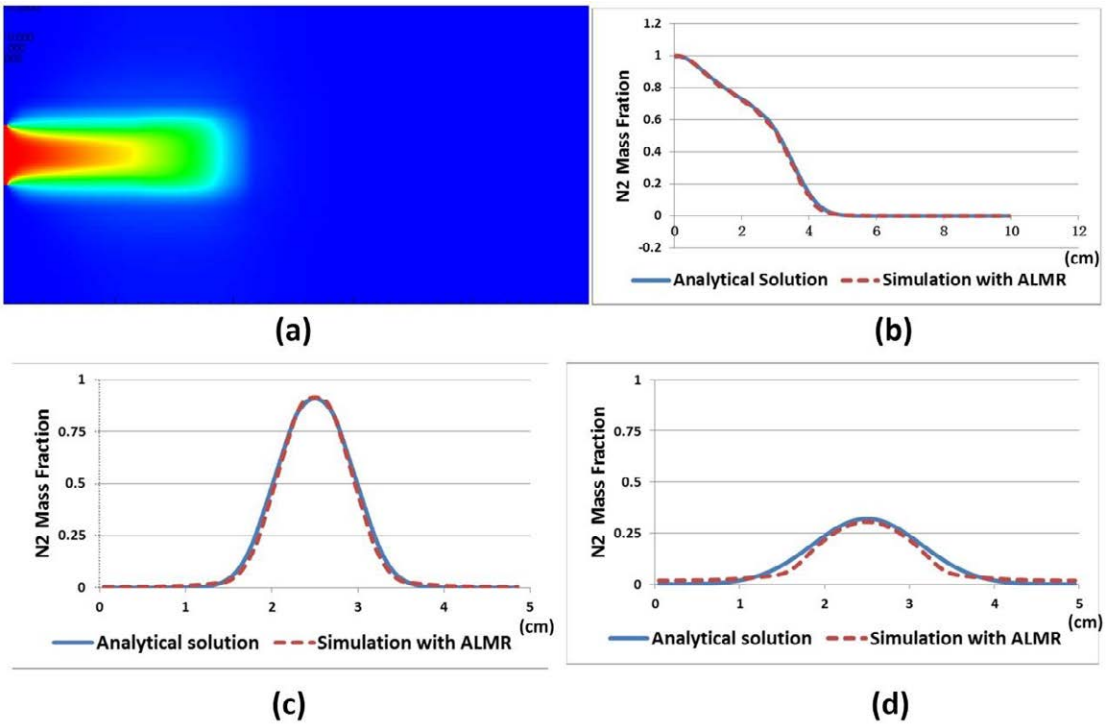



Figure 5.15 Results of simulation with ALMR at T=0.018s

In the part (a) of Figure 5.15, propagation of nitrogen has reached half length of the computational domain. The part (b) shows comparison of nitrogen distribution at the position Y=0.5mm between

simulation with ALMR and analytical solution. In the parts (c) and (d), results in Y direction have been shown, the highest resolution in position X=9.5mm (c) is 0.125mm and the highest resolution in position X=35.5mm (d) is 0.25mm. In all the comparisons, results of simulation with ALMR are quite close to the analytical solution.

Under the maintenance of the physical properties, the local mesh refinement technique can balance the resolution and computation saving successfully in this 2D advection diffusion simulation. In addition, the analytical solution can be used as the verification for the implementation of LMR in the solution of NS equations. The simulation in the section also shows us the importance of the LMR for the V&V. V&V is a very important part of the code development and requires huge amount of calculations to make support, so the high efficiency brought by the new technique can greatly improve the diversity and repeatability of the detailed verification and validation.

### 5.2. Local mesh refinement for jet simulation

In some big practical simulations, cell sizes should be big enough for preventing too much computation cost, but some important boundary conditions might be lost in this situation. With LMR, detailed boundary can be established in the coarse grids domain. In the industrial applications, such usage of local mesh refinement may be very important to the simulation of jet.

Take the simulation in EPR containment as an example, the cell size is usually around 30-50cm to control the totally cells number not exceeding 3 million. However, to the simulation of Loss Of Coolant Accident (LOCA) in which the leakage diameter might vary form 0.01m to 1m, the cell size 30-50cm sometimes is too coarse to make accurate simulation.

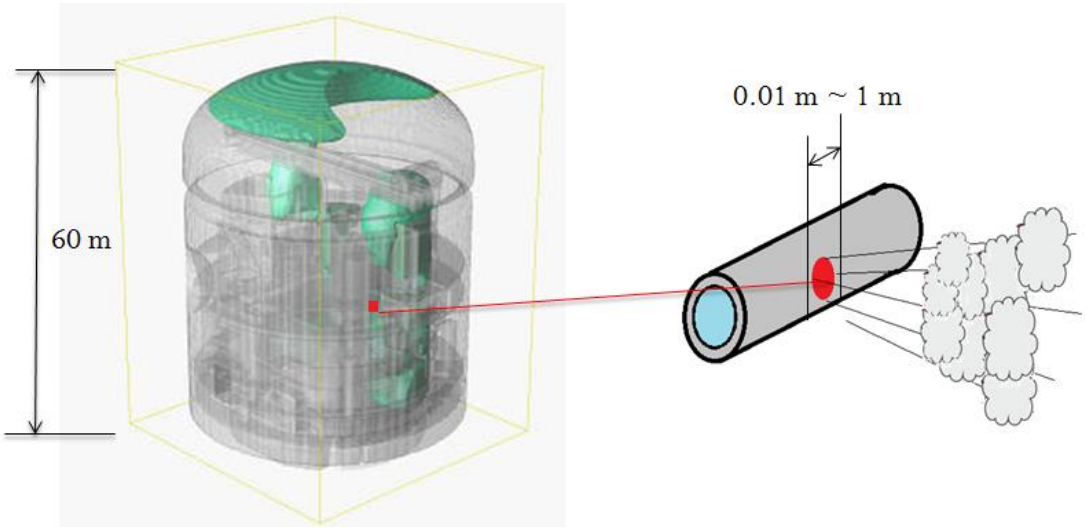



Figure 5.16 Loss Of Coolant Accident (LOCA) inside EPR containment

In some software with the uniform structured grid, the technique of virtual nozzle [9] [92] is used to construct equivalent boundary in computational domain. In such method, the supersonic underexpanded jet is transformed to the subsonic expanded jet. The detailed formulas and the theory of the virtual nozzle are shown as following.

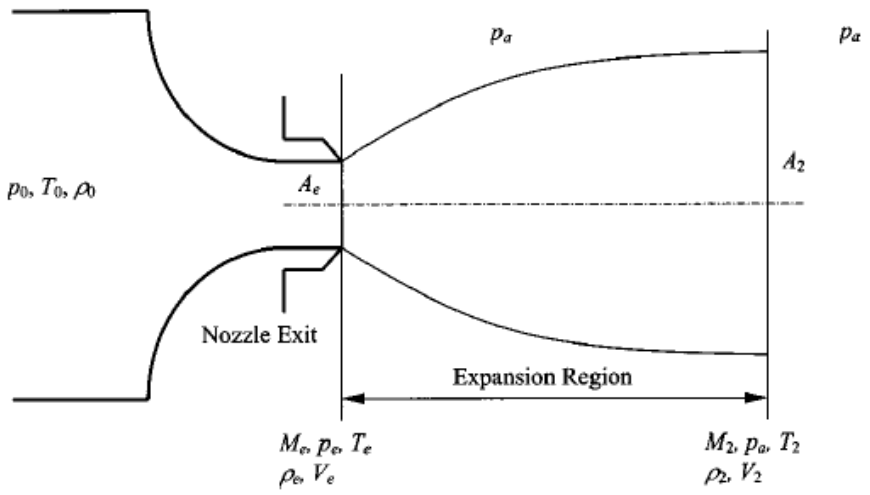



Figure 5.17 Schematic depicting the initial expansion process of an underexpanded jet

In the virtual nozzle, conservation of mass (5-3) is in very high priority:

$$\rho_e A_e V_e = \rho_2 A_2 V_2 \tag{5-3}$$

Under the assumption that pressure at the virtual nozzle is approximately the same as the atmosphere pressure, velocity of the virtual nozzle (5-5) can be established based on the conservation of momentum (5-4).

$$p_e A_e + \rho_e A_e V_e^2 = p_2 A_e + \rho_2 A_2 V_2^2 \tag{5-4}$$

$$\frac{V_2}{V_e} = 1 + \frac{p_e - p_a}{\rho_e V_e^2} \tag{5-5}$$

The other parameters of the virtual nozzle such as temperature, density and area (shown in (5-6)) can be gotten by the similar method. Some new symbols are introduced in (5-6), $u = p_e / p_a$ is the underexpansion ratio (also the NPR in latter discussion), $M_e$ is the March number of the actual nozzle and $\gamma$ is the heat ratio of gas.

$$\frac{T_2}{T_e} = 1 + \frac{\gamma - 1}{2} M_e^2 [1 - (\frac{V_2}{V_e})^2]$$

$$\frac{\rho_2}{\rho_e} = \frac{2\gamma^2 M_e^2 u}{2\gamma M_e^2 u(\gamma + u - 1) - (\gamma - 1)(u - 1)^2}$$

$$\frac{A_2}{A_e} = \frac{2\gamma M_e^2 u(\gamma + u - 1) - (\gamma - 1)(u - 1)^2}{2\gamma^2 M_e^2 u + 2\gamma(u - 1)}$$

(5-6)

With above formulas, all the parameters of equivalent outlet can be gotten. However, since the virtual nozzle does not consider the real physics carefully and just make the transition between the two different jets through simple conservation law, values gotten from above formulas may be far

from the simulation results. In addition, the technique of virtual nozzle cannot fully help the simulation getting rid of the limitation of grid size, it may also fail when quite small diameter nozzle is simulated under very coarse resolutions.

### 5.2.1. Comparison between real jet and virtual nozzle

In order to show the shortage of virtual nozzle, some jet simulations are made. The computational domain, grid information and initial condition of the simulations are shown in Figure 5.18 and Table 5.7 (NPR values in Table 5.7 indicate the ratio of pressures between jet orifice and atmosphere). In this section, local mesh refinement is done for approaching the circle structured orifice and more detailed expression of shock wave, LMR is given statically near the orifice.

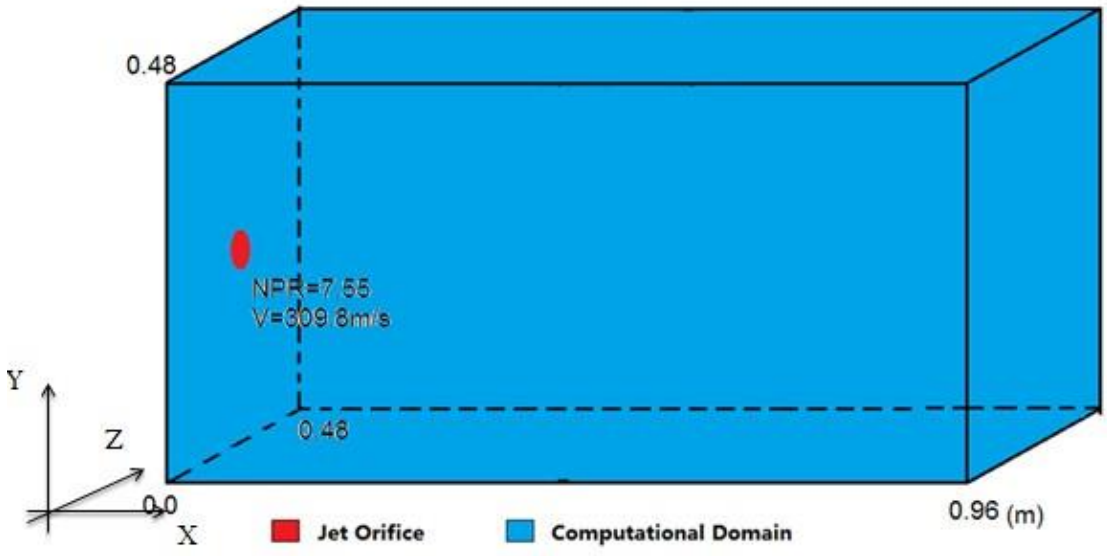



Figure 5.18 Computational domain of the jet simulation

Table 5.7 Grid information and initial condition of the jet simulation

| Grids Information | | |
|---|---|---|
| NPR=7.55 | Base Level: (0, 0, 0)X(59, 29, 29) Resolution: 0.016m<br>First Level covered range: (0, 5, 5)X(29, 24, 24) Resolution: 0.004m<br>Second Level covered range: (0, 7, 7)X(19, 21, 21) Resolution: 0.002m | |
| NPR=15.53 | Base Level: (0, 0, 0)X(59, 49, 49) Resolution: 0.016m<br>First Level covered range: (0, 10, 10)X(29, 39, 39) Resolution: 0.004m | |
| Initial conditions | | |
| NPR | Jet Orifice | Atmosphere |
| 7.55 | $D_e$=0.064 m<br>$p_e$=754431.5 Pa<br>$(V_x, V_y, V_z)_e$=(309.8, 0, 0) m/s<br>$T_e$=234 K<br>$M_e$=1.01<br>$H_2 : O_2 : N_2 : H_2O = 0 : 1 : 3.76 : 0$ | $p_a$=101300 Pa<br>$(V_x, V_y, V_z)_a$=(17.3, 0, 0) m/s<br>$T_a$=298 K<br>$M_a$=0.05<br>$H_2 : O_2 : N_2 : H_2O = 0 : 1 : 3.76 : 0$ |
| 15.53 | $D_e$=0.064 m<br>$p_e$=1574020 Pa | $p_a$=101300 Pa<br>$(V_x, V_y, V_z)_a$=(17.3, 0, 0) m/s |

| | | |
|---|---|---|
| | $(V_x, V_y, V_z)_e$=(310.6, 0, 0) m/s<br>$T_e$=235 K<br>$M_e$=1.01<br>$H_2 : O_2 : N_2 : H_2O = 0 : 1 : 3.76 : 0$ | $T_a$=298 K<br>$M_a$=0.05<br>$H_2 : O_2 : N_2 : H_2O = 0 : 1 : 3.76 : 0$ |

As shown in above table, two supersonic jets with different NPR values should be simulated. Normally, simulation with bigger NPR value has a larger expansion ratio, so computational domains and local mesh refinement are given differently due to the different expansion sizes of jets. In the case NPR=7.55, size of the jet, such as the shock wave part, is small, so higher resolution is required to get details. In contrast, the size of jet in the case NPR=15.53 is much bigger and one finer level with the resolution 4mm is enough to get detailed information of shock wave.

Before the comparison with the virtual nozzle, results of the simulation are compared to publish numerical results [54] to show the simulation results are valid. The testing for the case NPR=7.55 is given below.

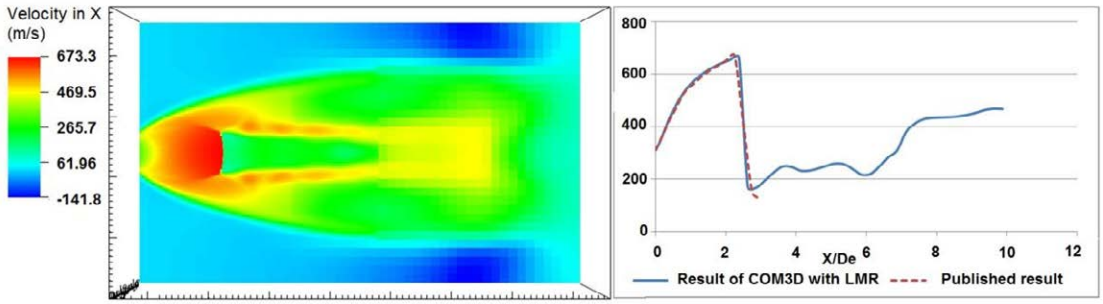



Figure 5.19 Result of velocity in X direction in case NPR=7.55

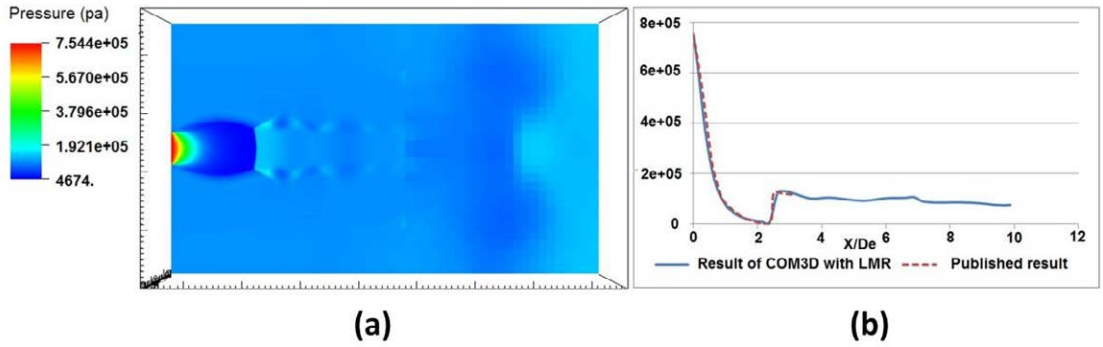
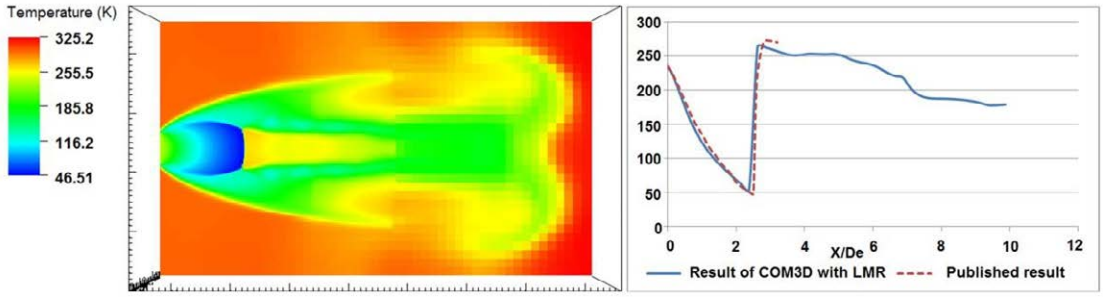



Figure 5.20 Result of pressure in case NPR=7.55




(a) (b)

Figure 5.21 Result of temperature in case NPR=7.55

The above figures show the simulation of jet under the condition NPR=7.55. In part (a) of above figures, distributions of velocity in X direction, pressure and temperature at the slice Z=0.24m are given, and the central values of COM3D's simulation are compared with the results in [54] in parts (b). Since the Euler equations are used to simulate the shock wave, comparisons are limited to the shock part only. In (b) part of above figures, $X / D_e$ is used as the abscissa axis, therefore two simulations which use different diameter nozzles can be compared with each other. In the comparisons of central values, simulation in COM3D with local mesh refinement (shown by the solid curves) shows almost the same results as the simulation in [54]. Diameter of shock wave part is about 3 times as the orifice in COM3D's simulation, which is the same as the published simulation. Therefore, comparison between the simulation of jet in COM3D and the virtual nozzle can be done in the following. The virtual nozzle corresponds to the position where the average pressure is close to the atmosphere, so the average values at the shock wave is compared with virtual nozzle in Table 5.8.

Table 5.8 Comparison between real jet simulation and virtual nozzle (NPR=7.55)

| Comparison between real jet simulation and virtual nozzle | |
|---|---|
| Result in COM3D | Virtual nozzle |
| Average velocity in X direction: 191.433 m/s<br>Average temperature: 250.953 K<br>Diameter of the jet at this position: 3 $D_e$<br>Averaged density: 2.01 $kg / m^3$ | Velocity in X direction: 498 m/s<br>Temperature: 164 K<br>Diameter of the virtual nozzle: 1.8 $D_e$<br>Density: 2.15 $kg / m^3$ |

In the comparison, except the density, other important parameters are far different from each other. Indeed, virtual nozzle does not consider the shock wave generated in front of the orifice, so the virtual nozzle can hardly keep all the physical properties of jet and the detailed parameter of the virtual nozzle may be far from the real simulation.

Beside the jet simulation with NPR=7.55, another simulation which has NPR=15.53 is used to make the comparison. Similarly to the case NPR=7.55, simulation of jet with local mesh refinement is compared with the published simulation [54]. The results at the slice Z=0.4m and the comparison of central quantities are shown in following figures.

