

博士論文

Improving Accuracy of Building-Cube Method Using
Upwind Interpolation at Hanging Node Interfaces in
Compressible Euler Equation Simulations

(圧縮性オイラー方程式シミュレーションのハンギング
ノードインターフェースでの風上補間を使ったビルディン
グキューブ法の精度向上について)

Surendranath Srikanth

スレンドラナート スリカンス

Acknowledgement

I would like to express my sincere gratitude towards my supervisor **Dr. Akira Oyama**, Department of Space Flight Systems, Institute of Space and Astronautical Science, JAXA, for his guidance during the research work. I am thankful to him for giving me an opportunity to do research under him.

I am grateful to Prof. Ryoji Takaki, JAXA for his invaluable support and guidance for my research and also to Prof. Kenichi Rinoie, Prof. Katsuhiro Nishinari and Prof. Taro Imamura for all being a part of my Doctoral Thesis evaluation committee and for their valuable suggestions.

I would like to thank my senior Dr. Hiroaki Fukumoto for all his help regarding my stay and life in Japan. I would also like to thank all Oyama Lab members for their direct and indirect support for my research life.

I am deeply indebted to my friend Arjun John Kaithakkal, who has been a pillar of support in my difficult times. No words can describe my gratitude to Mrs. and Mr. Shibata for their affection and care and in making my stay in Japan a memorable one.

My hearty gratitude goes to MEXT, Japan for providing me with a scholarship and also to MHRD India for giving me an opportunity to study in Japan. Lastly, and certainly not the least, my highest gratitude and love goes to my parents **M. Saroja** and **P. A. Surendranath** who have borne sacrifices for fulfilling my dreams.

I present my research work at the Lotus feet of the Supreme Lord Vishnu and his eternal consort Goddess Lakshmi Devi.

Srikanth Surendranath

Abstract

Cartesian grids are gaining importance because of the fewer efforts to be invested in different aspects of simulation like grid generation, higher order scheme implementation and post-processing. Simulations which use block structured Cartesian grid approach and where the entire domain consists of different sized cubes/squares with the same number of cells in each cube/square are the Building-Cube Method (BCM) simulations. The presence of adjacent different sized cubes/squares creates a hanging node interface between them, and the cubes/squares in such a domain mutually exchange information with the help of their ghost cells. This research work is being carried out to propose a new interpolation method for obtaining ghost cell values to reduce the bad effect of hanging nodes in the simulation and to confirm its performance by conducting simulations of

- sine wave propagation using two-dimensional wave equation with one-dimensional hanging node interface,
- vortex convection using two-dimensional isentropic vortex equations with one-dimensional and two-dimensional hanging node interfaces,
- and shock wave propagation in a two-dimensional shock tube with two-dimensional hanging node interface.

Roe scheme with second-order MUSCL extrapolation and Roe scheme with fifth-order WENO extrapolation are used in this research. In all the simulations, the entire computational domain is divided into regions having different grid spacing. This is done to analyze the flow field when the fluid flows from a fine to a coarse region. The two ways of interpolation used are described below.

The standard way of interpolating values at ghost cells is referred to as standard interpolation in this research. The standard interpolation used for ghost cells of fine region at the hanging node interface is a direct transfer of value from the coarse cell to the fine ghost cells. The standard interpolation

used for ghost cells of coarse region cells is a simple averaging of all values in the fine cells that the coarse ghost cell encloses.

The new upwind way of interpolation proposed here uses the characteristic equations at the hanging node interface. Eigenvalues and eigenvectors are used to determine the flow characteristics and then for transforming the primitive values to characteristic variable values before transferring it to ghost cells either directly or by using higher-order interpolations. Then the characteristic values are transformed back to primitive values for using in simulation calculations.

In sine wave propagation simulation with a simple sine wave travelling from fine to coarse region in the domain, a reflected wave is observed to the left of the one-dimensional hanging node interface while using standard interpolation. This reflected wave is diminished by using a new upwind S2L interpolation, thereby maintaining the magnitude of the original propagating sine wave. From the L2 error values and plots, it is confirmed that upwind S2L interpolation performs better than standard interpolation.

For isentropic vortex convection, separate cases of computational domain having one-dimensional hanging node interface and a computational domain having two-dimensional hanging node interfaces are both simulated. It is observed that using upwind interpolation generates less error in the domain than while using standard interpolation in both these cases. This is also confirmed from error in pressure contour plots, L2 error in pressure values, time-averaged L2 error in pressure values in the domain.

In a two-dimensional shock tube simulation, with the propagation of a shock wave in the computational flow domain having two-dimensional hanging node interface, it is observed that using upwind interpolation near the hanging node interface gives more accurate solutions in the domain. The L2 error in pressure values, its time history, time-averaged L2 error values in the domain also confirms that upwind interpolation performs better than standard interpolation.

From these simulations conducted in this research, it can be inferred that accuracy of solutions in simulations using upwind interpolation near hanging node interfaces has improved than while using standard interpolation.

Nomenclature

Subscripts

i	refers to index pointing to x -axis direction of the grid in an iteration
j	refers to index pointing to y -axis direction of the grid in an iteration
n	refers to index pointing to the current time level of an iteration

Abbreviations

1-D	One-Dimensional
2-D	Two-Dimensional
3-D	Three-Dimensional
BCM	Building-Cube Method
CFD	Computational Fluid Dynamics
CFL	Courant-Friedrichs-Lewy
i.e.	id est (in other words)
L2S	Large to Small (direction of value transfer among cells)
MUSCL	Monotone Upstream Scheme for Conservation Law
S2L	Small to Large (direction of value transfer among cells)
WENO	Weighted Essentially Non-Oscillatory
w.r.t.	with respect to

Symbols

U	displacement of a wave
c	speed of a wave
Λ	wavelength of a wave
ω	angular frequency of a wave
ρ	density in a flow field
u	velocity along x -axis direction in a flow field

v	velocity along y -axis direction in a flow field
p	pressure in a flow field
T	temperature in a flow field
R	specific gas constant
C_v	specific heat of gas at constant volume
ρ_∞	free-stream density in flow field
u_∞	free-stream velocity along x -axis direction in flow field
v_∞	free-stream velocity along y -axis direction in flow field
x_c	x -axis coordinate of vortex center
y_c	y -axis coordinate of vortex center
r	radial distance of any point in flow field from the vortex center
b	strength of the vortex
e	Euler's number, a mathematical constant
ρ_L	density in left side of Sod shock tube
u_L	velocity along x -axis direction in left side of Sod shock tube
v_L	velocity along y -axis direction in left side of Sod shock tube
p_L	pressure in left side of Sod shock tube
ρ_R	density in right side of Sod shock tube
u_R	velocity along x -axis direction in right side of Sod shock tube
v_R	velocity along y -axis direction in right side of Sod shock tube
p_R	pressure in right side of Sod shock tube
\mathbf{U}	conservative variable matrix
\mathbf{V}	primitive variable matrix
\mathbf{W}	characteristic variable matrix
λ_x	characteristic speeds for x -axis calculations
λ_y	characteristic speeds for y -axis calculations
\mathbf{L}_x	left eigenvector matrix for x -axis calculations
\mathbf{L}_y	left eigenvector matrix for y -axis calculations
\mathbf{R}_x	right eigenvector matrix for x -axis calculations
\mathbf{R}_y	right eigenvector matrix for y -axis calculations
\mathbf{F}	fluxes along x -axis in a flow field
\mathbf{G}	fluxes along y -axis in a flow field
Δ	difference matrix calculated from \mathbf{U}
Φ	slope limiter function
\mathbf{r}	slope limiter function variable
\mathbf{K}	eigenvalue matrix
τ	a constant parameter used to find slope limiter value
l	bracketed index ranging from 1 to 4 in 2-D calculations
\tilde{u}	u -velocity by Roe averaging
\tilde{v}	v -velocity by Roe averaging

\tilde{H}	total enthalpy by Roe averaging
\tilde{a}	sound speed by Roe averaging
\tilde{V}	kinetic energy by Roe averaging
$\tilde{\alpha}_i$	wave strengths calculated using eigenvalue matrices
β	smoothness indicator matrix
U^L	reconstructed state matrix at left face of a cell
U^R	reconstructed state matrix at right face of a cell
ω	weights matrix in WENO scheme
α	non-linear weights matrix in WENO scheme

Contents

1	Introduction	1
1.1	Background	1
1.2	Research motivation	2
1.3	Previous studies	4
1.4	Objectives	10
1.5	Thesis outline	11
2	Interpolations	13
2.1	Introduction	13
2.2	Standard interpolation	14
2.3	Upwind interpolation	18
3	Sine wave propagation	23
3.1	Introduction	23
3.2	Problem setting	23
3.3	Results and discussion	26
3.3.1	Standard interpolation	26
3.3.2	Upwind S2L interpolation	28
3.3.3	Analytic solution	29
3.3.4	Comparing sine waves at different times	30
3.3.5	L2 error	35
3.3.6	Summary	36
4	Isentropic vortex convection	37
4.1	Introduction	37
4.2	Problem setting	37
4.3	Results and discussion	40
4.3.1	Comparing standard and upwind interpolation	40
4.3.2	Comparison of Δ Pressure distribution at multiple times	44
4.3.3	L2 error	47
4.3.4	Summary	48

4.4	Vortex convection with 2-D hanging node interfaces	49
4.4.1	Results and discussion	51
4.4.2	Summary	59
5	Shock wave propagation	61
5.1	Introduction	61
5.2	Sod shock tube Riemann problem	61
5.3	Problem setting	62
5.4	Results and discussion	64
5.4.1	Comparing standard and upwind interpolation for Roe with MUSCL	65
5.4.2	Comparing standard and upwind interpolation for Roe with WENO	66
5.4.3	L2 error	67
5.4.4	Summary	70
5.5	Shock propagation from right side to left side of the Sod shock tube	71
5.5.1	Problem setting	71
5.5.2	Results and discussion	72
5.5.3	Summary	75
6	Concluding remarks	77
6.1	Conclusions	77
6.2	Future work	78
A	Governing equations and schemes	81
A.1	Wave equation	81
A.2	Euler equation	83
A.3	Roe with MUSCL scheme	84
A.4	Roe with WENO scheme	87
A.5	Code validations	89
A.5.1	Sine wave propagation	89
A.5.2	Vortex convection	90
A.5.3	Shock wave propagation	91
B	More simulations	93
B.1	Shock wave propagation with 1-D hanging node interface . . .	94
B.1.1	Results and discussion	95
B.1.2	Summary	98
B.2	1-D sine wave propagation	99
B.2.1	Results and discussion	100

List of Figures

1.1	BCM computational domain for flow around airfoil [4]	2
1.2	Initial and final stages of cube generation for flow around a body in BCM [4]	2
1.3	Hanging node interface between different sized cubes	3
1.4	1-D interpolation for random data points [5]	4
1.5	1-D interpolation for a randomly produced ten point table of data [6]	5
1.6	Computational block boundary with hanging node. Closed and open circle denote values at fluid cells and ghost cells respectively [8]	6
1.7	Lagrange interpolation [9]	7
1.8	L_∞ error plot [9]	7
1.9	Interpolation between finite difference grids (1-D)[10]	8
1.10	Interpolation between finite difference grids (2-D)[10]	8
1.11	Standard interpolation for S2L and L2S with two layers of ghost cells	10
2.1	Computational flow domain with jump in square size along x axis direction	13
2.2	Standard interpolation for S2L and L2S with two layers of ghost cells and jump in square size along x axis direction . . .	14
2.3	Standard interpolation for S2L and L2S with three layers of ghost cells and jump in square size along x axis direction	16
2.4	Standard interpolation for S2L and L2S with two layers of ghost cells and jump in square size along y axis direction . . .	17
2.5	Upwind interpolation (low order) for ghost cells $g1, g2$ of small square $S1$	19
2.6	Upwind interpolation (higher order) for ghost cell $g1$ of small square $S1$	20
2.7	Upwind interpolation with two layers of ghost cells and jump in square size along y axis direction	22

3.1	Simulation domain for sine wave propagation	25
3.2	Standard interpolation	26
3.3	Sine wave at time $t = 80$ using standard interpolation	27
3.4	Sine wave at time $t = 160$ using standard interpolation	27
3.5	Upwind S2L interpolation	28
3.6	Sine wave at time $t = 80$ using upwind S2L interpolation	29
3.7	Sine wave at time $t = 160$ using upwind S2L interpolation	29
3.8	Analytic solution at $t = 80$	30
3.9	Analytic solution at $t = 160$	30
3.10	Comparing sine waves at time $t = 40$	31
3.11	Comparing sine waves at time $t = 60$	31
3.12	Comparing sine waves at time $t = 80$	32
3.13	Comparing sine waves at time $t = 100$	32
3.14	Comparing sine waves at time $t = 120$	33
3.15	Comparing sine waves at time $t = 140$	33
3.16	Comparing sine waves at time $t = 160$	34
3.17	Comparing sine waves at time $t = 180$	34
3.18	L2 error plot	35
4.1	Grid specifications in the regions 1 and 2 of the entire domain	38
4.2	Fine and coarse regions in the domain	38
4.3	Initial pressure distribution	39
4.4	Error in pressure (p) throughout the computational domain with standard interpolation (left side plots) and upwind interpolation (right side plots) at $t = 8, 9, 12, 16$	41
4.5	Error in u -velocity throughout the computational domain with standard interpolation (left side plots) and upwind interpolation (right side plots) at $t = 8, 9, 12, 16$	42
4.6	Error in v -velocity throughout the computational domain with standard interpolation (left side plots) and upwind interpolation (right side plots) at $t = 8, 9, 12, 16$	43
4.7	Comparing pressure (p) profiles obtained from simulations using standard, upwind interpolations at $t = 16$	44
4.8	Grid specifications for uniform fine grid simulation	45
4.9	Grid specifications for uniform coarse grid simulation	45
4.10	Δ Pressure distribution at time $t = 6$	46
4.11	Δ Pressure distribution at time $t = 16$	46
4.12	L2 error of pressure (p) in the domain	47
4.13	Grid specifications in all the regions of the entire domain	49
4.14	Fine and coarse regions in the domain	50
4.15	Initial pressure distribution	51

4.16	Error in pressure (p) throughout the computational domain with standard interpolation (left side plots) and upwind interpolation (right side plots) at $t = 8, 9, 12, 16$	52
4.17	Error in u -velocity throughout the computational domain with standard interpolation (left side plots) and upwind interpolation (right side plots) at $t = 8, 9, 12, 16$	53
4.18	Error in v -velocity throughout the computational domain with standard interpolation (left side plots) and upwind interpolation (right side plots) at $t = 8, 9, 12, 16$	54
4.19	Comparing pressure (p) profiles obtained from simulations using standard, upwind interpolations at $t = 16$	55
4.20	Grid specifications for uniform fine grid simulation	56
4.21	Grid specifications for uniform coarse grid simulation	56
4.22	Δ Pressure distribution at time $t = 6$	57
4.23	Δ Pressure distribution at time $t = 16$	57
4.24	L2 error of pressure (p) in the domain	58
5.1	Waves in shock tube	61
5.2	Initial states in the shock tube	62
5.3	Grid distribution	63
5.4	Error in pressure (p) throughout the computational domain with standard interpolation at $t = 0.60$	65
5.5	Error in pressure (p) throughout the computational domain with upwind interpolation at $t = 0.60$	65
5.6	Error in pressure (p) throughout the computational domain with standard interpolation at $t = 0.60$	66
5.7	Error in pressure (p) throughout the computational domain with upwind interpolation at $t = 0.60$	66
5.8	Error in pressure (p) while using both schemes with standard interpolation at $t = 0.60$	67
5.9	L2 error of pressure (p) in the domain at different times for Roe scheme with MUSCL	68
5.10	L2 error of pressure (p) in the domain at different times for Roe scheme with WENO	69
5.11	Initial states in the shock tube	71
5.12	Grid distribution	71
5.13	Error in pressure (p) throughout the computational domain with standard interpolation at $t = 0.60$	73
5.14	Error in pressure (p) throughout the computational domain with upwind interpolation at $t = 0.60$	73

5.15	L2 error of pressure (p) in the domain at different times for Roe scheme with WENO	74
A.1	Grid specifications in the domain	89
A.2	Grid specifications in all the regions of the entire domain . . .	90
A.3	Grid specifications in all the regions of the entire domain . . .	91
B.1	Initial States in the shock tube	94
B.2	Grid distribution	94
B.3	Error in pressure throughout the computational domain with standard interpolation at $t = 0.75$	96
B.4	Error in pressure throughout the computational domain with upwind interpolation at $t = 0.75$	96
B.5	L2 error of pressure in the domain at different times for Roe scheme with MUSCL	97
B.6	Simulation domain for sine wave propagation	99
B.7	Analytic solution at $t = 80$	100
B.8	Analytic solution at $t = 160$	101
B.9	Standard interpolation	101
B.10	Sine wave at time $t = 80$ using standard interpolation	102
B.11	Sine wave at time $t = 160$ using standard interpolation	102
B.12	Upwind S2L interpolation	103
B.13	Sine wave at time $t = 80$ using upwind S2L interpolation . . .	104
B.14	Sine wave at time $t = 160$ using upwind S2L interpolation . .	104
B.15	Comparing sine waves at time $t = 80$	105
B.16	Comparing sine waves at time $t = 160$	105
B.17	L2 error plot	106

List of Tables

3.1	Simulation conditions for sine wave propagation	25
4.1	Grid sizes in regions 1 and 2	40
4.2	Time averaged L2 error of pressure (p) in the domain for Roe scheme with WENO	48
4.3	Grid sizes in regions 1,2,3,4	50
4.4	Time averaged L2 error of pressure (p) in the domain for Roe scheme with WENO	59
5.1	Grid sizes in regions 1,2,3,4	63
5.2	Time averaged L2 error of pressure (p) in the domain for Roe scheme with MUSCL	69
5.3	Time averaged L2 error of pressure (p) in the domain for Roe scheme with WENO	70
5.4	Grid sizes in regions 1,2,3,4	72
5.5	Time averaged L2 error of pressure (p) in the domain for Roe scheme with WENO	75
A.1	L2 error for different grid levels in uniform domain sine wave propagation simulation	89
A.2	L2 error for different grid levels in uniform domain vortex con- vection simulation	90
A.3	L2 error for different grid levels in uniform domain shock wave propagation simulation	91
B.1	Time averaged L2 error of pressure in the domain for Roe scheme with MUSCL	98
B.2	Simulation conditions for sine wave propagation	99

Chapter 1

Introduction

1.1 Background

Large-scale fluid flow computations like the simulation of supercritical four-element airfoil, full airplane simulations [1] in CFD requires easy and quick generation of grid around complex geometries, easy adaptation of local resolution to local flow characteristic length, easy implementation of spatially higher order schemes, massive parallel computations, easy post-processing for huge data output, algorithm simplicity for software maintenance and updation [2][3]. Building-Cube Method (BCM) [3] which is a Cartesian-mesh approach was thus proposed to address these needs of large-scale computations. The strategies are building up of square/cubic subdomains in 2-D/3-D, uniform Cartesian mesh in each square/cube for easy implementation of higher-order schemes, same grid size in all squares/cubes for easy parallel computations and staircase representation of wall boundaries for algorithm simplicity [3]. Figure 1.1 shows a BCM domain for flow around an airfoil.

BCM mesh generation is implemented by two procedures, square/cube generation and cell generation in each cube [3][4]. Initially, the flow field is divided into a number of subdomains named cube in 3-D or square in 2-D. The geometrical size of each cube/square is determined by adapting to the geometry and the flow features. Figure 1.2 depict the stages in cube generation for a simulation using BCM for flow around a body. In each cube/square, a uniform spacing Cartesian mesh is used. Also, local computational resolution can be determined by the size of the cube/square since same number of Cartesian mesh is used in all the squares/cubes [3].

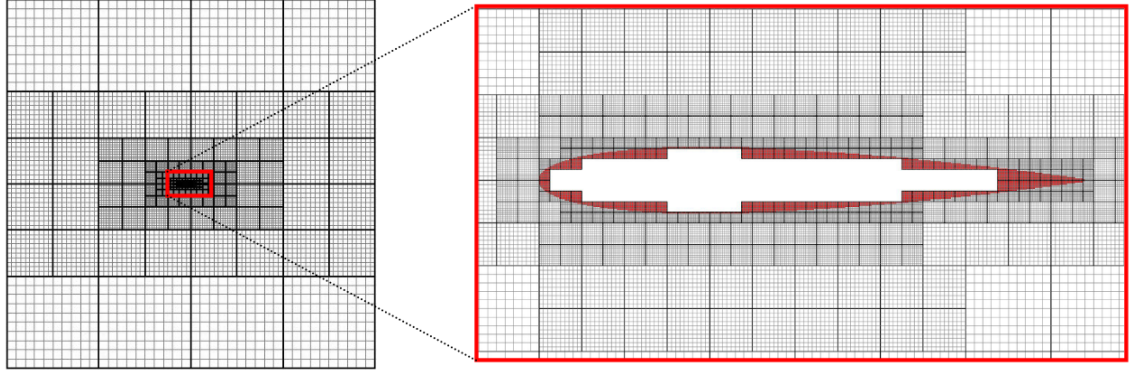


Figure 1.1: BCM computational domain for flow around airfoil [4]

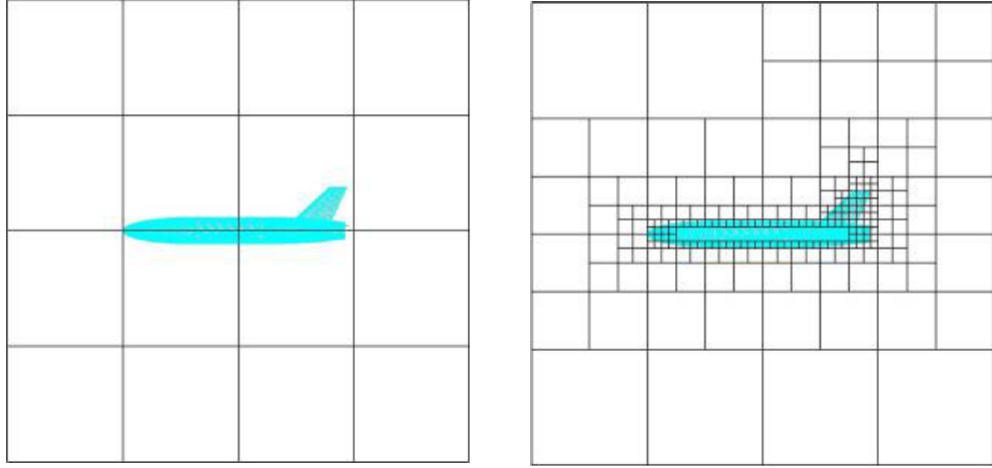


Figure 1.2: Initial and final stages of cube generation for flow around a body in BCM [4]

1.2 Research motivation

In BCM, after the generation of cubes/squares in the entire flow field, it is observed that some of these cubes/squares have different sized cube/square as their neighbor because of the difference in the refinement level. But, since each of these cube/square has the same number of cells in them, a jump in the cell size occurs across the boundary of these adjacent cubes in the flow field.

Hanging node interface

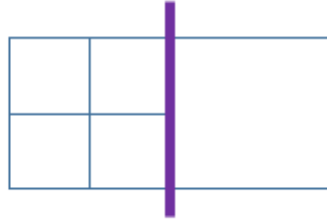


Figure 1.3: Hanging node interface between different sized cubes

A boundary existing between different sized cubes is referred to as a hanging node interface. Figure 1.3 shows a hanging node interface that exists in the domain between neighboring cubes of different sizes. Multiple hanging node interfaces in a computational domain can be observed from Figure 1.1 since the flow field contains cubes/squares having different sized neighbors.

At a cell near the boundary between different sized cubes, calculation of quantities require value from the adjacent different sized cell. The calculation of quantities at a smaller cell near the boundary requires some values from the adjacent larger cell, but the grid point here is largely spaced, and therefore the values can't just be taken and used for computations. Instead, an interpolation has to be devised for using the values. Similarly, for the computation of quantities at larger cell requires values from adjacent smaller cell and an appropriate interpolation has to be implemented here too. Therefore, two different interpolations are a necessity to handle the situation.

Interpolation doesn't give accurate values, it gives values close to the actual values. The usage of interpolation across different adjacent boundaries throughout the field may create errors which may compromise the accuracy of simulations. In simulations where there are many levels of refinement of the cubes/squares, in complex full body airplane simulations the interpolation used at the hanging node interfaces play an important role in deciding the accuracy of the results. In the next section, some interpolations which are considered in fluid simulations, some approaches to transfer the information to ghost cells are discussed.

1.3 Previous studies

Steffen[5] proposed a simple method for 1-D interpolation of a given set of data points. In each interval, the interpolation function is assumed to be a third-order polynomial passing through the data points (piecewise cubic interpolation function that passes through N data points in a monotonic way). Slope is determined to give monotonicity to the interpolating function (i.e. by changing slope at junction points) and results in a smooth curve ensuring convexity of the data. This method gives exact results when the data points correspond to a second-order polynomial.