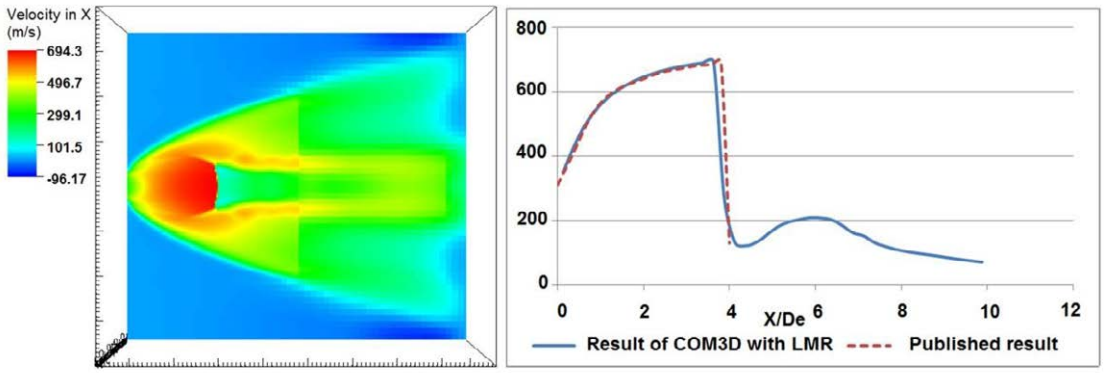



Figure 5.22 Result of velocity in X direction in case NPR=15.53

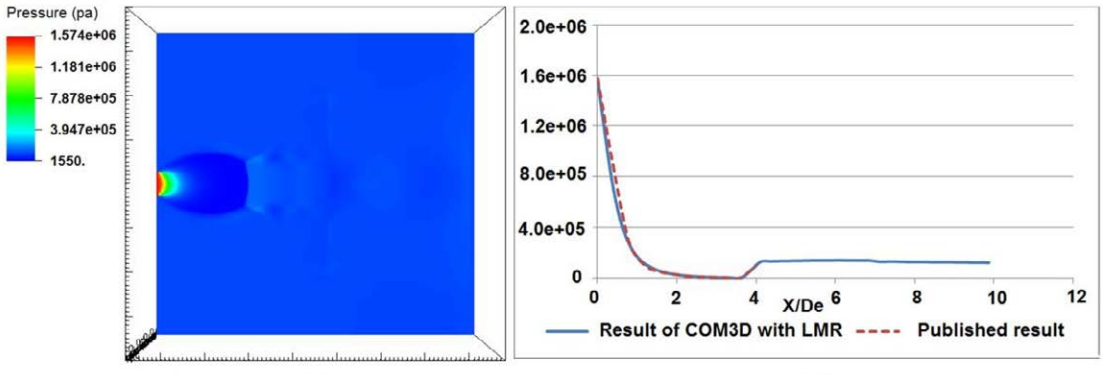



Figure 5.23 Result of pressure in case NPR=15.53

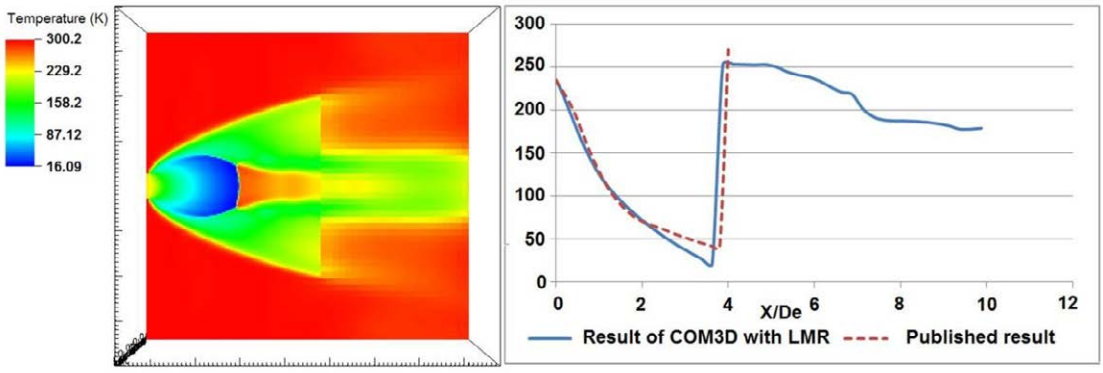



Figure 5.24 Result of temperature in case NPR=15.53

As shown in the comparisons, central values of COM3D's simulation show very good compatibility to [54]. Similarly to the result in [54], diameter of the jet at the position where virtual nozzle should be located is about 5 $D_e$ . Comparison between the simulation with LMR and virtual nozzle is done in Table 5.9.

Table 5.9 Comparison between real jet simulation and virtual nozzle (NPR=15.53)

| Comparison between real jet simulation and virtual nozzle | |
|---|---|
| Result in COM3D | Virtual nozzle |
| Average velocity in X direction: 148.48 m/s<br>Average temperature: 273.43 K<br>Diameter of the jet at this position: 5 $D_e$<br>Averaged density: 1.94 $kg/m^3$ | Velocity in X direction: 514 m/s<br>Temperature: 157 K<br>Diameter of the virtual nozzle: 2.6 $D_e$<br>Density: 2.05 $kg/m^3$ |

The parameters in virtual nozzle are quite different from the real simulation, some differences become even larger than the NPR=7.55 case. So, in the case NPR=15.53, the virtual nozzle is unable to give convincing results either.

After the comparisons in the cases NPR=7.55 and NPR=15.53, it is clear that the virtual nozzle is not an efficient solution to the problem of tiny nozzle. The parameters in the virtual nozzle are far from the real simulation, so the results in the region near the nozzle may be different from the real situation.

5.2.2. LMR for the simulation of small jet

In this sub-section, the technique of LMR is used to construct the small diameter nozzle for jet simulation. Detailed grid information, boundary conditions and initial conditions of this simulation are given in Table 5.10.

Table 5.10 Grid information and initial condition of small jet problem

| Grid information | |
|---|---|
| Base Level: (0, 0, 0)X(59, 29, 29) | Resolution: 0.064m |
| First Level covered range: (0, 11, 11)X(7, 18, 18) | Resolution: 0.016m |
| Second Level covered range: (0, 12, 12)X(3, 17, 17) | Resolution: 0.008m |
| **Initial conditions** | |
| Jet orifice | Atmosphere |
| $D_e$=0.032 m<br>$p_e$=754431.5 Pa<br>$(V_x, V_y, V_z)_e$=(309.8, 0, 0) m/s<br>$T_e$=234 K<br>$M_e$=1.01<br>$H_2 : O_2 : N_2 : H_2O = 0 : 1 : 3.76 : 0$ | $p_a$=101300 Pa<br>$(V_x, V_y, V_z)_a$=(17.3, 0, 0) m/s<br>$T_a$=298 K<br>$M_a$=0.05<br>$H_2 : O_2 : N_2 : H_2O = 0 : 1 : 3.76 : 0$ |

As shown in Table 5.10, diameter of the jet orifice is half as the cell size on base level. It is impossible to simulate jet on the base level directly, and local mesh refinement or virtual nozzle is required. The resolution 0.064m is too coarse to construct the equivalent virtual nozzle, because it has already been shown that the virtual nozzle has the diameter about 1.8 times as the real nozzle in the case NPR=7.55. Since the details of jet are not important here, the distance of four cells is used as diameter of the nozzle on the finest level (if necessary, more refinement levels can be used to simulate the details of jet).

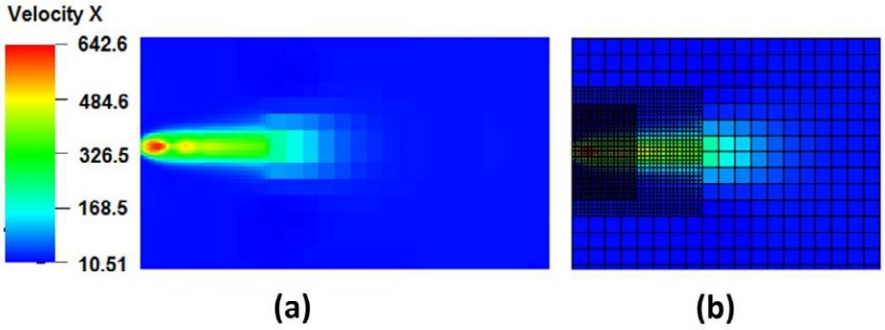



Figure 5.25 Simulation of small jet at T=0.0039s

Figure 5.25 shows the result at T=0.0039s. Distribution of velocity in X direction at the slice Z=0.96m is shown in part (a), since the relative resolution is not high enough, results cannot be as good as the former jet simulation. In part (b), detailed grid structured near the jet orifice is given.

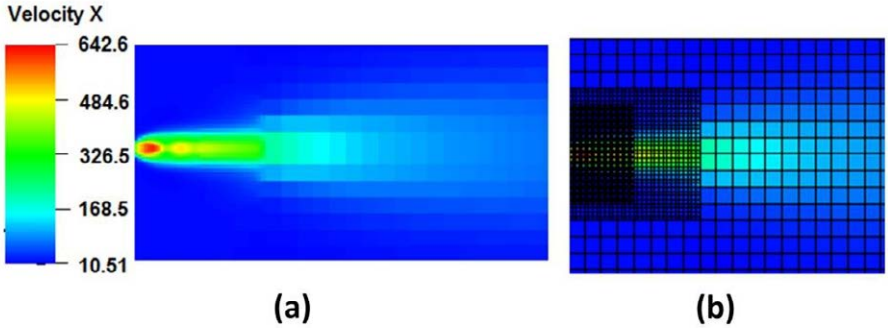



Figure 5.26 Simulation of small jet at T=0.0319s

Figure 5.26 is the result at T=0.0319s. In longer time calculation, simulation with local mesh refinement still shows good stability. Both figures show the new technique managed in constructing the small nozzle for jet simulation. In reproduction of physics, simulation with LMR shows very high compatibility to the former detailed jet simulations.

Such LMR simulation case shows the function of local mesh refinement in the construction of tiny nozzles. Differently from the virtual nozzle which still has strict requests in cell sizes, in principle, the technique of local mesh refinement can construct the real size nozzle independently without considering the cell size on the base level. The more important is the parameters of the jet with LMR are more reasonable than parameters achieved by virtual nozzle.

### 5.3. Detonation simulation with LMR

In experiments, visible cellular structure on sooted foil (shown by Figure 5.27) is one important feature of detonation [15] [36] [68]. In numerical simulations, such special feature should be performed as well. In model, chemical reaction part in COM3D is given by one step Arrhenius (5-7).

$$\frac{\partial Y}{\partial t} = -K_{ch} Y \exp(-\frac{E_{ch}}{T}) \tag{5-7}$$

$Y$ indicates the density of gas components, $T$ stands for the temperature, parameter $K_{ch}$ is the pre-exponential factor which relates to the intensity of the chemical reaction and $E_{ch}$ is the activation energy. Proved by some numerical researches about the one step Arrhenius approached detonation [36] [68], the appearance of cellular structures strongly depends on the resolution used in the computational domain and the pre-exponential factor [15]. In order to reproduce the cell size ~1cm observed in experiment, the pre-exponential factor $K_{ch} = 10^7$ is used in the code COM3D.

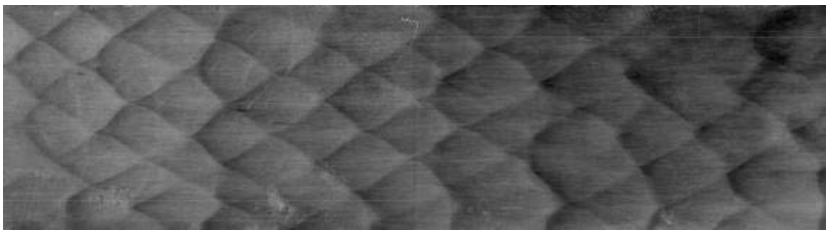


Figure 5.27 cellular structure of detonation wave

Therefore, very high resolutions are required in detonation simulation to reproduce the real physics. In simulations with uniform structured grid, especially for the large-scale practical problems, high resolution required by the simulation of detonation cells can result in very huge or unacceptable computational efforts.

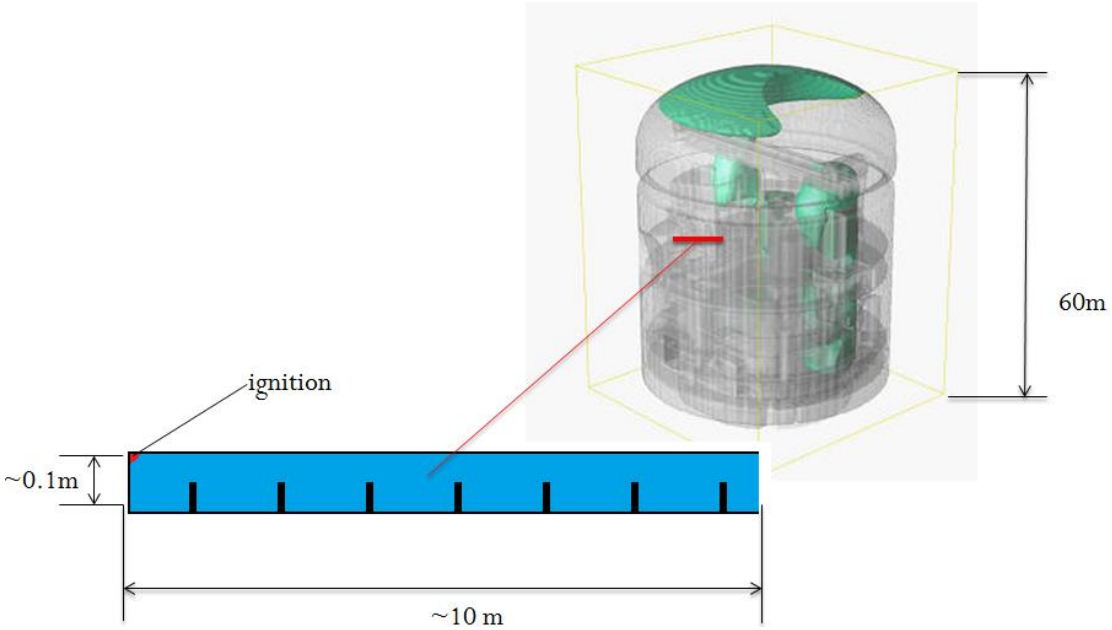



Figure 5.28 Detonation simulation inside EPR containment

Such as the detonation simulation inside EPR containment shown in Figure 5.28, simulation of detonation cell is unbelievable under uniform grid. Even for the 0.1m×10m sub-domain shown in the Figure 5.28, simulation of cellular structure with uniform grid also causes huge amount of computational efforts. However, with the technique of LMR in this thesis, computational efforts for simulation of detonation cellular structure can be reduce to an acceptable level.

In the following parts, 3 detonation simulations with different scales and different dimension numbers are proposed. In the first small-scale detonation simulation, both uniform grid and LMR are used in simulation to prove the simulation with LMR can reproduce the same physical property of detonation as the uniform grid calculation. Then, a 2D large-scale detonation interacting with complex geometry is performed with LMR to show the benefit of the technique in computational saving. Finally, a real 3D detonation simulation is accomplished with the help of the new technique, which shows the revolutionary progress in the code COM3D by the implementation of LMR.

### 5.3.1. Simple 2D detonation simulation

Firstly, a 2D detonation simulation based on the grid with resolution 0.001m is given. The computational domain and the initial condition are show by Figure 5.29 and Table 5.11.

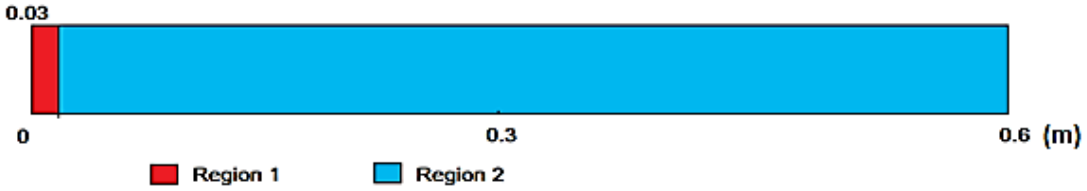



Figure 5.29 Computational domain of 2D detonation simulation

Table 5.11 Initial condition and grid information of the 2D detonation

| **Grid Information** | |
|---|---|
| **Computational Domain:** $(0,0,0)\times(599,29,1)$<br>**Cell Size: 0.001m** | |
| **Initial Condition** | |
| **Region 1** | **Region 2** |
| **Domain:** $(0,0,0)\times(19,29,1)$<br>**Pressure:100Bar**<br>**Temperature:3000K**<br>**Velocity: 0, 0, 0**<br>**Gas Spices (mole ratio):**<br>$H_2:O_2:N_2:H_2O=2:1:3.76:0$ | **Domain:** $(20,0,0)\times(599,29,1)$<br>**Pressure:1Bar**<br>**Temperature:298K**<br>**Velocity: 0, 0, 0**<br>**Gas Spices (mole ratio):**<br>$H_2:O_2:N_2:H_2O=2:1:3.76:0$ |

In the simulation in COM3D, in order to simulate the cellular structures successfully and avoid possible unphysical oscillation, the solver van Leer with CFL number 0.2 is used in the solution of the gas part.

Shown by the numerical smoke-foil record in Figure 5.30, the shock front keep flat till the end of the simulation and no cellular structures can be found in the record. By considering the volume fraction of hydrogen gas and the results in experiments, the computational domain with the width 0.03m should contain about 3 detonation cells in numerical smoke-foil record. It is clear that the 0.001m cell size is not fine enough to express such intensive chemistry, no triple point [84] can be generated in the simulation and no cellular structure can be found.


Figure 5.30 Numerical smoke-foil record for 2D detonation with resolution 1 mm

For the simulation of cellular structures, finer resolution 0.00025m is used uniformly, the computational domain is (0, 0, 0)×(2399, 119, 1). Comparing to the former coarser grid case, the number of cells has been increased by 16 times. The total computational efforts are increased by 64 times, for the 4 times finer time step is also required for stability. Figure 5.31 is the numerical smoke-foil record for 2D detonation under finer resolution after 12000 steps' calculation, corresponding to the smoke-foil record for 2D detonation under coarse resolution after 3000 steps' calculation. As shown in the figure, some cellular structure can be observed at the 2/3 length of the whole computational domain.


Figure 5.31 Numerical smoke-foil record of the finer case

Proved by above smoke-foil record, detonation cells can be simulated successfully under higher resolution. However, the full grid refinement actually is not necessary in simulation of detonation cells. According to the properties of detonation wave, high resolution is necessary only in the chemical reaction zone which is just behind the detonation shock wave front is the key to simulate the triple points and cellular structures [84]. In the other regions, such as the burnt or unburned ones, high resolution is not needed. So, local mesh refinement can be used to satisfy the requirements of fine resolutions around the chemical reaction zone and reduce the computational efforts. As the propagation of the shock wave in the detonation simulation, the technique of local mesh refinement should be used adaptively. Referencing to the works of some predecessors about ASMR (adaptive structured mesh refinement) [24] [35] [66], detailed process of re-gridding in adaptive local mesh refinement (ALMR) is shown by the Figure 5.32.

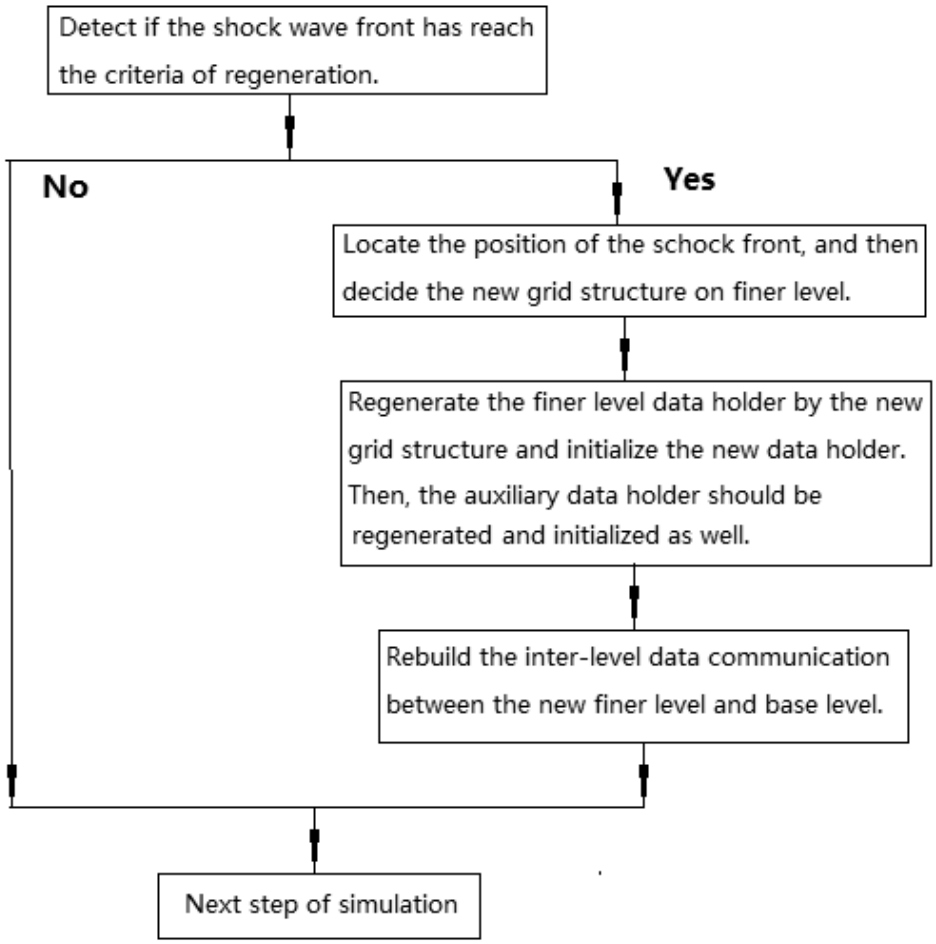



Figure 5.32 Process of re-gridding in detonation simulation

The detailed process of re-gridding in detonation simulation with adaptive local mesh refinement is as below.

1) Detect if the shock front (through the gradients of pressure or temperature) has reached the criteria of the finer level regeneration.
2) If the regeneration criteria are satisfied, position of the detonation wave is located, and the new refinement region should be made based on this location.

3) In order to keep the current chemical reaction zone in the finer grids and save the total computational efforts on finer level, lower bound of the refinement region is made several cells (normally 5-10 cells) behind the shock front and the upper bound is made about 10-20 coarse cells forwards the front (too many finer grids may increase the total computational efforts largely).
4) After the reconstruction of finer grids, redefinition of data holder is made based on the new finer grids. The new data holder is initialized by coarse data through the coarse-to-fine constant interpolation and the finer data transferred from the auxiliary data holder. Latterly, the auxiliary data holder is also redefined by the new finer grid.
5) After the initialization on fine level, inter-level data communications between the new finer data and coarse data are re-built.


Figure 5.33 Numerical smoke-foil record for 2D detonation with ALMR

Figure 5.33 shows the smoke-foil record for the calculation with adaptive local mesh refinement. Similarly to the finer uniform grid calculation case, detonation cellular structures do not appear at the very beginning and appear at about the 2/3 of the whole domain length. The number of detonation cells is 3.5 and the performance of these cells has been kept stable till the shock reaches the right end of the computational domain.

Besides the simulating of detonation cellular structure successfully, simulating the detonation cellular structures with correct size is also very important [15]. If the simulation with adaptive local mesh refinement can have the same cellular size as the uniform grid simulation, implementation of local mesh refinement in simulation of detonation wave can be verified. Figure 5.34 shows the maximum pressure records in horizontal direction of the simulation with uniform resolution 0.25mm (a) and the simulation with LMR (b), the two results show the peak pressure values in the distance of 7.5cm.

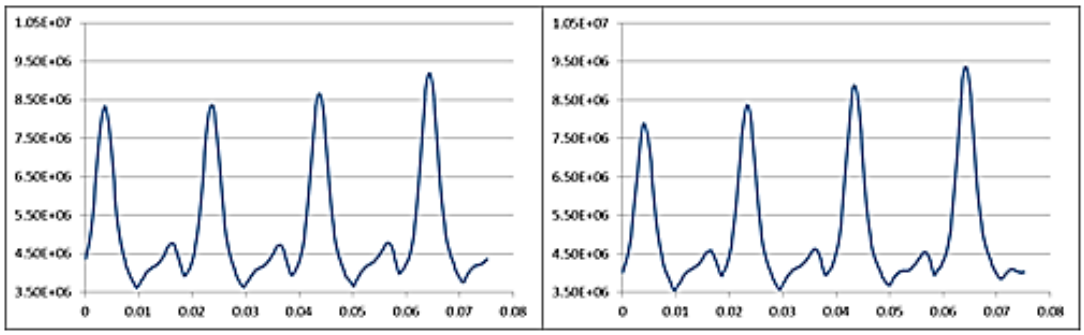



Figure 5.34 Smoke-foil records in the X direction of two simulations

Both fully refined grid calculation (a) and locally refined grid calculation (b) show 4 peaks in the length of 7.5cm, so the lengths of detonation cells in two simulations are the same. Then, as both smoked-foil records above have 3.5 detonation cells in Y direction, the two simulations produce the same cell width. Therefore, implementation of LMR in simulation of detonation cellular structures is verified.

Table 5.12 Comparison of the total computational effort

| | Computational Grid | Total Computational Cost |
|---|---|---|
| Uniform Grid | (0, 0, 0)X(2399, 119, 1) | 3 CPUs, 40hour |
| Adaptive LMR | Coarse Grid: (0, 0, 0)X(599, 29, 1)<br>Finer grid covered ratio: 5.5% | 2 CPUs, 18hour |

Table 5.12 shows the grid structures of the two simulations and their computational cost. The simulation with adaptive local mesh refinement costs about 30% of the full grid refinement case. Comparing to the 5% fine level covered ratio, the time saving of local mesh refinement is not efficient. Indeed, 2D simulations in COM3D are accomplished by the 3D computation, and the computational domain always contains two cells in Z direction. The local mesh refinement in such domain has to be 3D, and the finer level contains 8 cells in Z direction. As a result, the computational efforts on the finer region in the simulation with adaptive local mesh refinement are about 22% of the uniform grid case. In addition, the cost of calculation on the coarse level and inter-level data communication should be included in the total computational cost. Therefore, the ALMR approach saves about 70% work of the full grid refinement case only.

### 5.3.2. Decay and re-initiation of detonation simulation with ALMR

As shown in the simple 2D example, detonation requires very high resolution in the chemical reaction zone to simulate the detonation cells. The adaptive local mesh refinement meets the requirement of resolution in chemical reaction zone and cost much less than the full grid refinement technique. The 2D example above can verify the LMR in simulation of detonation cells. In order to show the advantage of local mesh refinement in time saving, more complicated and larger scale of 2D detonation simulation should be done with LMR. The simulation of decay and re-initiation of the detonation wave interacting with obstacles is given. The computational domain and initial condition of this calculation are shown in the Figure 5.35 and Table 5.13.

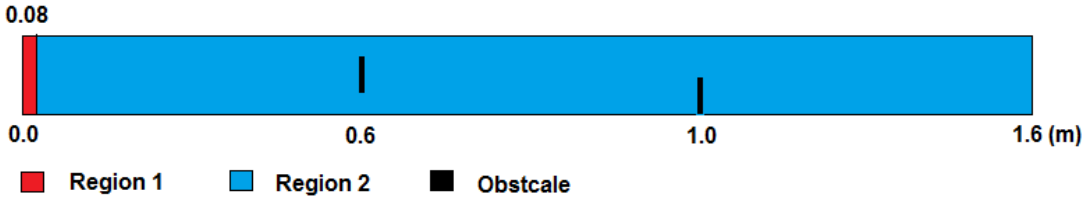



Figure 5.35 Domain of the simulation of decay and re-initiation

Table 5.13 Initial condition and grid information of decay and re-initiation

| **Grid Information** | |
|---|---|
| **Computational Domain:** $(0,0,0)\times(399,19,1)$<br>**Cell Size: 0.004m** | |
| **Initial Condition** | |
| **Region 1** | **Region 2** |
| **Domain:** $(0,0,0)\times(4,19,1)$<br>**Pressure:100Bar**<br>**Temperature:3000K**<br>**Velocity: 0, 0, 0**<br>**Gas Spices (mole ratio):**<br>$H_2:O_2:N_2:H_2O=2:1:3.76:0$ | **Domain:** $(5,0,0)\times(399,19,1)$<br>**Pressure:1Bar**<br>**Temperature:298K**<br>**Velocity: 0, 0, 0**<br>**Gas Spices (mole ratio):**<br>$H_2:O_2:N_2:H_2O=2:1:3.76:0$ |

In simulation of decay and re-initiation, the parameters in Region 1 are given the same as the former 2D case to generate detonation wave, and the 0.6m long region before the first obstacle is provided for the generation of detonation cellular structures. In the computational domain, two obstacles are installed, one of them is located at the position 0.6m (central position) and the other is located at 1.0m (bottom position). Installations of the two obstacles are made differently in the computational domain to get different performances of decay and re-initiation of detonation wave.

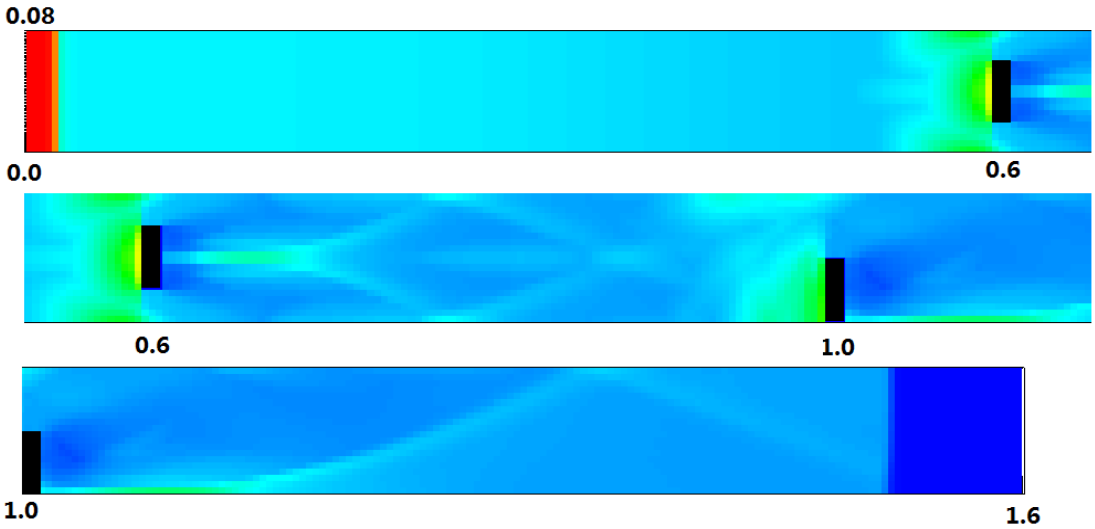



Figure 5.36 Smoke-foil record for resolution 0.004m

Firstly, the simulation without any refinement is done. Since the resolution 0.004m is coarser than the former 2D coarse case, no cellular structures can be found before the first obstacle. Between the first and second obstacles, one big cellular trace has been generated. However, this cellular trace is very obscure and can hardly be maintained along the propagation of detonation wave, which is different from the detonation cellular structure. After the second obstacle, a half cellular trace has been formed. Similarly to the trace between the first and second obstacles, this cellular trace cannot be maintained as the real detonation cellular structure. Therefore, the resolution 0.004m is too coarse to simulate intensive chemical reaction of detonation, and resulted in the failure of reproduction of decay and re-initiation of detonation.

Differently from the former 2D detonation simulation case, since the computational domain is very big, it is unrealistic to simulate the detonation cellular structures by the full grid refinement technique. Normally, the cell size 0.0002m is used in COM3D to make the simulation of detonation, and the number of computational cells will be 400X8000X2=6400000. The time step should be 1/20 as the calculation with 0.004m grid size for stability. The simulation of decay and re-initiation of detonation with full grid refinement may require about 2 months with 8 CPUs. In order to reduce the total computational efforts to an acceptable level, the adaptive local mesh refinement is necessary in the simulation.

In the simulation with ALMR, similarly to section 5.3.1, the shock front should be simulated under higher resolutions. Based on the base level with resolution 0.004m, in order to reach the resolution 0.00025m, two finer refinement levels with the refinement ratio four are used in the adaptive local mesh refinement. In the re-gridding, the regeneration of the grid structure in this case is similar to the former one: on the second level (finest level), the re-gridding criteria is the same as the former ALMR; on the first level, the new grids need to cover the new finest level and leave enough regions to make the coarse-to-fine interpolation. Detailed information of the local mesh refinement in the 2D simulation is shown in Table 5.4.

Table 5.14 Information about adaptive local mesh refinement in the decay and re-initiation

| | |
|---|---|
| Coarse domain | Region: (0, 0, 0)X(399, 19, 1)<br>Cell Size: 0.001m |
| Initial Refined domain | First level: (0, 0, 0)X(44, 79, 3)<br>Refinement Ratio: 4<br>Cell size: 0.001m<br>Second level: (0, 0, 0)X(159, 319, 15)<br>Refinement Ratio: 4<br>Cell size: 0.00025m |
| Refined domain | 5 base cells forward the shock front and 3 base cells behind the shock front |
| Fine grids ratio | 2%-6% |

Influenced by the obstacles in the computational domain, the shock wave does not only propagate along the X direction. Differently from the refined grids in former ALMR, the fine grid ratio may change a lot.

Figure 5.37 shows the numerical smoke-foil record for decay and re-initiation with adaptive local mesh refinement. Similarly to the former 2D case, there are no detonation cellular structures in the numerical smoke-foil record at the very beginning. Latterly, due to the instability of the detonation wave, some cellular structures can be generated. It is worth to note that the cell size before the first obstacle is about 9mm in width, which is quite near to the experiments of detonation under the pre-mixed hydrogen-air mixture.

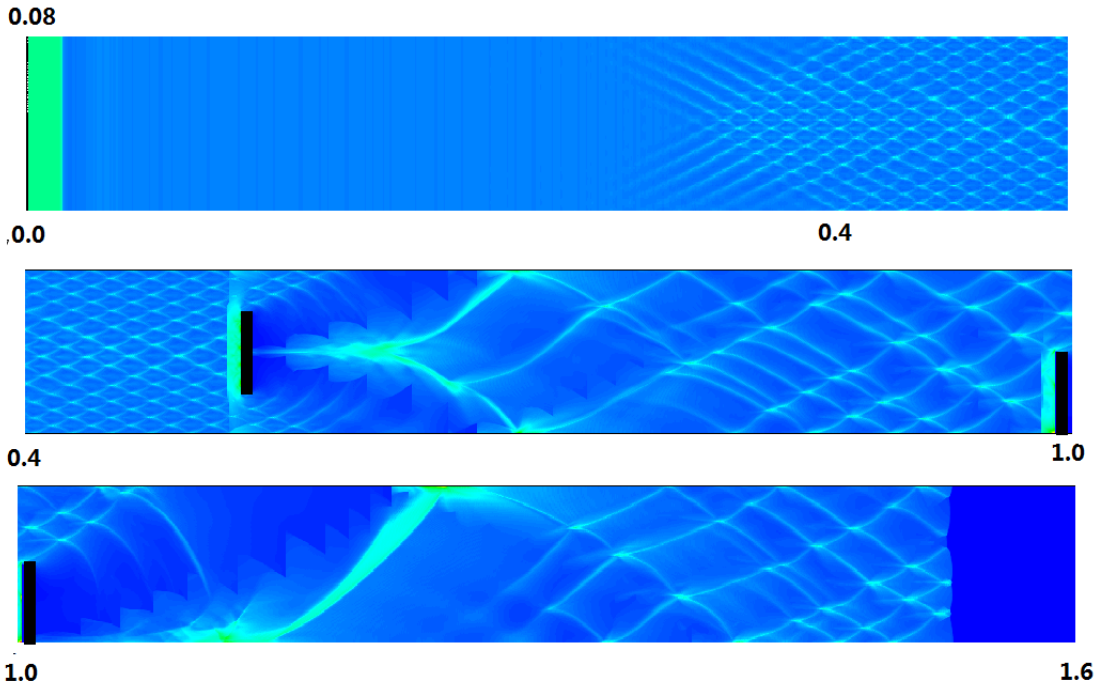



Figure 5.37 Smoke-foil record for simulation with ALMR

When the shock wave meets the first obstacle, regularity of this shock is broken and the cellular structures collapse, thus detonation cells disappear (decay) after the obstacle. However, the overlap of the pressure wave has sufficient strength after the obstacle and the instability of the detonation can be simulated successfully with the fine resolution, the cellular structures are regenerated after the obstacle, which is also called the re-initiation of the detonation cells. In this simulation, the detonation cellular structures have been regenerated successfully after the two obstacles.

As mentioned in former part, the 2 cells in Z direction on base level can make the time saving of LMR inefficient. Worse than the 2D case in section 5.3.1., the performance of ALMR in computation saving may be limited largely by the finest level with 32 cells in Z direction. In order to show the benefit of local mesh refinement in time saving, simulation of the decay and re-initiation with the base resolution 0.001m is made. In this case, one finer level is required and the total computational efforts on the finest level are 1/4 as the one with base resolution 0.004m. The base grid structure in this calculation case is shown in Table 5.15.

Table 5.15 Initial condition and grid information of another simulation

| **Grid Information** |
|---|
| **Computational Domain:** $(0,0,0)\times(1599,79,1)$ |

| **Cell Size: 0.001m** | |
|---|---|
| **Initial Condition** | |
| **Region 1** | **Region 2** |
| **Domain:** $(0,0,0)\times(19,79,1)$<br>**Pressure:100Bar**<br>**Temperature:3000K**<br>**Velocity: 0, 0, 0**<br>**Gas Spices (mole ratio):**<br>$H_2:O_2:N_2:H_2O=2:1:3.76:0$ | **Domain:** $(20,0,0)\times(1599,79,1)$<br>**Pressure:1Bar**<br>**Temperature:298K**<br>**Velocity: 0, 0, 0**<br>**Gas Spices (mole ratio):**<br>$H_2:O_2:N_2:H_2O=2:1:3.76:0$ |

In the Figure 5.38, the smoke-foil records of the simulation with ALMR under the base resolution 0.001m is given. Since the finest resolution is the same as the former simulation of decay and re-initiation with ALMR, the cellular structures in the smoke-foil record is almost the same as the former one.

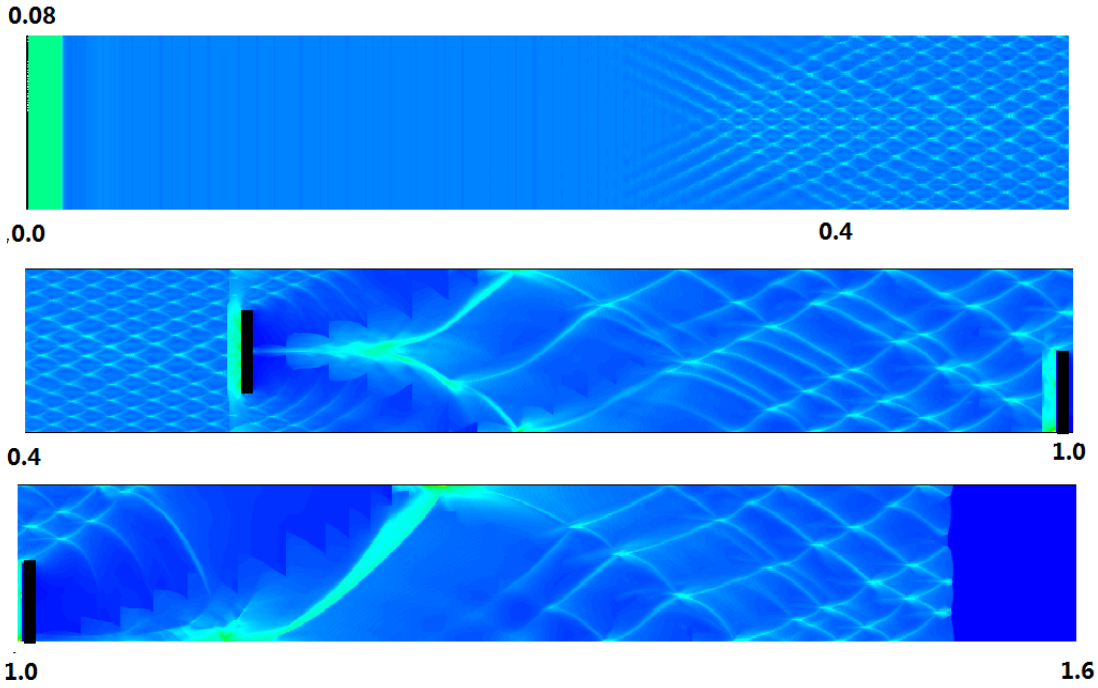



Figure 5.38 Numerical smoke-foil record for simulation with one finer level

After the comparison among the results of the simulation with ALMR and coarse resolutions, it is clear that the simulation with higher resolution can express more physical phenomenon, especially in simulation of detonation cellular structures. The maximum pressure records in the simulation with ALMR are different from the coarse grid simulation. Sometimes, maximum pressure given by the simulation with local mesh refinement can be 2 times as the simulation with coarse grid. In detonation simulation of industrial relevant problems such as the nuclear containments or big plants, the maximum pressure values are very important parameters for the design of the concrete body or safety equipment. With the local mesh refinement, very fine resolutions can be used efficiently in the huge industrial computational domains to get more accurate data.

In Table 5.16, the comparison of the time costs of all three kinds of simulation in the decay and re-initiation of detonation is given.

Table 5.16 Comparison of computational costs in simulation of decay and re-initiation

| | Computational domain | Computational cost |
|---|---|---|
| Full grids refinement | (0, 0, 0)X(6399, 319, 1) | 8 CPUs, 40 days (by estimation) |
| Adaptive local mesh refinement | Coarse grid: (0, 0, 0)X(399, 19, 1)<br>Finer levels: 2<br>Refinement ratio: 4, 4<br>Finest grid covered ratio: 2% | 8 CPUs, 7 days 12 hours |
| | Coarse grid: (0, 0, 0)X(1599, 79, 1)<br>Finer levels: 1<br>Refinement ratio: 4<br>Finest grid covered ratio: 2% | 8 CPUs, 2 days 20 hours |

In Table 5.16, computation with fully refined domain may require about 40 days to finish the whole simulation (since the total computational amount for one step of calculation in the uniform grid always requires the same time and 1000 steps of calculation costs 24 hours, the total computational cost will be about 40X24(hours)=40(days)). In the former part, it has been mentioned that the total computational efforts on finest level in the calculation with two finer levels are about four times as the case with 1 finer level. As shown in the table, the real time cost of the 2 finer levels calculation is less than three times as the case with 1 finer level. The result shows that the computational efforts on the base level with resolution 0.001m also take a considerable ratio in the total computational efforts. If the uniform time step is used, according to the computational efforts contained by the base level, there is no doubt that the time cost of the calculation with 1 finer level will be increased by about 50%. So, the above results can also show the importance of the multiple time steps method in computation saving.

In the section 5.3.2, complex physical phenomena of detonation wave in large-scale domain has been reproduced efficiently with the help of adaptive LMR. In hydrogen safety issues, detonation is the most destructive combustion mode and may bring great damages to the building structures. Studying of decay and re-initiation of the detonation wave is quiet meaningful for the hydrogen safety. Now, with the technique of local mesh refinement, studying of whether and how the obstacle can stop the propagation of detonation wave becomes much more efficient. With the help of the new technique, studying of the detailed detonation behavior in the facilities with larger physical size is also possible now.

### 5.3.3. 3D detonation simulation with ALMR

In the uniform grid computation simulation of 3D detonation cells may cost hundreds times more than the 2D cases, both the memory and computational resource of the cluster should withstand enormous loads. With the technique of local mesh refinement, the total computational efforts can be reduced to an acceptable level. Computational domain and initial conditions of the simulation of detonation in square cross-section tube are shown in Figure 5.39 and Table 5.17.

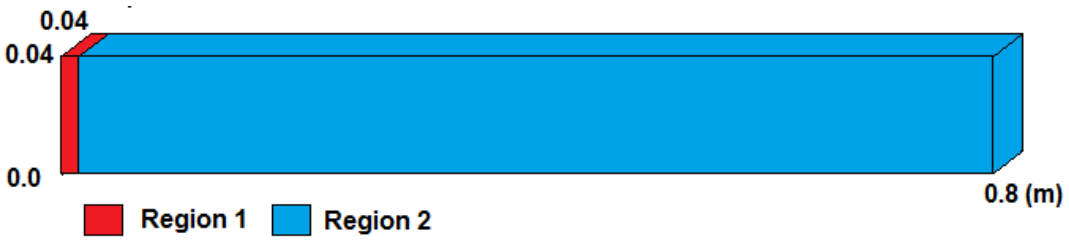



Figure 5.39 3D computational domain for detonation

Table 5.17 grid information and initial condition of the 3D detonation simulation

| **Grid Information** | |
|---|---|
| **Computational Domain:** $(0,0,0)\times(199,9,9)$<br>**Cell Size: 0.004m** | |
| **Initial Condition** | |
| **Region 1** | **Region 2** |
| **Domain:** $(0,0,0)\times(4,9,9)$<br>**Pressure:100Bar**<br>**Temperature:3000K**<br>**Velocity: 0, 0, 0**<br>**Gas Spices (mole ratio):**<br>$H_2:O_2:N_2:H_2O=2:1:3.76:0$ | **Domain:** $(5,0,0)\times(199,9,9)$<br>**Pressure:1Bar**<br>**Temperature:298K**<br>**Velocity: 0, 0, 0**<br>**Gas Spices (mole ratio):**<br>$H_2:O_2:N_2:H_2O=2:1:3.76:0$ |

In this 3D detonation simulation, finer resolutions are used adaptively. Similarly to the former simulation of decay and re-initiation, two finer levels with refinement ratio four are used in simulation.