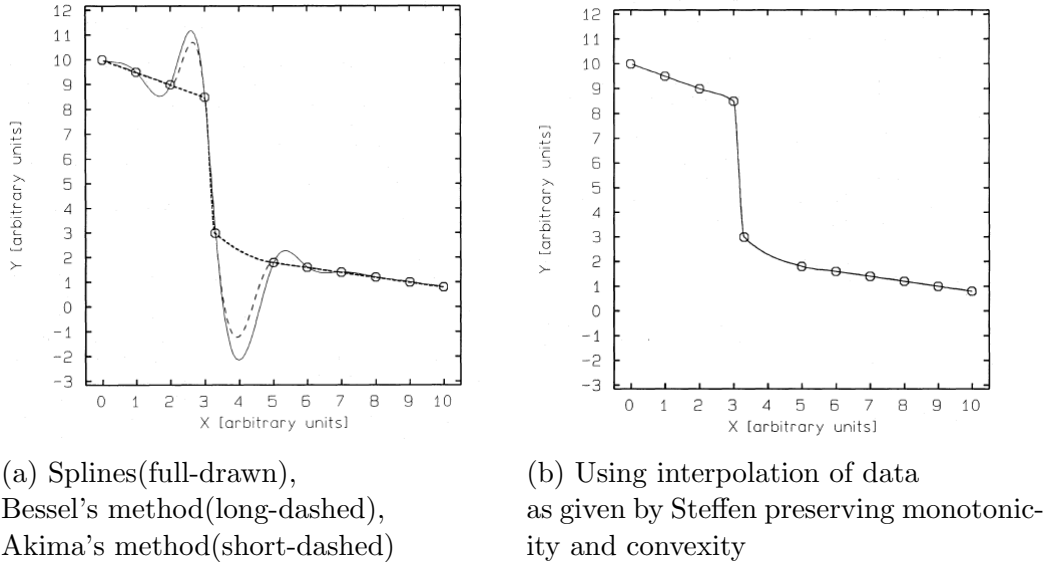
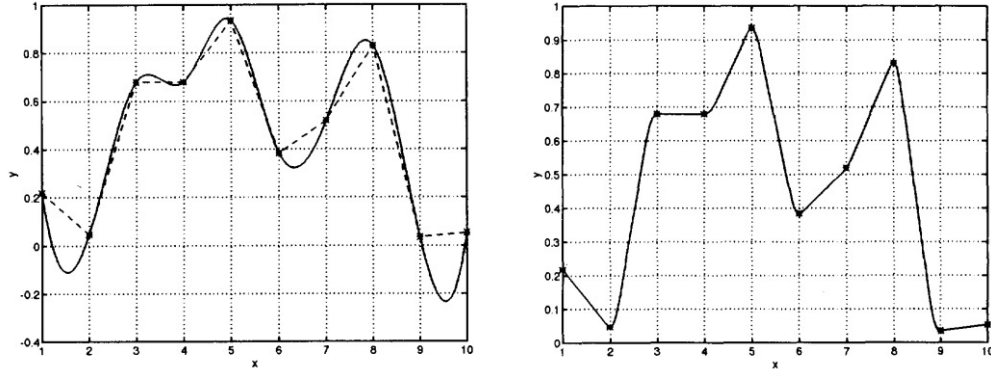


Figure 1.4: 1-D interpolation for random data points [5]

Figure 1.4a compares different interpolations namely, splines method, Bessel's method, Akima's method for the random 1-D data. Figure 1.4b shows the same data being approximated by using interpolation given by Steffen, which preserves the convexity and monotonicity of the data.

Bless and Moerder[6] developed a method for multi-variable data interpolation to obtain second-derivative continuity across data points and have a computationally efficient method for obtaining function evaluations. They achieved this by using multilinear interpolating functions over most of the data set and by using higher-order polynomial surfaces in the vicinity of

the data points to connect multilinear points smoothly. In this method, the interpolating functions are found locally and not globally, which allows N -dimensional tables to be handled efficiently. Bless and Moerder[6] observed and concluded that while using quartic patches, first derivative continuity is obtained and when quadratic patches are used, second derivative continuity is obtained. The research also mentions the special case of one-dimensional interpolation, where it is possible to preserve the convexity and monotonicity of the data.



(a) Comparison of cubic spline (solid line) (b) Using quartic patches, interpolation and straight line approximation (dashed preserving convexity and monotonicity of line) data

Figure 1.5: 1-D interpolation for a randomly produced ten point table of data [6]

Figure 1.5a compares a cubic and a straight line approximation to a randomly produced ten point table of 1-D data. Figure 1.5b shows the same data being approximated by using a quartic patch, which preserves the convexity and monotonicity of this 1-D data.

Kuya et al.[7] developed a numerical scheme for stable and non-dissipative compressible flow simulations on hierarchical Cartesian grids. This scheme is based on the extension of KEEP scheme[8], which is performed to satisfy conservation at computational block boundaries associated with hanging nodes. Here, the values of ghost cells are obtained not from using any particular interpolation method but by satisfying the kinetic energy and entropy conservation in a discrete sense. In this extended scheme, the formulations are built in such a way that the sum of the outflux from a fine grid block is the same as the influx to a coarse grid block.

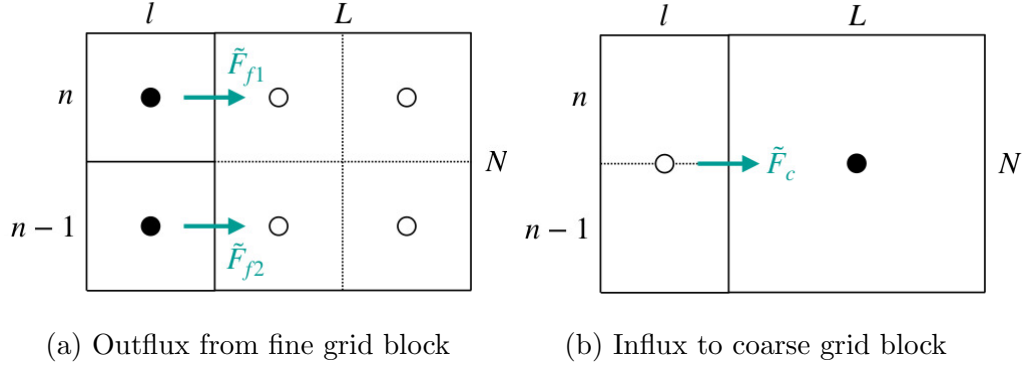


Figure 1.6: Computational block boundary with hanging node. Closed and open circle denote values at fluid cells and ghost cells respectively [8]

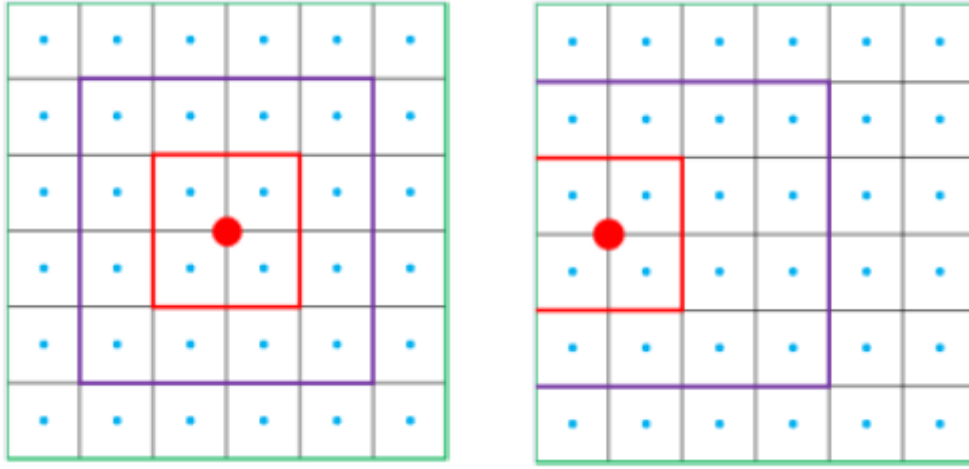
In Figure 1.6 for simplicity the x -direction flux exchange in a 2-D computational domain is considered. l , n , L , N denote the cell number in fine and coarse grid blocks. So, in order to satisfy conservation, numerical fluxes at hanging nodes should satisfy the following equation,

$$\tilde{F}_{f1} + \tilde{F}_{f2} = 2\tilde{F}_c$$

where,

\tilde{F}_{f1} , \tilde{F}_{f2} are the outfluxes at the two fine grid block faces in Figure 1.6a and \tilde{F}_c is the influx at coarse grid block face in Figure 1.6b

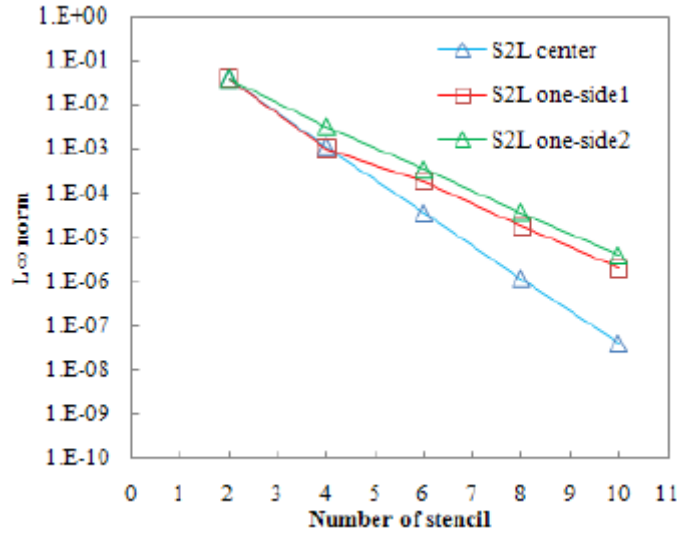
Ishida[9] studied the characteristics of Lagrange interpolation in 1-D and 2-D finite difference framework with the help of sine function. He compared the effects of central difference-like Lagrange interpolation and one-sided-like Lagrange interpolation, as shown in Figure 1.7. From the error plots in Figure 1.8 made by comparing with analytical values he concluded that central-difference like interpolation used for finding values at large cell by interpolating values from smaller cells (known as S2L interpolation) gives lower-order error and stable results, when compared to one-sided interpolation. This central-difference like interpolation is also found to be better for finding values at small cell by interpolating values from large cell (known as L2S interpolation).



(a) S2L central-like interpolation

(b) S2L one-sided interpolation

Figure 1.7: Lagrange interpolation [9]

Figure 1.8: L_∞ error plot [9]

Figueroa et al.[10] studied different order Lagrange interpolation schemes on nested Cartesian finite-difference grids of different size (similar to a BCM domain with hanging node interface). At the fine-coarse boundary, since information needs to be exchanged, ghost points were used. Low-order and high-order Lagrangian interpolation schemes were employed. In order to

avoid spurious oscillations or sharp changes while employing high-order interpolation, several limiters were also used.

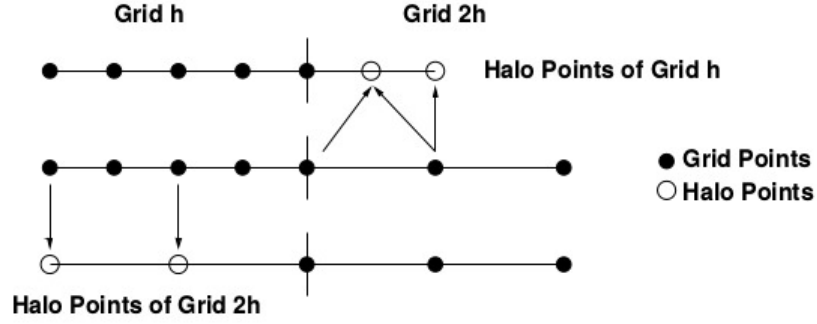


Figure 1.9: Interpolation between finite difference grids (1-D)[10]

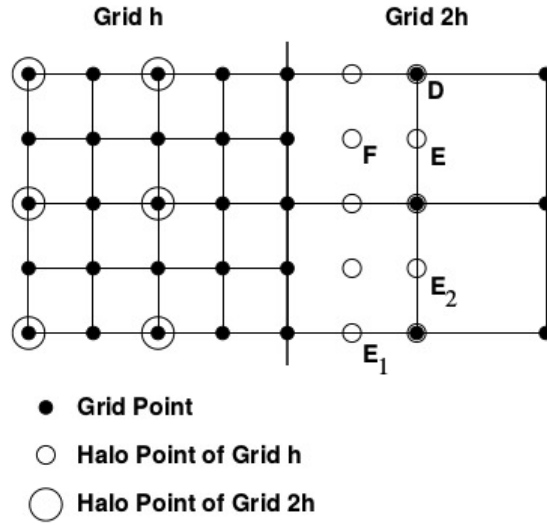


Figure 1.10: Interpolation between finite difference grids (2-D)[10]

Figure 1.9 shows interpolation between finite difference grids in 1-D. Halo/ghost points, which are fake points at which values are not computed directly, but by interpolation. For halo/ghost points of grid h that coincides grid point of $2h$, a direct transfer of value is done. For the other halo points of h which do

not coincide points from $2h$, an average of the surrounding values is transferred. For the halo points of grid $2h$, direct transfer of values is always possible from points of grid h .

Figure 1.10 shows interpolation between finite difference grids in 2-D. For halo points of grid h and type D which coincide with actual points, a direct value transfer is possible. For type E_1 halo points which are edge points aligned with grid-lines from grid h , low-order Lagrangian interpolations are normally used. For type E_2 halo points which are edge points not aligned with grid-lines from grid h , a high-order Lagrangian interpolations are used. Halo points of type F which are face points, maybe interpolated by weighted average of surrounding edge-points of type E_1 with extra information required obtained previously for points of type E_2 . The halo points of grid $2h$ takes value directly from points of grid h .

While simulating a 2-D Lamb vortex, Figueroa et al. observed that using a uniform coarse mesh throughout the domain yielded better results than while employing bilinear interpolation on the fine-coarse domain. But using higher order cubic interpolation yielded better results on fine-coarse simulation domain. They concluded that high-order (cubic and quartic) interpolations in conjunction with limiters gives considerable improvement for 2-D flows. For 3-D flows, they concluded that further research is required to improve the performance of interpolation schemes.

Nakahashi and Kim[3] in BCM simulations transferred physical quantity to ghost cells of larger cube/square (S2L) by assigning the average values in cells of smaller cubes/squares. The transfer of physical quantity from a larger cube/square to ghost cells of smaller cubes/squares (L2S) is done by just taking the value from the cell of the larger cube/square. This way of interpolation is referred to as standard interpolation.

Figure 1.11 shows the standard interpolation on a fine-coarse domain. $S1$, $S2$, $S3$ and $S4$ are the small squares and L is the large square. The large square L has two adjacent smaller squares $S1$, $S3$ and the ghost cells of $S1$, $S3$ squares fall in the space of the large square L . Therefore, L2S transfer of values is done by a zero-order interpolation of value at large cells. Each ghost cell of L encloses a small square having four cells. Therefore, S2L transfer of value is done after averaging of values at four small cells.

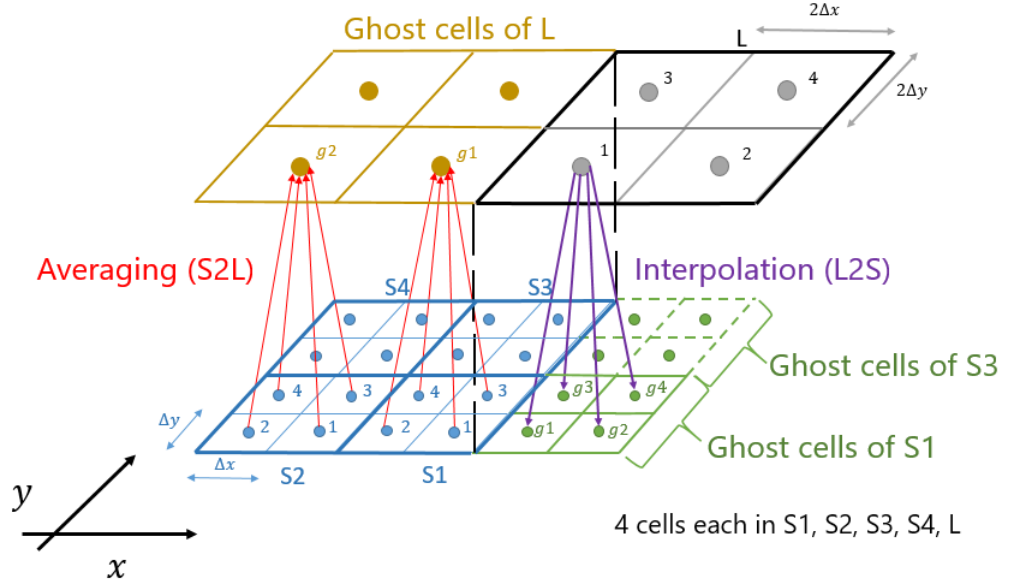


Figure 1.11: Standard interpolation for S2L and L2S with two layers of ghost cells

In standard interpolation, while transferring physical quantities to ghost cells near the hanging node interface, the flow characteristics are not considered, which could make the simulation less accurate.

In a uniform domain, there is no jump in grid size occurring anywhere in the domain. But, since across the hanging node interface in BCM domain a jump in grid size occurs, this interface is a matter of interest. Therefore, to understand what bad happens at the interface while using standard interpolation, a simple sine wave propagation simulation is to be conducted on a BCM domain. The details of this simulation are discussed ahead in Chapter 3.

1.4 Objectives

Based on the literature study and the limited research material available, it is understood that sufficient research to improve simulation accuracy in BCM while using ghost cells is not yet done. The accuracy of results from simulations depend on values assigned to ghost cells during computations. The present interpolation used for obtaining values at ghost cells is the standard interpolation. The inherent shortcoming of this interpolation results in less

accurate fluid flow simulations. So effort is made to increase the accuracy of the values transferred to ghost cells by proposing a new better interpolation. This interpolation is aimed to consider the upwind direction of the flow characteristics for obtaining better interpolated values in ghost cells, since information is propagated in a flow field from upwind side to downwind side. Therefore, the objectives of this research are,

- firstly, to understand how a simple wave travels through a BCM kind of domain having both fine and coarse grids. This is done to check what bad happens at the hanging node interface which reduces the accuracy of simulation and whether this solution accuracy can be improved further by a new interpolation. This is done by a simple sine wave propagation simulation,
- secondly, to propose an upwind interpolation method based on characteristic variables to get more accurate values of primitive variables in ghost cells near hanging node interfaces for better BCM compressible Euler equation simulations,
- and finally to confirm that the new upwind interpolation performs better than standard interpolation in vortex convection and shock wave propagation simulations by checking the errors in the simulation domain calculated by comparing with the analytical values. Isentropic vortex convection problem is used since it is a basic advection problem. The Sod shock tube problem is used because it is a widely used basic problem in Riemann solvers.

1.5 Thesis outline

This thesis contains six chapters, including this introductory chapter. The remaining work is categorized as follows,

- Chapter 2, the existing interpolation used in BCM i.e. standard interpolation used for obtaining the primitive variables inside the ghost cells of larger and smaller cubes/squares is explained, then the newly formulated upwind interpolation is presented.
- In Chapter 3, a simple sine wave propagation is simulated on a domain with 1-D hanging node interface. The results obtained from simulations employing standard interpolation and upwind interpolation separately are shown and discussed.

- In Chapter 4, a convection test case problem, an isentropic vortex convection is simulated. Two separate scenarios with 1-D, 2-D hanging node interfaces in the domain are simulated. The results obtained in both the scenarios while using standard interpolation and upwind interpolation separately are compared and discussed.
- In Chapter 5, a shock propagation problem, namely Sod shock tube problem, is described and simulated using standard and upwind interpolation separately on a domain having 2-D hanging node interface. Results obtained from both the cases are analyzed.
- In Chapter 6, the conclusions of this research and the scope for future research in this domain are presented.

Chapter 2

Interpolations

2.1 Introduction

In this chapter, the existing interpolation namely, standard interpolation is explained and then the newly proposed upwind interpolation based on characteristic variables is also detailed for compressible Euler equation simulations. A sample computational domain considered for simulations when there is a jump in size of squares only along x axis direction is shown in Figure 2.1.

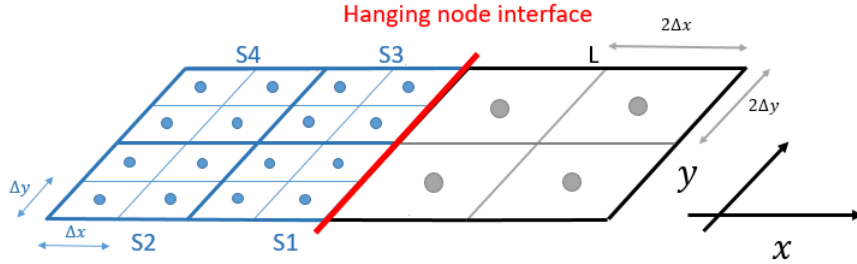


Figure 2.1: Computational flow domain with jump in square size along x axis direction

The hanging node interface is therefore present only along the y axis direction, as shown. This domain with jump in square size along x axis direction is primarily used for explaining both the standard and upwind interpolations.

2.2 Standard interpolation

Standard interpolation when two layers of ghost cells (of squares $S1, S3$) are employed is shown in Figure 2.2. Here, the fluid flow computation is considered in a single direction i.e. along x axis direction and the flow encounters hanging node interface (since there is a jump in size of squares along x axis direction).

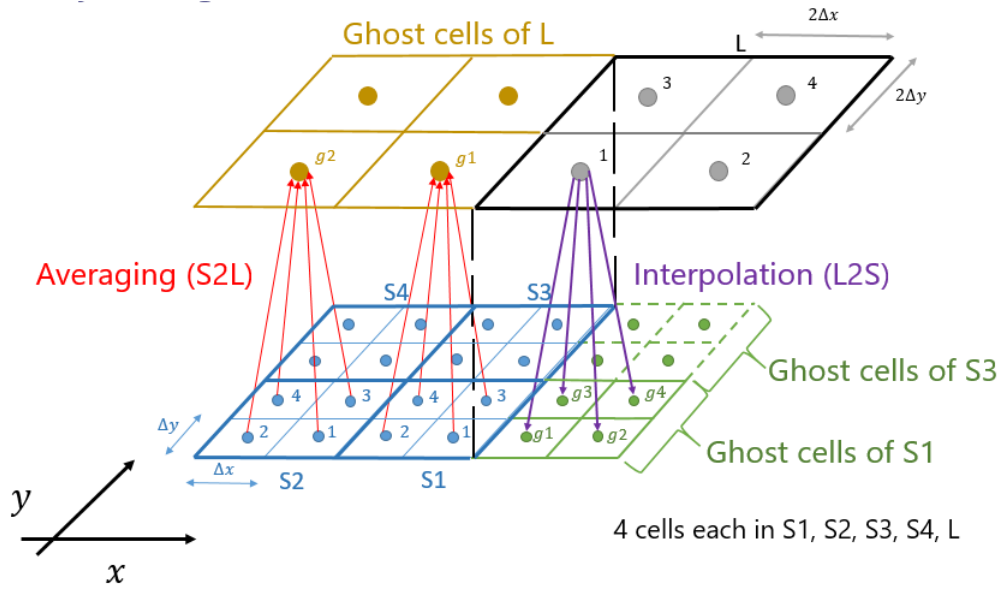


Figure 2.2: Standard interpolation for S2L and L2S with two layers of ghost cells and jump in square size along x axis direction

From this Figure 2.2, $S1, S2, S3$ and $S4$ are the small squares and L is the large square. The larger square L has two adjacent smaller squares $S1, S3$. Therefore, the ghost cells of $S1, S3$ squares fall in the space of large square L .

In standard interpolation, values that are transferred are the primitive variable value matrix \mathbf{V} . The primitive variable matrix \mathbf{V} is defined as,

$$\mathbf{V}(4) = \begin{bmatrix} \rho & u & v & p \end{bmatrix} \quad (2.1)$$

where ρ, u, v, p are the density, u -velocity, v -velocity and pressure respectively.

The standard interpolation method for getting the primitive values at ghost cells $(g1, g2, g3, g4)$ of $S1$ as shown in the Figure 2.2 is by transferring the primitive values from cells in L to the ghost cells of $S1$, $S3$ (L2S). To get the primitive values at a ghost cell of L is by averaging the cell values of fine cells (S2L).

The subscript indices for matrix \mathbf{V} in the following equations are used to identify the square, cell respectively in the Figure 2.2.

$$\mathbf{V}_{(S1,g1)} = \mathbf{V}_{(L,1)} \quad (2.2)$$

$$\mathbf{V}_{(S1,g2)} = \mathbf{V}_{(L,1)} \quad (2.3)$$

$$\mathbf{V}_{(S1,g3)} = \mathbf{V}_{(L,1)} \quad (2.4)$$

$$\mathbf{V}_{(S1,g4)} = \mathbf{V}_{(L,1)} \quad (2.5)$$

$$\mathbf{V}_{(L,g1)} = \frac{1}{4} \sum_{k=1}^4 \mathbf{V}_{(S1,k)} \quad (2.6)$$

$$\mathbf{V}_{(L,g2)} = \frac{1}{4} \sum_{k=1}^4 \mathbf{V}_{(S2,k)} \quad (2.7)$$

With three layers of ghost cells employed as shown in Figure 2.3, $S1$, $S2$, $S3$, $S4$, $S5$, $S6$ are the small squares and L is the large square. For two adjacent squares of different sizes (L and $S1$ or L and $S4$), the ghost cells of $S1$, $S4$ squares fall into the space of L .

The standard interpolation method for getting the primitive values at ghost cells of $S1$, $S4$ as shown in the Figure 2.3 is by transferring the primitive values from cells in L to the ghost cells of $S1$, $S4$ (L2S). To get the values at a ghost cell of L is by averaging the cell values of fine cells (S2L).

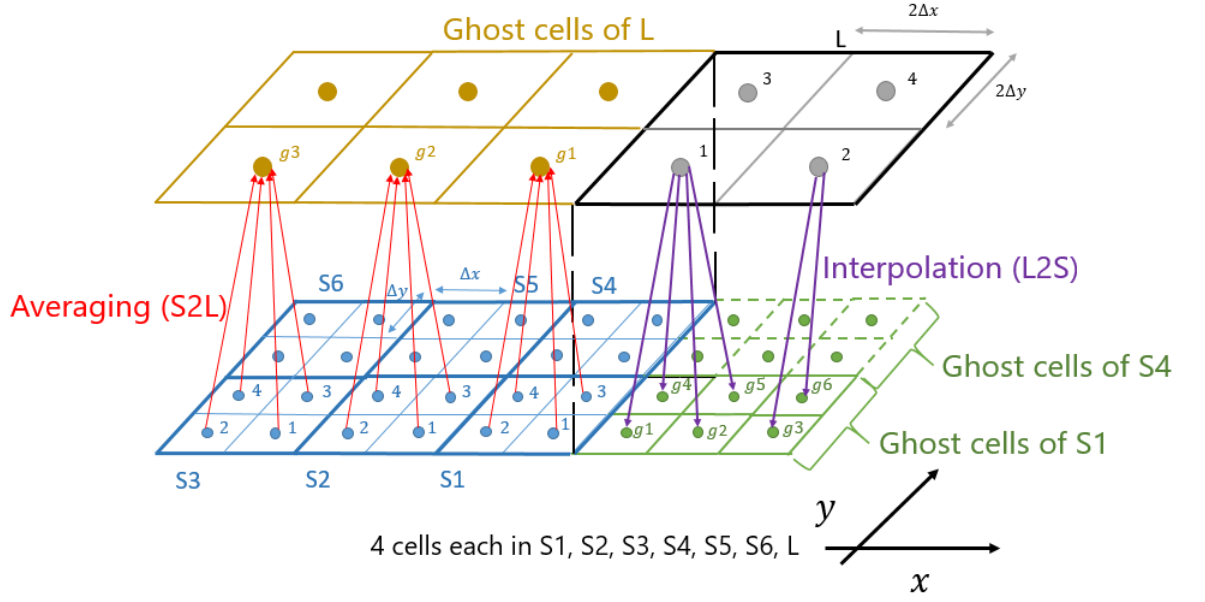


Figure 2.3: Standard interpolation for S2L and L2S with three layers of ghost cells and jump in square size along x axis direction

$$V_{(S1,g1)} = V_{(L,1)} \quad (2.8)$$

$$V_{(S1,g2)} = V_{(L,1)} \quad (2.9)$$

$$V_{(S1,g3)} = V_{(L,2)} \quad (2.10)$$

$$V_{(S1,g4)} = V_{(L,1)} \quad (2.11)$$

$$V_{(S1,g5)} = V_{(L,1)} \quad (2.12)$$

$$V_{(S1,g6)} = V_{(L,2)} \quad (2.13)$$

$$V_{(L,g1)} = \frac{1}{4} \sum_{k=1}^4 V_{(S1,k)} \quad (2.14)$$

$$V_{(L,g2)} = \frac{1}{4} \sum_{k=1}^4 V_{(S2,k)} \quad (2.15)$$

$$V_{(L,g3)} = \frac{1}{4} \sum_{k=1}^4 V_{(S3,k)} \quad (2.16)$$

When there is a jump in the size of squares in the y axis direction (i.e. hanging node interface is present along x axis direction), the following Figure 2.4

shows the ghost cells and the standard interpolation used, a scenario similar to when the jump in square size was along x axis direction.

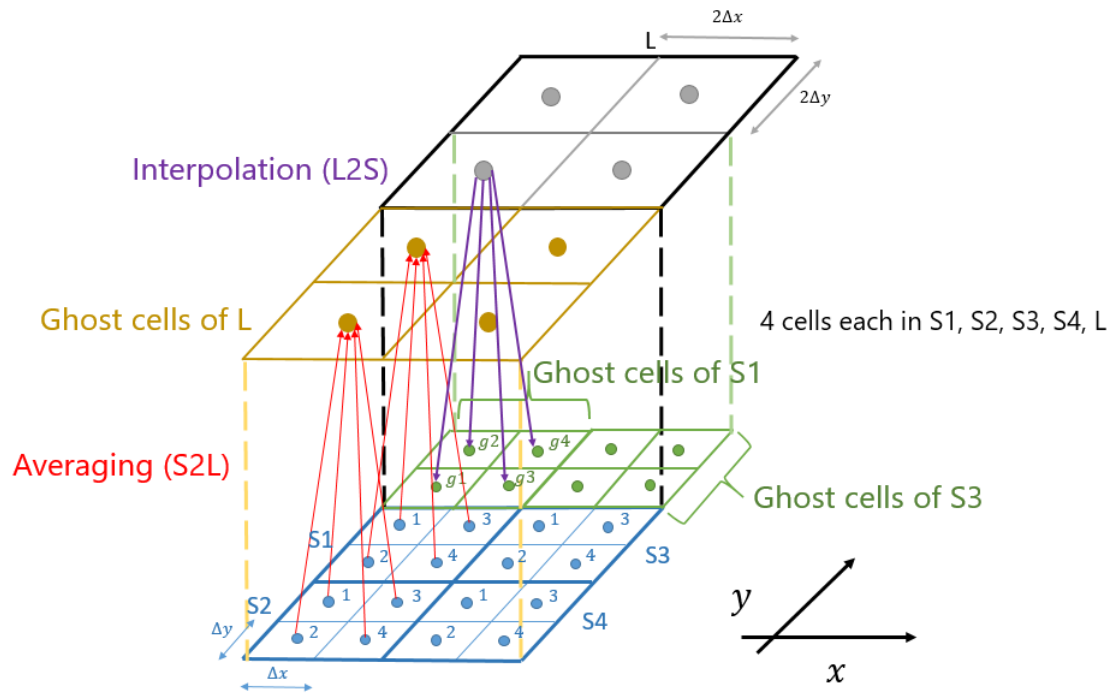


Figure 2.4: Standard interpolation for S2L and L2S with two layers of ghost cells and jump in square size along y axis direction

2.3 Upwind interpolation

Upwind interpolation is done by taking into account the direction of fluid flow characteristics while interpolating. This is made possible by using the eigenvalue matrix.

$$\boldsymbol{\lambda}_x = \begin{bmatrix} u - c & u & u & u + c \end{bmatrix} \quad (2.17)$$

Equation 2.17 shows the eigenvalues along x axis direction. In this interpolation, characteristic variable values are transferred to ghost cells instead of primitive values. The characteristic variable matrices $\mathbf{W}_x, \mathbf{W}_y$ in the x, y directions are obtained from the primitive variable \mathbf{V} by transformation [12] using left Eigenvector matrices $\mathbf{L}_x, \mathbf{L}_y$ [13] in x, y directions according to the Equations 2.18, 2.19.

$$\mathbf{W}_x = \mathbf{L}_x \mathbf{V} \quad (2.18)$$

$$\mathbf{W}_y = \mathbf{L}_y \mathbf{V} \quad (2.19)$$

and therefore,

$$\mathbf{W}_x = \begin{bmatrix} u - \frac{p}{\rho c} \\ \rho - \frac{p}{c^2} \\ v \\ u + \frac{p}{\rho c} \end{bmatrix} \quad \mathbf{W}_y = \begin{bmatrix} v - \frac{p}{\rho c} \\ \rho - \frac{p}{c^2} \\ u \\ v + \frac{p}{\rho c} \end{bmatrix}$$

Figure 2.5 shows a low order i.e. a zero order upwind interpolation to ghost cells of smaller squares. Ghost cell $g1$ of $S1$ takes characteristic variable value from cell 1 of the small square $S1$ if the characteristic flow direction is in positive x axis direction and takes value from cell 1 of the large square L if the characteristic flow direction is in negative x axis direction. For flow computations along x axis direction, this characteristic upwind direction is decided from eigenvalues inside the matrix $\boldsymbol{\lambda}_x$ of cell 1 in the larger square L (which contain the ghost cells of $S1$). The interpolation to ghost cells of smaller squares (L2S from standard interpolation) are modified in this new upwind interpolation.

The ghost cells of L get their values (S2L) by averaging, as in standard interpolation. Since the area of ghost cell of L encompasses all four cells in

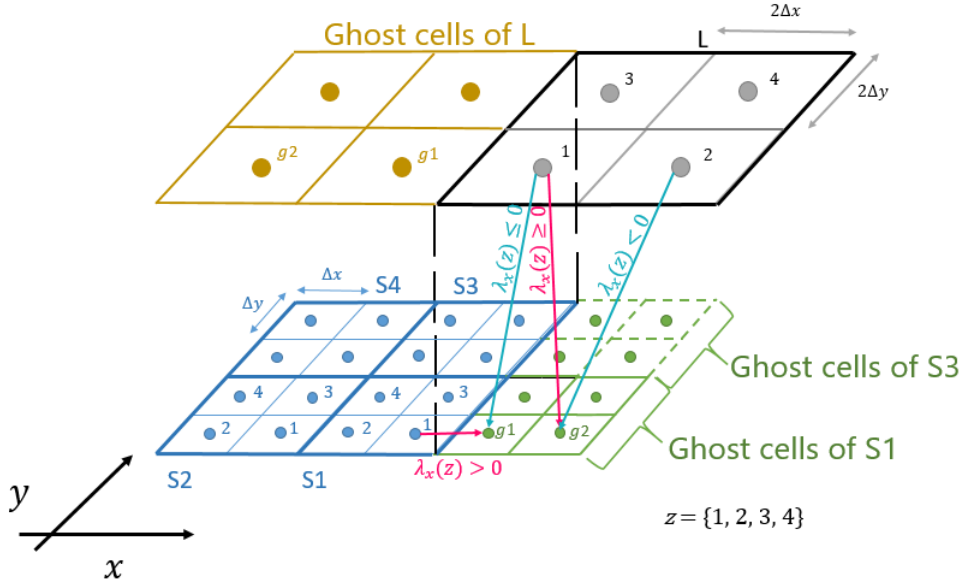


Figure 2.5: Upwind interpolation (low order) for ghost cells $g1, g2$ of small square $S1$

a small square, averaging is preferred here.

The Equations 2.20, 2.21 show the mathematical representation of the upwind interpolation as used in simulation. In x axis direction computation,

$$\mathbf{W}_{x(S1,g1)}(z) = \begin{cases} \mathbf{W}_{x(S1,1)} & \lambda_x(z) > 0 \\ \mathbf{W}_{x(L,1)} & \lambda_x(z) \leq 0 \end{cases} \quad (2.20)$$

$$\mathbf{W}_{x(S1,g2)}(z) = \begin{cases} \mathbf{W}_{x(L,1)} & \lambda_x(z) \geq 0 \\ \mathbf{W}_{x(L,2)} & \lambda_x(z) < 0 \end{cases} \quad (2.21)$$

$$z = \{1, 2, 3, 4\}$$

The subscript indices for matrix \mathbf{W}_x in the above equations are used to identify the square, cell respectively in the Figure 2.5.

The characteristic variable values thus obtained at ghost cells needs to be transformed into the primitive values at the ghost cells for the scheme computations in x, y directions. For this purpose, the right Eigenvector matrices $\mathbf{R}_x, \mathbf{R}_y$ of ghost cells in x, y directions are used according to the Equations 2.22, 2.23 and the values for variables used in $\mathbf{R}_x, \mathbf{R}_y$ are obtained from the

large cell in square L .

$$\mathbf{V} = \mathbf{R}_x \mathbf{W}_x \quad (2.22)$$

$$\mathbf{V} = \mathbf{R}_y \mathbf{W}_y \quad (2.23)$$

Thus, upwind interpolation approach is different from the standard interpolation (where values from cell 1 of large square L is passed to the ghost cell $g1$, irrespective of the flow characteristics direction).

In upwind interpolation, for calculating characteristic values in ghost cells of smaller squares, apart from using zero order interpolations, higher order Lagrange interpolations can also be used. With higher order Roe with WENO schemes[14] explained in appendix A.4 while using three layers of ghost cells, higher order interpolations using larger stencil as shown in Figure 2.6 can also be utilized for calculating the characteristic variables in the ghost cell. Such a higher order Lagrange interpolation used in our research are shown by the Equations 2.24, 2.25 and 2.26.

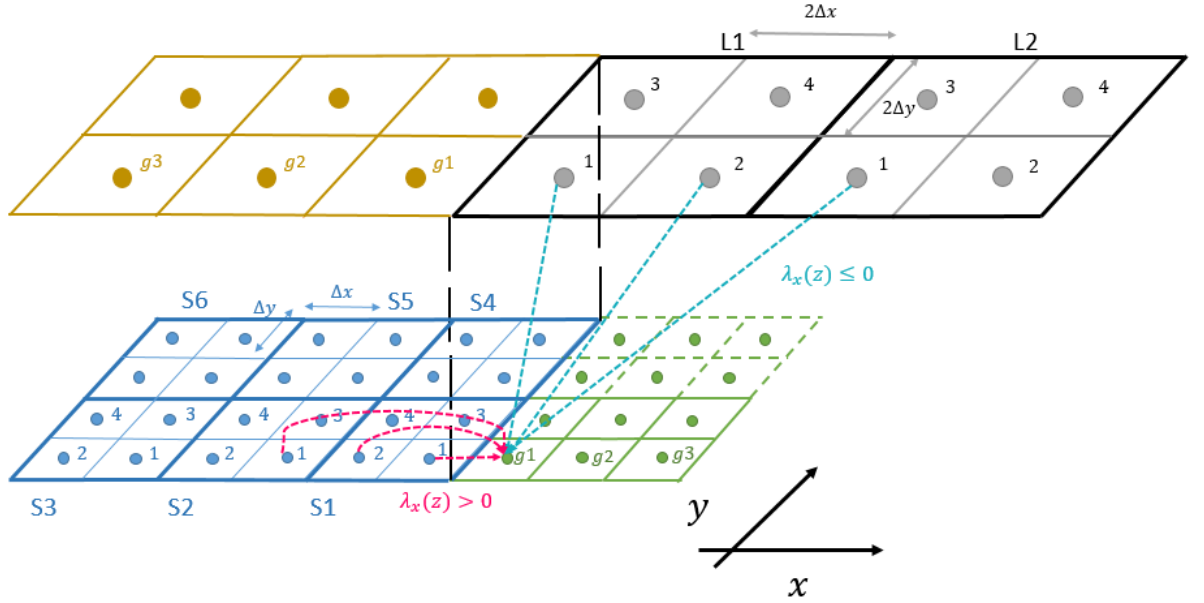


Figure 2.6: Upwind interpolation (higher order) for ghost cell $g1$ of small square $S1$

$$\mathbf{W}_{x(S1,g1)}(z) = \begin{cases} \mathbf{W}_{x(S2,1)} - 3\mathbf{W}_{x(S1,2)} + 3\mathbf{W}_{x(S1,1)} & \lambda_x(z) > 0 \\ \frac{45}{32}\mathbf{W}_{x(L1,1)} - \frac{18}{32}\mathbf{W}_{x(L1,2)} + \frac{5}{32}\mathbf{W}_{x(L2,1)} & \lambda_x(z) \leq 0 \end{cases} \quad (2.24)$$

$$\mathbf{W}_{x(S1,g2)}(z) = \begin{cases} \frac{2}{5}\mathbf{W}_{x(S1,2)} - \mathbf{W}_{x(S1,1)} + \frac{8}{5}\mathbf{W}_{x(L1,1)} & \lambda_x(z) \geq 0 \\ \frac{77}{32}\mathbf{W}_{x(L1,2)} - \frac{66}{32}\mathbf{W}_{x(L2,1)} + \frac{21}{32}\mathbf{W}_{x(L2,2)} & \lambda_x(z) < 0 \end{cases} \quad (2.25)$$

$$\mathbf{W}_{x(S1,g3)}(z) = \begin{cases} \frac{9}{5}\mathbf{W}_{x(S1,2)} - 4\mathbf{W}_{x(S1,1)} + \frac{16}{5}\mathbf{W}_{x(L1,1)} & \lambda_x(z) > 0 \\ \frac{45}{32}\mathbf{W}_{x(L1,2)} - \frac{18}{32}\mathbf{W}_{x(L2,1)} + \frac{5}{32}\mathbf{W}_{x(L2,2)} & \lambda_x(z) \leq 0 \end{cases} \quad (2.26)$$

$$z = \{1, 2, 3, 4\}$$

Figure 2.7: Upwind interpolation with two layers of ghost cells and jump in square size along y axis direction

Chapter 3

Sine wave propagation

3.1 Introduction

A sine wave propagation is chosen initially to understand how a simple wave travels through a BCM like domain which has both fine and coarse grids, i.e. how it propagates across the interface between these two regions in the domain. A fine grid to coarse grid flow is chosen, since in real world problems the flow features move from finer regions closer to the body to the coarser outer regions of the computational fluid domain. Two interpolations, namely, standard interpolation and upwind S2L interpolation used in this non-Euler equation simulation are detailed and results obtained from the simulations using these interpolations are discussed.

3.2 Problem setting

The classical wave equation used for propagating the sine wave is a second-order linear partial differential equation and is as follows on a 2-D domain,

$$\frac{\partial^2 U}{\partial t^2} = c^2 \left(\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} \right) \quad (3.1)$$

where U is the displacement of the wave, x, y are the axes of the 2-D domain and c is the wave speed. The central differencing scheme is used for this simulation, and the equation used for updating the U value in the next iteration level $n + 1$ from the current level n is given by,

$$\begin{aligned}
U_{i,j}^{n+1} = & 2U_{i,j}^n - U_{i,j}^{n-1} + \frac{c^2 \Delta t^2}{\Delta x^2} (U_{i+1,j}^n - 2U_{i,j}^n + U_{i-1,j}^n) \\
& + \frac{c^2 \Delta t^2}{\Delta y^2} (U_{i,j+1}^n - 2U_{i,j}^n + U_{i,j-1}^n) \quad (3.2)
\end{aligned}$$

This scheme is derived in a finite-volume framework in section A.1 of appendix A.

A single sine wave is propagated with wave speed $c = 1$ and wavelength $\Lambda = 40$ units from $x = 0$ (left side boundary) according to the Equation 3.3,

$$\begin{aligned}
U(0, t) &= \sin(\omega t), \quad \omega t \leq 2\pi \quad (3.3) \\
\omega &= \frac{2\pi}{\Lambda/c}
\end{aligned}$$

where ω is the angular frequency as defined and non-reflecting boundary conditions are imposed on all other boundaries.

Other simulation conditions are as shown in Table 3.1. To mimic the BCM kind of simulation which have hanging node interfaces, the entire domain length of 200 units in x direction is divided into two halves at $x = 100$ by allocating grid resolution as shown in Figure 3.1. Therefore, now the domain consists of a hanging node interface at $x = 100$ in the y axis direction.

Figure 3.1 shows the two regions separated by the hanging node interface. The filled circular dots represent the cell centers in each region where the equation is calculated in each iteration. One layer of ghost cells in each region is used here, as shown by the colored circular dots. The sine wave propagates from fine region on the left side of the hanging node interface to coarse region on the right side, since the wave speed c here is a positive quantity. Simulation outputs are taken from the time $t = 20$ (i.e. the leading front of the wave has reached $x = 20$ since $c = 1$) when the complete sine wave is in the fine region until a time $t = 180$ when the wave is well-placed in the coarse region after having crossed the hanging node interface at intervals of 20 time units.

Domain	Length - 200 Width - 100
Mesh	Fine grid - 100 x 100, x=0 to 100 Coarse grid - 50 x 50, x=100 to 200
CFL	Fine grid - 1.0 Coarse grid - 0.5
Simulation end time	180
Governing equation	2-D Wave equation
Scheme	2 nd order Central differencing scheme

Table 3.1: Simulation conditions for sine wave propagation

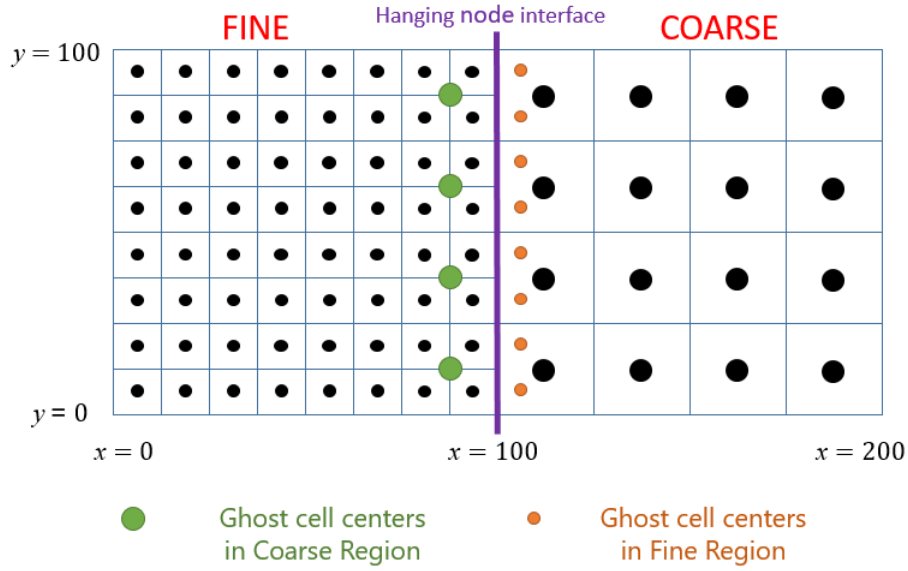
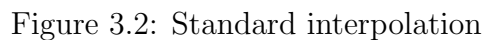


Figure 3.1: Simulation domain for sine wave propagation

To understand the behavior of sine wave in a fine to coarse BCM like domain, the wave is initially simulated with the help of standard interpolation (used for obtaining the values at the ghost cell centers). The standard interpolation used in this case is explained ahead.

The transfer of values to ghost cells while using this interpolation are as shown in Figure 3.2. The ghost cells in fine region takes average of the values from surrounding small cells and the ghost cells in coarse region takes value from the nearest large cell.



It is observed that standard interpolation produces a reflected wave from

the hanging node interface which travels back into the fine region. This interpolation also reduces the amplitude of the sine wave after the interface.

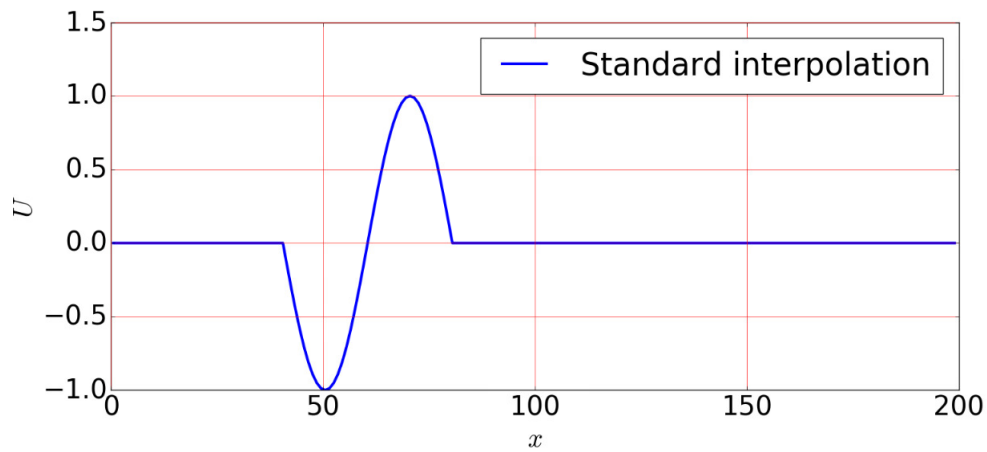


Figure 3.3: Sine wave at time $t = 80$ using standard interpolation

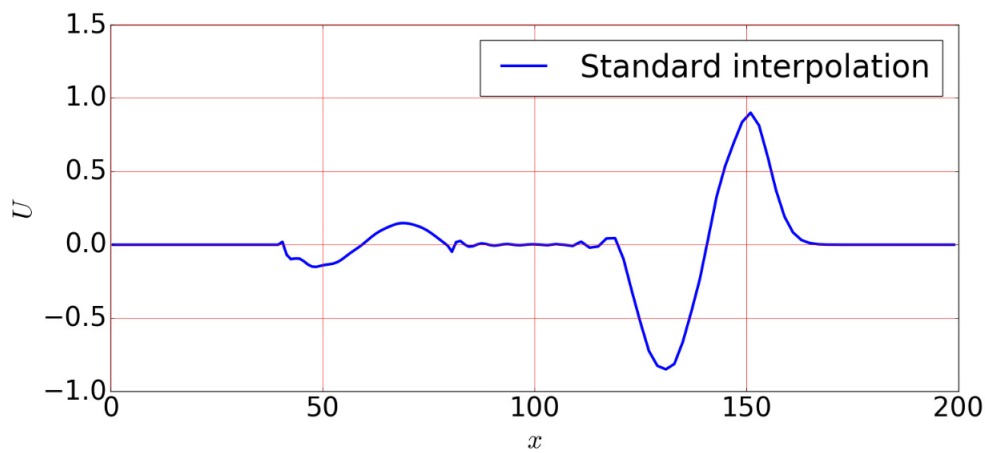


Figure 3.4: Sine wave at time $t = 160$ using standard interpolation

3.3.2 Upwind S2L interpolation

The sine wave propagation is then simulated using upwind like interpolation for the ghost cells to check whether this interpolation provides better wave in the coarse region. The transfer of values to ghost cells while using this interpolation are as shown in Figure 3.5. The ghost cells in fine region takes average of the values from small cells in the upwind direction and the ghost cells in coarse region takes value from the nearest large cell.