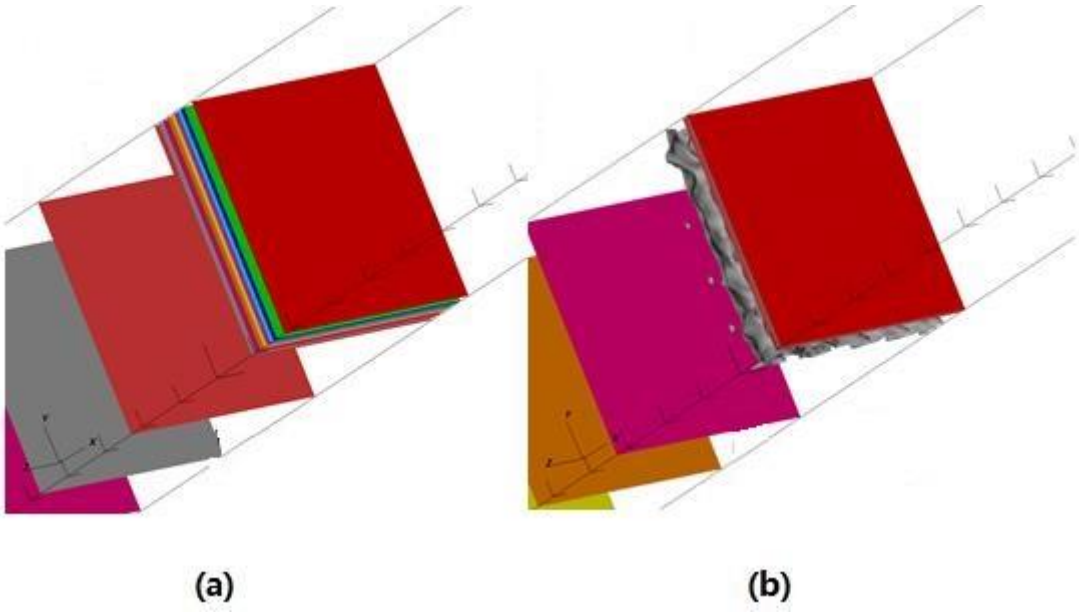



Figure 5.40 Contour figure of $V_x$ in coarse grid (a) and ALMR (b) simulation at step 400

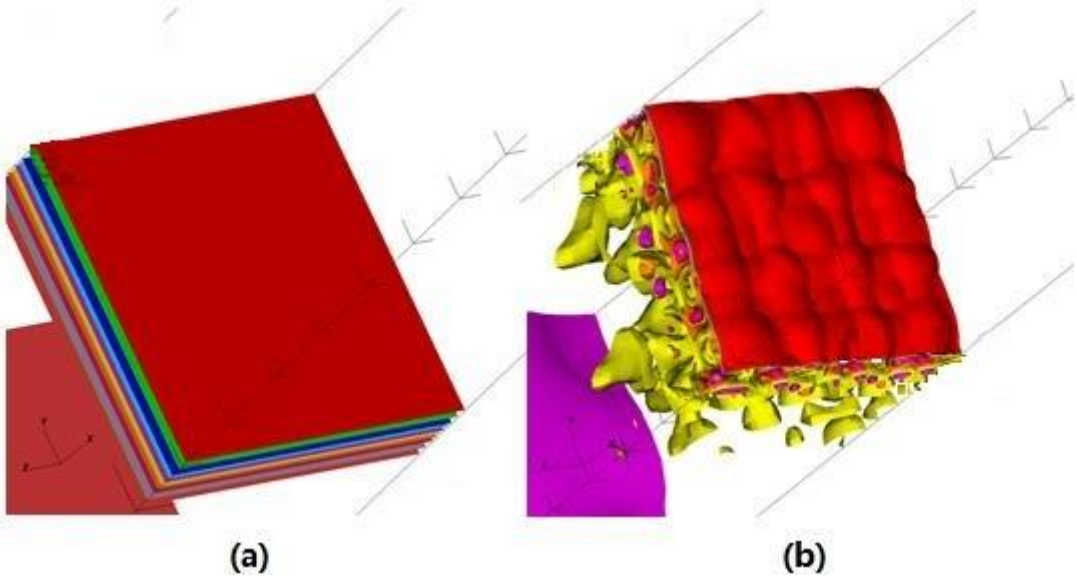



Figure 5.41 Contour figure of $V_x$ in coarse grid (a) and ALMR (b) simulation at step 800

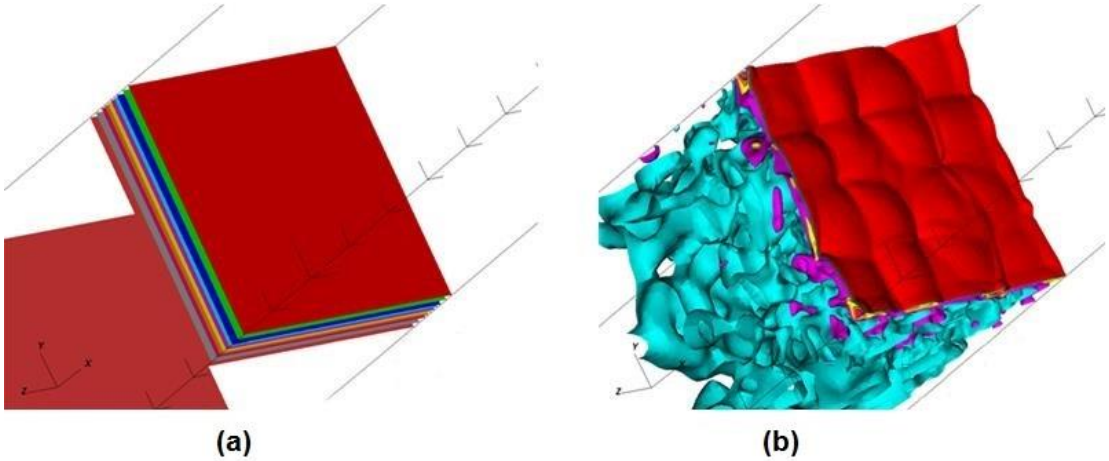



Figure 5.42 Contour figure of $V_x$ in coarse grid (a) and ALMR (b) simulation at step 1000

In the figures above, the comparison between coarse grid simulation and simulation with ALMR are given in three different time steps. In the coarse grid calculation, the resolution is not fine enough to simulate the triple points and no detonation cellular structure has been generated. In contrast, the simulation with ALMR can reach the resolution 0.00025m locally and some perturbation has been generated since the step 400. In the longer time simulation, about 16 detonation cells have been generated. The importance of ALMR in simulation of detonation cells is quite obvious. The more important is the time saving of LMR in 3D simulation. The comparison of computational costs between full grid refinement and ALMR is shown in Table 5.18. Differently from the 2D cases, benefits of the new technique in time saving can be performed perfectly in 3D cases: the finest region covered ratio in the ALMR case is 5%, the real calculation also costs about 5% of the full grid refinement case. Since the most practical applications require 3D computational domain, the local mesh refinement technology can large increase the efficiency of the code COM3D.

Table 5.18 Comparison of computational cost in 3D detonation simulations

| | Computational domain | Computational cost |
|---|---|---|
| Full grid refinement | (0, 0, 0)X(3199, 159, 159) | 64 CPUs , about 10 hours for 400 steps (about 20 days for 19200 steps) |
| Adaptive local mesh refinement | Coarse grid: (0, 0, 0)X(199, 9, 9)<br>Finer levels: 2<br>Refinement ratio: 4, 4<br>Finest region covered ratio: 5% | 8 CPUs, 6 days 18 hours (for 19200 steps' calculation on the finest level) |

Moreover, for the code COM3D, ability to simulate the 3D detonation cellular structure is a quite meaningful progress. Experimental results are all collected in 3D situation, using the detonation cells in 3D experiments to verify the 2D simulation sometimes may be unpersuasive. After the introduction of the LMR, COM3D can now make the V&V for the simulation of detonation cells in 3D, which helps discovering problems or errors hided by 2D geometry and improving the current detonation models.

.

# 6. Conclusions and future work

In this article, implementation of the block-structured patch-based local mesh refinement in basic gas dynamics and detonation simulation has been established in combustion code COM3D. In this final chapter the main results of this work are summarized and suggestions for future research are present.

## 6.1. Conclusions

### 6.1.1. Contribution to COM3D

The currently used uniform structured grid in the combustion code COM3D has greatly restricted the application of the program in large-scale industrial problems and the improvement of the code itself for a long time. The object of the work is to implement the technique of block-structured patch-based local mesh refinement (LMR) in COM3D for optimizing the computational grids structure and achieving a good balance between resolution and total computational efforts.

First of all, the implementations of the specific LMR addressed in this thesis emphasize on the use of temporal refinement, apart from spatial refinement, which is an important feature of the refinement technique. Differently from the normal LMR investing the most efforts on the finest refinement level, the LMR in the industrial relevant simulations occupies only a small fraction of computational efforts on the finer level. So the high efficiency produced by the refinement in time is highly desired by COM3D. In the implementations of LMR, the ghost region calculation in the alternating direction scheme and the further decoupled time-splitting operator in detonation simulations are established to maintain the usage of multiple time steps function. The advantages of local mesh refinement with temporal refinement (called also full-scale LMR) are very prominent in the case of detonation simulations with LMR in the region of pressure wave front and the case of jet simulations with the LMR in the adjacent region of a tiny nozzle.

Secondly, one of the biggest characteristics of the CFD code COM3D is that, it contains a variety of solvers and algorithms providing different accuracies. While implementing LMR, it should also be concerned to maintain the high accuracy of those high order schemes and algorithms. However, besides the spatial truncation errors as usual, the temporal truncation errors are introduced by keeping the temporal refinement. So, sometimes the implementation of full-scale local mesh refinement should be cautious. Unlike some other implementations of LMR ignoring analyses about accuracy, strict mathematical analyses are provided in the thesis to support in mathematics the reasonableness of the implementation. Especially, based on the strict mathematical analyses, the scheme of the parabolic and linear hybrid interpolation has been developed to solve the Navier-Stokes equations, in which both accuracy and conservation of physical quantities are guaranteed theoretically in the implementation of LMR.

In addition, the several practical applications are given in this thesis to show the advantages of the full-scale LMR technique. The motivation of the introduction of the local mesh refinement technique is to improve the current grids structures and to increase the calculation efficiency. After all, if the local mesh refinement cannot show big advantages in time saving, the implementation is

unnecessary at all no matter how good the implementation is in maintaining the accuracy of solvers and the feature of technique. In the 2D applications, even influenced by the minimal cell number in Z direction, the technique of local mesh refinement can still show 5 to 10 times faster solution. In the 3D applications, 20 times or higher efficiency can be achieved. Moreover, the applications also show that the technique of local mesh refinement can not only provide high efficiency in solving problems but also maintain the same capability as the full grid refinement, to disclose numerically the physical phenomena such as complicated detonation cells. The several application examples manifested perfectly the capability of LMR in balancing the high resolutions and total computation saving.

Supported by the mathematical analyses and application results, the patch-based local mesh refinement with multiple time steps has been implemented successfully in the solution of basic gas dynamics and detonation simulation in COM3D. Moreover, as proved in the practical applications, the technique of patch-based local mesh refinement solves successfully the historical problem of COM3D, namely, the confliction between resolution and total computational volume.

#### 6.1.2. Contribution to general implementation of LMR

In the thesis, the further operator splitting treatment is established in the implementation of LMR for the detonation simulation with time-splitting operator. Such method provides a new methodology in coupling solution of PDE and solution of ODE in LMR, and solves the redundant chemical reaction in detonation simulation. For the simulations with ODE style source terms, the further operator splitting treatment may become the key to the successful implementation of LMR.

### 6.2. Future Work

After the implementation of local mesh refinement in the solution of basic gas dynamics and detonation simulation, the implementation of LMR in more complex models should be considered.

#### 6.2.1. Future application of LMR for detonation simulation

In detonation simulations, local mesh refinement technique shows excellent performances in balancing the resolution and total computational efforts. With such technique, physical properties of the detonation wave can be expressed as the same as the uniform grid calculation. The new technique is also meaningful for implementing more complicated detonation chemical models in COM3D efficiently.

In the current version, chemical reaction in simulation of detonation is accomplished by the 1-step Arrhenius method, and the species $H_2, O_2, H_2O$ are included in the chemical process. However, the detailed process of the chemical reaction in the real detonation is more complicated, and the possible gas species in the chemical process may be more than the above three species. In some other simulations, besides the simple 1-step chemical simulation, the more complicated models such as 48 steps-8 species chemical reaction [69] or 19 steps-9 species chemical reaction [40] are used. In those more complicated chemical reactions, the unstable species such as $H, O, OH, H_2O_2, HO_2$ can exist only in the intensive chemical reaction area which is a very thin shear behind the detonation front, and high resolution is quite necessary to simulate the thin chemical reaction zone and express the distribution of the gas species. Normally, in the detonation

simulations with more complicated chemical processes, the cell size should be 0.01mm-0.2mm and the simulation of detonation may result in very huge computational efforts. In addition, the extra computational efforts introduced by the more complicated chemical reaction may increase the total computational efforts further.

The key to implement complex chemical reactions in COM3D is to improve the grids structure. With the technique of adaptive local mesh refinement, the fine grid region can be achieved in the chemical reaction zones locally [30]. The total cell number in the computational domain is much less than the uniform grid calculation. Consequently, the total computational cost may reduce to an acceptable level, although the usage of complex chemical reaction may increase the total computational efforts in chemical part largely. The distribution of those unstable gas species and the calculation for complex chemical reactions may not need to be considered on the coarse resolution level to save the memory and computational efforts further.

### 6.2.2. Future application of LMR for turbulent flow and other combustion regimes

Then, besides the implementation of LMR in the solution of basics gas dynamics, the more complicated turbulent models should be considered as well. Actually, the turbulent models used in COM3D such as the two equation models, one equation models and algebraic models are mostly based on the basic Navier-Stokes equations. So, the simple idea of the implementation of LMR in those turbulent models is making the implementation based on the implementations in NS equations and introducing new turbulent related parameters to the calculation. However, under the calculation with multiple time steps, the production of the spatial directives in the transport equations of those turbulent related parameters may bring some difficulties to the flux correction, especially in the high order algorithms which contain several intermediate phases. Although the calculation with uniform time step can solve such difficulties easily, the good efficiency brought by the multiple time steps is lost. Moreover, by the influence of very strong turbulent effects, the coupling of different refinement levels may also become a very big problem in the implementation of full-scale local mesh refinement, because the time step used on different refinement levels may be dominated by different factors.

In addition, future attention should also be put on the implementation of LMR in normal combustion simulation. In the detonation simulation, further operator splitting method is used to maintain the use of multiple time steps, is that possible to use such method in the others more complicated combustion models? In detonation, propagation of the flame is dominated by the propagation of high pressure. However, in some other combustion, the propagation of the flame is dominated by both the pressure and heat conduction, and sometimes the heat conduction may be more important. So, the governing equation of the combustion is quite different from the detonation, and the validation of the further operator splitting in normal combustion simulation may become doubtful. Moreover, differently from the point-wised chemical work in detonation simulation, the chemical reaction of some combustion models may contain the turbulent effects. Therefore, the neighbor cells also should be included in simulation of chemistry, and then the boundary updating may also become a problem under the full-scale local mesh refinement.

All in all, the work in the future and the problems may be encountered can be summarized into one point - implementation of the full-scale local mesh refinement to all the fluid and combustion models in COM3D.

### 6.3. Ongoing implementation of the technique

The technique of local mesh refinement has partially been implemented in combustion CFD code COM3D. However, limited by the license of library Chombo, the technique of local mesh refinement will become available for commercial projects by the end of 2014. At that time, COM3D with LMR will become available for large-scale industrial simulations. Some specific implementations of the technique in practical ongoing studies are listed below, which could not be systematically investigated without LMR.

#### 6.3.1. Implementation in large-scale nuclear problems

One of its practical applications is in nuclear safety analysis. Since the Fukushima nuclear accident, all existing nuclear power plants and plants in design need comprehensive studying of hydrogen safety. Taking the hydrogen combustion analysis for European Pressurized Reactor (EPR) as an example, combustion simulation with super computer Cray T3E is supposed to cost 5 to 10 days currently, but the implementation of local mesh refinement can greatly reduce the simulation time.

According to the nuclear containment layout, propagation of combustion can be predicted roughly by experience or coarse grids calculations. As indicated in Figure 6.1, only the predicted combustion region is needed to be covered with fine resolution and the others can be covered with coarse resolution. Under such grid structure, simulation can be boosted by 8-10 times compared to the uniform calculation. Therefore, more accidents scenarios can be simulated in the same period of time and more comprehensive studying of nuclear safety can be finally achieved.

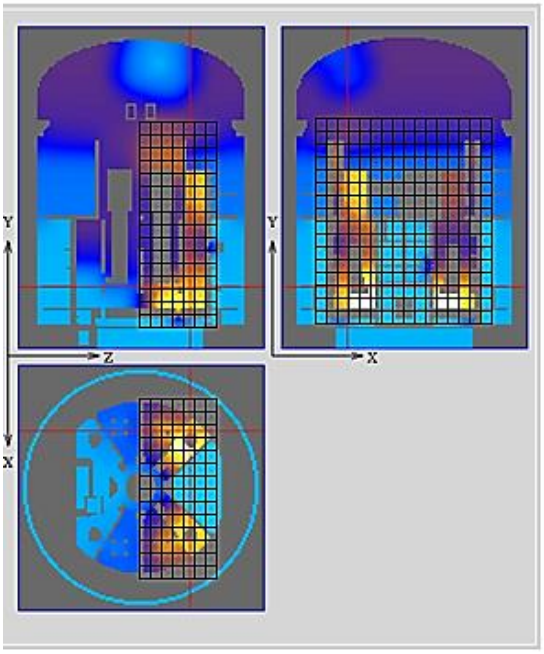



Figure 6.1 Inferred implementation of LMR in combustion inside EPR containment

In addition, similarly to the methodology used in 2D advection diffusion simulation, local mesh refinement technique can be used adaptively in the nuclear containment. In that situation, simulation efficiency of COM3D can be improved yet further.

### 6.3.2. Implementation in V&V projects

For the European project SUSANA (SUpport to Safety ANAlysis of Hydrogen and Fuel Cell Technologies), the technique of local mesh refinement is of great importance. The project is built on the complementarities of expertise of leading European experts in the field of CFD use for provision of hydrogen safety to achieve the synergy and consolidate the CFD excellence in application to safety design of FCH systems and infrastructure. The final outcome of the project is a comprehensive validation and verification (V&V) system, including numerical studying of CFD models, verification database, validation database and the protocol for V&V.

In the project, KIT leads the work on building the V&V database and the review of the database. In detailed data review, the deflagration and detonation cases shall be tested through the simulation with CFD combustion code COM3D. With the help of local mesh refinement technique, some V&V cases can be tested more efficiently and more comprehensively than before. Figure 6.2 shows the inferred implementation of the technique in detonation benchmarking simulation.

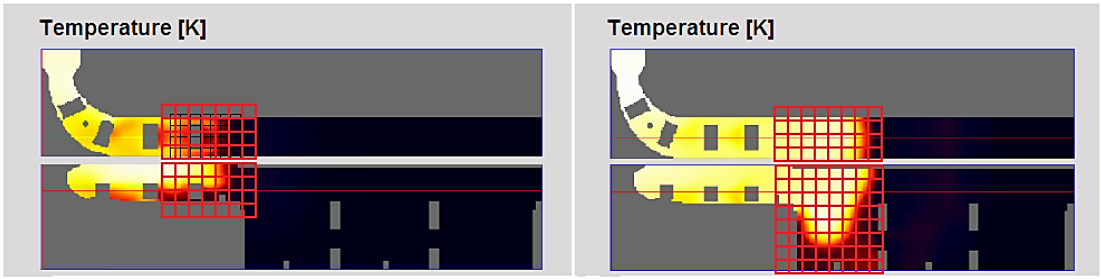



Figure 6.2 Inferred implementation of LMR in detonation verification

The validation case shown in Figure 6.2 is the detonation experiment made in RUT facility. According to the experience in detonation simulation in Chapter 5, 10-20 times higher simulation efficiency can be accomplished by the implementation of adaptive local mesh refinement in the V&V. Therefore, much more detonation and deflagration databases can be tested than before.

In addition, more complete evaluation of the experimental data can be achieved by the implementation of LMR. RUT facility has the size of 6.5 m × 2.5 m × 65 m. Numerical studying for the experiment in the RUT facility is a typical large-scale simulation. In order to control total computational efforts, chemical models such as one-step Arrhenius model and Crebcom model requiring fine resolutions cannot be tested under the current uniform grid structure. In contrast, with the technique of local mesh refinement and practical load balancing tool, those models can be validated by the experimental data as well. Therefore, for some large-scale experiment data, their V&V range can be studied more completely.

# References


[1] AMROC, http://amroc.sourceforge.net/

[2] ANSYS CFX, http://www.ansys.com/Products/Simulation+Technology/Fluid+Dynamics/Fluid+Dynamics+Products/ANSYS+CFX.

[3] Baker J. G. and van Meter J. R., "Reducing reflections from mesh refinement interfaces in numerical relativity", Physical Review D 72, 104010, 2005.

[4] Barad M. and Colella P., "A fourth-order accuracy local mesh refinement method for Poisson's equation", Journal of Computational Physics 209, 1-18, 2005.