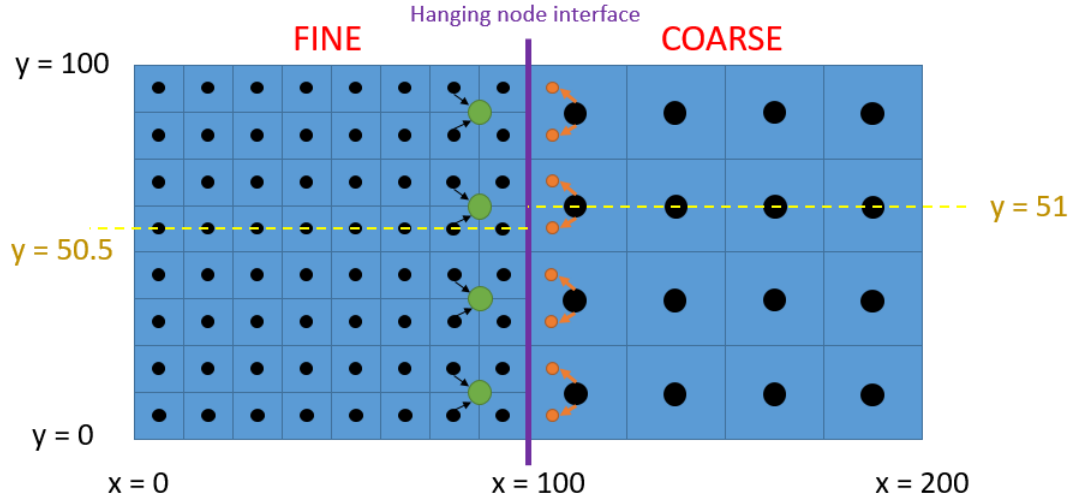
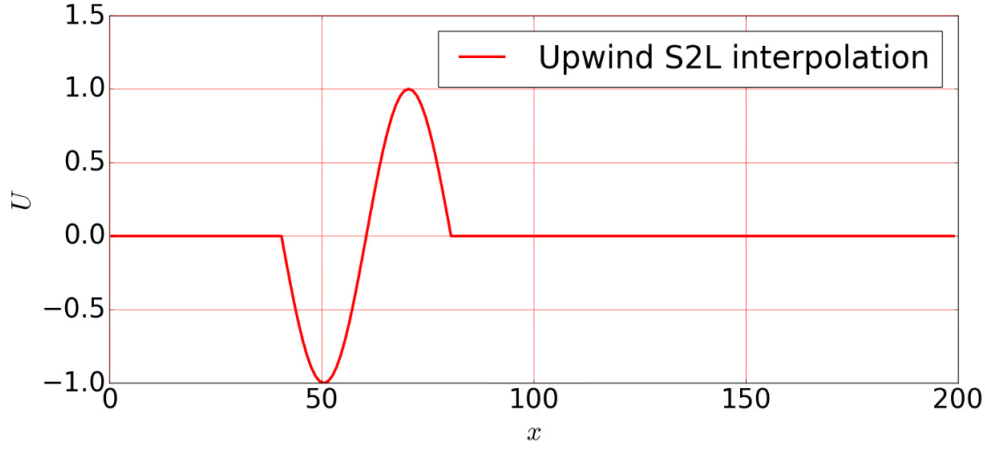
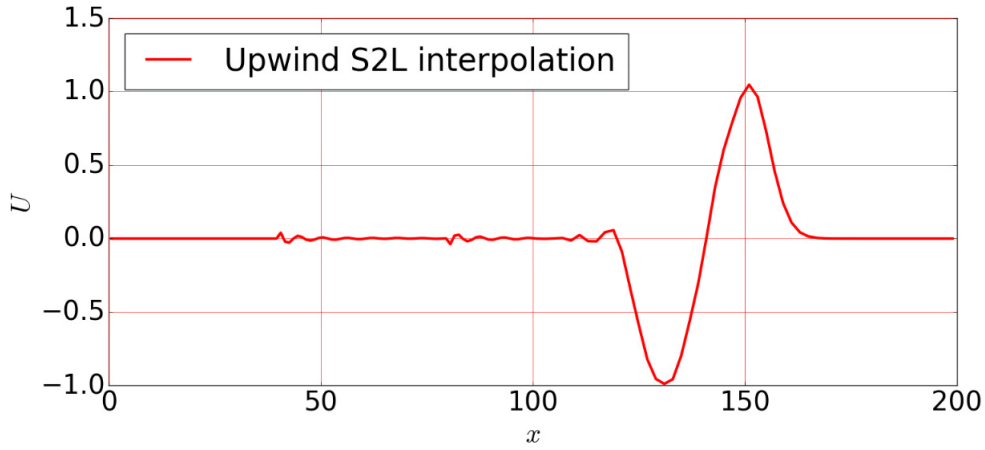


Figure 3.5: Upwind S2L interpolation

While using upwind S2L interpolation shown in Figure 3.5 at the hanging node interface located at the boundary between fine and coarse region, the wave at times $t = 80$ and $t = 160$ are plotted in Figure 3.6 and in Figure 3.7. The plots are made using values at $y = 50.5$ and $y = 51$ in the fine and coarse regions respectively, since the cell centers do not lie in the same y value across the entire domain.

Although very small magnitude reflected waves (in the form of wiggles) can be seen in the fine region, it is observed that upwind S2L interpolation effectively reduces the reflected wave from the hanging node interface. Therefore, this interpolation is able to maintain the original amplitude of the sine wave even after the interface.

Figure 3.6: Sine wave at time $t = 80$ using upwind S2L interpolationFigure 3.7: Sine wave at time $t = 160$ using upwind S2L interpolation

3.3.3 Analytic solution

To compare the simulation results while using standard and upwind S2L interpolation, the analytic solution is used and is calculated from sine wave equation 3.4 for all y values on the fine-coarse domain.

$$U(x, t) = U(x - ct) = \sin(\omega(x - ct)) \quad (3.4)$$

The analytic waves at times $t = 80$ (a randomly chosen time when the wave has not reached the hanging node interface and is completely in fine region) and $t = 160$ (a randomly chosen time when the wave has completely crossed the hanging node interface and is completely in coarse region) are plotted as shown in Figures 3.8, 3.9.

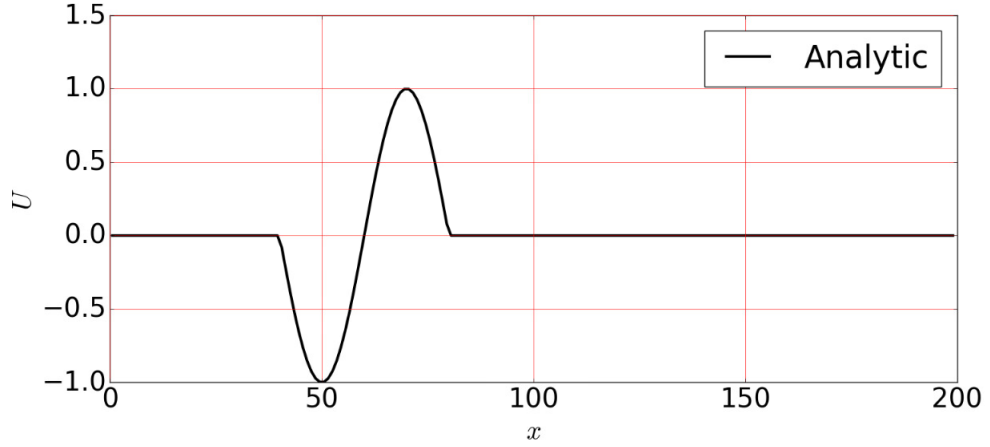


Figure 3.8: Analytic solution at $t = 80$

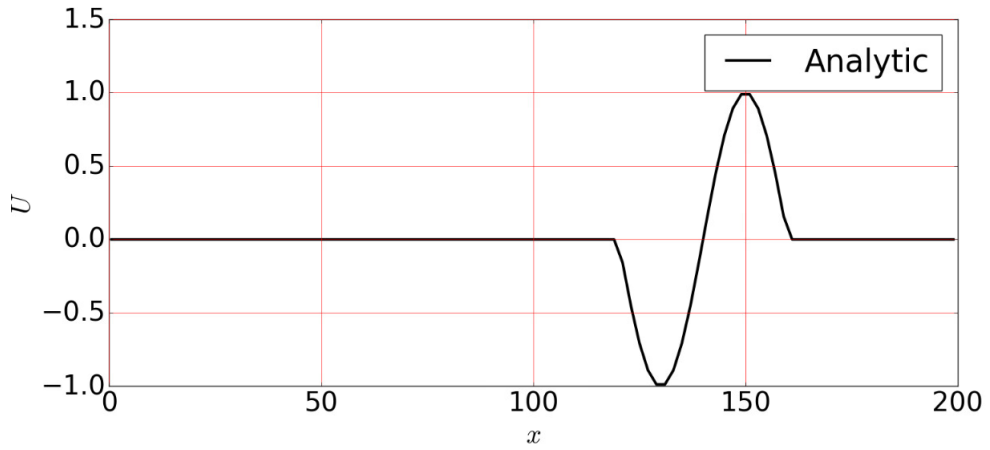


Figure 3.9: Analytic solution at $t = 160$

3.3.4 Comparing sine waves at different times

Figures 3.10 to 3.17 compares waves obtained from simulations using standard interpolation, upwind S2L interpolation and the analytic solution for

times $t = 40$ to $t = 180$.

The waves thus obtained are the same at any time before they reach the hanging node interface (at $x = 100$ at time $t = 100$) as shown in Figures 3.10 to 3.13.

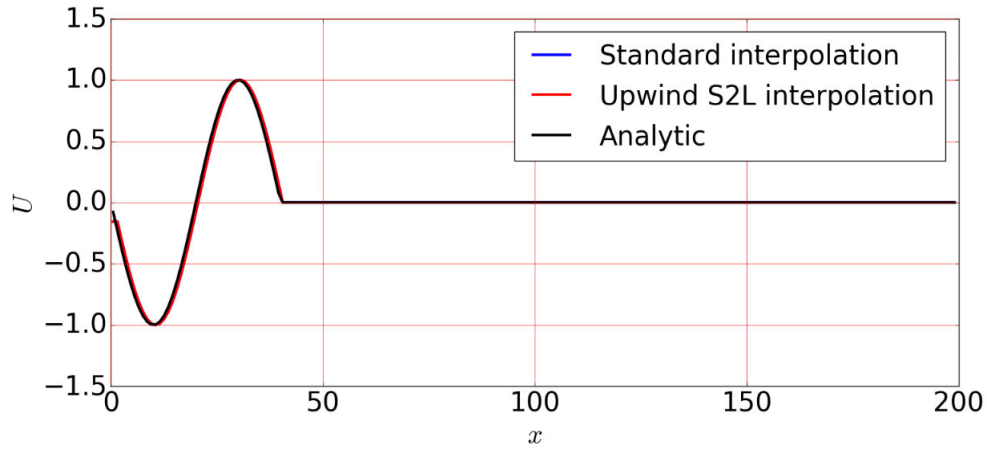


Figure 3.10: Comparing sine waves at time $t = 40$

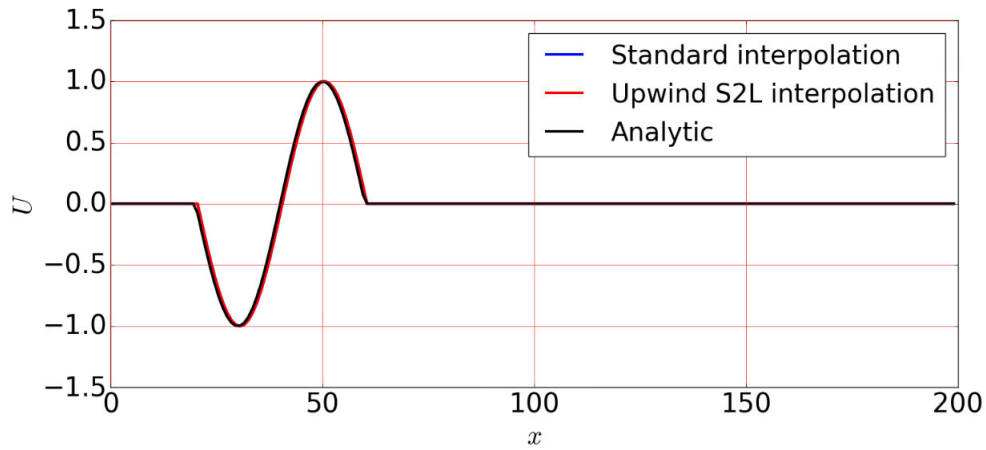
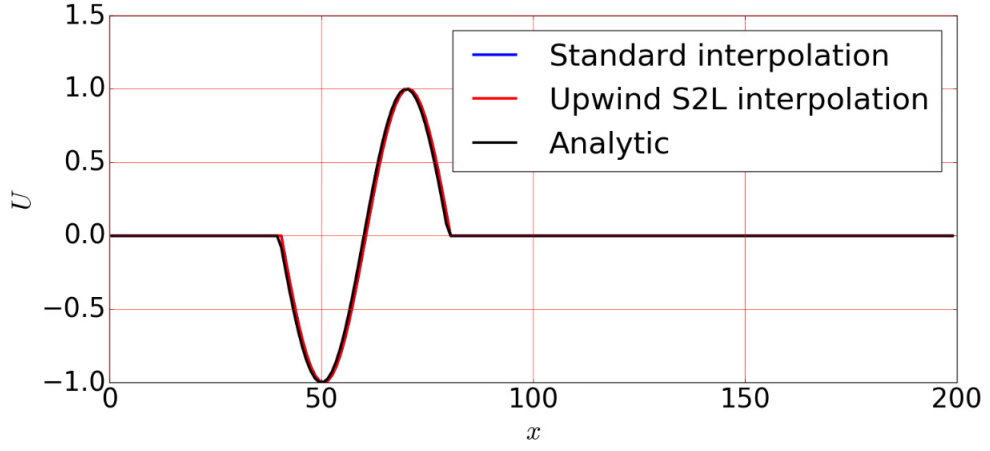
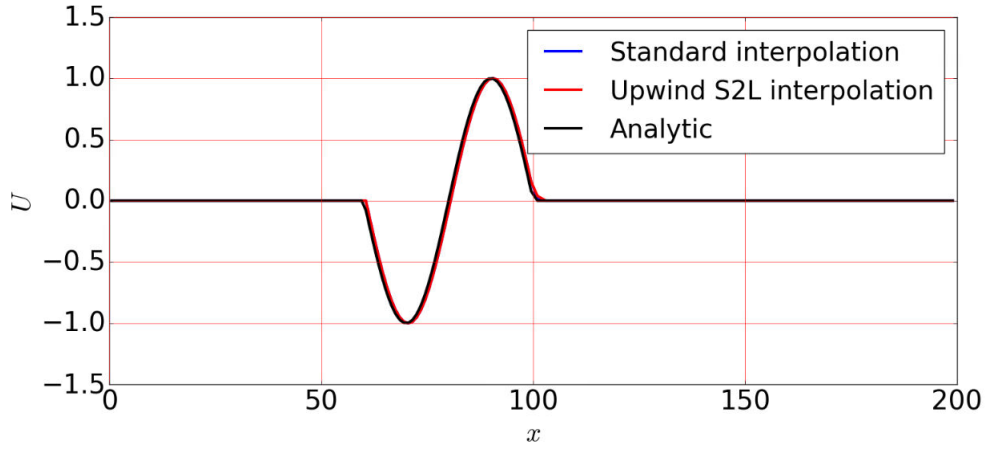
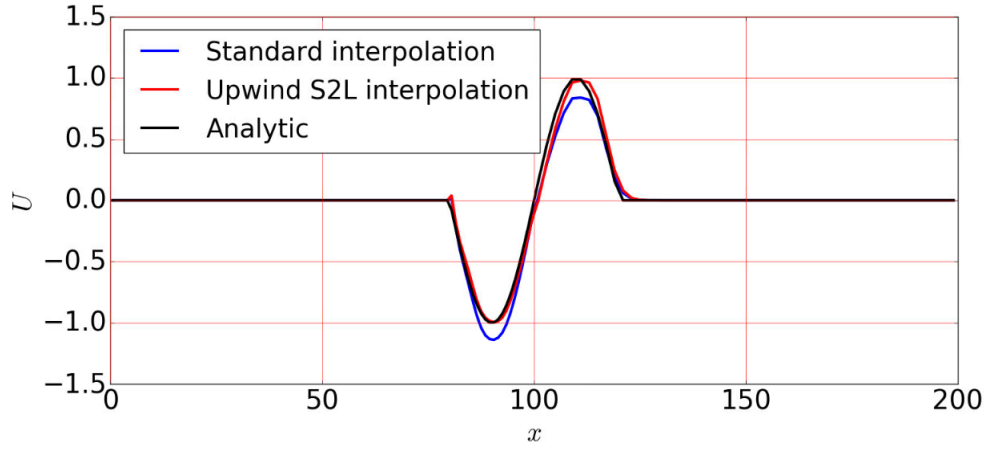


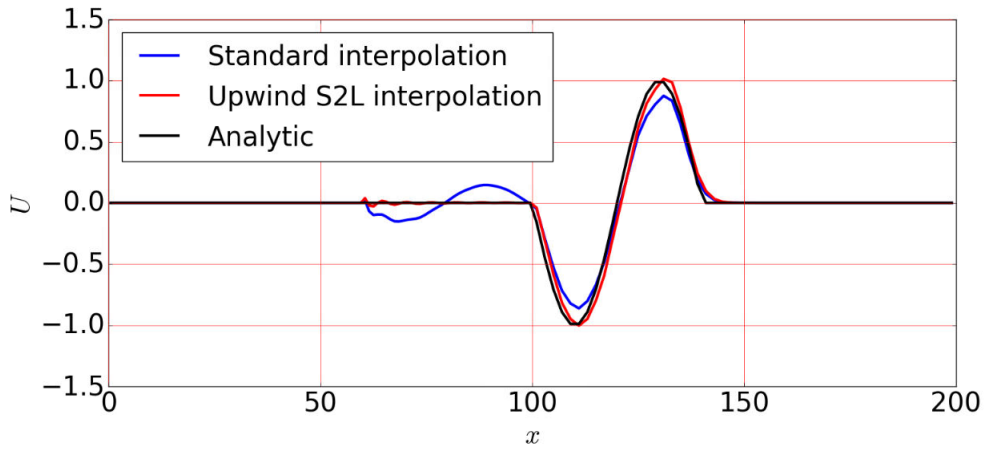
Figure 3.11: Comparing sine waves at time $t = 60$

Figure 3.12: Comparing sine waves at time $t = 80$ Figure 3.13: Comparing sine waves at time $t = 100$

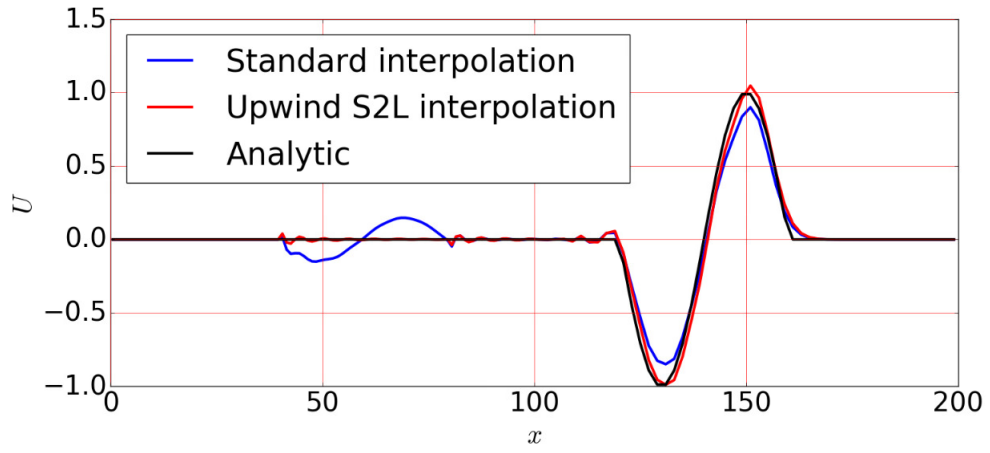
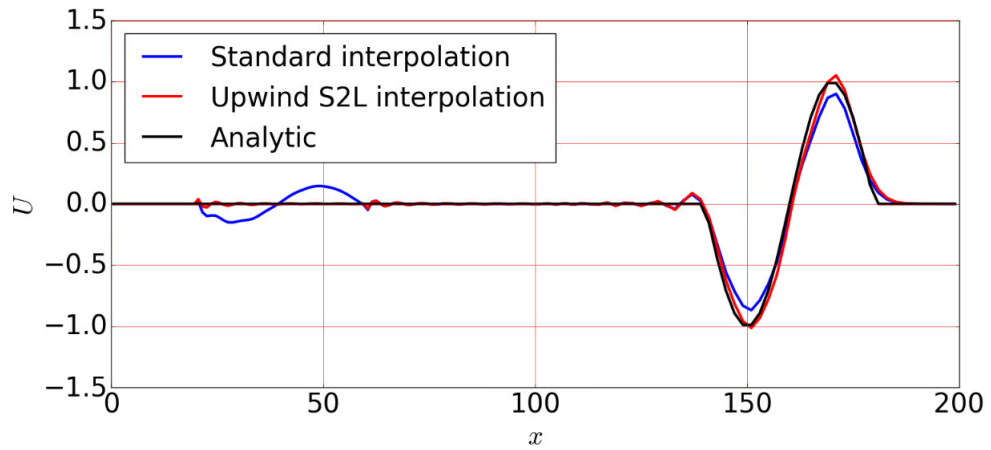
In Figure 3.14, at $t = 120$ the first half of the wave has crossed into the coarse region. For standard interpolation (indicated by blue line) the reflected wave of this half gets added to the latter half of the original wave and the magnitude of the wave amplitude in the trailing half is found to be greater than 1.

Figure 3.14: Comparing sine waves at time $t = 120$

In Figure 3.15, the standard interpolation wave has magnitude lesser than 1 since the reflected component of the sine wave travels back into the fine region.

Figure 3.15: Comparing sine waves at time $t = 140$

Figures 3.16, 3.17 compares the waves at later times $t = 160, 180$ when the wave is completely in the coarse region. It can be confirmed that the wave while using upwind interpolation closely approximates the analytical wave and therefore reduces the error in the domain.

Figure 3.16: Comparing sine waves at time $t = 160$ Figure 3.17: Comparing sine waves at time $t = 180$

While using upwind S2L interpolation, the wave closely approximates the analytical wave. However, a negligible positive phase shift of the wave is observed compared to the analytical wave while using both standard and upwind S2L interpolations.

3.3.5 L2 error

L2 error in U is defined as

$$\frac{\sqrt{\sum_{i,j} \left[(U_{fine-coarse} - U_{analytic})^2 \right]_{i,j}}}{\sqrt{\sum_{i,j} \left[(U_{analytic})^2 \right]_{i,j}}} \quad (3.5)$$

Errors are calculated w.r.t. analytical values in the domain. L2 error in U is calculated for the entire domain while employing both standard and upwind S2L interpolations according to the Equation 3.5. These error values are plotted in Figure 3.18 for better visualization of error variation with time.

The L2 error increases in value while using both interpolations after the leading edge of the wave passes the hanging node interface at $x = 100$ (at $t = 100$) can be observed from Figure 3.18.

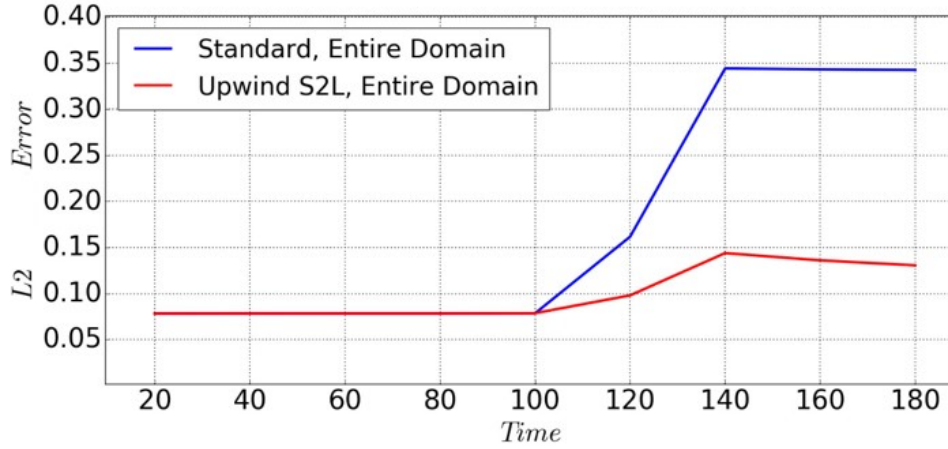


Figure 3.18: L2 error plot

It can also be observed that L2 error values (after the hanging node interface in the domain) while using upwind S2L interpolation are less than half the error value obtained while using standard interpolation. This reduction in error might have been possible because of considering the upwind direction in calculating values at ghost cell in the coarse region.

While using standard interpolation, when the wave passes the hanging node

interface i.e. from $t = 100$ to, $t = 140$ the error in the domain increases since the reflected wave is generated and moves back into the fine domain. Slope is found to increase at, $t = 120$ since the reflected wave from the latter part of the original wave gets added to the domain. These errors and slope are found to be of smaller magnitude in the case of upwind S2L interpolation, since the magnitude of reflected wave generated here is very small.

3.3.6 Summary

A sine wave is propagated in a 2-D domain from fine to coarse region. A reflected wave is observed from the boundary between fine and coarse grid section, i.e. from the 1-D hanging node interface in the domain and travelling back into the finer region while using standard interpolation. The new upwind S2L interpolation successfully reduces the reflected wave from the hanging node interface and maintains the amplitude of the originally propagating sine wave. This is also confirmed from the L2 error values, L2 error plots, which indicates lower error values in the domain while using upwind S2L interpolation.

The successful usage of upwind kind of interpolation for obtaining values at ghost cells in this case could help us in designing better interpolations for other basic flow problems in fluid dynamics like Isentropic vortex convection and Sod shock tube problem. The simulation of these problems on a BCM like domain is discussed in the chapters ahead.

Chapter 4

Isentropic vortex convection

4.1 Introduction

Isentropic vortex convection is chosen since it is one of the basic phenomena in fluid flows. Here, the exact solution corresponding to a pure advection of the vortex at the free stream velocity can be used for comparison of results. Two interpolations, namely, standard interpolation and upwind interpolation are used in Euler equation simulation and results obtained from simulations using these interpolations are discussed.

4.2 Problem setting

An isentropic vortex in the domain is defined by,

$$\begin{aligned}\rho &= \rho_\infty - \frac{(\gamma - 1)b^2}{8\gamma\pi^2} e^{(1-r^2)\frac{1}{(\gamma-1)}}, \\ u &= u_\infty - \frac{b}{2\pi} e^{(1-r^2)/2} (y - y_c), \\ v &= v_\infty + \frac{b}{2\pi} e^{(1-r^2)/2} (x - x_c), \\ p &= \rho^\gamma\end{aligned}\tag{4.1}$$

$$b = 5.0$$

$$r = \sqrt{(x - x_c)^2 + (y - y_c)^2}$$

where in the Equations 4.1, ρ , u , v , p are the density, u -velocity, v -velocity, pressure in the domain. (x_c, y_c) are the coordinates of the center of the vortex. b is the vortex strength and r is the distance of a point in the domain from the vortex center. Grid numbers in different regions of the domain are as shown in Figure 4.1.

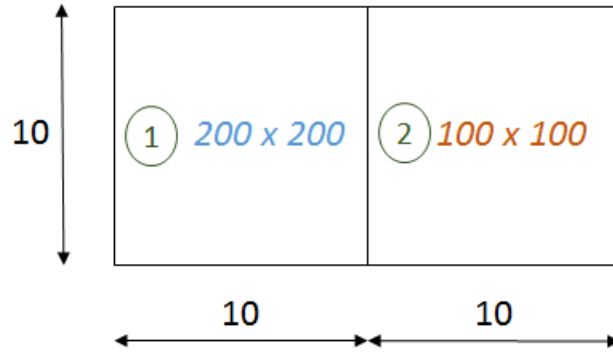


Figure 4.1: Grid specifications in the regions 1 and 2 of the entire domain

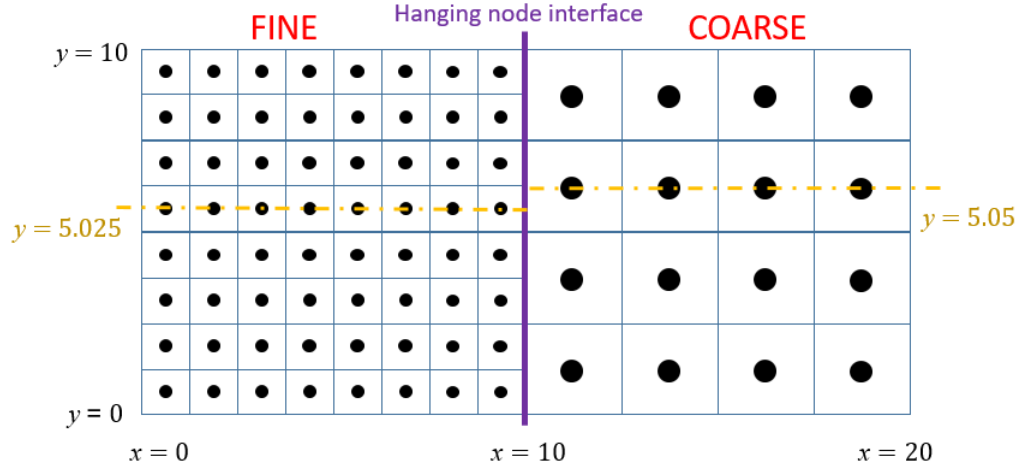


Figure 4.2: Fine and coarse regions in the domain

The vortex is convected through the domain with a jump in size of the squares

along x axis direction (i.e. a hanging node interface along y axis direction) as shown in Figure 4.2. Free-stream density $\rho_\infty = 1$, and free stream velocities $u_\infty = 0.5$, $v_\infty = 0$ are used in the x , y directions, respectively. CFL number 0.1 in the fine regions is used for conducting the simulations.

The initial conditions of the vortex are obtained by substituting values in Equation 4.1 with $x_c = 7.025$, $y_c = 5.025$. Non-reflecting boundary conditions are imposed on all boundaries of the computational domain. The initial pressure distribution of the vortex flow can be seen from Figure 4.3.

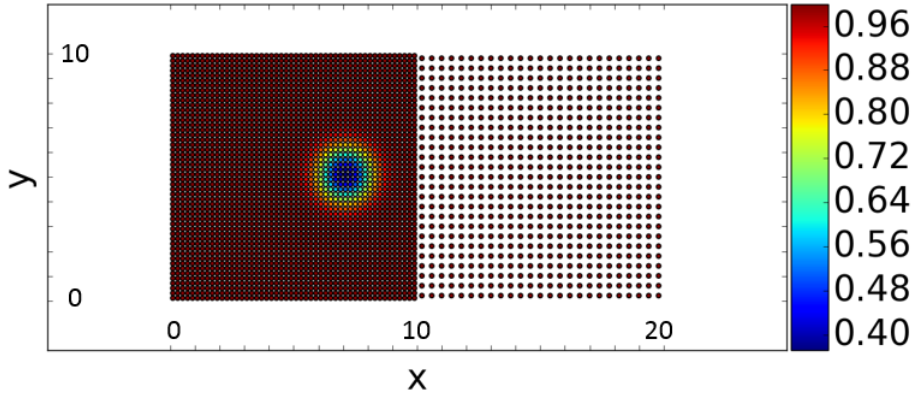


Figure 4.3: Initial pressure distribution

Roe with fifth-order WENO scheme is used to simulate the above-mentioned vortex on a fine (region 1) to coarse region (region 2) with grid sizes as in Table 4.1 which represents the BCM kind domain, using both standard and upwind interpolations. In upwind interpolation, for transfer of values to ghost cells, higher order interpolation is used as discussed in Section 2.3.

Region 1	$\Delta x_1 = 0.05$ $\Delta y_1 = 0.05$
Region 2	$\Delta x_2 = 0.1$ $\Delta y_2 = 0.1$

Table 4.1: Grid sizes in regions 1 and 2

4.3 Results and discussion

4.3.1 Comparing standard and upwind interpolation

In this section, error contours throughout the domain for different variables like pressure (p), u -velocity (u), v -velocity (v) are plotted. The error values for these variables are obtained by comparing the values from simulation (i.e. values obtained from using standard and upwind interpolation) with the analytical values. These analytical values are obtained by advecting the initial vortex with free-stream velocities u_∞, v_∞ in x, y axes directions, respectively. The analytical values are used for comparison since they are the actual values in the domain and therefore most accurate to compare with.

These contours are plotted at random times ($t = 8, 9, 12, 16$) after the vortex center has crossed on to the coarse region of the domain (i.e. the vortex center has advected past the hanging node interface). At time $t = 8$ the vortex center reaches $x = 11.025$ and at time $t = 16$ the vortex center reaches $x = 15.025$.

Comparing the left side and right side plots in Figure 4.4, showing the variation of error in pressure values in the domain while using standard and upwind interpolation, it can be observed that the error contours have smaller magnitude values while using upwind interpolation.

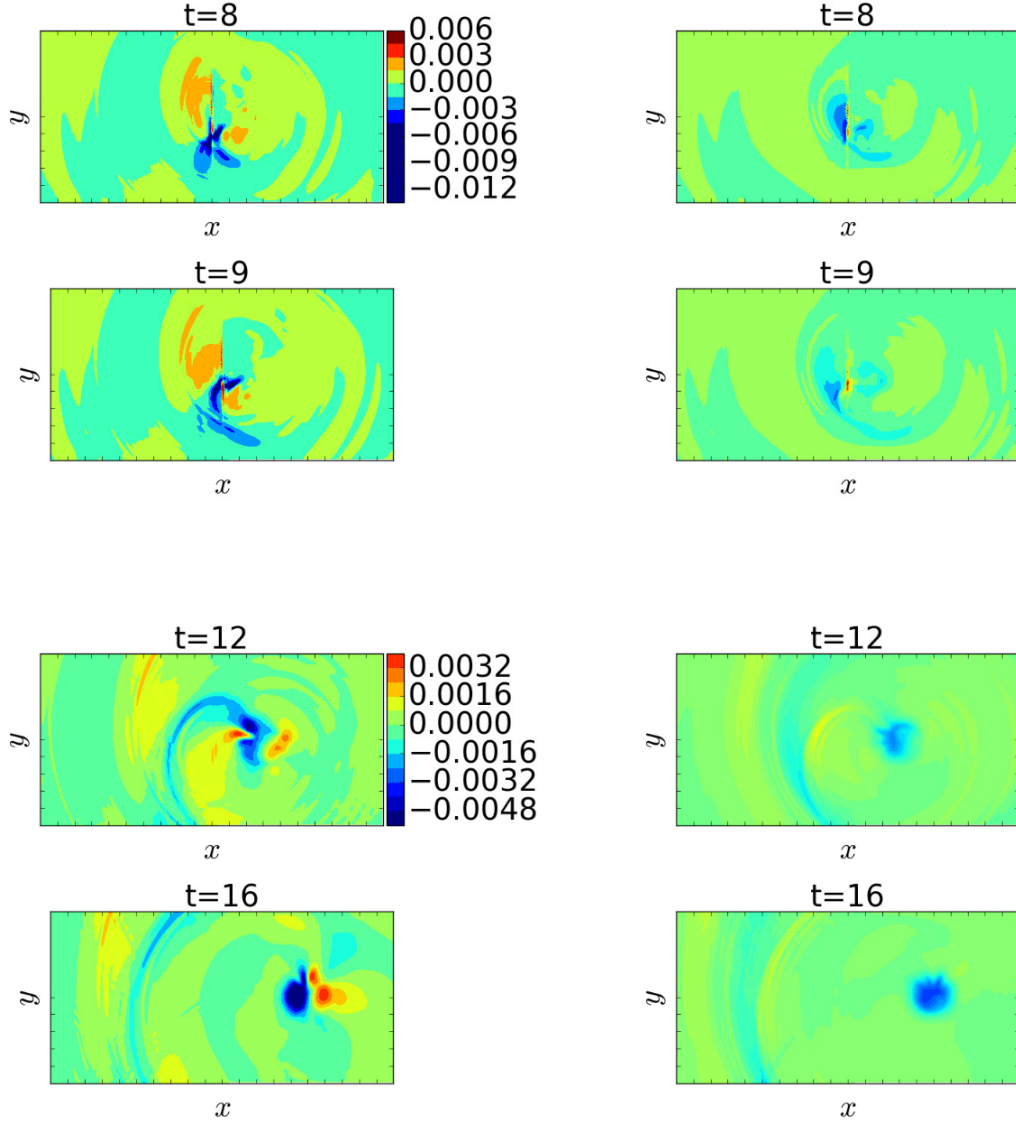


Figure 4.4: Error in pressure (p) throughout the computational domain with standard interpolation (left side plots) and upwind interpolation (right side plots) at $t = 8, 9, 12, 16$

In Figure 4.5, the error in primitive variable u -velocity throughout the domain is compared while using standard and upwind interpolations. The error contours indicates that at all times the error in u -velocity has decreased while using upwind interpolation.

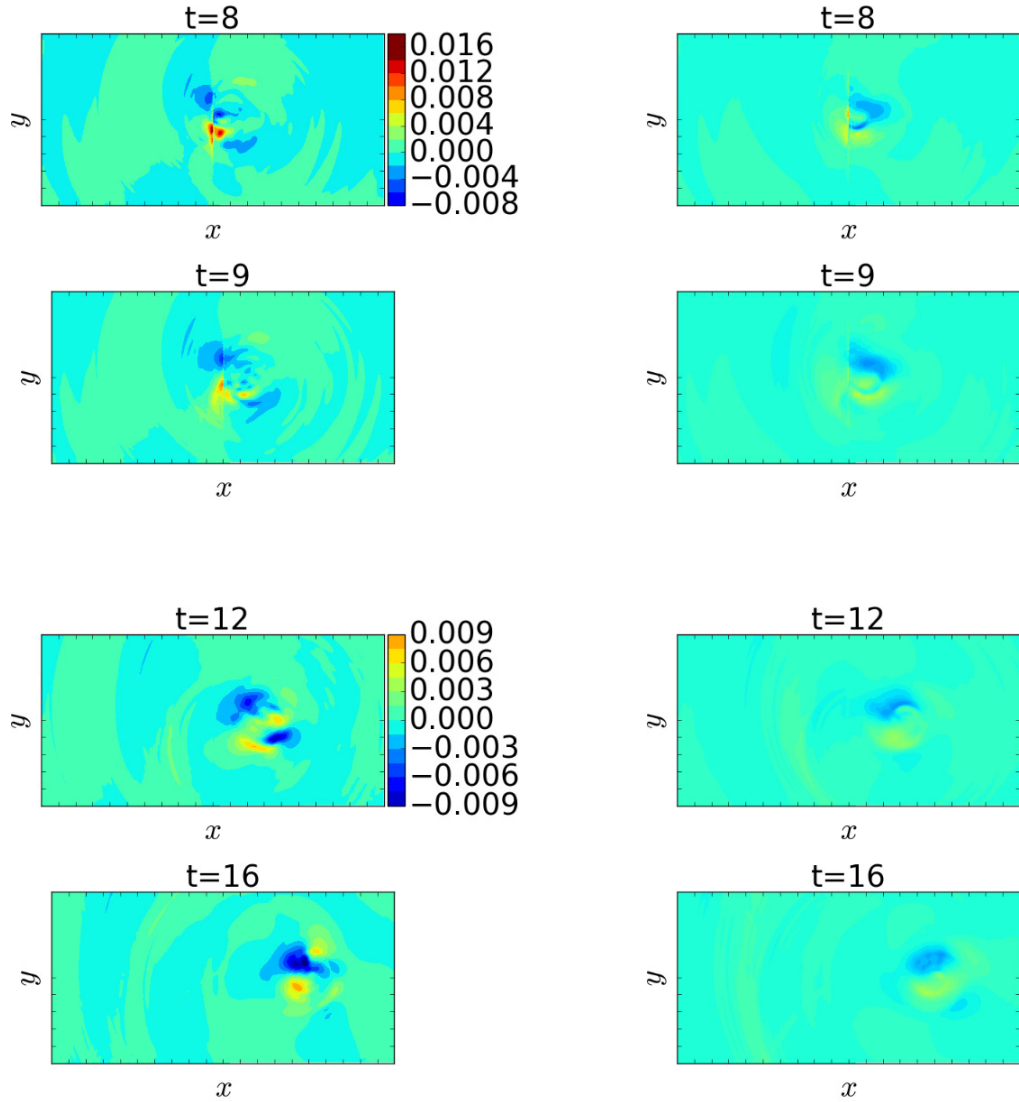


Figure 4.5: Error in u -velocity throughout the computational domain with standard interpolation (left side plots) and upwind interpolation (right side plots) at $t = 8, 9, 12, 16$

In Figure 4.6, the error in primitive variable v -velocity throughout the domain is compared while using standard and upwind interpolations. As evident from the error contours at all times, the error in v -velocity has decreased while using upwind interpolation.

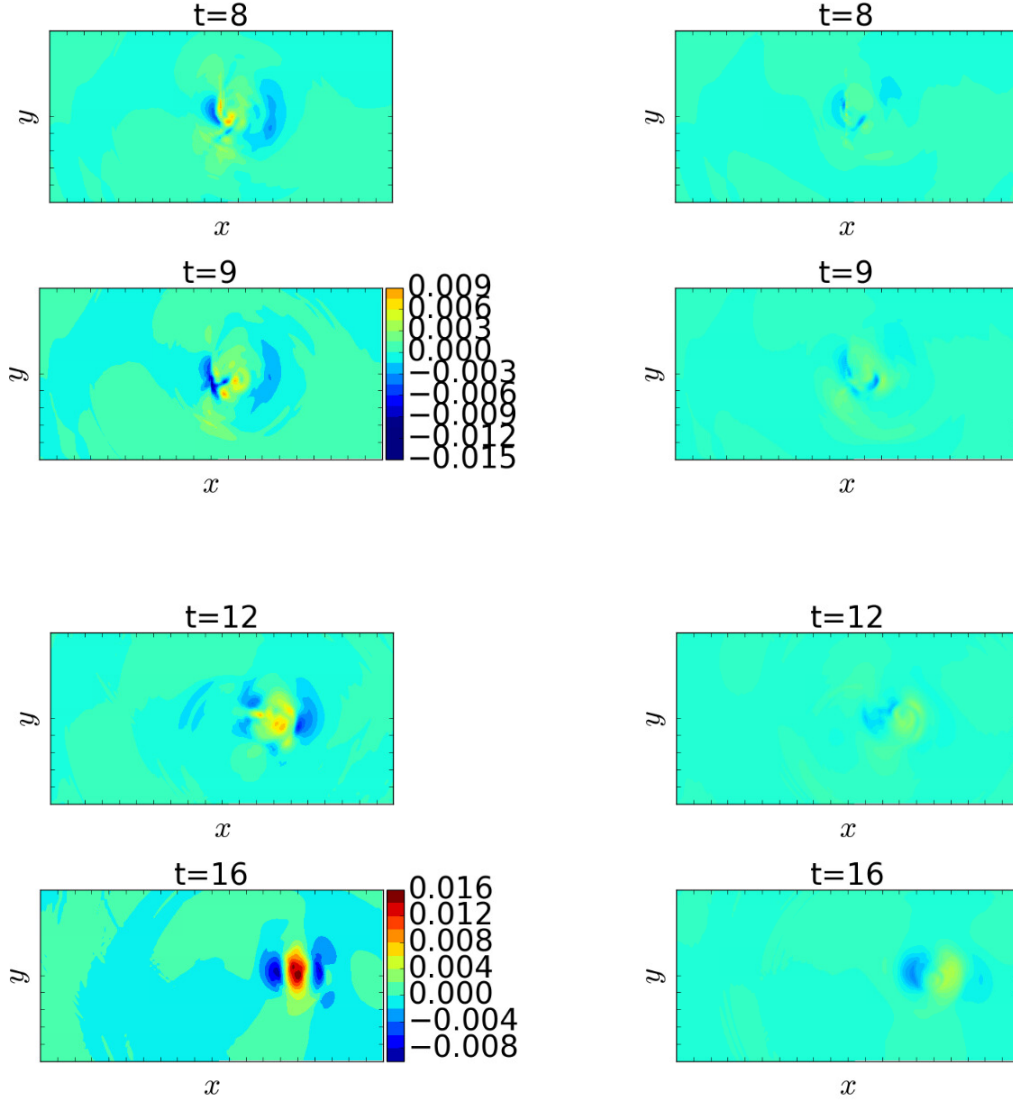


Figure 4.6: Error in v -velocity throughout the computational domain with standard interpolation (left side plots) and upwind interpolation (right side plots) at $t = 8, 9, 12, 16$

To understand whether there is any change in position of vortex while using upwind interpolation as compared to when using standard interpolation, the pressure contour lines from both the cases are plotted. From Figure 4.7, it is confirmed that the vortex center from both the simulations coincide and is at $x = 15.025, y = 5.025$ at time $t = 16$ when the vortex has convected in the x -direction with a u -velocity of 0.5 units.

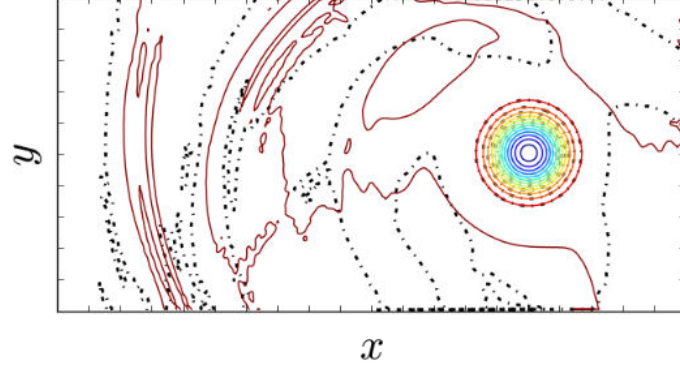


Figure 4.7: Comparing pressure (p) profiles obtained from simulations using standard, upwind interpolations at $t = 16$

4.3.2 Comparison of Δ Pressure distribution at multiple times

For Δ Pressure line plots, pressure values along $y = 5.025$ in region 1 and $y = 5.050$ in region 2 (as shown in Figure 4.2) from the domain are used. The values from different y coordinates in both regions are used, since values along the same y coordinate value cannot be directly obtained (as the cell centers lie on different y coordinate values). So the adjacent y coordinate values which contain cell centers are used.

Different Δ Pressure values are calculated and plotted in the Figures 4.10, 4.11 which are the following,

1. the pressure values obtained from uniform grid simulation values compared w.r.t. exact or analytical pressure values
2. the pressure values obtained from simulation using standard interpolation compared w.r.t. exact or analytical pressure values.
3. the pressure values obtained from simulation using upwind interpolation compared w.r.t. exact or analytical pressure values.
4. the pressure values obtained from simulation using standard interpolation compared w.r.t. values obtained from uniform grid simulations.
5. the pressure values obtained from simulation using upwind interpolation compared w.r.t. values obtained from uniform grid simulations.

The pressure values obtained from simulations using standard and upwind interpolations are also compared with pressure values from uniform grid simulations to confirm that the error in the domain exclusively due to the presence of hanging node interface have reduced while using upwind interpolation.

Uniform grid simulations results are obtained for the fine-coarse BCM like domain by conducting separate simulations for uniform fine grid throughout the entire domain and then for uniform coarse grid throughout the entire domain. Figures 4.8, 4.9 shows the grid details for separate uniform grid simulation.

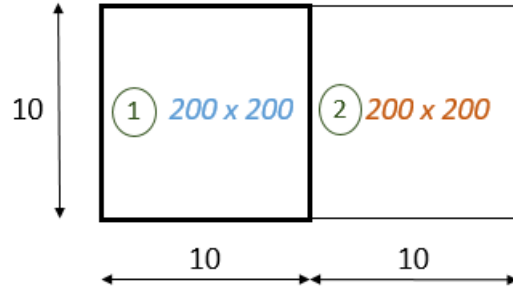


Figure 4.8: Grid specifications for uniform fine grid simulation

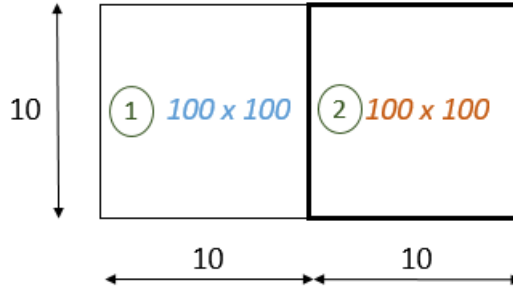


Figure 4.9: Grid specifications for uniform coarse grid simulation

Pressure values from region 1 in Figure 4.8 are combined with pressure values from region 2 in Figure 4.9 to get uniform grid values on a BCM like fine-coarse domain.

The different Δ Pressure values in the domain discussed above are plotted

at time $t = 6$ (when the vortex center has just advected past the hanging node interface at $x = 10$) and at time $t = 16$ (when the whole vortex is well-placed in the coarse region of the domain) as shown in Figures 4.10, 4.11 respectively.

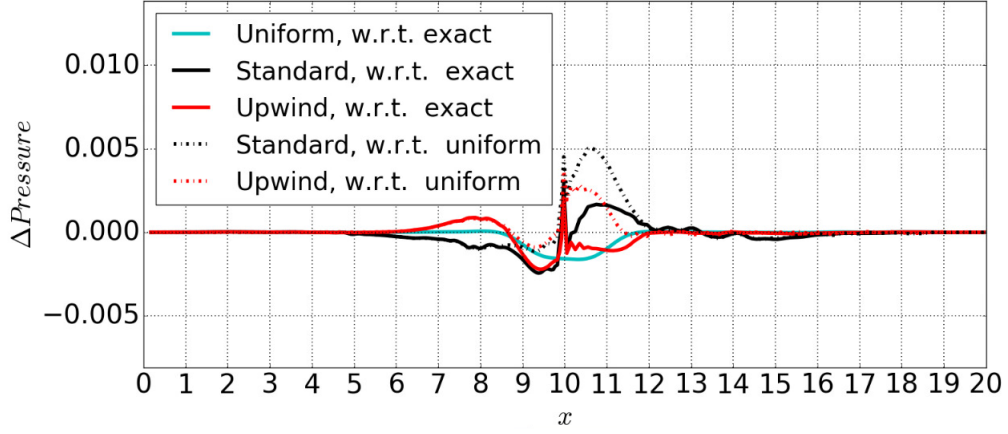


Figure 4.10: Δ Pressure distribution at time $t = 6$

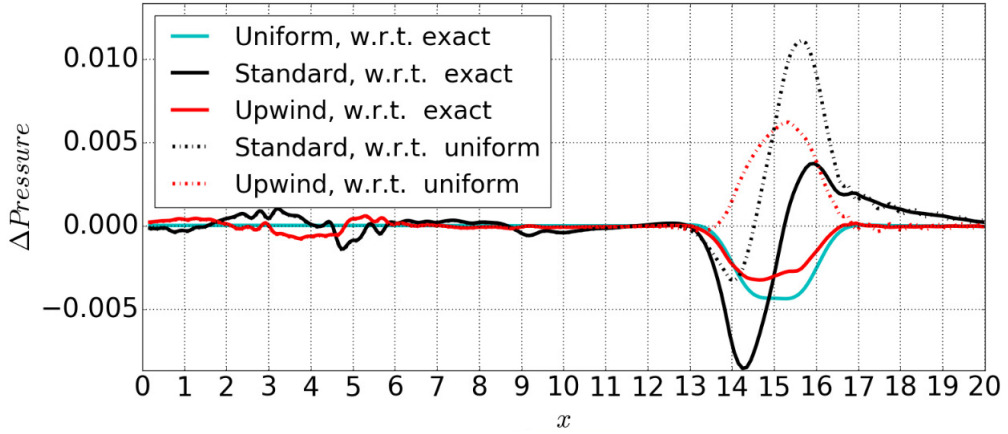


Figure 4.11: Δ Pressure distribution at time $t = 16$

From Figures 4.10, 4.11 we observe that the Δ Pressure values are lesser for upwind interpolation relative to standard interpolation in both the cases, when the comparison is made with exact (or analytical) values and also when the comparison is made with uniform grid simulation values at both time instances.