[5] Bell J., Berger M., Saltzman J. and Welcome M., "Three-Dimensional Adaptive Mesh Refinement for hyperbolic Conservation Laws", SIAM J. SCI. COMPUT 15-1, 127-138, 1994.

[6] Berger M. J. and Oliger J., "Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations", Journal of Computational Physics 53, 484-512, 1984.

[7] Berger M. J. and Colella P., "Local Adaptive Mesh Refinement for Shock Hydrodynamics", Journal of Computational Physics 82, 64-84, 1989.

[8] Bielert U., Kotchourko A., Veser A. and Breitung W., "Multidimensional simulation of hydrogen distribution and turbulent combustion in severe accidents", Wissenschaftliche Berichte, FZKA-6696, 2002.

[9] Birch A. D., Brown D. R., Dodson M. G. and Swaffield F., "The Structure and Concentration Decay of High Pressure Jets of Natural Gas", Combustion Science and Technology 36, 249-261, 1984.

[10] Bjerketvedt D., Bakke J. R. and van Wingerden K., "Gas Explosion Handbook", Journal of Harzardous Materials 52, 1-150,1997.

[11] Boersma B. J., Kooper M. N., Nieuwstadt F. T. M. and Wesseling P., "Local Grid Refinement in Large-Eddy Simulations", Journal of Engineering mathematics 32, 167-175,1997.

[12] Cai Z., Gland F. L. and Zhang H., "An Adaptive Local Grid Refinement Method for Nonlinear Filtering", Research Report, Institut National de Recherche en Informatique et en Automatique, N° 2679, 1995.

[13] Chen C., Liu H. and Beardsley R., "An Unstructured Grid, Finite-Volume, Three-Dimensional, Primitive Equations Ocean Model: Application to Coastal Ocean and Estuaries", Journal of Atmospheric and Oceanic Technology 20, 159-186, 2003.

[14] Choi D., Brown J. D., Imbiriba B., Centrella J. and MacNeice P., "Interface Condition for Wave Propagation Through Mesh Refinement Boundary", Journal of Computational Physics 193, 398-425, 2004.

[15] Choi J. Y., Ma F. H. and Yang V., "Some Numerical Issues on Simulation of Detonation Cell Structures", Combustion, Explosion, and Shock Wave 44-5, 560-578, 2008.

[16] Chiang Y. Li, van Leer B. and Powell K. G., "Simulation of Unsteady Inviscid Flow on an Adaptive Refined Cartesian Grid", 30th Aerospace Science Meeting & Exhibit, AIAA 92-0443, 1992.

[17] Chombo, https://commons.lbl.gov/display/chombo.

[18] Colella P., Graves D. T., Keen N. D., Ligocki T. J., Martin D. F., McCoquodale P. W., Modiano D., Schwartz P. O., Sternberg T. D. and van Straalen B., "Chombo Software Packge for AMR Applications Design Document", Applied Numerical Algorithms Group,

Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, CA, 2009.

[19] Colella P., Dorr M. R. and Wake D. D., "Numerical Solution of Plasma Fluid Equations Using Local Refined Grids", Working Report, Lawrence Berkeley National Laboratory, UCRL-ID-126401, 1997.

[20] Colella P., Dorr M. R. and Wake D. D., "Numerical Solution of Plasma Fluid Equations Using Locally Refined Grids", Journal of Computational Physics 152, 550-583, 1999.

[21] COM3D, http://hycodes.net/com3d/.

[22] Courant R., Isaacson E. and Rees M., "On the Solution of Nonlinear Hyperbolic Differential Equation by Finite Differences", Comm. Pure Appl. Math. 5, 243-255, 1952.

[23] Crandall M. and Majda A., "The Method of Fractional Steps for Conservation Laws", Numer. Math. 34, 285-314, 1980.

[24] Crutchfield W. Y. and Welcom M. L., "Object-Oriented Implementation of Adaptive Mesh Refinement Algorithms", Scientific Programming 2, 145-156, 1993.

[25] Deiterding R., "Numerical Simulation of Transient Detonation Structures in $H_2$-$O_2$ Mixtures in Smooth Pipe Bends", 21st ICDERS, Poitier, 2007.

[26] Deiterding R., "Detonation Simulation with the AMROC Framework", ASCI Center for Simulation of Dynamic Response of Materials Site Visit, Oct 28-29, 2003.

[27] Deiterding R., "High-Resolution Numerical Simulation and Analysis of Mach Reflection Structures in Detonation Waves in Low-Pressure $H_2$-$O_2$-Air Mixture: A Summary of Results Obtained with the Adaptive Mesh Refinement Framework AMROC", Journal of Combustion, 2011: 738969, 2011.

[28] Deiterding R., "Detonation Structure Simulation with AMROC", High Performance Computing and Communciations, 916- 927, 2005.

[29] Deiterding R., "Construction and Application of an AMR Algorithm for Distributed Memory Computers", Adaptive Mesh Refinement – Theory and Applications, Proc. Of the Chicago Workshop on Adaptive Mesh Refinement Method Sept. 3-5, 361-372, 2003.

[30] Deiterding R., "High-resolution Simulation of Detonation with Detail Chemistry", Analysis and Numerics for Conservation Laws, 69-91, Springer, Berlin, 2005.

[31] Deiterding R., "A High-resolution Method for Realistic Detonation Structure Simulation", Hyperbolic Problems: Theory, Numerics, Applications, Sep. 13-17, Vol. I, 343-350, 2006.

[32] Deiterding R., "Parallel Adaptive Simulation of Weak and Strong Transverse-Wave Structures in $H_2$-$O_2$ Detonation", Parallel CFD 2009: Recent Advances and Future Directions, 519-534, DEStech Publications, Lancaster, 2010.

[33] Dickinson J. E., James S. C., Mehl S., Hill M. C., Leake S. A., Zyvoloski G. A., Faunt C. C. and Eddebbarh A., "A New Ghost-node method for linking different models and initial investigations of heterogeneity and nonmatching grids", Advances in Water Resources 30, 1722-1736, 2007.

[34] Droege M., "Local Mesh Refinement", Master's Thesis, Department of Mathematics, University of Groningen, 2000.

[35] Durbin P. A. and Iaccarino G., "An Approach to Local Refinement of Structured Grid", Journal of Computational Physics 181, 639-653, 2002.

[36] Farshchi M. and Hossainpour S., "Simulation of Detonation Initiation in Straight and Baffled Channels", Scientia Iranica 11-1&2, 37-49, 2004.

[37] Gropp W. D., “Local Uniform Mesh Refinement for Elliptic Partial Differential Equations”, Research Report, YALEU/DCS/RR-278, 1983.

[38] Harten A., ”High Resolution Schemes for Hyperbolic Conservation Laws”, Journal of Computational Physics 135, 260-278, 1997.

[39] Hino T., “A Finite-Volume Method with Unstructured Grid for Free Surface Flow Simulations”, 6th International Conference on Numerical Ship Hydrodynamics, 1993.

[40] Hu X. Y., Khoo B. C., Zhang D. L. and Jiang Z. L., “The Cellular Structure of a Two-Dimensional $H_2/O_2$/Air Detonation Wave”, Combustion Theory Model 8-2, 339-359, 2004.

[41] Huamo W., “On the Possible Accuracy of TVD Scheme”, Research Report, Konrad-Zuse-Zentrum fuer Informationstechnik Berlin, ISSN 0933-7911, 1989.

[42] IAEA, “Mitigation of Hydrogen Hazards in Severe Accidents in Nuclear Power Plants”, IAEA Safety Standards Series, IAEA-TECDOC-1661, 2011.

[43] Johansson H. and Steensland J., “A Characterization of a Hybrid and Dynamics Partitioner for SAMR Applications”, 16th IASTED International Conference on Parallel and Distributed Computing and System, 2004.

[44] Johansson H. and Vakili Abbas, “A Patch-based Partitioner for Parallel SAMR Applications”, Proceedings of the International Conference on Parallel and Distributed Computing and Systems, 2008.

[45] Johansson H., “Design and Implementation of an Adaptive Meta-Partitioner for SAMR Grid Hierarchies”, Report 2008-017, Department of Information Technology, Uppsala University, Sweden, 2008. Available at http://www.it.uu.se/research/reports/2008-017/.

[46] Johansson H., “A Meta-Partitioner for Run-time and Evaluation of Multiple Partitioning Algorithms for SAMR Grid Hierarchies”, Report 2009-008, Department of Information Technology, Uppsala University, Sweden, 2009. Available at http://www.it.uu.se/research/reports/2009-008/.

[47] Jones D. A., Kemister G. and Borg R. A. J., “Numerical Simulation of Detonation in Condensed Phase Explosives”, Weapons Systems Division, Aeronautical and Maritime Research Laboratory, DSTO-TR-0705, 1998.

[48] KeLP V1.4, http://cseweb.ucsd.edu/groups/hpcl/scg/KeLP1.4/.

[49] Kort A. J. A., Werstappen R. W. C. P., Wubs F. W. and Veldman A. E. P., “Fully Conservative Discretizations for Local Grid Refinement”, 3rd IASME/WSEAS Int. Conference on Fluid Dynamics & Aerodynamics, August 20-22, 261-266, 2005.

[50] Kotchourko, A., “Methodology of CFD safety analysis for large-scale industrial structures”, 1st International Conference on Hydrogen Safety, Pisa 8-10 September, 2005.

[51] Kotchourko A., Lelyakin A., Yanez J., Xu Z., Ren K.: COM3D: Turbulent Combustion Code Tutorial Guide Version 4.2, KIT, 2011.

[52] Kudriakov1 S., Dabbene1 F., Studer1 E. and Beccantini A., “THE TONUS CFD CODE FOR HYDROGEN RISK ANALYSIS: PHYSICAL MODELS, NUMERICAL SCHEMES AND VALIDATION MATRIX”, CFD4NRS, Garching, Munich, 2006.

[53] Lea C. J. and Ledin H. S., “A Review of the State-of-the-Art in Gas Explosion Modeling”, technical report, HSL/2002/02, 2002.

[54] Lehnasch G. “thèse réalisée et soutenue en 2005 au laboratoire de Combustion et de Détonique UPR 9028 CNRS”, Poitiers, 2005.

[55] Li S. and Hyman J. M., “Adaptive Mesh Refinement for Finite Difference WENO Schemes”, Los Alamos Report, LA-UR-03-8927, 2003.

[56] Lie K. A., Haugse V. and Karlsen K. H., “Dimensional Splitting with Front Tracking and Adaptive Grid Refinement”, Numerical Methods for PDEs 14-5, 627-648, 1998.

[57] Liou M. and Steffen C. J., “A New Flux Splitting Scheme”, Journal of Computational Physics 107, 23-39, 1993.

[58] Liou M., “A Sequel to AUSM: AUSM+”, Journal of Computational Physics 129, 364-382, 1996.

[59] Liou M., “A Sequel to AUSM, Part II: AUSM$^{+}$-up”, Journal of Computational Physics 214, 137-170, 2006.

[60] Lisitsa V., Reshetova G. and Tcheverda V., “Local Time-Space Mesh Refinement for Finite Difference Simulation of Waves”, Numerical Mathematics and advanced Applications 2009, Proceedings of ENUMATH 2009, 609-616. Springer, Heidelberg, 2010.

[61] Martin D. F. and Colella P., “A Cell-Centered Adaptive Projection Method for the Incompressible Euler equations”, Journal of Computational Physics 163, 271-312, 2000.

[62] Martin D. F., Colella P. and Graves D. T., “A Cell-Centered Adaptive Projection Method for the Incompressible Navier-Stokes Equations in Three Dimensions”, Journal of Computational Physics 227, 1863-1886, 2008.

[63] Martin D. F. and Cartwright K. L., “Solving Poisson’s Equation using Adaptive Mesh Refinement”, Technical Report UCB/ERI M96/66 UC Berkeley, 1996.

[64] Mavriplis D. J., “Unstructured Grid Technique ”, Annual Reviews of Fluid Mechanism 29, 473-514, 1997.

[65] Mehl S., Hill M. C. and Leake S., “Comparison of Local Grid Refinement Methods for MODEFLOW”, Ground Water 44-6, 792-796, 2006.

[66] Minion M. L., “A Projection Method for Locally Refined Grids”, Journal of Computational Physics 127, 158-178, 1996.

[67] Olstad B. and Manne F., “Efficient Partitioning of Sequences”, IEEE Transactions on Computers 44-11, 1322-1326, 1995.

[68] Oran E. S., Boris J. P., Young T., Flanigan M., Burks T. and Picone M., “Numerical Simulation of Detonations in Hydrogen-Air and Methane-Air Mixture”, Eighteenth Int. Symp. On Combustion, 1641-1649, 1981.

[69] Oran E. S., Weber J. W., Stefaniw E. I., Lefebvre M. H. and Anderson J. D., “A Numerical Study of a Two-Dimensional $H_2$-$O_2$-Air Detonation Using a Detailed Chemical Reaction Model”, Combustion Flame 116 (1-2), 154-165, 1999.

[70] Owen S. J., “A Survey of Unstructured Mesh Generation Technology”, 7$^{th}$ International Meshing Roundtable, 239-267, 1998.

[71] Rendleman C. A., Beckner V. E., Lijweski M, Crutchfield W. and Bell J. B., “Parallelization of Structured, Hierarchical Adaptive Mesh Refinement Algorithms”, Computing and Visualization, 1999.

[72] Rohde M., Kandhai D., Derksen J. J. and van den Akker H. E. A., “A generic, mass conservative local mesh refinement technique for lattice-Boltzmann schemes”, International Journal for Numerical Methods in Fluids 51, 439-468, 2006.

[73] SAMRAI, https://computation.llnl.gov.casc/SAMRAI/index.html.

[74] Sharma S., Singh S. and Sharma M., “Performance Analysis of Load Balancing Algorithms”,

World Academy of Science, Engineering and Technology 38, 269-272, 2008.
[75] Strang G. “ON THE CONSRUCTION AND COMPARION OF DIFFERENCE SCHEMES”, SIAM Journal on Numerical Analysis 5-3, 506-517, 1968.
[76] Shen C., Qiu J. and Christlieb A., “Adaptive mesh refinement based on high order finite difference WENO scheme for multi-scale simulation”, Journal of Computational Physics 230, 3780-3802, 2011.
[77] Steensland J. and Ray J., “A Heuristic Re-Mapping Algorithm Reducing Inter-Level Communication in SAMR Applications”, 15th IASTED International Conference Parallel and Distributed Computing and System, 2003.
[78] Steensland J., “Dynamic Structured Grid Hierarchy Partitioners Using Inverse Spacing-Filling Curves”, Technical Report 2001-002, Dept. of Scientific Computing, Uppsala University, Uppsala, Sweden, 2001.
[79] Steensland J., Chandra and Parashar M., “An Application-Centric Characterization of Domain-Based SFC Partitioners for Parallel SAMR”, IEEE Transactions on Parallel and Distributed System, Vol. 13, No. 12, 1275-1289, 2002.
[80] Steensland J. and Ray J., “A Partitioner-Centric Model for SAMR Partitioning Trade-off Optimization: Part I”, 4th Symposium of the Los Alamos Computer Science Institute (LACSI04), 2003.
[81] Steensland J. and Ray J., “A Partitioner-Centric Model for SAMR Partitioning Trade-off Optimization: Part II”, 6th Workshop on High Performance Scientific and Engineering Computing (HPSEC-04), 2004.
[82] Steensland J., “Efficient Partitioning of Dynamics Structured Grid Hierarchies”, PhD thesis, Department of Scientific Computing, Information Technology, Uppsala University, Oct. 2002.
[83] Takasuo E. and Huhtanen R., “Application of TONUS V2006 and FLUENT 6.2.16 CFD codes to ENACCEF hydrogen combustion tests”, technical report, STUK-TR 1, 2007.
[84] Taki S. and Fujiwara T., “Numerical Simulation of Triple Shock Behavior of Gaseous Detonation”, 18th Symposium (International) on Combustion, 1671-1681, 1981.
[85] Trompert R. A. and Verwer J. G., “Runge-Kutta Methods and Local Uniform Grid Refinement”, Mathematics of Computation 60-202, 591-616, 1993.
[86] Tullio M. D., Palma P. D. Iaccarino G., Pascazio G. and Napolitano M., “An immersed boundary method for compressible flows using local grid refinement”, Journal of Computational Physics 225, 2098-2117, 2007.
[87] van Leer B., “Towards the Ultimate Coservative Difference Scheme II Monotonicity and Conservation Combined in a Second-Order Scheme”, Journal of Computational Physics 14, 361-370, 1974.
[88] van Leer B., “Towards the Ultimate Coservative Difference Scheme III Upstream-Centered Finite-Difference Schemes for Ideal Compressible Flow”, Journal of Computational Physics 23, 263-275, 1977.
[89] van Leer B., “Towards the Ultimate Coservative Difference Scheme V A Second-Order Sequel to Godunov’s Method”, Journal of Computational Physics 32, 101-136, 1979.
[90] Wilkening H. and Huld T., “An adaptive 3–D CFD solver for explosion modeling on large scales”, Combustion Science and Technology, Vol.149, No.1-6, 361-388, 1999.
[91] Xu Z., Travis J. R., Breitung W. “Green’s Function Method and its Application to Verification

of Diffusion Models of GASFLOW Code", Wissenschaftliche Berichte, FZKA-7293, 2007.
[92] Yueceil K. B. and Oetuegen M. V., "Scaling parameters for underexpanded supersonic jets", Physics of Fuids 14-12, 4206-4215, 2002.
[93] Yes¸im YAZICI, "OPERATOR SPLITTING METHODS FOR DIFFERENTIAL EQUATIONS", master thesis, Izmir Institute of Technology, 2010.
[94] Zakharian A. R., Brio M. and Moloney J. V., "FDTD Based second-Order Accurate Local mesh Refinement Method for Maxwell's Equation in Two Space Dimensions", Comm. Math. Sci. 2-3, 497-513, 2004.

# Appendix A

In the construction of the TVNI solver, the second order solver Lax-Wendroff (LW) was used in the proof of high order accuracy of TVNI. The alternating direction scheme was also firstly introduced and implemented in LW solver [23]. The truncation error analyses for the AD based LW solver is made as the first step, and then the proof for AD based TVNI is trivial.

$$\frac{\partial u}{\partial t} = A\frac{\partial u}{\partial x} + B\frac{\partial u}{\partial y} \tag{A-1}$$

The equation (A-1) is one 2D linear hyperbolic problem, the matrixes A and B in the equation are constant. Its second order Taylor series in time is,

$$u(T+\Delta t) = u(T) + \Delta t\frac{\partial u}{\partial t}|_T + \frac{(\Delta t)^2}{2}\frac{\partial^2 u}{\partial t^2}|_T + O((\Delta t)^3) \tag{A-2}$$

Then, following the 2D PDE equation (A-1), the Taylor series (A-2) can be transmitted into the format (A-3), which is also the format of LW solver.

$$u(T+\Delta t) = u + \Delta t(Au_x + Bu_y) + \frac{(\Delta t)^2}{2}(A^2u_{xx} + ABu_{xy} + BAu_{xy} + B^2u_{yy}) = LWu(T) \tag{A-3}$$

In (A-3), since the matrix space is not Abelian, $AB \neq BA$ and the two cross derivatives terms cannot be added together directly. In the alternating direction scheme, the 2D linear hyperbolic differential equation is decomposed into two 1D linear hyperbolic equations.

$$\frac{\partial v}{\partial t} = A\frac{\partial v}{\partial x}, \quad \frac{\partial w}{\partial t} = B\frac{\partial w}{\partial y} \tag{A-4}$$

For the two 1D problems in (A-4), their LW based solutions are,

$$\begin{aligned} v_{n+1} &= v_n + \Delta t\cdot Av_x + (\Delta t)^2/2\cdot A^2v_{xx} = (I + \Delta t\cdot A\frac{\partial}{\partial x} + \frac{(\Delta t)^2}{2}\cdot A^2\frac{\partial^2}{\partial x^2})v_n = L_x^{\Delta t}v_n \\ w_{n+1} &= w_n + \Delta t\cdot Bw_y + (\Delta t)^2/2\cdot B^2w_{yy} = (I + \Delta t\cdot B\frac{\partial}{\partial y} + \frac{(\Delta t)^2}{2}\cdot B^2\frac{\partial^2}{\partial y^2})w_n = L_y^{\Delta t}w_n \end{aligned} \tag{A-5}$$

At the very beginning, the alternating direction scheme was introduced for more relaxed time step and used as (A-6).

$$\begin{aligned} u_{n+1} &= L_y^{\Delta t}L_x^{\Delta t}u_n = (I + \Delta t\cdot B\frac{\partial}{\partial y} + \frac{(\Delta t)^2}{2}\cdot B^2\frac{\partial^2}{\partial y^2})(I + \Delta t\cdot A\frac{\partial}{\partial x} + \frac{(\Delta t)^2}{2}\cdot A^2\frac{\partial^2}{\partial x^2})u_n \\ u_{n+1} &= u_n + \Delta t\cdot(Au_x + Bu_y) + (\Delta t)^2/2\cdot(A^2u_{xx} + 2BAu_{xy} + B^2u_{yy}) \end{aligned} \tag{A-6}$$

By subtracting with the original LW scheme, the result (A-7) shows that some second order terms are remained.

$$LWu_n - L_y^{\Delta t}L_x^{\Delta t}u_n = \frac{1}{2}\cdot(\Delta t)^2(AB - BA)u_{xy} \tag{A-7}$$