4.3.3 L2 error

L2 error in p is defined as

$$\frac{\sqrt{\sum_{i,j} \left[(p_{fine-coarse} - p_{analytic})^2 \right]_{i,j}}}{\sqrt{\sum_{i,j} \left[(p_{analytic})^2 \right]_{i,j}}} \quad (4.2)$$

The L2 error of pressure values in the entire domain with respect to analytical pressure values are plotted in Figure 4.12 and the time averaged L2 error values are shown in Table 4.2.

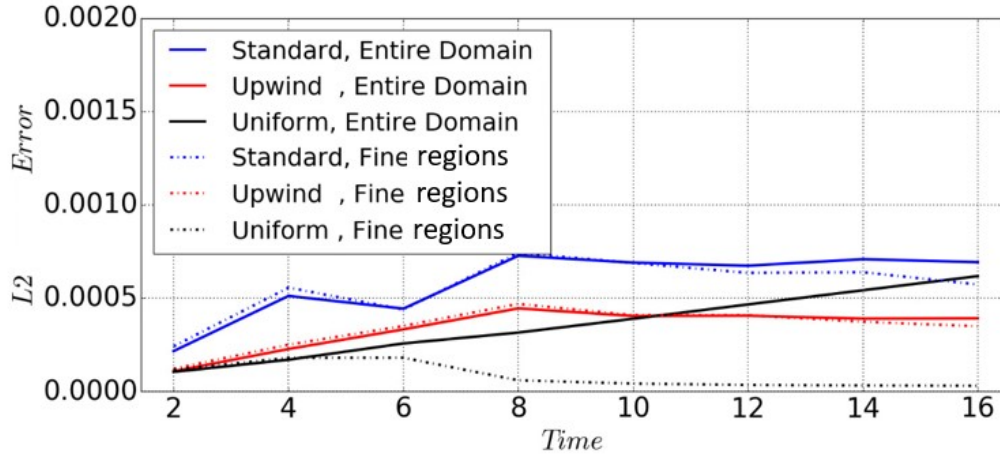


Figure 4.12: L2 error of pressure (p) in the domain

	Standard Interpolation	Upwind Interpolation
Entire domain	5.832568 e-04	3.383543 e-04
Fine regions	5.645808 e-04	3.416721 e-04
Coarse region	6.201789 e-04	3.055668 e-04

Table 4.2: Time averaged L2 error of pressure (p) in the domain for Roe scheme with WENO

Figure 4.12, Table 4.2 confirms that simulations conducted using upwind interpolation have considerably lesser error in the domain (about half the error value) compared to simulations conducted using standard interpolation throughout the entire domain and in fine, coarse regions.

4.3.4 Summary

An isentropic vortex is convected in a 2-D computational domain using standard and upwind interpolation for calculating ghost cell values near the 1-D hanging node interface. Error values in pressure are found to increase as the vortex convects through the domain. From the variable error contour plots, Δ Pressure plots, L2 error in pressure plot and the time-averaged L2 error values, it is observed that the errors in the domain decrease while using upwind interpolation as compared to while using standard interpolation. Therefore, using upwind interpolation for calculating ghost cell value gives more accurate results in simulations.

4.4 Vortex convection with 2-D hanging node interfaces

An isentropic vortex as defined in Section 4.2 is now convected on a domain where there is a change in grid size along both x and y axes of the domain, i.e. hanging node interface is present along both y and x axes directions respectively. Roe with fifth order WENO scheme is used. Simulations are conducted using both standard and upwind interpolations. In upwind interpolation, for transfer of values to ghost cells, higher order interpolation is used as discussed in Section 2.3. Grid numbers in different regions of the domain are as shown in Figure 4.13. Fine, coarse regions and the hanging node interfaces in the domain can be seen from Figure 4.14. The grid spacing in different regions are shown in Table 4.3.

Free-stream density $\rho_\infty = 1$, and free stream velocities $u_\infty = 0.5$, $v_\infty = 0$ in the x , y directions, respectively. The initial conditions of the vortex are obtained by substituting values in Equation 4.1 with $x_c = 7.025$, $y_c = 5.025$. Non-reflecting boundary conditions are imposed on all boundaries of the computational domain. CFL number 0.1 in the fine regions is used for conducting the simulations. The initial pressure distribution of the vortex can be seen from Figure 4.15.

③ <i>200 x 100</i>	④ <i>100 x 50</i>
① <i>200 x 100</i>	② <i>200 x 100</i>

Figure 4.13: Grid specifications in all the regions of the entire domain

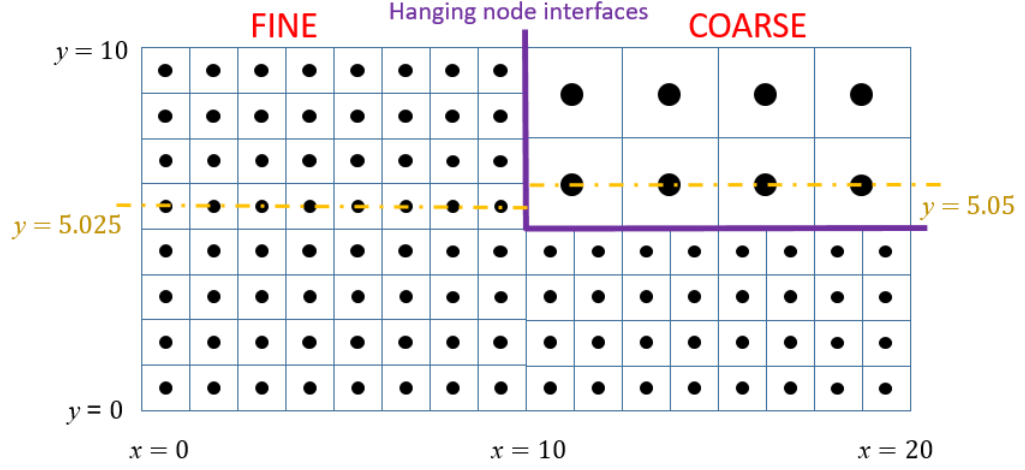


Figure 4.14: Fine and coarse regions in the domain

Region 1	$\Delta x_1 = 0.05$ $\Delta y_1 = 0.05$
Region 2	$\Delta x_2 = 0.05$ $\Delta y_2 = 0.05$
Region 3	$\Delta x_3 = 0.05$ $\Delta y_3 = 0.05$
Region 4	$\Delta x_4 = 0.1$ $\Delta y_4 = 0.1$

Table 4.3: Grid sizes in regions 1,2,3,4

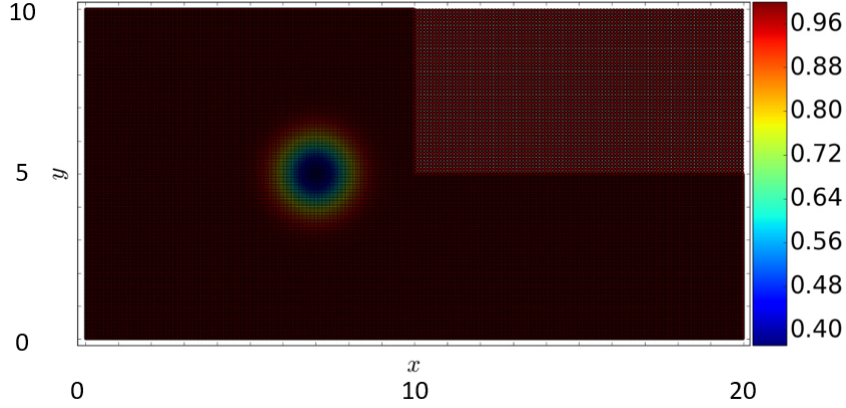


Figure 4.15: Initial pressure distribution

4.4.1 Results and discussion

Comparing standard and upwind interpolation

In this section, error contours throughout the domain for different variables like pressure (p), u -velocity (u), v -velocity (v) are plotted. The error values for these variables are obtained by comparing the values from simulation (i.e. values obtained from using standard and upwind interpolation) with the analytical values. These analytical values are obtained by advecting the initial vortex with free-stream velocities u_∞, v_∞ in x, y axes directions, respectively. The analytical values are used for comparison since they are the actual values in the domain and therefore most accurate to compare with.

These contours are plotted at random times ($t = 8, 9, 12, 16$) after the vortex center has crossed on to the coarse region of the domain (i.e. the vortex center has advected past the hanging node interface along y direction). At time $t = 8$ the vortex center reaches $x = 11.025$ and at time $t = 16$ the vortex center reaches $x = 15.025$.

Comparing the left side and right side plots in Figure 4.16, shows that the variation of error in pressure values in the domain while using standard and upwind interpolation, it can be observed that the error contours have lesser magnitude values while using upwind interpolation.

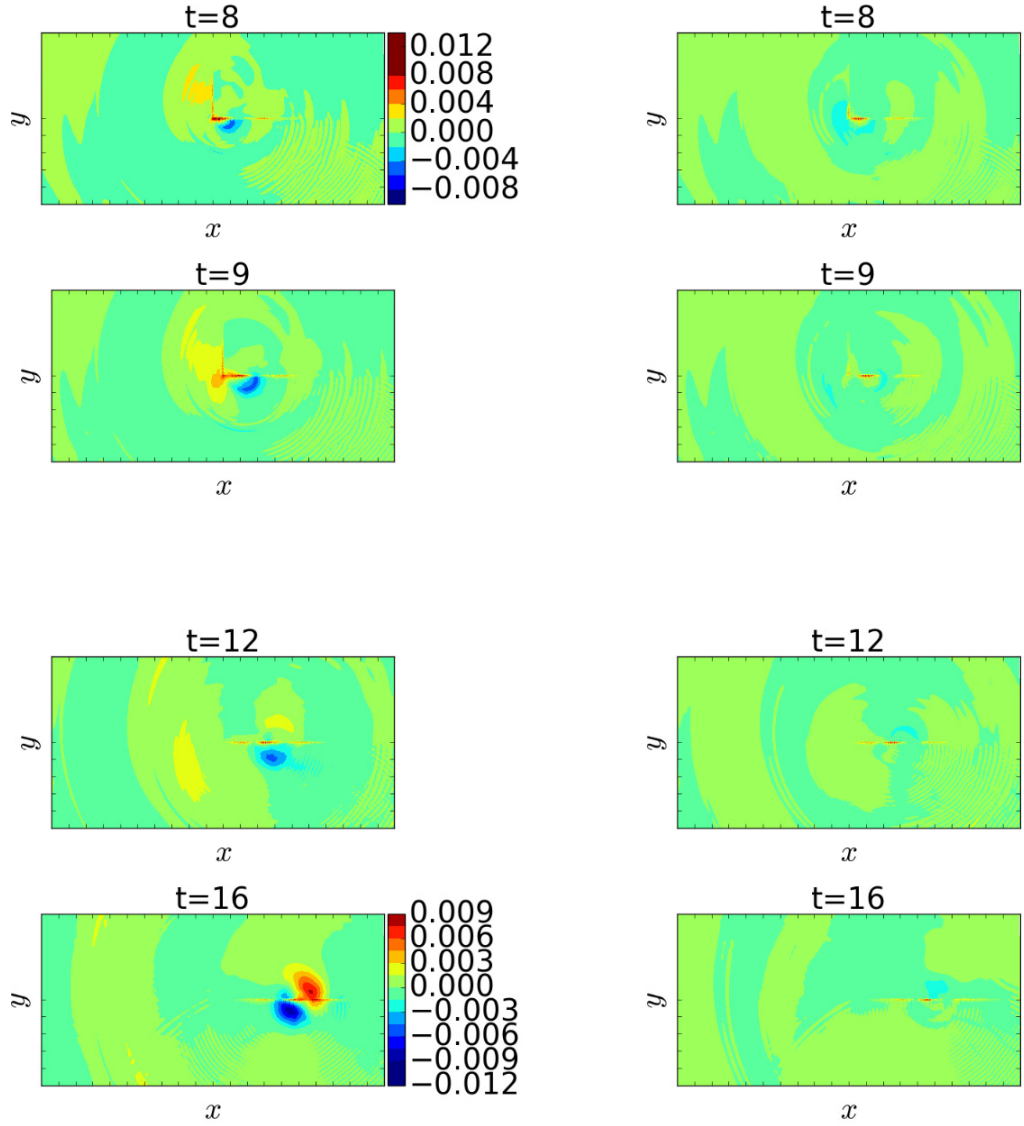


Figure 4.16: Error in pressure (p) throughout the computational domain with standard interpolation (left side plots) and upwind interpolation (right side plots) at $t = 8, 9, 12, 16$

In Figure 4.17, the error in primitive variable u -velocity throughout the domain is compared while using standard and upwind interpolations. The error contours indicate that at all times the error in u -velocity has decreased while using upwind interpolation.

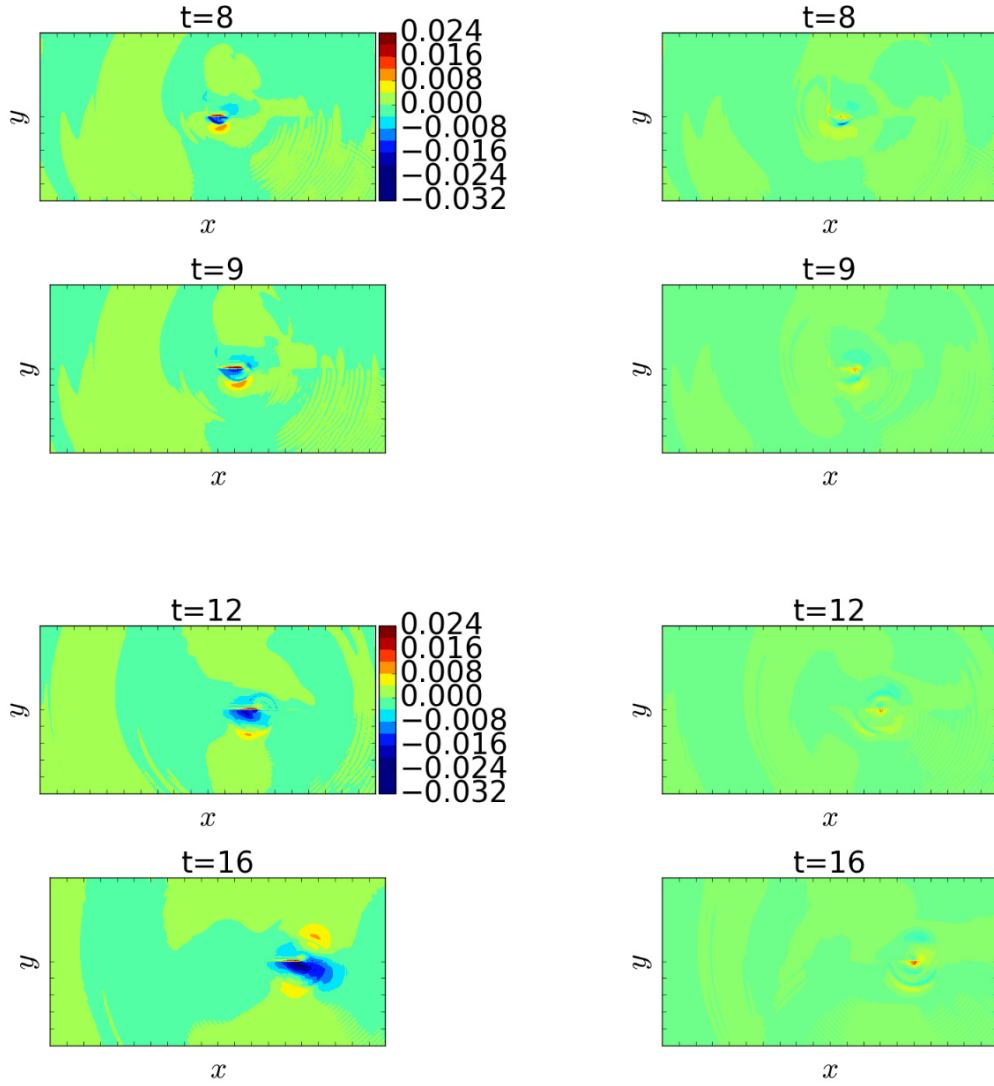


Figure 4.17: Error in u -velocity throughout the computational domain with standard interpolation (left side plots) and upwind interpolation (right side plots) at $t = 8, 9, 12, 16$

In Figure 4.18, the error in primitive variable v -velocity throughout the domain is compared while using standard and upwind interpolations. As evident from the error contours at all times, the error in v -velocity has decreased while using upwind interpolation.

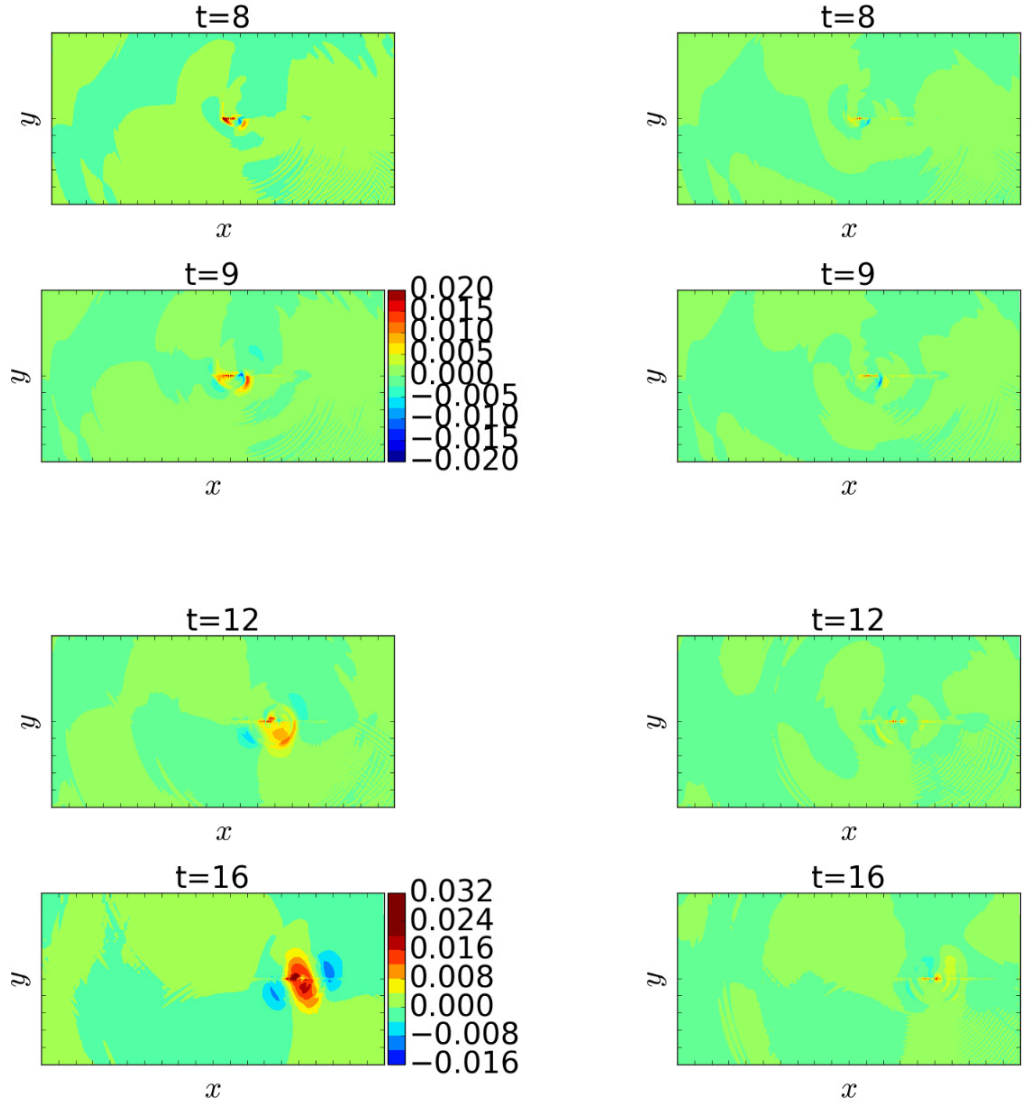


Figure 4.18: Error in v -velocity throughout the computational domain with standard interpolation (left side plots) and upwind interpolation (right side plots) at $t = 8, 9, 12, 16$

To understand whether there is any change in position of vortex while using upwind interpolation as compared to when using standard interpolation, the pressure contour lines from both the cases are plotted. From Figure 4.19, it is confirmed that the vortex center from both the simulations coincide and is at $x = 15.025, y = 5.025$ at time $t = 16$ when the vortex has convected in the x -direction with a u -velocity of 0.5 units.

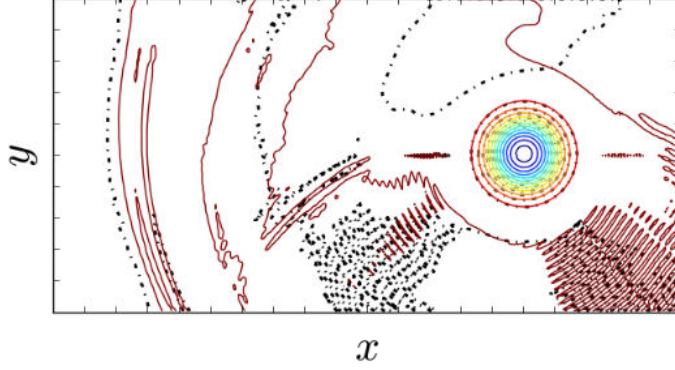


Figure 4.19: Comparing pressure (p) profiles obtained from simulations using standard, upwind interpolations at $t = 16$

Comparison of Δ Pressure distribution at multiple times

For Δ Pressure line plots, pressure values along $y = 5.025$ in region 1 and $y = 5.050$ in region 4 (as shown in Figure 4.14) from the domain are used. The values from different y coordinates in both regions are used, since values along the same y coordinate value cannot be directly obtained (as the cell centers lie on different y coordinate values). So the adjacent y coordinate values which contain cell centers are used.

Different Δ Pressure values are calculated and plotted in the Figures 4.22, 4.23 which are the following,

1. the pressure values obtained from uniform grid simulation values compared w.r.t. exact or analytical pressure values
2. the pressure values obtained from simulation using standard interpolation compared w.r.t. exact or analytical pressure values.
3. the pressure values obtained from simulation using upwind interpolation compared w.r.t. exact or analytical pressure values.
4. the pressure values obtained from simulation using standard interpolation compared w.r.t. values obtained from uniform grid simulations.
5. the pressure values obtained from simulation using upwind interpolation compared w.r.t. values obtained from uniform grid simulations.

The pressure values obtained from simulations using standard and upwind interpolations are also compared with pressure values from uniform grid simulations to confirm that the error in the domain exclusively due to the presence of hanging node interface have reduced while using upwind interpolation. Uniform grid simulations results are obtained for the fine-coarse BCM like domain by conducting separate simulations for uniform fine grid throughout the entire domain and then for uniform coarse grid throughout the entire domain. Figures 4.20, 4.21 shows the grid details for separate uniform grid simulation.

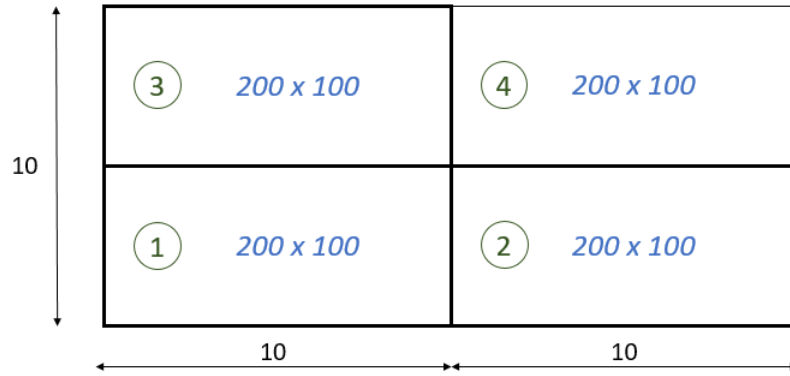


Figure 4.20: Grid specifications for uniform fine grid simulation

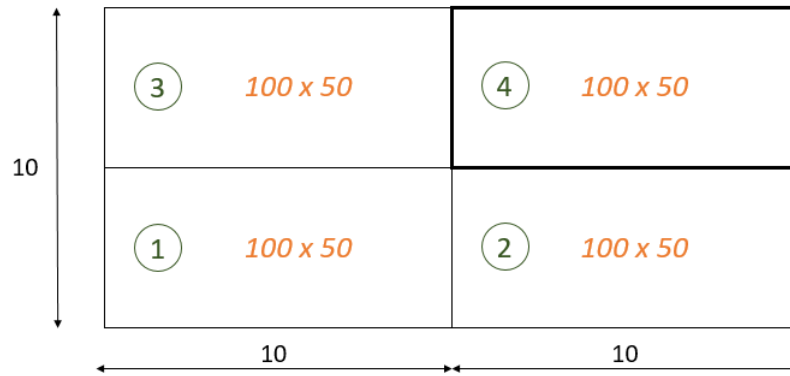


Figure 4.21: Grid specifications for uniform coarse grid simulation

Pressure values from regions 1,2,3 in Figure 4.20 are combined with pressure values from region 4 in Figure 4.21 to get uniform grid values on a BCM like fine-coarse domain.