So the original alternating direction scheme can only provide second order accuracy in time for each step of calculation. The whole calculation scheme is reduced to first order (which means that

the total accumulation error is first order in time). In order to maintain the temporal accuracy of LW scheme, the modified alternating direction scheme is present,

$$u_{n+1} = L_x^{\Delta t/2} L_y^{\Delta t/2} L_y^{\Delta t/2} L_x^{\Delta t/2} u_n$$

$$\begin{aligned} L_x^{\Delta t/2} L_y^{\Delta t/2} L_y^{\Delta t/2} L_x^{\Delta t/2} u_n &= (I + \frac{\Delta t}{2} \cdot (A\frac{\partial}{\partial x} + B\frac{\partial}{\partial y}) + \frac{1}{2}(\frac{\Delta t}{2})^2 \cdot (A^2 \frac{\partial^2}{\partial x^2} + 2AB\frac{\partial^2}{\partial xy} + B^2 \frac{\partial^2}{\partial y^2})) \\ &\quad (I + \frac{\Delta t}{2} \cdot (A\frac{\partial}{\partial x} + B\frac{\partial}{\partial y}) + \frac{1}{2}(\frac{\Delta t}{2})^2 \cdot (A^2 \frac{\partial^2}{\partial x^2} + 2BA\frac{\partial^2}{\partial xy} + B^2 \frac{\partial^2}{\partial y^2}))u_n \\ &= (I + \Delta t \cdot (A\frac{\partial}{\partial x} + B\frac{\partial}{\partial y}) + \frac{1}{2}(\Delta t)^2 \cdot (A^2 \frac{\partial^2}{\partial x^2} + AB\frac{\partial^2}{\partial xy} + BA\frac{\partial^2}{\partial xy} + B^2 \frac{\partial^2}{\partial y^2}))u_n \\ &= LWu_n + O(\Delta t^3) \end{aligned} \tag{A-8}$$

So, for the linear hyperbolic equation the modified alternating directions scheme can reach the same accuracy in time as the original LW scheme. Next, for the 2D nonlinear hyperbolic problem:

$$\frac{\partial u}{\partial t} = \frac{\partial f(u)}{\partial x} + \frac{\partial g(u)}{\partial y} \tag{A-9}$$

Its second order Taylor series is shown as the equation (A-10).

$$u(T + \Delta t) = u(T) + \Delta t \frac{\partial u}{\partial t}|_T + \frac{(\Delta t)^2}{2}\frac{\partial^2 u}{\partial t^2}|_T + O((\Delta t)^3) \tag{A-10}$$

Similarly, following the relation shown by the hyperbolic partial differential equation (A-9), the LW scheme for the nonlinear problem is established.

$$\begin{aligned} &\frac{\partial u}{\partial t} = \frac{\partial f}{\partial x} + \frac{\partial g}{\partial y} \\ &\frac{\partial^2 u}{\partial t^2} = \frac{\partial}{\partial t}\frac{\partial u}{\partial t} = \frac{\partial}{\partial t}(\frac{\partial f(u)}{\partial x} + \frac{\partial g(u)}{\partial y}) = (\frac{\partial f(u)}{\partial x} + \frac{\partial g(u)}{\partial y})_t = (\frac{\partial f(u)}{\partial t})_x + (\frac{\partial g(u)}{\partial t})_y \\ &(\frac{\partial f(u)}{\partial t})_x = (\frac{\partial f}{\partial u}\frac{\partial u}{\partial t})_x = (\frac{\partial f}{\partial u}\frac{\partial f}{\partial x} + \frac{\partial f}{\partial u}\frac{\partial g}{\partial y})_x = ((\frac{\partial f}{\partial u})^2 \frac{\partial u}{\partial x} + \frac{\partial f}{\partial u}\frac{\partial g}{\partial u}\frac{\partial u}{\partial y})_x \\ &(\frac{\partial g(u)}{\partial t})_y = (\frac{\partial g}{\partial u}\frac{\partial u}{\partial t})_y = (\frac{\partial g}{\partial u}\frac{\partial f}{\partial x} + \frac{\partial g}{\partial u}\frac{\partial g}{\partial y})_y = (\frac{\partial g}{\partial u}\frac{\partial f}{\partial u}\frac{\partial u}{\partial x} + (\frac{\partial g}{\partial u})^2 \frac{\partial u}{\partial y})_y \end{aligned} \tag{A-11}$$

In formulas (A-11), $\partial g/\partial u$ and $\partial f/\partial u$ are the two Jacobi matrices. In the alternating direction scheme for the no-linear hyperbolic problem, the nonlinear 2D hyperbolic problem is also decomposed into two 1D hyperbolic problems.

$$\frac{\partial v}{\partial t} = \frac{\partial f(v)}{\partial x}, \quad \frac{\partial w}{\partial t} = \frac{\partial g(w)}{\partial y} \tag{A-12}$$

For the two nonlinear PDEs in (A-12), their LW schemes are:

$$\begin{aligned} v_{n+1} &= v_n + \Delta t \cdot f_x(v) + (\Delta t)^2/2 \cdot ((f_v)^2 \cdot v_x)_x = (I + \Delta t \cdot \frac{\partial f}{\partial x} + \frac{(\Delta t)^2}{2} \cdot \frac{\partial}{\partial x}((\frac{\partial f}{\partial v})^2 \frac{\partial}{\partial x}))v_n = L_x^{\Delta t} v_n \\ w_{n+1} &= w_n + \Delta t \cdot g_y(w) + (\Delta t)^2/2 \cdot ((g_w)^2 \cdot w_y)_y = (I + \Delta t \cdot \frac{\partial g}{\partial y} + \frac{(\Delta t)^2}{2} \cdot \frac{\partial}{\partial y}((\frac{\partial g}{\partial w})^2 \frac{\partial}{\partial y}))w_n = L_y^{\Delta t} w_n \end{aligned} \tag{A-13}$$

Similarly to the linear case, the original alternating directions scheme $u_{n+1} = L_y^{\Delta t} L_x^{\Delta t} u_n$ can only provide second order accuracy for single step of calculation and the modified alternating direction scheme $u_{n+1} = L_x^{\Delta t/2} L_y^{\Delta t/2} L_y^{\Delta t/2} L_x^{\Delta t/2} u_n$ can keep the third order accuracy in time. Consequently, by using the same mathematical analyses, it can be proved that the 3D AD scheme used in COM3D can maintain the high temporal accuracy.

Finally, based on the discussion about the AD based LW solver, the discussion of AD based TVNI solver can be made. Normally, the numerical solution of 1 D problem is expressed as the calculation of fluxes as (A-14.)

$$u_j^{n+1} = u_j^n + \frac{\Delta t}{\Delta x}(f(u_{j+1/2}^n) - f(u_{j-1/2}^n)) \tag{A-14}$$

In the LW scheme, the flux is as (A-15).

$$F_{LW} = f + \frac{\Delta t}{2} \cdot (\frac{\partial f}{\partial u})^2 u_x \tag{A-15}$$

When the TVNI solver is established, comparison of the fluxes between LW scheme and TVNI scheme is made to show the newly established solver can reach the same accuracy as the LW in general. The differences between the LW fluxes and TVNI fluxes are second order.

$$F_{TVNI} = F_{LW} + O(\Delta^2) \tag{A-16}$$

As a result, the TVNI solver can reach the same accuracy as the LW scheme in 1D calculation (in some special point, in order to keep the monotonicity, the TVNI may reduce to first order scheme in space and time [41]). The calculation with AD based TVNI solver can reach the general second order accuracy as well.

# Appendix B

In the patch-based local mesh refinement, coarse-to-fine interpolation, flux correction and the fine-to-coarse data transfer have been used. In order to prove if proper inter-level data communications have been implemented, mathematical analyses are done to search the truncation errors brought by the communication.

Similarly to the analyses in TVNI solver, if the fluxes on the finer level can satisfy (B-1),

$$F_{TVD}(\tilde{u}_i) = F_{TVD}(u_i) + O(\Delta^2)$$

Or (B-1)

$$F_{TVD}(\tilde{u}_i) = F_{LW}(u_i) + O(\Delta^2)$$

the calculation on finer level can keep the same accuracy as the uniform grid calculation in general. In equation (B-1), the item $u_i$ stands for the normal data on the finer level and the item $\tilde{u}_i$ stands for the data which include the influence of inter-level data communication.

Each patch contains internal region $\Omega$ and the ghost region, the symbol $\Omega(-N)$ means the block-structured domain achieved by shirking $N$ cells in each direction in $\Omega$. After the interpolation or some directions' calculation, the data on the finer level can at least satisfy (B-2).

$$\tilde{u}_i = \begin{cases} u_i & i \in \Omega(-4) \\ u_i + O(\Delta^2) & otherwise \end{cases} \tag{B-2}$$

Since the data contained in region $\Omega(-4)$ are not influenced by the truncation error produced by the coarse-to-fine interpolation, the data in this region are always assumed to be "real". The data in other regions are gotten by either coarse-to-fine interpolation or calculation with interpolated data, they may contain second order truncation errors. There is no need to worry about whether this assumption may cause the logical error circular reasoning, because TVNI can at least provide the accuracy $O(\Delta t \cdot \Delta x) + O(\Delta t^2) \sim O(\Delta x^2) + O(\Delta t^2)$ with the second order interpolated data. In the TVNI solver, the expression of the flux in [38] for the scalar mode is,

$$F_{i+1/2}^{TVNI} = \frac{1}{2}[F_i^M + F_{i+1}^M - \frac{1}{\lambda} Q(\bar{v}_{j+1/2}^M)\Delta_{i+1/2}u] \tag{B-3}$$

In equation (B-3), detailed terms are,

$$\begin{aligned}
&F_i^M = f(u_i) + \frac{1}{\lambda} g_i, \quad \bar{v}_{j+1/2}^M = \bar{v}_{i+1/2} + \gamma_{i+1/2}, \quad \gamma_{i+1/2} = (g_{i+1} - g_i)/\Delta_{i+1/2}u \\
&g_i = s_{i+1/2} \max[0, \min(|\tilde{g}_{i+1/2}|, \tilde{g}_{i-1/2} \cdot s_{i+1/2})] \\
&\tilde{g}_{i+1/2} = \frac{1}{2}[Q(\bar{v}_{i+1/2}) - (\bar{v}_{i+1/2})^2]\Delta_{i+1/2}u, \quad s_{i+1/2} = sign(\tilde{g}_{i+1/2}) \\
&\bar{v}_{i+1/2} = \lambda(f_{i+1} - f_i)/\Delta_{i+1/2}u, \quad \Delta_{i+1/2}u = u_{i+1} - u_i, \quad \lambda = \Delta t/\Delta x
\end{aligned} \tag{B-3}$$

In COM3D, the special map $Q(x)$ in (B-3) is,

$$Q(x)=\begin{cases}(x^2/(4\xi))+\xi & |x|<2\xi=1\times10^{-300}\\ |x| & otherwise\end{cases} \tag{B-4}$$

One important property of the function $Q$ is its continuous satisfies the Lipschitz continuous, which mean,

$$\exists K>0,\forall x,y\in\Omega\Rightarrow|Q(x)-Q(y)|\le K|x-y| \tag{B-5}$$

From the definition of $Q(x)$, it is not difficult to prove that $[Q(\overline{v}_{i+1/2})-(\overline{v}_{i+1/2})^2]/(2\lambda)$ and $\gamma_{i+1/2}$ are all bounded. In the following analyses, with the property of $Q$ and the definition of the flux in TVNI solver, the analyses are done.

For $\overline{v}_{i+1/2}$, the truncation errors in it are,

$$\begin{aligned}&\tilde{\overline{v}}_{i+1/2}=\lambda(f(\tilde{u}_{i+1})-f(\tilde{u}_i))/\Delta_{i+1/2}\tilde{u},\quad \Delta_{i+1/2}\tilde{u}=\tilde{u}_{i+1}-\tilde{u}_i\\ &f(\tilde{u})=f(u+O(\Delta^2))=f(u)+f'(u)\cdot O(\Delta^2)+O(\Delta^4)=f(u)+O(\Delta^2)\\ &\Delta_{i+1/2}\tilde{u}=\tilde{u}_{i+1}-\tilde{u}_i=u_{i+1}-u_i+O(\Delta^2)=\Delta_{i+1/2}u+O(\Delta^2)\end{aligned} \tag{B-6}$$

Since the item $\Delta_{i+1/2}u=u_{i+1}-u_i=u'\cdot\Delta x+O(\Delta x^2)>>O(\Delta^2)$ and it appears as a divider in the (B-3), $1/\Delta_{i+1/2}\tilde{u}$ can be turned into the format in (B-7).

$$\begin{aligned}&\Delta_{i+1/2}\tilde{u}=\Delta_{i+1/2}u+O(\Delta^2)=\Delta_{i+1/2}u(1+O(\Delta^2)/\Delta_{i+1/2}u)\\ &\frac{1}{\Delta_{i+1/2}\tilde{u}}=\frac{1}{\Delta_{i+1/2}u}\frac{1}{1+O(\Delta^2)/\Delta_{i+1/2}u}=\frac{1}{\Delta_{i+1/2}u}\cdot\left(1+\sum_{n=1}^{+\infty}\left(-\frac{O(\Delta^2)}{\Delta_{i+1/2}u}\right)^n\right)=\frac{1}{\Delta_{i+1/2}u}+\frac{1}{\Delta_{i+1/2}u}\cdot\frac{O(\Delta^2)}{\Delta_{i+1/2}u}\end{aligned} \tag{B-7}$$

Then, errors in $\tilde{\overline{v}}_{i+1/2}$ is,

$$\begin{aligned}\tilde{\overline{v}}_{i+1/2}&=\lambda\frac{\Delta_{1+1/2}f+O(\Delta^2)}{\Delta_{i+1/2}u}\left(1+\frac{O(\Delta^2)}{\Delta_{i+1/2}u}\right)=\lambda\frac{\Delta_{1+1/2}f}{\Delta_{i+1/2}u}+\lambda\frac{O(\Delta^2)}{\Delta_{i+1/2}u}+\lambda\frac{\Delta_{1+1/2}f}{\Delta_{i+1/2}u}\frac{O(\Delta^2)}{\Delta_{i+1/2}u}+\frac{O(\Delta^3)}{\Delta_{i+1/2}u}\\ &=\lambda\frac{\Delta_{1+1/2}f}{\Delta_{i+1/2}u}+\lambda\frac{O(\Delta^2)}{\Delta_{i+1/2}u}=\overline{v}_{i+1/2}+\lambda\frac{O(\Delta^2)}{\Delta_{i+1/2}u}\end{aligned} \tag{B-8}$$

As shown in the equation (B-9), truncation error analysis for $g_i$ should be based on the analysis for $\tilde{g}$.

$$\begin{aligned}&g_i(\tilde{u}_i)=s_{i+1/2}\max[0,\min(|\tilde{g}_{i+1/2}(\tilde{u})|,\tilde{g}_{i-1/2}(\tilde{u})\cdot s_{i+1/2})]\\ &\tilde{g}_{i+1/2}(\tilde{u})=\frac{1}{2}[Q(\tilde{\overline{v}}_{i+1/2})-(\tilde{\overline{v}}_{i+1/2})^2]\Delta_{i+1/2}\tilde{u}\end{aligned} \tag{B-9}$$

For the map $Q$ is Lipschitz continuous and $\lambda$ is bounded, the truncation errors in $\tilde{g}$ is,

$$\begin{aligned}\tilde{g}_{i+1/2}(\tilde{u}) &= \frac{1}{2}[Q(\overline{v}_{i+1/2}) + K'\cdot\lambda\frac{O(\Delta^2)}{\Delta_{i+1/2}u} - (\overline{v}_{i+1/2})^2 - 2\overline{v}_{i+1/2}\lambda\frac{O(\Delta^2)}{\Delta_{i+1/2}u} - \frac{O(\Delta^3)}{\Delta_{i+1/2}u}][\Delta_{i+1/2}u + O(\Delta^2)] \\ &= \frac{1}{2}[Q(\overline{v}_{i+1/2}) - (\overline{v}_{i+1/2})^2]\Delta_{i+1/2}u + \lambda\frac{O(\Delta^2)}{\Delta_{i+1/2}u}\cdot\Delta_{i+1/2}u + \lambda\frac{1}{2\lambda}[Q(\overline{v}_{i+1/2}) - (\overline{v}_{i+1/2})^2]O(\Delta^2) + O(\Delta^3) \\ &= \tilde{g}_{i+1/2}(u) + \lambda\cdot O(\Delta^2)\end{aligned}$$

(B-10)

As a result, the errors in $g$ is also $\lambda\cdot O(\Delta^2)$, then the analysis in the $\overline{v}^M_{j+1/2}$ shown by (B-11) can be made.

$$\tilde{\overline{v}}^M_{j+1/2} = \tilde{\overline{v}}_{i+1/2} + \tilde{\gamma}_{i+1/2},\quad \tilde{\gamma}_{i+1/2} = (g_{i+1}(\tilde{u}) - g_i(\tilde{u}))/\Delta_{i+1/2}\tilde{u} \tag{B-11}$$

Meantime, according to the property of $Q$, the accuracy of the item $\frac{1}{\lambda}Q(\overline{v}^M_{j+1/2})\Delta_{i+1/2}u$ can be gotten as well.

$$\tilde{\gamma}_{i+1/2} = (g_{i+1}(\tilde{u}) - g_i(\tilde{u}))/\Delta_{i+1/2}\tilde{u} = [(g_{i+1} - g_i) + \lambda\cdot O(\Delta^2)]\frac{1}{\Delta_{i+1/2}u}\cdot(1 + \frac{O(\Delta^2)}{\Delta_{i+1/2}u}) = \gamma_{i+1/2} + \lambda\cdot\frac{O(\Delta^2)}{\Delta_{i+1/2}u}$$

$$\tilde{\overline{v}}^M_{j+1/2} = \tilde{\overline{v}}_{i+1/2} + \tilde{\gamma}_{i+1/2} = \overline{v}_{i+1/2} + \lambda\frac{O(\Delta^2)}{\Delta_{i+1/2}u} + \gamma_{i+1/2} + \lambda\frac{O(\Delta^2)}{\Delta_{i+1/2}u} = \overline{v}^M_{j+1/2} + \lambda\frac{O(\Delta^2)}{\Delta_{i+1/2}u}$$

(B-12)

$$\begin{aligned}\frac{1}{\lambda}Q(\tilde{\overline{v}}^M_{j+1/2})\Delta_{i+1/2}\tilde{u} &= \frac{1}{\lambda}[Q(\overline{v}^M_{j+1/2}) + \lambda\frac{O(\Delta^2)}{\Delta_{i+1/2}u}][\Delta_{i+1/2}u + O(\Delta^2)] \\ &= \frac{1}{\lambda}Q(\overline{v}^M_{j+1/2})\Delta_{i+1/2}u + O(\Delta^2) + \frac{1}{\lambda}Q(\overline{v}^M_{j+1/2})\cdot O(\Delta^2) + O(\Delta^3) \\ &= \frac{1}{\lambda}Q(\overline{v}^M_{j+1/2})\Delta_{i+1/2}u + O(\Delta^2)\end{aligned} \tag{B-13}$$

The truncation error in the item $F^M_i$ is as (B-14).

$$F^M_i(\tilde{u}) = f(\tilde{u}_i) + \frac{g_i(\tilde{u})}{\lambda} = f(u_i) + f'(u_i)\cdot O(\Delta^2) + O(\Delta^3) + \frac{g_i(u)}{\lambda} + O(\Delta^2) = F^M_i(u) + O(\Delta^2)$$

(B-14)

So, the extra error brought by the linear coarse-to-fine interpolation is,.

$$\begin{aligned}F^{TVNI}_{i+1/2}(\tilde{u}) &= \frac{1}{2}[F^M_i(\tilde{u}) + F^M_{i+1}(\tilde{u}) - \frac{1}{\lambda}Q(\tilde{\overline{v}}^M_{j+1/2})\Delta_{i+1/2}\tilde{u}] = \frac{1}{2}[F^M_i + F^M_{i+1} - \frac{1}{\lambda}Q(\overline{v}^M_{j+1/2})\Delta_{i+1/2}u] + O(\Delta^2) \\ &= F^{TVNI}_{i+1/2}(u) + O(\Delta^2)\end{aligned}$$

(B-15)

By considering the relation between the flux in TVNI solver and the flux in LW scheme, the TVNI flux on finer level satisfy (B-16).

$$F^{TVNI}_{i+1/2}(\tilde{u}) = F^{LW}_{i+1/2}(u) + O(\Delta^2) \tag{B-16}$$

For the conservation laws, the problems can be solve by the scalar TVNI mode scalarly trough some linear transformation [38]. In normal conservation problems, the TVNI flux value of the problem with $m$ variables is as following,

$$\begin{aligned}
&\bar{f}^n_{j+1/2} = \frac{1}{2}[f(u_j) + f(u_{j+1})] + \frac{1}{2\lambda}\sum_{k=1}^{m} R^k_{j+1/2}[g^k_j + g^k_{j+1} - Q^k(v^k_{j+1/2} + \gamma^k_{j+1/2})\alpha^k_{j+1/2}] \\
&v^k_{j+1/2} = \lambda a^k(u_{j+1/2}) \\
&g^k_i = s^k_{i+1/2}\max[0, \min(|\tilde{g}^k_{i+1/2}|, \tilde{g}^k_{i-1/2}\cdot s^k_{i+1/2})] \\
&\tilde{g}^k_{i+1/2} = \frac{1}{2}[Q^k(v^k_{i+1/2}) - (v^k_{i+1/2})^2]\alpha^k_{j+1/2}, \quad s^k_{i+1/2} = sign(\tilde{g}^k_{i+1/2}) \\
&\gamma^k_{j+1/2} = (g^k_{i+1} - g^k_i)/\alpha^k_{j+1/2} \quad when\ \alpha^k_{j+1/2} \neq 0 \\
&\qquad = 0 \qquad\qquad\qquad otherwise \\
&\alpha^k_{j+1/2} = L^k_{j+1/2}\Delta_{j+1/2}u
\end{aligned} \tag{B-17}$$

The $R^k$ is the eigenvectors of the Jacobi matrix $A(u)$. In the linear algebra, it is not difficult to prove one matrix $S(u) = (R^1(u), R^2(u), \cdots, R^m(u))$ to satisfy (B-18).

$$\begin{aligned}
&S(u) = (R^1(u), R^2(u), \cdots, R^m(u)) \\
&S^{-1}(u) = (L^1(u), L^2(u), \cdots, L^m(u)) \\
&S^{-1}AS = \Lambda, \quad \Lambda_{ij} = a^i(u)\delta_j
\end{aligned} \tag{B-18}$$

By considering the format of the gas dynamics matrix [38] and the calculation of inverse matrix, it is easy to prove the linear transformation in Euler problem can all keep the second order accuracy. With the similar truncation error analyses for (B-17), final result (B-19) can be achieved.

$$F^{TVNI}_{i+1/2}(\tilde{u}) = F^{TVNI}_{i+1/2}(u) + O(\Delta^2) = F^{LW}_{i+1/2}(u) + O(\Delta^2) \tag{B-19}$$

Since the initial condition (B-2) considers the truncation error of any intermediate phases, the above proof is suitable for the calculation begin with any intermediate phases. The linear interpolation method can satisfy the requirement of TVNI solver is proved.