The different $\Delta\text{Pressure}$ values in the domain discussed above are plotted at time $t = 6$, when the vortex center has just advected past the hanging node interface at $x = 10$ and at time $t = 16$ when a part of the vortex is placed in the coarse region of the domain is plotted in Figures 4.22, 4.23 respectively.

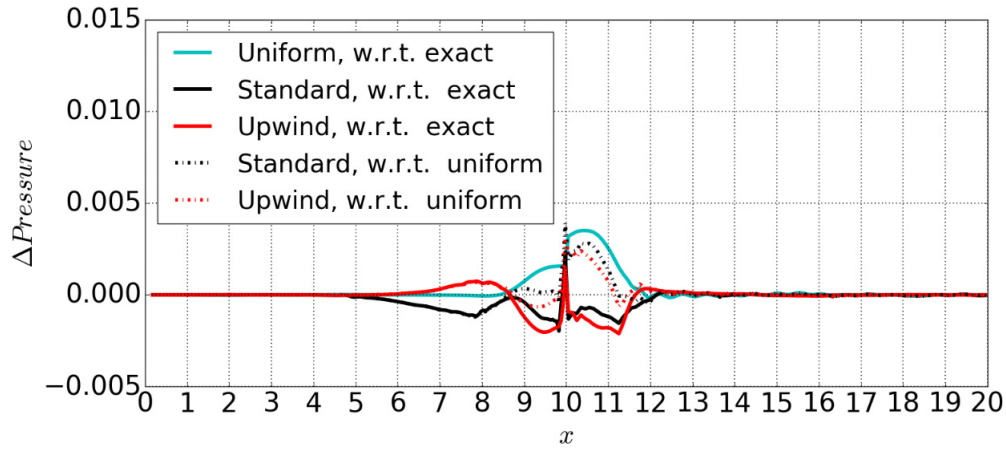


Figure 4.22: $\Delta\text{Pressure}$ distribution at time $t = 6$

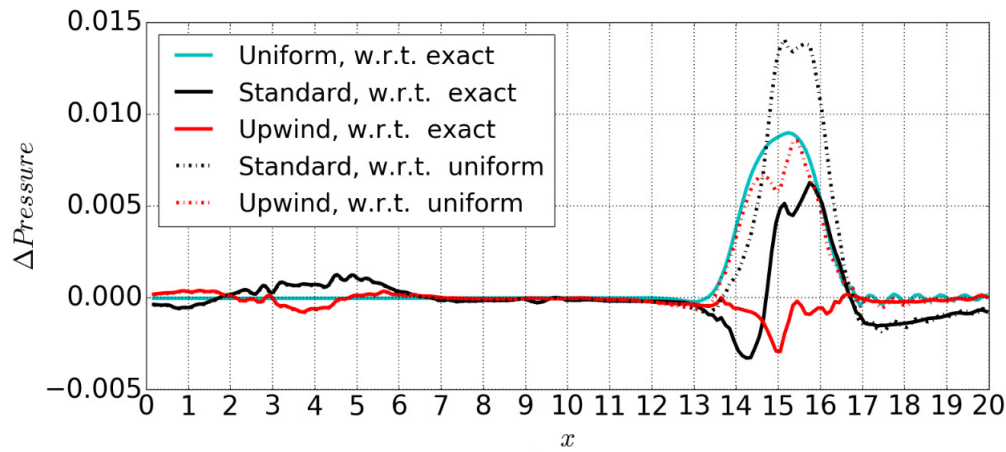


Figure 4.23: $\Delta\text{Pressure}$ distribution at time $t = 16$

From Figures 4.22, 4.23 we observe that the Δ Pressure values are lesser for upwind interpolation relative to standard interpolation in both the cases, when the comparison is made with exact (or analytical) values and also when the comparison is made with uniform grid simulation values at both time instances.

L2 error

L2 error in p is defined as

$$\frac{\sqrt{\sum_{i,j} \left[(p_{fine-coarse} - p_{analytic})^2 \right]_{i,j}}}{\sqrt{\sum_{i,j} \left[(p_{analytic})^2 \right]_{i,j}}} \quad (4.3)$$

The L2 error of pressure values in the entire domain with respect to analytical pressure values are plotted in Figure 4.24 and the time averaged L2 error values are shown in Table 4.4.

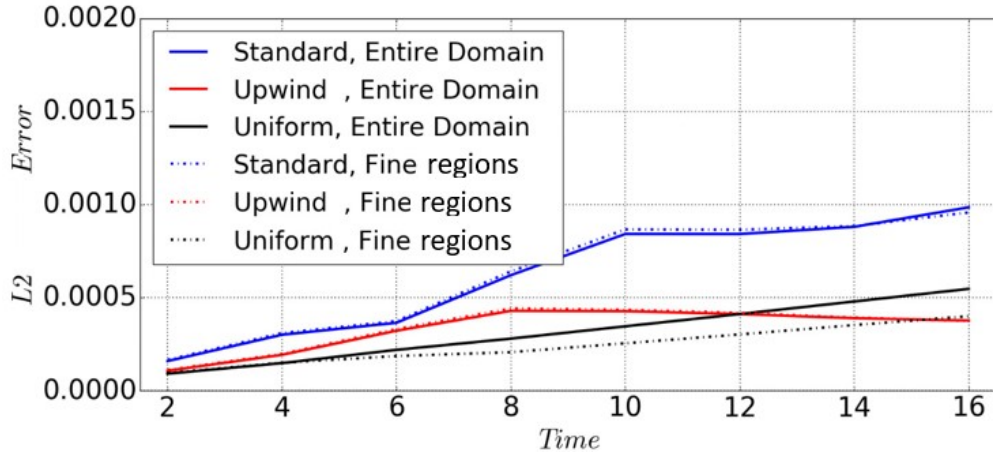


Figure 4.24: L2 error of pressure (p) in the domain

	Standard Interpolation	Upwind Interpolation
Entire domain	6.244498 e-04	3.322682 e-04
Fine regions	6.330758 e-04	3.379522 e-04
Coarse region	4.628784 e-04	2.445045 e-04

Table 4.4: Time averaged L2 error of pressure (p) in the domain for Roe scheme with WENO

Figure 4.24, Table 4.4 confirms that simulations conducted using upwind interpolation have lesser error in the domain (about half the error value) compared to simulations conducted using standard interpolation throughout the entire domain and in fine, coarse regions.

4.4.2 Summary

An isentropic vortex is convected in the 2-D computational domain using standard and upwind interpolation for calculating ghost cell values near the 2-D hanging node interface. Error values in pressure are found to increase as the vortex convects through the domain. From the variable profile plots, Δ Pressure plots, L2 error in pressure plot and the time-averaged L2 error values, it is observed that the errors in the domain decrease while using upwind interpolation as compared to while using standard interpolation and therefore using upwind interpolation for calculating ghost cell values gives more accurate results in simulations.

Chapter 5

Shock wave propagation

5.1 Introduction

Sod shock tube problem is chosen since it is one of the standard problems in gas dynamics and a widely used problem in Riemann solvers. Since the exact solution is known, it can be used for simulation results' comparison[16]. A 2-D shock tube is considered here for investigating the 2-D hanging node interface problem. Two interpolations, namely, standard interpolation and upwind interpolation are used in Euler equation simulation and results obtained from the simulations using these interpolations are discussed.

5.2 Sod shock tube Riemann problem

Sod shock tube is originally a 1-D tube closed at both the ends and divided into equal areas or regions by a thin diaphragm. Each region is filled with the same gas having different thermodynamic properties, namely density, pressure and velocity. Upon sudden breaking of the thin diaphragm, there occurs a high speed flow of gas, which propagates into the low-pressure region[19].

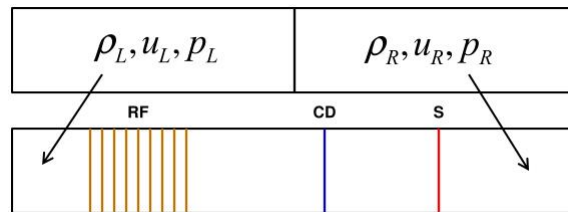


Figure 5.1: Waves in shock tube

At time, $t = 0$ the diaphragm breaks, generating a process that naturally tends to equalize the pressure in the tube. The gas at high pressure (on the left) expands through an expansion or rarefaction wave and flows into the low-pressure region (on the right) pushing the gas here. The rarefaction is a continuing process and takes place inside a well-defined region (the expansion fan) that propagates to the left; the width of the expansion fan grows in time[15]. All the viscous effects are negligible along the tube

The compression of low-pressure gas generates a shock wave which propagates to the right. The expanded gas is separated from the compressed gas by a contact discontinuity, which can be conceived as a fictitious membrane travelling to the right at a constant speed. The point to be observed is that some of the thermodynamic parameters across the shock wave and contact discontinuity are discontinuous, and these discontinuities makes this problem difficult[15].

5.3 Problem setting

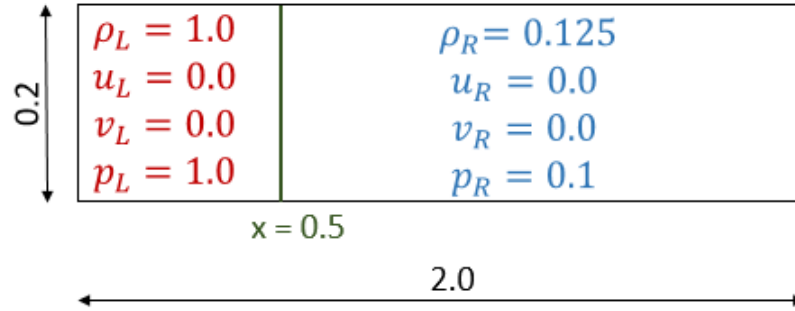


Figure 5.2: Initial states in the shock tube

In this research, a 2-D shock tube is considered. The initial states inside the shock tube are as shown in Figure 5.2. The left and right states in the shock tube are separated at $x = 0.5$. In order to replicate the BCM domain, the computational domain for shock tube is divided into 4 regions as shown in Figure 5.3. The regions 1, 2, 3 are the fine regions which means they have the smaller cells (smaller cell spacing) while region 4 is the coarse region which has larger cells (with double cell spacing as smaller cells).

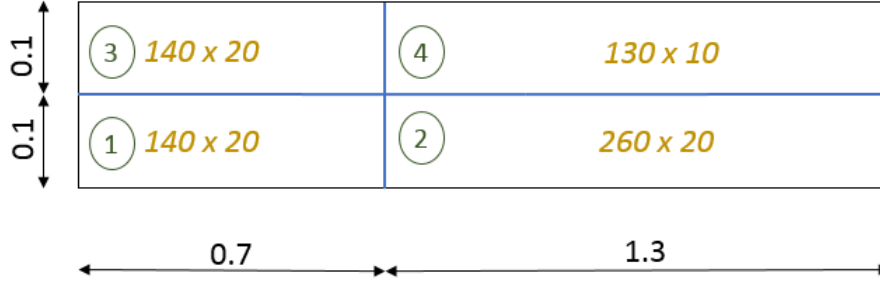


Figure 5.3: Grid distribution

Region 1	$\Delta x_1 = 0.005$ $\Delta y_1 = 0.005$
Region 2	$\Delta x_2 = 0.005$ $\Delta y_2 = 0.005$
Region 3	$\Delta x_3 = 0.005$ $\Delta y_3 = 0.005$
Region 4	$\Delta x_4 = 0.01$ $\Delta y_4 = 0.01$

Table 5.1: Grid sizes in regions 1,2,3,4

The simulations are carried out by using Roe scheme with MUSCL initially and then using Roe scheme with WENO. Both standard and upwind interpolation simulations are conducted. For upwind interpolation, a low order interpolation is used for transferring values to ghost cells, as discussed in Section 2.3. CFL number used in fine region is 0.1. The simulation is conducted till time $t = 0.85$ when the shock wave reaches approximately $x = 1.9925$. The shock wave reaches the interface of fine and coarse grids $x = 0.7$ at approximately $t = 0.11$.

Grid spacing in different regions are shown in Table 5.1. Since there is grid size jump along both x and y axes, 2-D hanging node interface is present, i.e. hanging node interfaces along both y and x axes directions, respectively.

5.4 Results and discussion

In this section, the results of simulation using standard and upwind interpolation for calculating ghost cell values are compared with analytical values in the domain, since they are the most accurate values to be compared with. In Section 5.4.1, 5.4.2, error in pressure (p) contours in the domain at a random time $t = 0.60$ after the shock wave has crossed the hanging node interface is used for comparison.

5.4.1 Comparing standard and upwind interpolation for Roe with MUSCL

Figures 5.4, 5.5 compares the error contours in pressure (p) at time $t = 0.60$ when shock wave reaches $x = 1.5575$ for Roe scheme with MUSCL. The difference in error contour is not clearly visible from these plots, but the maximum error observed is lower in simulation using upwind interpolation.

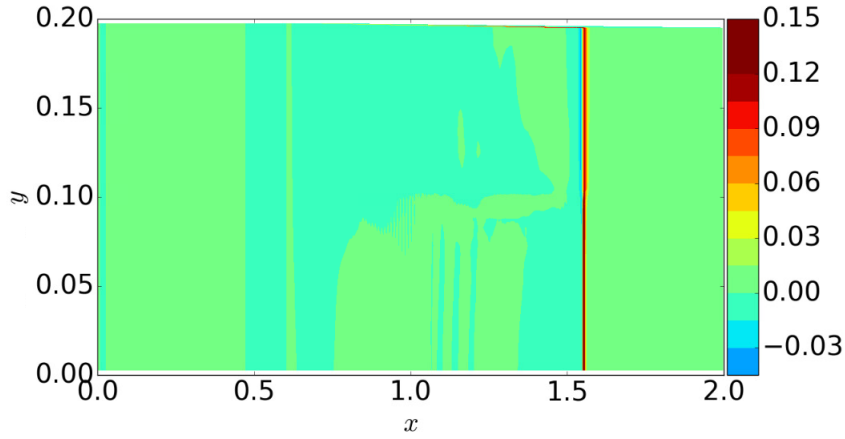


Figure 5.4: Error in pressure (p) throughout the computational domain with standard interpolation at $t = 0.60$

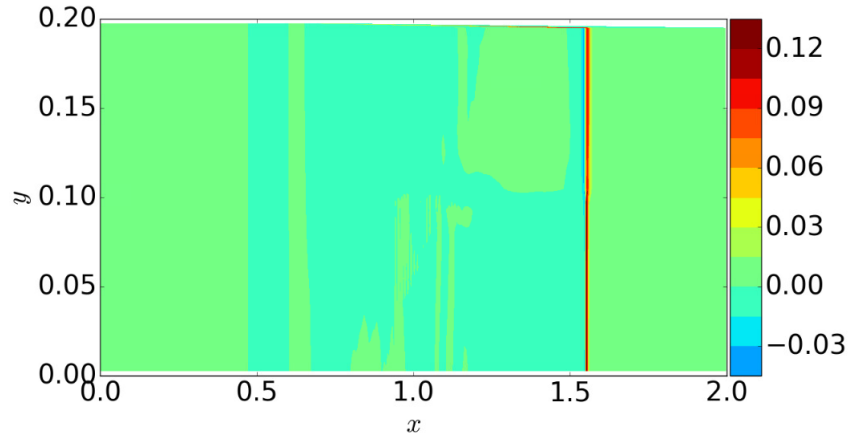


Figure 5.5: Error in pressure (p) throughout the computational domain with upwind interpolation at $t = 0.60$

5.4.2 Comparing standard and upwind interpolation for Roe with WENO

Figures 5.6, 5.7 compares the error contours in pressure (p) at time $t = 0.60$ when shock wave reaches $x = 1.5575$ for Roe scheme with WENO. The difference in error contour is not clearly visible from these plots, but the maximum error observed is lower in simulation using upwind interpolation.

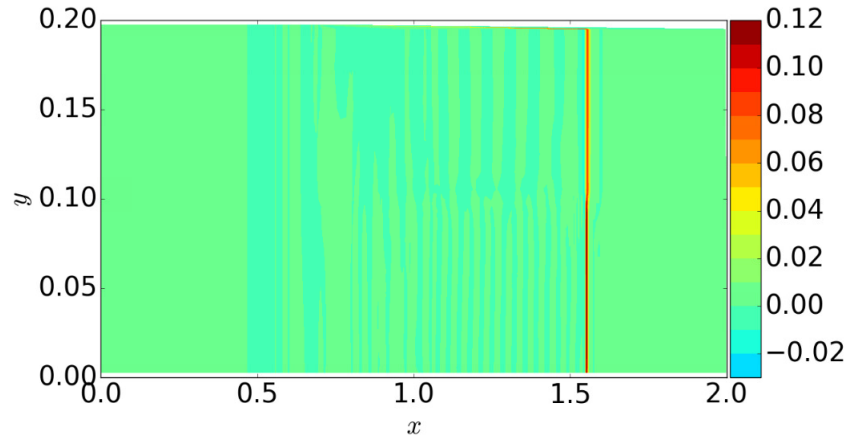


Figure 5.6: Error in pressure (p) throughout the computational domain with standard interpolation at $t = 0.60$

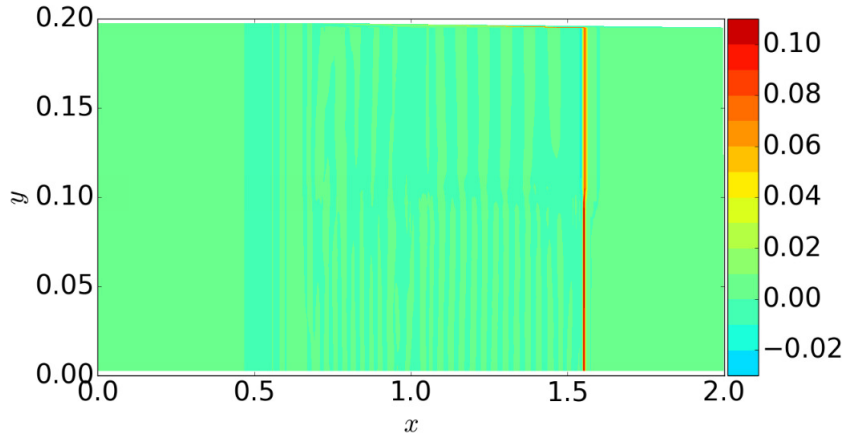


Figure 5.7: Error in pressure (p) throughout the computational domain with upwind interpolation at $t = 0.60$

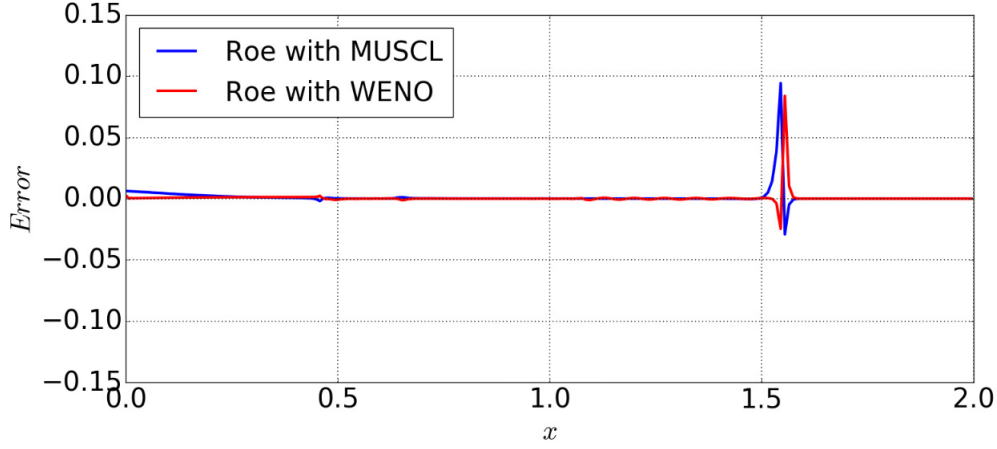


Figure 5.8: Error in pressure (p) while using both schemes with standard interpolation at $t = 0.60$

Figure 5.8 shows the error in pressure (p) obtained along a constant y value line across the regions 3 (fine), 4 (coarse) in the domain while using Roe with MUSCL, a low order scheme and Roe with WENO, a higher order scheme. The error in the fine region and also the reflection observed (to the left of hanging node interface at $x = 0.7$) is of the same order, from which we conclude that the higher order scheme doesn't produce a higher reflection onto the fine region in this shock tube simulation.

5.4.3 L2 error

L2 error in p is defined as

$$\frac{\sqrt{\sum_{i,j} \left[(p_{fine-coarse} - p_{analytic})^2 \right]_{i,j}}}{\sqrt{\sum_{i,j} \left[(p_{analytic})^2 \right]_{i,j}}} \quad (5.1)$$

Errors are calculated w.r.t. analytical values in the domain.

L2 error for simulations using Roe scheme with MUSCL

Figure 5.9 shows the L2 error in pressure (p) throughout the domain from time $t = 0$ to $t = 0.85$ when the shock wave has almost reached the end of the shock tube. Roe scheme with MUSCL implementation is used for simulations conducted separately with standard and upwind interpolation. This figure also shows the L2 error history in the fine regions in the computational domain. It can be observed that the error values decrease while using upwind interpolation in both the cases.

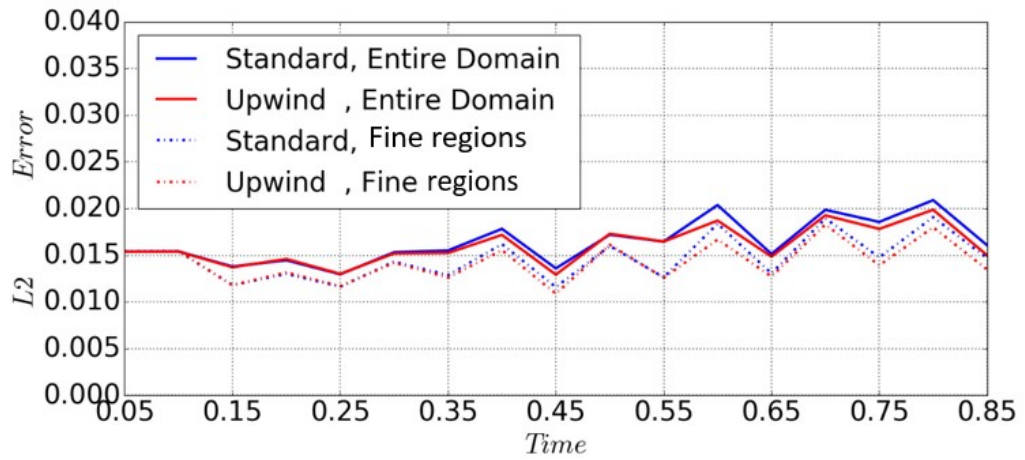


Figure 5.9: L2 error of pressure (p) in the domain at different times for Roe scheme with MUSCL

Table 5.2 shows the time-averaged L2 errors of pressure throughout the entire domain and in the fine regions. These values help us to understand that the errors in the entire domain and fine regions have reduced while using upwind interpolation.

	Standard Interpolation	Upwind Interpolation
Entire domain	0.016403	0.015997
Fine regions	0.014697	0.014267

Table 5.2: Time averaged L2 error of pressure (p) in the domain for Roe scheme with MUSCL

L2 error for simulations using Roe scheme with WENO

Figure 5.10 shows the L2 error in pressure (p) throughout the domain from time $t = 0$ to $t = 0.85$. Here, Roe scheme with WENO implementation is used for simulations conducted separately with standard and upwind interpolation.

This figure also shows the L2 error history in the fine regions in the computational domain. It can be observed that the error values decrease while using upwind interpolation in both the cases.

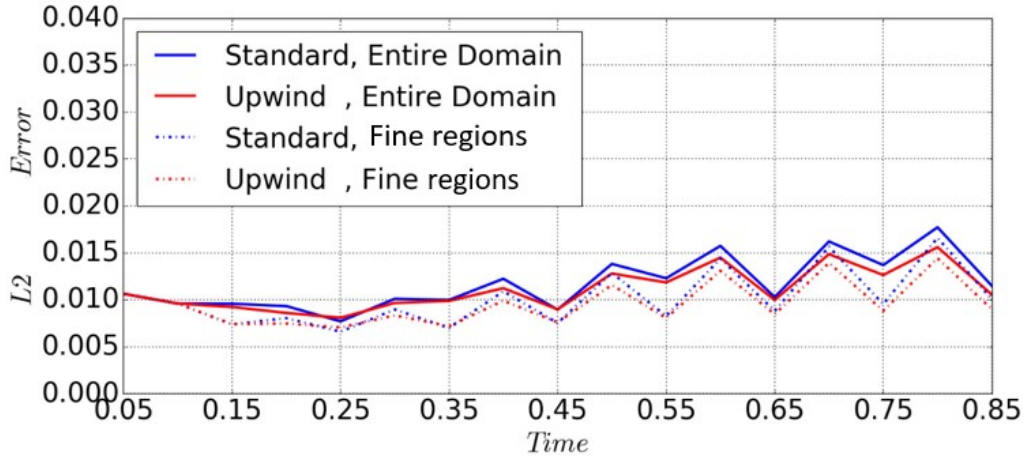


Figure 5.10: L2 error of pressure (p) in the domain at different times for Roe scheme with WENO

Table 5.3 shows the time-averaged L2 errors of pressure throughout the entire domain and in the fine regions. These values help us to understand and confirm that the errors in the entire domain and fine regions have reduced while using upwind interpolation.