In the fine-to-coarse data transfer, the finer data are averaged and filled to the covered coarse region. Since this work is done in the synchronization phase, only the second order errors made by the spatial averaging should be considered. By using the similar analysis, it is clear that the averaging used in fine-to-coarse data transfer can satisfy the TVNI scheme.

In the implementation of LMR in AD based calculation, numerical solution keeps the same calculation order on all the refinement levels. The analyses of the flux correction is based on the calculation process (the refinement ratio is 2) shown by Figure B.1.

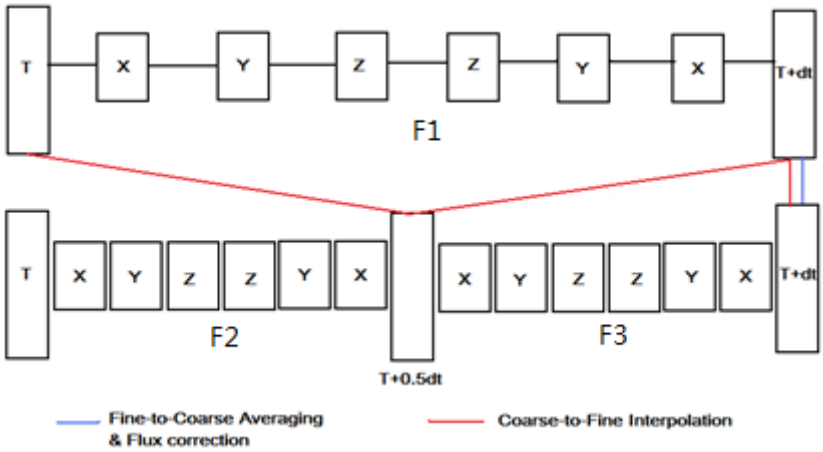



Figure B.1 AD scheme with LMR

In the above computational process, all the alternating direction based TVNI calculations are given in the standard way. For simplicity, only the flux correction in X direction is analyzed in the following part, and the analyses for the other directions can be made in the same way.

By considering (B-19), all the analyses of TVNI solver can be transmitted into the analyses in the fluxes of LW scheme. In the AD scheme, the calculations of the fluxes in X direction are based on $u_n$ and $L_y^{LW}L_z^{LW}L_z^{LW}L_y^{LW}L_x^{LW}u_n$. Firstly, treatments are done to make $L_y^{LW}L_z^{LW}L_z^{LW}L_y^{LW}L_x^{LW}u_n$ more readable.

$$L_y^{LW}L_z^{LW}L_z^{LW}L_y^{LW}L_x^{LW}u_n=(L_x^{LW})^{-1}L_x^{LW}L_y^{LW}L_z^{LW}L_z^{LW}L_y^{LW}L_x^{LW}u_n=(L_x^{LW})^{-1}u_{n+1}$$
$$(L_x^{LW})^{-1}=\frac{I}{L_x^{LW}}=\frac{I}{I+\Delta t/2\cdot(f_x+O(\Delta t))} \tag{B-20}$$

By considering the CFL condition used in the explicit solver, it is quite easy to prove that all the norms of eigenvalues of the $\Delta t\cdot f_x$ are less than 1. The results in (B-20) can be turned into (B-21).

$$(L_x^{LW})^{-1}=\frac{I}{L_x^{LW}}=\frac{I}{I+\Delta t/2\cdot(f_x+O(\Delta t))}=I+\sum_{i=1}^{+\infty}[-\Delta t/2\cdot(f_x+O(\Delta t))]^i=I-\Delta t/2\cdot f_x+O(\Delta t^2) \tag{B-21}$$

As a result, calculations in X direction are based on the data shown by (B-22).

$u_n$

$$(I-\Delta t/2\cdot f_x+O(\Delta t^2))u_{n+1}=(I-\Delta t/2\cdot f_x+O(\Delta t^2))(u_n+\Delta t\cdot f_x+\Delta t\cdot g_y+\Delta t\cdot k_z+O(\Delta t^2))$$
$$=u_n+\Delta t/2\cdot f_x+\Delta t\cdot g_y+\Delta t\cdot k_z+O(\Delta t^2) \tag{B-22}$$

The fluxes in X direction on coarse level can be expressed as (B-23).

$$F(u_n) = f(u_n) + \frac{\Delta t}{4}\frac{\partial f}{\partial u}\frac{\partial f}{\partial x}$$

$$\begin{aligned} F((L_{x_C}^{LW})^{-1}u_{n+1}) &= f(u_n + \Delta t/2 \cdot f_x + \Delta t \cdot g_y + \Delta t \cdot k_z + O(\Delta t^2)) + \frac{\Delta t}{4}\frac{\partial f}{\partial u}(u_n + O(\Delta t))\frac{\partial f}{\partial x}(u_n + O(\Delta t)) \\ &= f(u_n) + \frac{\Delta t}{2}\frac{\partial f}{\partial u}\frac{\partial f}{\partial x} + \Delta t\frac{\partial f}{\partial u}\frac{\partial g}{\partial y} + \Delta t\frac{\partial f}{\partial u}\frac{\partial k}{\partial z} + O(\Delta t^2) + \frac{\Delta t}{4}\frac{\partial f}{\partial u}\frac{\partial f}{\partial x} + O(\Delta t^2) \\ &= f(u_n) + \frac{3\Delta t}{4}\frac{\partial f}{\partial u}\frac{\partial f}{\partial x} + \Delta t\frac{\partial f}{\partial u}\frac{\partial g}{\partial y} + \Delta t\frac{\partial f}{\partial u}\frac{\partial k}{\partial z} + O(\Delta t^2) \end{aligned}$$

(B-23)

Similarly, fluxes in X direction on the finer are shown in (B-24).

$$F(u_n) = f(u_n) + \frac{\Delta t}{8}\frac{\partial f}{\partial u}\frac{\partial f}{\partial x} + O(\Delta^2)$$

$$F((L_{x_F}^{LW})^{-1}u_{n+1/2}) = f(u_n) + \frac{3\Delta t}{8}\frac{\partial f}{\partial u}\frac{\partial f}{\partial x} + \frac{\Delta t}{2}\frac{\partial f}{\partial u}\frac{\partial g}{\partial y} + \frac{\Delta t}{2}\frac{\partial f}{\partial u}\frac{\partial k}{\partial z} + O(\Delta^2)$$

$$\begin{aligned} F(u_{n+1/2}) &= f(u_{n+1/2}) + \frac{\Delta t}{8}\frac{\partial f}{\partial u}(u_{n+1/2})\frac{\partial f}{\partial x}(u_{n+1/2}) + O(\Delta^2) \\ &= f(u_n + \frac{\Delta t}{2}\frac{\partial f}{\partial x} + \frac{\Delta t}{2}\frac{\partial g}{\partial y} + \frac{\Delta t}{2}\frac{\partial k}{\partial z} + O(\Delta t^2)) + \frac{\Delta t}{8}\frac{\partial f}{\partial u}(u_n + O(\Delta t))\frac{\partial f}{\partial x}(u_n + O(\Delta t)) \\ &= f(u_n) + \frac{5\Delta t}{8}\frac{\partial f}{\partial u}\frac{\partial f}{\partial x} + \frac{\Delta t}{2}\frac{\partial f}{\partial u}\frac{\partial g}{\partial y} + \frac{\Delta t}{2}\frac{\partial f}{\partial u}\frac{\partial k}{\partial z} + O(\Delta^2) \end{aligned}$$

$$\begin{aligned} F((L_{x_F}^{LW})^{-1}u_{n+1}) &= f(u_n + \frac{3\Delta t}{4} \cdot f_x + \Delta t \cdot g_y + \Delta t \cdot k_z + O(\Delta t^2)) + \frac{\Delta t}{8}\frac{\partial f}{\partial u}\frac{\partial f}{\partial x}(u_n + O(\Delta t)) \\ &= f(u_n) + \frac{7\Delta t}{8}\frac{\partial f}{\partial u}\frac{\partial f}{\partial x} + \Delta t\frac{\partial f}{\partial u}\frac{\partial g}{\partial y} + \Delta t\frac{\partial f}{\partial u}\frac{\partial k}{\partial z} + O(\Delta^2) \end{aligned}$$

(B-24)

In the comparison between (B-23) and (B-24), it is clear that:

$$\frac{F(u_n) + F((L_{x_F}^{LW})^{-1}u_{n+1/2}) + F(u_{n+1/2}) + F((L_{x_F}^{LW})^{-1}u_{n+1})}{2} = F(u_n) + F((L_{x_C}^{LW})^{-1}u_{n+1}) + O(\Delta t^2)$$

(B-25)

The averaging work in flux correction introduces second order errors to the coarser level in AD based LW scheme. By considering (B-19) again, the flux correction in local mesh refinement can keep the same accuracy as the uniform grid calculation. The flux corrections under other refinement ratios can be analyzed by the same way, and the second order accuracy can be maintained on the coarse level as well.

In all, there is no doubt that all communications used in current LMR can satisfy the requirements of TVNI with AD scheme.

# Appendix C

For the second order RK scheme, the truncation error analyses are as (C-1).

$$\begin{aligned}
&\frac{\partial x}{\partial t} = f(x) \\
&k_1 = f(x_n) = f \\
&k_2 = f(x_n + \Delta t \times k_1) = f(x_n + \Delta t \times f(x_n)) = f + \Delta t \times f_x f + O(\Delta t^2) \\
&x_{n+1} = x_n + \frac{1}{2}\Delta t(k_1 + k_2) = x_n + \Delta t \times f + \frac{\Delta t^2}{2} \times f_x f + O(\Delta t^3)
\end{aligned} \tag{C-1}$$

The second order RK scheme can achieve the same accuracy as the third order Taylor series (C-2) in time.

$$\begin{aligned}
x_{n+1} &= x_n + \Delta t \times \frac{\partial x}{\partial t} + \frac{\Delta t^2}{2} \times \frac{\partial^2 x}{\partial t^2} + O(\Delta t^3) \\
&= x_n + \Delta t \times f(x_n) + \frac{\Delta t^2}{2} \times \frac{\partial f}{\partial x}|_{x_n} f(x_n) + O(\Delta t^3) \\
&= x_n + \Delta t \times f + \frac{\Delta t^2}{2} \times f_x f + O(\Delta t^3)
\end{aligned} \tag{C-2}$$

For the fourth order RK scheme, analyses are much more complicated, but the process is quite similar. The fifth order Taylor series is shown by equation (C-3).

$$\begin{aligned}
x_{n+1} &= x_n + \Delta t \cdot \frac{\partial x}{\partial t} + \frac{\Delta t^2}{2} \cdot \frac{\partial^2 x}{\partial t^2} + \frac{\Delta t^3}{6} \cdot \frac{\partial^3 x}{\partial t^3} + \frac{\Delta t^4}{24} \cdot \frac{\partial^4 x}{\partial t^4} + O(\Delta t^5) \\
&= x_n + \Delta t \cdot f + \frac{\Delta t^2}{2} \cdot f_x f + \frac{\Delta t^3}{6} \cdot (f_{xx} f^2 + f_x^2 f) + \\
&\frac{\Delta t^4}{24} \cdot (f_{xxx} f^3 + f_{xx} f_x f^2 + 2 f_{xx} f f_x f + f_x f_{xx} f^2 + f_x^3 f) + O(\Delta t^5)
\end{aligned} \tag{C-3}$$

Truncation error analyses for the four intermediate phases in RK4 are given in (C-4).

$$\begin{aligned}
&x_1 = x_n + \Delta t \cdot f(x_n) \\
&x_2 = x_n + \Delta t \cdot f(x_n + \frac{\Delta t}{2} \cdot f(x_n)) \\
&= x_n + \Delta t \cdot [f(x_n) + \frac{\Delta t}{2} \cdot f_x f(x_n) + \frac{1}{2} f_{xx} (\frac{\Delta t}{2} \cdot f(x_n))^2 + \frac{1}{6} f_{xxx} (\frac{\Delta t}{2} \cdot f(x_n))^3 + O(\Delta t^4)] \\
&= x_n + \Delta t \cdot f + \frac{\Delta t^2}{2} \cdot f_x f + \frac{\Delta t^3}{8} f_{xx} f^2 + \frac{\Delta t^4}{48} f_{xxx} f^3 + O(\Delta t^5)
\end{aligned}$$

$$
\begin{aligned}
x_3 &= x_n + \Delta t \cdot f(x_n + \frac{\Delta t}{2} \cdot [f(x_n) + \frac{\Delta t}{2} \cdot f_x f(x_n) + \frac{1}{2} f_{xx} (\frac{\Delta t}{2} \cdot f(x_n))^2 + \frac{1}{6} f_{xxx} (\frac{\Delta t}{2} \cdot f(x_n))^3]) \\
&= x_n + \Delta t \cdot [f(x_n) + \frac{\Delta t}{2} \cdot f_x (f + \frac{\Delta t}{2} \cdot f_x f + \frac{1}{2} f_{xx} (\frac{\Delta t}{2} \cdot f)^2) + \frac{1}{2} f_{xx} (\frac{\Delta t^2}{4} (f^2 + \frac{\Delta t}{2} \cdot f f_x f + \frac{\Delta t}{2} \cdot f_x f^2)) + \\
&\quad \frac{1}{6} f_{xxx} (\frac{\Delta t}{2} \cdot f(x_n))^3] \\
&= x_n + \Delta t \cdot f + \frac{\Delta t^2}{2} \cdot f_x f + \frac{\Delta t^3}{4} \cdot f_x^2 f + \frac{\Delta t^3}{8} \cdot f_{xx} f^2 + \frac{\Delta t^4}{16} (f_{xx} f f_x f + f_{xx} f_x f^2 + f_x f_{xx} f^2) + \frac{\Delta t^4}{48} \cdot f_{xxx} f^3 + O(\Delta t^5) \\
x_4 &= x_n + \Delta t \cdot f(x_3) \\
&= x_n + \Delta t \cdot [f + f_x (\Delta t \cdot f + \frac{\Delta t^2}{2} \cdot f_x f + \frac{\Delta t^3}{4} \cdot f_x^2 f + \frac{\Delta t^3}{8} \cdot f_{xx} f^2) + \frac{1}{2} f_{xx} (\Delta t \cdot f + \frac{\Delta t^2}{2} \cdot f_x f)^2 + \\
&\quad \frac{1}{6} f_{xxx} (\Delta t \cdot f)^3] \\
&= x_n + \Delta t \cdot f + \Delta t^2 \cdot f_x f + \frac{\Delta t^3}{2} \cdot f_x^2 f + \frac{\Delta t^3}{2} \cdot f_{xx} f^2 + \frac{\Delta t^4}{4} \cdot f_x^3 f + \frac{\Delta t^4}{8} \cdot f_x f_{xx} f^2 + \frac{\Delta t^4}{4} f_{xx} f f_x f + \\
&\quad \frac{\Delta t^4}{4} f_{xx} f_x f^2 + \frac{\Delta t^4}{6} \cdot f_{xxx} f^3 + O(\Delta t^5)
\end{aligned}
\tag{C-4}
$$

In the software package COM3D, four intermediate phases are combined with the coefficients 1/6, 1/3, 1/3 and 1/6. The result on the new time step n+1 achieved by the RK4 scheme is (C-5).

$$
\begin{aligned}
x_{n+1} &= \frac{1}{6} x_1 + \frac{1}{3} x_2 + \frac{1}{3} x_3 + \frac{1}{6} x_4 \\
&= x_n + \Delta t \cdot f + \frac{\Delta t^2}{2} \cdot f_x f + \frac{\Delta t^3}{6} \cdot (f_{xx} f^2 + f_x^2 f) + \\
&\quad \frac{\Delta t^4}{24} \cdot (f_{xxx} f^3 + f_{xx} f_x f^2 + 2 f_{xx} f f_x f + f_x f_{xx} f^2 + f_x^3 f) + O(\Delta t^5)
\end{aligned}
\tag{C-5}
$$

So, the RK4 scheme can have the same expression as the 5$^{th}$ order Taylor series in time, which means that 5$^{th}$ order temporal accuracy can be achieved in each single step of calculation.

# Appendix D

Differently from the truncation error analyses in Appendix B, the RK scheme is used to improve the temporal accuracy and has little influence to the spatial accuracy, the analyses for the implementation of LMR in RK schemes only focus on the temporal accuracy.

The analyses for RK2 are given first. In the implementation of LMR in RK2, the truncation errors brought by the interpolation come from the initial state of each step. As the parabolic interpolation is used in time to achieve higher accuracy, initial conditions of each step on finer level are more accurate (only considering the temporal truncation error).

$$\tilde{x}_i = \begin{cases} x_i & i \in \Omega \\ x_i + O(\Delta t^3) & otherwise \end{cases} \tag{D-1}$$

With the initial condition (D-1), the truncation errors of two intermediate phases and final result are,

$$\begin{aligned} k_1 &= \begin{cases} f(x_n) = f & x_n \in \Omega(-2) \\ f(x_n + O(\Delta t^3)) = f + O(\Delta t^2) & x_n \in \Omega(2)/\Omega(-2) \end{cases} \\ k_2 &= \begin{cases} f(x_n + 0.5\Delta t \times f) = f + \Delta t \times f_x f + O(\Delta t^2) & x_n \in \Omega(-4) \\ f(x_n + 0.5\Delta t(f + O(\Delta t^2))) = f + \Delta t \times f_x f + O(\Delta t^2) & x_n \in \Omega/\Omega(-4) \end{cases} \\ x_{n+1} &= x_n + \frac{1}{2}\Delta t(k_1 + k_2) = x_n + \Delta t \times f + \frac{\Delta t^2}{2} \times f_x f + O(\Delta t^3) \end{aligned} \tag{D-2}$$

In (D-1) and (D-2), $\Omega$ is the internal region of the patch and the $\Omega(N)$ means area achieved by increasing N cells in all directions in $\Omega$. The analyses for the second order RK is very simple, it is clear that the third order accuracy of RK2 can be maintained in the implementation of LMR. The fine-to-coarse data transfer does not bring any temporal errors, detailed analyses can be avoided here. The truncation error brought by the flux correction in RK2 is discussed latterly with the flux correction in RK4.

For four layers of ghost cells are located in each finer patch and the ghost region calculation are used in the computation, the truncation errors of the first and second intermediate phases in RK 4 are the same as the uniform case if $x_n$ is in the same physical time as the next coarser level data.

$$\begin{aligned} x_1 &= x_n + \Delta t \cdot f(x_n) \\ x_2 &= x_n + \Delta t \cdot f + \frac{\Delta t^2}{2} \cdot f_x f + \frac{\Delta t^3}{8} f_{xx} f^2 + \frac{\Delta t^4}{48} f_{xxx} f^3 + O(\Delta t^5) \end{aligned} \tag{D-3}$$

After the calculation for the second intermediate phase, the ghost cells on finer level should be updated by parabolic interpolation. Then, the data in the intermediate phase 2 is,

$$x_2 = \begin{cases} x_n + \Delta t \cdot f + \frac{\Delta t^2}{2} \cdot f_x f + \frac{\Delta t^3}{8} f_{xx} f^2 + \frac{\Delta t^4}{48} f_{xxx} f^3 + O(\Delta t^5) & x_2 \in \Omega \\ x_n + \Delta t \cdot f + \frac{\Delta t^2}{2} \cdot f_x f + O(\Delta t^3) & otherwise \end{cases} \tag{D-4}$$

Then, truncation errors of the intermediate phase 3 and 4 are,

$$x_3=\begin{cases} x_n+\Delta t\cdot f+\dfrac{\Delta t^2}{2}\cdot f_x f+\dfrac{\Delta t^3}{4}\cdot f_x^2 f+\dfrac{\Delta t^3}{8}\cdot f_{xx}f^2+\dfrac{\Delta t^4}{16}\cdot(f_{xx}f_x f^2+f_{xx}ff_x f+f_x f_{xx}f^2)+\dfrac{\Delta t^4}{48}\cdot f_{xxx}f^3 \\ +O(\Delta t^5) & x_3\in\Omega(-2) \\ x_n+\Delta t\cdot f+\dfrac{\Delta t^2}{2}\cdot f_x f+O(\Delta t^3) & otherwise \end{cases}$$

$$x_4=\begin{cases} x_n+\Delta t\cdot f+\Delta t^2\cdot f_x f+\dfrac{\Delta t^3}{2}\cdot f_x^2 f+\dfrac{\Delta t^3}{2}\cdot f_{xx}f^2+\Delta t^4(\dfrac{f_x^3 f}{4}+\dfrac{f_x f_{xx}f^2}{8}+\dfrac{f_{xx}f_x f^2+f_{xx}ff_x f}{4}+\dfrac{f_{xxx}f^3}{6}) \\ +O(\Delta t^5) & x_4\in\Omega(-4) \\ x_n+\Delta t\cdot f+\Delta t^2\cdot f_x f+O(\Delta t^3) & otherwise \end{cases}$$

(D-5)

Combined with the proper coefficients, the final result is,

$$x_{n+1}=\begin{cases} x_n+\Delta t\cdot f+\dfrac{\Delta t^2}{2}\cdot f_x f+\dfrac{\Delta t^3}{6}\cdot(f_{xx}f^2+f_x^2 f)+\dfrac{\Delta t^4}{24}\cdot(f_x f_{xx}f^2+f_{xx}f_x f^2+f_{xx}ff_x f+f_{xxx}f^3+f_x^3 f) \\ +O(\Delta t^5) & x_2\in\Omega(-4) \\ x_n+\Delta t\cdot f+\dfrac{\Delta t^2}{2}\cdot ff_x+O(\Delta t^3) & otherwis \end{cases}$$

(D-6)

When the $x_n$ does not correspond to any coarse level data in time, the ghost cells may contained the third order temporal truncation errors at the very beginning.

$$\tilde{x}_i=\begin{cases} x_i & i\in\Omega \\ x_i+O(\Delta t^3) & otherwise \end{cases} \quad \text{(D-7)}$$

By using the same analyses, it can also be proved that some cells may reduce to third order accuracy in time.

Therefore, the 4th order RK4 scheme is reduce to a second order algorithm. Similarly, the analyses can be made as well for another implementation of LMR in RK4 [55], in which the uniform time step and simplified coarse-to-fine interpolation are used. Detailed analyses for all intermediate phases and final result are given by (D-8).

$$x_1=\begin{cases} x_n+\Delta t\cdot f(x_n) & x_1\in\Omega \\ x_n+\Delta t\cdot f(x_n)+O(\Delta t^2) & otherwise \end{cases}$$

$$x_2=\begin{cases} x_n+\Delta t\cdot f+\dfrac{\Delta t^2}{2}\cdot f_x f+\dfrac{\Delta t^3}{8}f_{xx}f^2+\dfrac{\Delta t^4}{48}f_{xxx}f^3+O(\Delta t^5) & x_2\in\Omega(-2) \\ x_n+\Delta t\cdot f+O(\Delta t^2) & otherwise \end{cases}$$

$$x_3=\begin{cases} x_n+\Delta t\cdot f+\dfrac{\Delta t^2}{2}\cdot f_x f+\dfrac{\Delta t^3}{4}\cdot f_x^2 f+\dfrac{\Delta t^3}{8}\cdot f_{xx}f^2+\dfrac{\Delta t^4}{16}\cdot(f_x f_{xx}f^2+f_{xx}f_x f^2+f_{xx}ff_x f)+\dfrac{\Delta t^4}{48}\cdot f_{xxx}f^3 \\ +O(\Delta t^5) & x_3\in\Omega(-4) \\ x_n+\Delta t\cdot f+O(\Delta t^2) & otherwise \end{cases}$$

$$x_4=\begin{cases} x_n+\Delta t\cdot f+\Delta t^2\cdot ff_x+\dfrac{\Delta t^3}{2}\cdot ff_x^2+\dfrac{\Delta t^3}{2}\cdot f^2 f_{xx}+\Delta t^4(\dfrac{ff_x^3}{4}+\dfrac{f_x f_{xx}f^2}{8}+\dfrac{f_{xx}f_x f^2+f_{xx}ff_x f}{4}+\dfrac{f_{xxx}f^3}{6}) \\ +O(\Delta t^5) & x_4\in\Omega(-6) \\ x_n+\Delta t\cdot f+O(\Delta t^2) & otherwise \end{cases}$$

$$x_{n+1}=\begin{cases} x_n+\Delta t\cdot f+\dfrac{\Delta t^2}{2}\cdot f_x f+\dfrac{\Delta t^3}{6}\cdot(f_{xx}f^2+f_x^2 f)+\dfrac{\Delta t^4}{24}\cdot(f_x f_{xx}f^2+f_{xx}f_x f^2+f_{xx}ff_x f+f_{xxx}f^3+f_x^3 f) \\ +O(\Delta t^5) & x_2\in\Omega(-6) \\ x_n+\Delta t\cdot f+O(\Delta t^2) & otherwis \end{cases}$$

(D-8)

Equations in (D-8) show that the RK4 algorithm is reduced to first order in time, which is worse than the implementation in COM3D. As mentioned in Chapter 2, each intermediate phase is not the simple first order result. Because of lacking the understanding of the phases in RK4 scheme, the temporal accuracy is lost seriously.

Besides the truncation errors on the finer levels, the truncation errors brought by the flux correction on the coarse level should also be analyzed. The truncation errors of the flux correction in RK2 and RK4 can be discussed together, for both two methods show third order accuracy in the regions which are adjacent to the fine-to-coarse interface. Normally, the van Leer solver is used through the RK4 scheme in COM3D, the discussion about the flux correction can be made based on this solver. The van Leer solver is shown as following.

$$\begin{aligned} &\frac{du_i}{dt}+\frac{1}{\Delta x}[F(u_{i+\frac{1}{2}})-F(u_{i-\frac{1}{2}})]=0 \\ &F(u_{i+\frac{1}{2}})=f^{low}_{i+\frac{1}{2}}-\Phi(r_i)(f^{low}_{i+\frac{1}{2}}-f^{high}_{i+\frac{1}{2}}) \\ &F(u_{i-\frac{1}{2}})=f^{low}_{i-\frac{1}{2}}-\Phi(r_{i-1})(f^{low}_{i-\frac{1}{2}}-f^{high}_{i-\frac{1}{2}}) \\ &r_i=\frac{u_i-u_{i-1}}{u_{i+1}-u_i},\quad \Phi(r)=\frac{r+|r|}{1+r} \end{aligned}$$

(D-9)

In (D-9), $f^{low}$ indicates the low precision flux, $f^{high}$ is the high precision flux, $r$ is the ratio of successive gradients on the solution mesh and function $\Phi(r)$ is the flux limiter.

In the van Leer solver, accuracy of the flux may be different for the differences of gradients on the solution mesh. Similarly to the TVNI solver, the van Leer flux limiter can provide the second order accuracy in space generally. The flux in the van Leer solver can provide the accuracy as (the small $f$ indicate the real value of the solution)

$$F_{i+\frac{1}{2}} = f_{i+\frac{1}{2}} + O(\Delta x^2) \tag{D-10}$$

Similarly to the AD based TVNI case, the analysis of the fluxes here focuses on if the second order accuracy can be maintained. The temporal accuracy in the finer cells which are adjacent to the fine-to-coarse interface is second order, the flux values of those cells (take the flux in the direction X as an example) are,

$$F_F = f + \frac{\Delta t_F}{2} \cdot (\frac{\partial f}{\partial u}\frac{\partial f}{\partial x} + \frac{\partial f}{\partial u}\frac{\partial g}{\partial y} + \frac{\partial f}{\partial u}\frac{\partial k}{\partial z}) + O(\Delta x^2) + \mathrm{O}(\Delta t^2) \tag{D-11}$$

The consideration of the second order mode of the flux on the next coarser level is enough, because the finer fluxes can only maintain second order accuracy.

$$F_C = f + \frac{\Delta t}{2} \cdot (\frac{\partial f}{\partial u}\frac{\partial f}{\partial x} + \frac{\partial f}{\partial u}\frac{\partial g}{\partial y} + \frac{\partial f}{\partial u}\frac{\partial k}{\partial z}) + O(\Delta x^2) + \mathrm{O}(\Delta t^2) \tag{D-12}$$

For any refinement ratio N, the flux on the finer level at the step k is,

$$F_F^k = f + \frac{\Delta t(2k-1)}{2N} \cdot (\frac{\partial f}{\partial u}\frac{\partial f}{\partial x} + \frac{\partial f}{\partial u}\frac{\partial g}{\partial y} + \frac{\partial f}{\partial u}\frac{\partial k}{\partial z}) + O(\Delta x^2) + \mathrm{O}(\Delta t^2) \tag{D-13}$$

The average of all the fluxes on the finer level in X direction is,

$$\begin{aligned} F_F^{average} &= \frac{\sum_{k=1}^{n} F_F^k}{N} = \frac{\sum_{k=1}^{n} (f + \frac{\Delta t(2k-1)}{2N} \cdot (\frac{\partial f}{\partial u}\frac{\partial f}{\partial x} + \frac{\partial f}{\partial u}\frac{\partial g}{\partial y} + \frac{\partial f}{\partial u}\frac{\partial k}{\partial z}) + O(\Delta x^2) + \mathrm{O}(\Delta t^2))}{N} \\ &= f + N \cdot \frac{2N-1+1}{2N \cdot 2N} \Delta t \cdot (\frac{\partial f}{\partial u}\frac{\partial f}{\partial x} + \frac{\partial f}{\partial u}\frac{\partial g}{\partial y} + \frac{\partial f}{\partial u}\frac{\partial k}{\partial z}) + O(\Delta x^2) + \mathrm{O}(\Delta t^2) \\ &= F_C + O(\Delta x^2) + \mathrm{O}(\Delta t^2) \end{aligned} \tag{D-14}$$

In other directions' flux corrections, the similar results can be gotten as well. So, after the flux correction on the coarse level, data on the coarse level show second order accuracy.

# Appendix E

The truncation error of molecular transportation part has been discussed in section 3.3.1 and the analyses for the convection have been established in the Appendix B, the truncation errors analysis in Appendix E is focused on the flux correction of the molecular transportation part.

Molecular transportation is expressed as the first order derivative terms shown by (E-1).

$$V(u)=\frac{\partial f(u)}{\partial x}+\frac{\partial g(u)}{\partial y}+\frac{\partial k(u)}{\partial z} \tag{E-1}$$

Following analysis of the flux correction is focused on the X direction, and the arbitrary refinement ratio N is considered. Since the high order interpolation and flux limiter are used in the solution of NS equations, second order accuracy can be guaranteed in the flux values on both coarse and finer levels. On the coarse level, the calculation of molecular transportation part are based on the data $U_C^1=L_z^{\Delta t/2}L_y^{\Delta t/2}L_x^{\Delta t/2}u_n$ and $U_C^2=L_V^{\Delta t/2}L_z^{\Delta t/2}L_y^{\Delta t/2}L_x^{\Delta t/2}u_n$. The analyses for the coarse fluxes are done in (E-2).

$$U_C^1=L_z^{\Delta t/2}L_y^{\Delta t/2}L_x^{\Delta t/2}u_n=u_n+\frac{\Delta t}{2}(X+Y+Z)+O(\Delta t^2)$$

$$U_C^2=L_V^{\Delta t/2}L_z^{\Delta t/2}L_y^{\Delta t/2}L_x^{\Delta t/2}u_n=u_n+\frac{\Delta t}{2}(X+Y+Z+V)+O(\Delta t^2)$$

$$\begin{aligned}F_X^{C,1}&=f(U_C^1)+\frac{\Delta t}{4}(f_u(U_C^1)f_x(U_C^1)+f_u(U_C^1)g_y(U_C^1)+f_u(U_C^1)k_z(U_C^1))\\&=(f+\frac{\Delta t}{4}(f_uf_x+f_ug_y+f_uk_z))\big|_{u_n+\frac{\Delta t}{2}(X+Y+Z)}+O(\Delta t^2)\\&=f+\frac{\Delta t}{4}f_u(2X+2Y+2Z+V)+O(\Delta t^2)\end{aligned} \tag{E-2}$$

$$\begin{aligned}F_X^{C,1}&=f(U_C^2)+\frac{\Delta t}{4}(f_u(U_C^2)f_x(U_C^2)+f_u(U_C^2)g_y(U_C^2)+f_u(U_C^2)k_z(U_C^2))\\&=(f+\frac{\Delta t}{4}(f_uf_x+f_ug_y+f_uk_z))\big|_{u_n+\frac{\Delta t}{2}(X+Y+Z+V)}+O(\Delta t^2)\\&=f+\frac{\Delta t}{4}f_u(2X+2Y+2Z+3V)+O(\Delta t^2)\end{aligned}$$

On the finer refinement level, the calculation of molecular transportation in the small time step k is based on the data $U_F^{1,k}=L_z^{\Delta t/2N}L_y^{\Delta t/2N}L_x^{\Delta t/2N}u_{n+k}$ and $U_F^{2,k}=L_V^{\Delta t/2N}L_z^{\Delta t/2N}L_y^{\Delta t/2N}L_x^{\Delta t/2N}u_{n+k}$. After proper analyses, they can be transmitted into the formats shown in (E-3),

$$
\begin{aligned}
U_F^{1,k} &= L_z^{\Delta t/2N} L_y^{\Delta t/2N} L_x^{\Delta t/2N} u_{n+k} \\
&= u_{n+k} + \frac{\Delta t}{2N}(X+Y+Z)|_{u_{n+k}} + O(\Delta t^2) \\
&= u_n + \frac{k\Delta t}{N}(X+Y+Z+V) + \frac{\Delta t}{2N}(X+Y+Z) + O(\Delta t^2) \\
&= u_n + \frac{2k+1\Delta t}{2N}(X+Y+Z) + \frac{k\Delta t}{N}V + O(\Delta t^2)
\end{aligned}
$$

$$
\begin{aligned}
U_F^{2,k} &= L_z^{\Delta t/2N} L_z^{\Delta t/2N} L_y^{\Delta t/2N} L_x^{\Delta t/2N} u_{n+k} \\
&= u_{n+k} + \frac{\Delta t}{2N}(X+Y+Z+V)|_{n+k} + O(\Delta t^2) \\
&= u_n + \frac{2k+1\Delta t}{2N}(X+Y+Z+V) + O(\Delta t^2)
\end{aligned}
$$

(E-3)

The flux values of small step k on the finer level and the average values of all the fine fluxes in X direction are shown in (E-4).

$$
\begin{aligned}
F_{X,1}^{F,k} &= f(U_F^{1,k}) + \frac{\Delta t}{4N}(f_u(U_F^{1,k}) f_x(U_F^{1,k}) + f_u(U_F^{1,k}) g_y(U_F^{1,k}) + f_u(U_F^{1,k}) k_z(U_F^{1,k})) + O(\Delta^2) \\
&= (f + \frac{\Delta t}{4N}(f_u f_x + f_u g_y + f_u k_z))|_{u_n + \frac{(2k+1)\Delta t}{2N}(X+Y+Z) + \frac{k\Delta t}{N}V} + O(\Delta^2) \\
&= f + \frac{(2k+1)\Delta t}{2N} f_u (X+Y+Z) + \frac{k\Delta t}{N} f_u V + \frac{\Delta t}{4N} f_u V + O(\Delta^2) \\
&= f + \frac{(2k+1)\Delta t}{2N} f_u (X+Y+Z) + \frac{(4k+1)\Delta t}{4N} f_u V + O(\Delta^2)
\end{aligned}
$$

$$
\begin{aligned}
F_{X,2}^{F,k} &= f(U_F^{2,k}) + \frac{\Delta t}{4N}(f_u(U_F^{2,k}) f_x(U_F^{2,k}) + f_u(U_F^{2,k}) g_y(U_F^{2,k}) + f_u(U_F^{2,k}) k_z(U_F^{2,k})) + O(\Delta^2) \\
&= (f + \frac{\Delta t}{4N}(f_u f_x + f_u g_y + f_u k_z))|_{u_n + \frac{(2k+1)\Delta t}{2N}(X+Y+Z+V)} + O(\Delta t^2) \\
&= f + \frac{(2k+1)\Delta t}{2N} f_u (X+Y+Z+V) + \frac{\Delta t}{4N} f_u V + O(\Delta t^2) \\
&= f + \frac{(2k+1)\Delta t}{2N} f_u (X+Y+Z) + \frac{(4k+3)\Delta t}{4N} f_u V + O(\Delta t^2)
\end{aligned}
$$

$$
F_X^{F,k} = F_{X,1}^{F,k} + F_{X,2}^{F,k} = 2f + \frac{(2k+1)\Delta t}{N} f_u (X+Y+Z+V) + O(\Delta t^2)
$$

$$
F_X^{C,1} + F_X^{C,2} = 2f + f_u (X+Y+Z+V) + O(\Delta t^2)
$$

$$
\Sigma_{k=0}^{N-1} F_X^{F,k} / N = 2f + f_u (X+Y+Z+V) + O(\Delta^2)
$$

$$
\Sigma_{k=0}^{N-1} F_X^{F,k} / N = F_X^{C,1} + F_X^{C,2} + O(\Delta^2)
$$

(E-4)

The above analyses show that the second order accuracy can be kept in calculation of molecular transportation. In the whole AD based calculation, the implementation of local mesh refinement can keep the second order accuracy in general.

# Appendix F

**Truncation errors of further decoupled detonation simulation:**

For convenience, the analyses for the second order scheme are made here only. The mathematical analysis of the second order scheme is begun with the step n, the truncation error analyses for n+1 and n+2 are shown by (F-1) (only focusing on temporal error ).

$$\begin{aligned} U_{n+1} &= U_n + \Delta t \cdot G + \Delta t^2 / 2 \cdot G_U G + O(\Delta t^3) \\ U_{n+2} &= U_{n+1} + \Delta t \cdot G(U_{n+1}) + \Delta t^2 / 2 \cdot G_U(U_{n+1}) G(U_{n+1}) + O(\Delta t^3) \\ &= U_n + \Delta t \cdot G + \Delta t^2 / 2 \cdot G_U G + \Delta t \cdot G(U_n + \Delta t \cdot G + O(\Delta t^2)) + \\ &\quad \Delta t^2 / 2 \cdot G_U(U_n + O(\Delta t)) G(U_n + O(\Delta t)) + O(\Delta t^3) \\ &= U_n + 2\Delta t \cdot G + 2\Delta t^2 \cdot G_U G + O(\Delta t^3) \end{aligned} \tag{F-1}$$

Assuming the result at the step n+k has the format (F-2).

$$U_{n+k} = U_n + k\Delta t \cdot G + k^2 \Delta t^2 / 2 \cdot G_U G + O(\Delta t^3) \tag{F-2}$$

Truncation errors at the step n+k+1 are shown in (F-3).

$$\begin{aligned} U_{n+k+1} &= U_{n+k} + \Delta t \cdot G(U_{n+k}) + \Delta t^2 / 2 \cdot G_U(U_{n+k}) G(U_{n+k}) + O(\Delta t^3) \\ &= U_n + k\Delta t \cdot G + k^2 \Delta t^2 / 2 \cdot G_U G + \Delta t \cdot G(U_n + k\Delta t \cdot G + O(\Delta t^2)) + \\ &\quad \Delta t^2 / 2 \cdot G_U(U_n + O(\Delta t)) G(U_n + O(\Delta t)) + O(\Delta t^3) \\ &= U_n + (k+1)\Delta t \cdot G + (k+1)^2 \Delta t^2 / 2 \cdot G_U G + O(\Delta t^3) \end{aligned} \tag{F-3}$$

As a result, the total truncation error of the k+1 steps' calculation is still third order in time. Similar analyses can be made for the first order scheme, and the result is shown in (F-4).

$$U_{n+k} = U_n + k\Delta t \cdot G + O(\Delta t^2) \tag{F-4}$$

After one step of chemical calculation with time span $k \times \Delta t$ , the final result is:

$$\begin{aligned} U_{new} &= U_{n+k} + k \times \Delta t \times C(U_{n+k}) + O(\Delta t^2) \\ &= U_n + k\Delta t \cdot G + k \times \Delta t \times C(U_n + k\Delta t \cdot G + O(\Delta t^2)) + O(\Delta t^2) \\ &= U_n + k\Delta t \cdot G + k\Delta t \cdot C + (k \times \Delta t)^2 \cdot C_U(U_n) G + O(\Delta t^2) \\ &= U_n + k\Delta t \cdot (G(U_n) + C(U_n)) + O((k \times \Delta t)^2) \end{aligned} \tag{F-5}$$

So, the detonation simulation with further operator splitting treatment can maintain the second order accuracy in time-scale.

**Analyses for the second order detonation simulation:**

In above, the accuracy of the gas part after k step of second order calculation has been analyzed.

$$U_{n+k} = U_n + k\Delta t \cdot G + k^2 \Delta t^2 / 2 \cdot G_U G + O(\Delta t^3) \tag{F-6}$$

The process of the scheme shown by Figure 3.41 is $U_{N+1} = L_C^{\Delta t/2} L_G^{\Delta t} L_C^{\Delta t/2} U_{N+1}$, so the calculation contains two intermediate phases $U_A = L_C^{\Delta t/2} U_{N+1}$ and $U_B = L_G^{\Delta t} L_C^{\Delta t/2} U_{N+1}$. Analyses for the two intermediate phases and the final result are done in (F-7).

$$U_A = U_n + \frac{\Delta t}{2}\cdot C(U_N) + \frac{\Delta t^2}{8} C_U(U_N)C(U_N)$$

$$\begin{aligned} U_B &= U_A + \Delta t \cdot G(U_A) + \frac{\Delta t^2}{2} G_U(U_A)G(U_A) + O(\Delta t^3) \\ &= U_N + \frac{\Delta t}{2}\cdot C(U_N) + \frac{\Delta t^2}{8} C_U(U_N)C(U_N) + \Delta t \cdot G(U_N + \frac{\Delta t}{2}\cdot C(U_N)) + \frac{\Delta t^2}{2} G_U(U_N)G(U_N) + O(\Delta t^3) \\ &= U_N + \Delta t \cdot (\frac{1}{2} C(U_N) + G(U_N)) + \frac{\Delta t^2}{2}(\frac{1}{4} C_U(U_N)C(U_N) + G_U(U_N)C(U_N) + G_U(U_N)G(U_N)) + O(\Delta t^3) \end{aligned}$$

$$\begin{aligned} U_{N+1} &= U_B + \frac{\Delta t}{2}\cdot C(U_B) + \frac{\Delta t^2}{8} C_U(U_B)C(U_B) \\ &= U_N + \Delta t \cdot (\frac{1}{2} C + G) + \frac{\Delta t^2}{2}(\frac{1}{4} C_U C + G_U C + G_U G) + \frac{\Delta t}{2} C + \frac{\Delta t^2}{2} C_U(\frac{1}{2} C + G) + \frac{\Delta t^2}{8} C_U C + O(\Delta t^3) \\ &= U_N + \Delta t \cdot (C + G) + \frac{\Delta t^2}{2}(C_U C + G_U C + C_U G + G_U G) + O(\Delta t^3) \end{aligned}$$

(F-7)

Since the calculation of chemistry is point-wised in detonation, the above work can stand for the analyses on all the refinement levels. It is clear that third order accuracy can be achieved in time in general by the method shown in Figure 3.41.

ISSN 1869-9669
ISBN 978-3-7315-0225-8

ISBN 978-3-7315-0225-8