	Standard Interpolation	Upwind Interpolation
Entire domain	0.011729	0.011100
Fine regions	0.010171	0.009566

Table 5.3: Time averaged L2 error of pressure (p) in the domain for Roe scheme with WENO

5.4.4 Summary

A 2-D Sod shock tube simulation is carried out using standard and upwind interpolation using Roe scheme with MUSCL and using Roe scheme with WENO. From the error in pressure plots, L2 error histories and the time-averaged L2 error tables, it is observed that upwind interpolation reduces the error in the simulation domain when compared to using standard interpolation.

5.5 Shock propagation from right side to left side of the Sod shock tube

The sod shock tube utilized in the previous simulation is considered, but with the higher initial values of the primitive variables on the right side of the tube, such that the shock wave propagates from the right side of the tube to the left side.

5.5.1 Problem setting

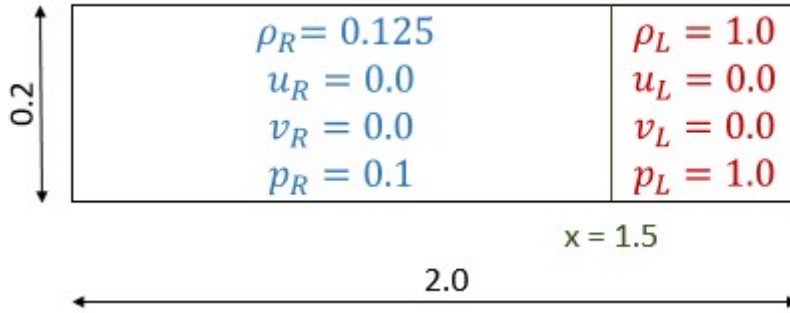


Figure 5.11: Initial states in the shock tube

A 2-D shock tube is considered with the initial states as shown in Figure 5.11. The left and right states in the shock tube are separated at $x = 1.5$. In order to replicate the BCM domain, the computational domain for shock tube is divided into 4 regions as shown in Figure 5.12. The regions 1, 2, 4 are the fine regions and region 3 is the coarse region.

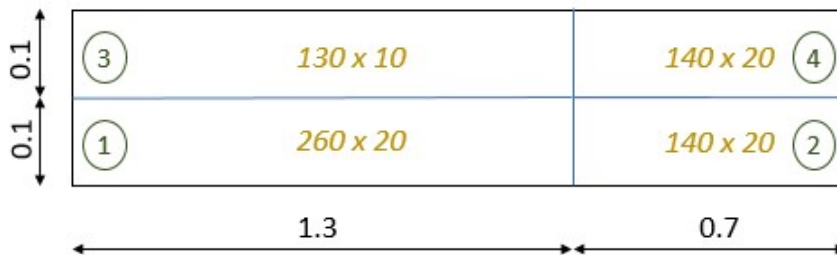


Figure 5.12: Grid distribution

Region 1	$\Delta x_1 = 0.005$ $\Delta y_1 = 0.005$
Region 2	$\Delta x_2 = 0.005$ $\Delta y_2 = 0.005$
Region 3	$\Delta x_3 = 0.01$ $\Delta y_3 = 0.01$
Region 4	$\Delta x_4 = 0.005$ $\Delta y_4 = 0.005$

Table 5.4: Grid sizes in regions 1,2,3,4

The simulations are carried out by using Roe scheme with WENO. The simulation is conducted till $t=0.85$. The shock wave reaches the interface of fine and coarse region $x = 1.3$ at approximately $t = 0.11$. Both standard and upwind interpolation simulations are conducted. For upwind interpolation, a low order interpolation is used for transferring values to ghost cells, as discussed in Section 2.3. CFL number used in fine regions is 0.1. Grid spacing in different regions are as shown in Table 5.4. Since grid size jump is present along both x and y axes, this is a 2-D hanging node interface simulation.

5.5.2 Results and discussion

In this section, the results obtained from simulations using standard and upwind interpolation for calculating ghost cell values are compared with analytical values. Here, error in pressure (p) contours in the domain at a random time after the shock wave has crossed the hanging node interface is used for comparison.

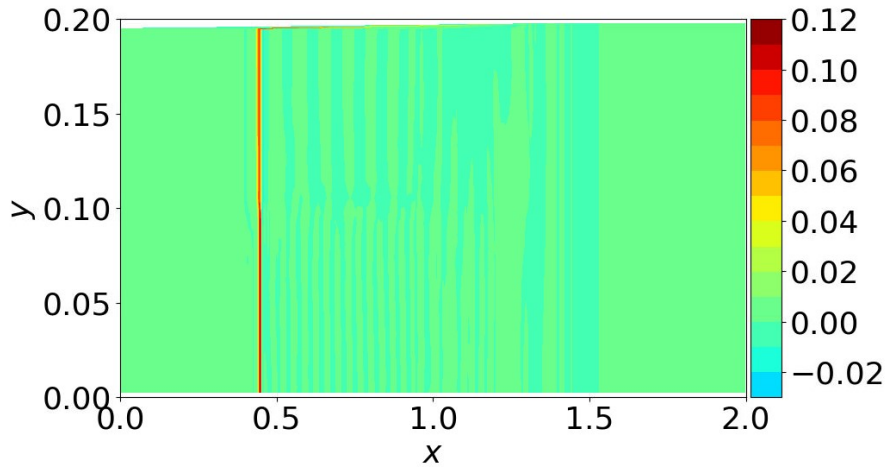


Figure 5.13: Error in pressure (p) throughout the computational domain with standard interpolation at $t = 0.60$

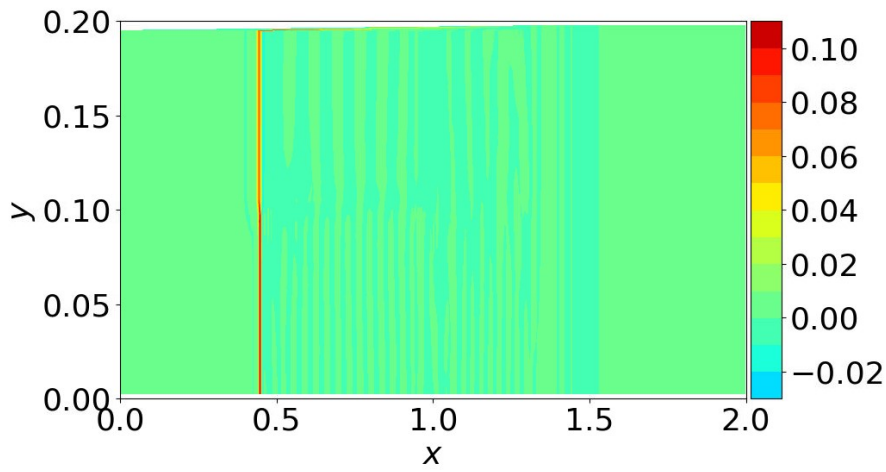


Figure 5.14: Error in pressure (p) throughout the computational domain with upwind interpolation at $t = 0.60$

Figures 5.13, 5.14, compares the error contours in pressure (p) at a time $t = 0.60$ when the shock wave reaches approximately $x = 0.44$. The difference in error contours is not clearly visible from these plots, but the maximum error observed is lower in simulation using upwind interpolation.

L2 error

L2 error in p is defined as

$$\frac{\sqrt{\sum_{i,j} \left[(p_{fine-coarse} - p_{analytic})^2 \right]_{i,j}}}{\sqrt{\sum_{i,j} \left[(p_{analytic})^2 \right]_{i,j}}} \quad (5.2)$$

Figure 5.15 shows the L2 error in pressure (p) throughout the domain from time $t = 0$ to $t = 0.85$. Here, Roe scheme with WENO implementation is used for simulations conducted separately with standard and upwind interpolation.

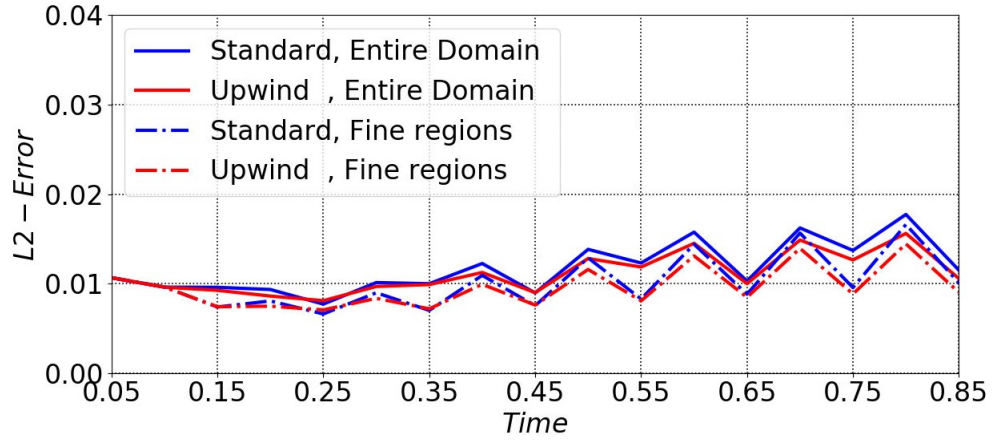


Figure 5.15: L2 error of pressure (p) in the domain at different times for Roe scheme with WENO

This figure also shows the L2 error history in the fine regions in the computational domain. It can be observed that the error values decrease while using upwind interpolation in both the cases.

Table 5.5 shows the time-averaged L2 errors of pressure throughout the entire domain and in the fine regions. These values help us to understand and confirm that the errors in the entire domain and fine regions have reduced while using upwind interpolation.

	Standard Interpolation	Upwind Interpolation
Entire domain	0.011729	0.011100
Fine regions	0.010171	0.009566

Table 5.5: Time averaged L2 error of pressure (p) in the domain for Roe scheme with WENO

5.5.3 Summary

A 2-D Sod shock tube simulation with shock wave propagating from the right side to the left side of the tube is carried out using standard and upwind interpolation using Roe scheme with WENO. From the error in pressure plots, L2 error histories and the time-averaged L2 error tables, it is observed that upwind interpolation reduces the error in the simulation domain when compared to using standard interpolation. The improvement obtained while using upwind interpolation is the same as in the previous case of shock wave propagating from the left side to the right side of the shock tube.

Chapter 6

Concluding remarks

6.1 Conclusions

In this research,

- first, a simple wave travel through a BCM kind of domain i.e. a domain having both fine and coarse grids was understood from the simple sine wave propagation simulation. This was performed to check what bad happens at the hanging node interface which reduces the accuracy of simulation and whether this solution accuracy can be improved further by a new interpolation. A reflected wave was observed from the interface while using standard interpolation for ghost cells near the hanging node interface. Therefore, an appropriate upwind S2L interpolation for obtaining ghost cell values was implemented which reduced this reflected wave and improved the accuracy of the simulation,
- second, an upwind interpolation method was proposed based on characteristic variables to get more accurate values of primitive variables in ghost cells near hanging node interfaces for better BCM compressible Euler equation simulations,
- and finally the performance of the proposed upwind interpolation was determined by comparing the errors in the simulation domain. The simulations were found to be more accurate while using upwind interpolation as compared to using standard interpolation in isentropic vortex convection, shock wave propagation simulations.

6.2 Future work

This research as a beginning has attempted to improve accuracy of Building-Cube Method simulations in the 2-D computational domain. However, this is only the stepping stone, since plethora of fluid dynamics problem simulations are done in a more realistic sense in the 3-D computational domain.

The upwind interpolation proposed in this research could be extended directly to be used in 3-D simulations to understand complex real-world problems. This is achieved by including the third dimension computations (along z -axis direction) in the simulations. The interpolations are carried out with the help of eigenvalues and characteristic variables along the z -direction in the same manner as along x, y axes discussed previously in our research.

Bibliography

- [1] K. Nakahashi, High-Density Mesh Flow Computations with Pre-/Post-Data Compressions, *AIAA 2005-4876*, 2005.
- [2] K. Nakahashi, Building Cube Method : A CFD Approach for Near-Future PetaFlops Computers, *8th World Congress on Computational Mechanics (WCCM8)*, 2008.
- [3] K. Nakahashi, L.S. Kim, Building Cube Method for Large-Scale, High Resolution Flow Computations, *AIAA 2004-434*, 2004.
- [4] T. Ishida, S. Takahashi, K. Nakahashi, Fast Cartesian Mesh Generation for Building-Cube Method using Multi-Core PC, *AIAA 2008-919*, 2008.
- [5] M. Steffen, A Simple Method for Monotonic Interpolation in One Dimension, *Astronomy and Astrophysics*, Vol.239, No.Nov(2), 1990.
- [6] R.R. Bless, D.D. Moerder, Computationally Efficient Method for Multidimensional Data Interpolation, *AIAA 95-3322-CP*, 1995.
- [7] Y. Kuya, Y. Fukushima, Y. Tamaki, S. Kawai, Kinetic Energy and entropy preserving schemes by split convective forms on hierarchical Cartesian grids with hanging nodes, *AIAA SciTech Forum 2019-0905*, 2019.
- [8] A. Jameson, Formulation of kinetic energy preserving schemes for gas dynamics and direct numerical simulation of one-dimensional viscous compressible flow in a shock tube using entropy and kinetic energy preserving schemes, *Journal of Scientific Computing*, Vol.34, No.2, 2008.
- [9] T. Ishida, Study of High-Order/High-Resolution Method for Flow Simulations with Cartesian Grid Method, *Doctoral Thesis*, Tohoku University, 2011.

- [10] A. Figueroa, R. Lohner, J. Sitaraman, On Interpolation schemes for Nested Cartesian Finite Difference Grids of Different Size, *AIAA SciTech Forum 2018-1561*, 2018.
- [11] J.D. Anderson, JR., *Computational Fluid Dynamics, The Basics with Applications*, McGraw Hill Publications, 1995.
- [12] C. Hirsch, *Numerical Computation of Internal and External Flows*, Wiley Publications, 1991.
- [13] A. Rohde, Eigenvalues and Eigenvectors of the Euler Equations in General Geometries, *AIAA Computational Fluid Dynamics Conference 2001-2609*, 2001.
- [14] V.A. Titarev, E.F. Toro, Finite-volume WENO schemes for three-dimensional conservation laws, *Journal of Computational Physics* 201, 2004.
- [15] I. Danaila, P. Joly, S.M. Kaber, M. Postel, *An Introduction to Scientific Computing*, Springer New York, 2007.
- [16] J.D. Anderson Jr., *Modern Compressible Flow with Historical Perspective*, McGraw Hill Publications, 1989.
- [17] E.F. Toro, *Riemann Solvers and Numerical methods for Fluid Dynamics* Springer, 2009.
- [18] K. Murawski, Jr.K. Murawski, P. Stpiczynski, Implementation of MUSCL-Hancock method into the C++ code for the Euler equations, *Bulletin of Polish Academy of Sciences*, Vol.60, No.1, 2012.
- [19] C.A.J. Fletcher, *Computational Techniques for Fluid Dynamics*, Springer-Verlag, 1991.
- [20] W. Xu, X. Kong, C. Zheng, W. Wu, Numerical Method for Predicting the Blast Wave in Partially Confined Chamber, *Hindawi Research Article*, 2018.
- [21] K. Onishi, S. Obayashi, Use of the Immersed Boundary Method within the Building Cube Method and its application to Real Vehicle CAD Data, *AIAA Computational Fluid Dynamics Conference 2013-2713*, 2013.

Appendix A

Governing equations and schemes

A.1 Wave equation

A sine wave is simulated using the wave equation introduced at the left boundary of the domain according to the value $\sin(\omega t)$ from time $t = 0$ for one wavelength of the wave, where angular frequency ω is defined as,

$$\omega = \frac{2\pi c}{\Lambda} \quad (\text{A.1})$$

Wavelength $\Lambda = 40$ units and wave speed $c = 1$ units.

The wave equation is a second-order linear partial differential equation. A classical 2-D wave equation is as follows

$$\frac{\partial^2 U}{\partial t^2} = c^2 \left(\frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} \right) \quad (\text{A.2})$$

Integrating equation A.2

$$\int_x \int_y \int_t \frac{\partial^2 U}{\partial t^2} dt dy dx = c^2 \int_t \int_y \int_x \frac{\partial^2 U}{\partial x^2} dx dy dt + c^2 \int_t \int_x \int_y \frac{\partial^2 U}{\partial y^2} dy dx dt \quad (\text{A.3})$$

Applying limits to equation A.3

$$\int_x \int_y \frac{\partial U}{\partial t} \Big|_{t_n}^{t_{n+1}} dy dx = c^2 \int_t \int_y \frac{\partial U}{\partial x} \Big|_{x_i}^{x_{i+1}} dy dt + c^2 \int_t \int_x \frac{\partial U}{\partial y} \Big|_{y_j}^{y_{j+1}} dx dt \quad (\text{A.4})$$

Upon discretising in the $x - y$ domain at time t , we obtain

$$\begin{aligned} \left(\frac{U_{i,j}^{n+1} - U_{i,j}^n}{\Delta t} - \frac{U_{i,j}^n - U_{i,j}^{n-1}}{\Delta t} \right) \Delta x \Delta y = c^2 \left(\frac{U_{i+1,j}^n - U_{i,j}^n}{\Delta x} - \frac{U_{i,j}^n - U_{i-1,j}^n}{\Delta x} \right) \Delta y \Delta t \\ + c^2 \left(\frac{U_{i,j+1}^n - U_{i,j}^n}{\Delta y} - \frac{U_{i,j}^n - U_{i,j-1}^n}{\Delta y} \right) \Delta x \Delta t \end{aligned} \quad (\text{A.5})$$

subscripts i, j, n represents the indexing for x, y, t respectively.

$$\begin{aligned} (U_{i,j}^{n+1} - 2U_{i,j}^n + U_{i,j}^{n-1}) \frac{\Delta x \Delta y}{\Delta t} = c^2 \frac{\Delta y \Delta t}{\Delta x} (U_{i+1,j}^n - 2U_{i,j}^n - U_{i-1,j}^n) \\ + c^2 \frac{\Delta x \Delta t}{\Delta y} (U_{i,j+1}^n - 2U_{i,j}^n - U_{i,j-1}^n) \end{aligned} \quad (\text{A.6})$$

Rearranging equation A.6

$$\begin{aligned} U_{i,j}^{n+1} = 2U_{i,j}^n - U_{i,j}^{n-1} + \frac{c^2 \Delta t^2}{\Delta x^2} (U_{i+1,j}^n - 2U_{i,j}^n - U_{i-1,j}^n) \\ + \frac{c^2 \Delta t^2}{\Delta y^2} (U_{i,j+1}^n - 2U_{i,j}^n - U_{i,j-1}^n) \end{aligned} \quad (\text{A.7})$$

In equation A.7

$$cfl_x = \frac{c\Delta t}{\Delta x} \quad cfl_y = \frac{c\Delta t}{\Delta y} \quad (\text{A.8})$$

At initial time $t = 0$, equation A.7 becomes

$$\begin{aligned} U_{i,j}^{n+1} = U_{i,j}^n + \frac{c^2 \Delta t^2}{2\Delta x^2} (U_{i+1,j}^n - 2U_{i,j}^n - U_{i-1,j}^n) \\ + \frac{c^2 \Delta t^2}{2\Delta y^2} (U_{i,j+1}^n - 2U_{i,j}^n - U_{i,j-1}^n) \end{aligned} \quad (\text{A.9})$$

A.2 Euler equation

In an unsteady 2-D compressible inviscid flow problem [15][16], the entire system of governing equations comprising the continuity, x -momentum, y -momentum and energy equations in the conservative form is given by

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} = 0 \quad (\text{A.10})$$

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho e \end{bmatrix} \quad \mathbf{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ (\rho e + p)u \end{bmatrix} \quad \mathbf{G} = \begin{bmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ (\rho e + p)v \end{bmatrix}$$

where \mathbf{U} is the solution vector and \mathbf{F} and \mathbf{G} are the flux vectors in the x and y directions, respectively. Here ρ , p , u , v , e are density, pressure, x -component of velocity, y -component of velocity and specific total energy respectively. These equations are also known as Euler's equations.

They are a coupled system of non-linear partial differential equations, and hence are very difficult to solve analytically[11]. From the above equations, we have 5 unknowns and 4 equations. To date, there are no general closed-form solutions to these equations. In aerodynamics, it is generally reasonable to assume the gas is a perfect gas. For a perfect gas, the equation of state is

$$p = \rho RT \quad (\text{A.11})$$

where, R is the specific gas constant. This equation provides a fifth equation but introduces a sixth unknown, namely, temperature T . A sixth equation to close the entire system is a thermodynamic relation between state variables. For a calorically perfect gas

$$\text{internal energy} = C_v T \quad (\text{A.12})$$

where C_v is the specific heat of gas at constant volume

A.3 Roe with MUSCL scheme

Euler equations in 2-D can be written in the form [17][18],

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} = 0 \quad (\text{A.13})$$

Upon discretizing,

$$\begin{aligned} \mathbf{U}_{i,j}^{n+1} = \mathbf{U}_{i,j}^n &+ \frac{\Delta t}{\Delta x} \left(\mathbf{F}_{i-\frac{1}{2},j}^n - \mathbf{F}_{i+\frac{1}{2},j}^n \right) \\ &+ \frac{\Delta t}{\Delta y} \left(\mathbf{G}_{i,j-\frac{1}{2}}^n - \mathbf{G}_{i,j+\frac{1}{2}}^n \right) \end{aligned} \quad (\text{A.14})$$

\mathbf{F} and \mathbf{G} are the fluxes in the x and y directions, respectively. The procedure for finding all required quantities along x -direction for calculating Equation A.14 as per Roe scheme is described below [19].

$$\mathbf{F}_{i+\frac{1}{2},j} = \frac{1}{2} (\mathbf{F}_{i,j}^L + \mathbf{F}_{i,j}^R) - \frac{1}{2} \sum_{i=1}^m \tilde{\alpha}_i \left| \tilde{\lambda}_i \right| \tilde{\mathbf{K}}^{(i)} \quad (\text{A.15})$$

where,

$$\begin{aligned} \mathbf{F}_{i,j}^L &= \mathbf{F}(\mathbf{U}_{i,j}^L) \\ \mathbf{F}_{i,j}^R &= \mathbf{F}(\mathbf{U}_{i,j}^R) \end{aligned} \quad (\text{A.16})$$

MUSCL extrapolations,

$$\begin{aligned} \mathbf{U}_{i,j}^L(l) &= \mathbf{U}_{i,j}(l) + \frac{1}{2} \Phi_{i,j}(l) * \Delta_{i,j}(l) \\ \mathbf{U}_{i,j}^R(l) &= \mathbf{U}_{i+1,j}(l) - \frac{1}{2} \Phi_{i+1,j}(l) * \Delta_{i+1,j}(l) \end{aligned} \quad (\text{A.17})$$

where Φ is the slope limiter,

$$\Phi_{i,j}(l) = \begin{cases} 0, & \mathbf{r}_{i,j}(l) \leq 0 \\ \mathbf{r}_{i,j}, & 0 \leq \mathbf{r}_{i,j}(l) \leq 1 \\ \min(1, \Phi_R(\mathbf{r}_{i,j}(l))), & \mathbf{r}_{i,j}(l) \geq 1 \end{cases} \quad (\text{A.18})$$

$$\Phi_R(\mathbf{r}_{i,j}(l)) = \frac{2}{1 - \tau + (1 + \tau)\mathbf{r}_{i,j}(l)} \quad (\text{A.19})$$

$$\mathbf{r}_{i,j}(l) = \frac{\Delta_{i-\frac{1}{2},j}(l)}{\Delta_{i+\frac{1}{2},j}(l)} \quad (\text{A.20})$$

$$\Delta_{i,j}(l) = \frac{1}{2} (1 + \tau) \Delta_{i-\frac{1}{2},j}(l) + \frac{1}{2} (1 - \tau) \Delta_{i+\frac{1}{2},j}(l) \quad (\text{A.21})$$

$$\Delta_{i-\frac{1}{2},j}(l) = \mathbf{U}_{i,j}(l) - \mathbf{U}_{i-1,j}(l) \quad (\text{A.22})$$

$$\Delta_{i+\frac{1}{2},j}(l) = \mathbf{U}_{i+1,j}(l) - \mathbf{U}_{i,j}(l) \quad (\text{A.23})$$

$$\tau \in [-1, 1]$$

$$l = 1 \text{ to } 4$$

To calculate Eigenvalues $\mathbf{K}^{(i)}$

$$\mathbf{K}^{(1)} = \begin{bmatrix} 1 \\ \tilde{u} - \tilde{a} \\ \tilde{v} \\ \tilde{H} - \tilde{u}\tilde{a} \end{bmatrix} \quad \mathbf{K}^{(2)} = \begin{bmatrix} 1 \\ \tilde{u} \\ \tilde{v} \\ \frac{1}{2}\tilde{V}^2 \end{bmatrix} \quad \mathbf{K}^{(3)} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ \tilde{v} \end{bmatrix} \quad \mathbf{K}^{(4)} = \begin{bmatrix} 1 \\ \tilde{u} + \tilde{a} \\ \tilde{v} \\ \tilde{H} + \tilde{u}\tilde{a} \end{bmatrix}$$

$$\tilde{u} = \frac{\sqrt{\rho_L}u_L + \sqrt{\rho_R}u_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \quad (\text{A.24})$$

$$\tilde{v} = \frac{\sqrt{\rho_L}v_L + \sqrt{\rho_R}v_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \quad (\text{A.25})$$

$$\tilde{H} = \frac{\sqrt{\rho_L}H_L + \sqrt{\rho_R}H_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \quad (\text{A.26})$$

$$\tilde{a} = \left((\gamma - 1) \left(\tilde{H} - \frac{1}{2} \tilde{V}^2 \right) \right)^{\frac{1}{2}} \quad (\text{A.27})$$

$$\tilde{V}^2 = \tilde{u}^2 + \tilde{v}^2 \quad (\text{A.28})$$

To calculate α_i solve Equation A.30

$$\Delta U = \sum_{i=1}^5 \tilde{\alpha}_i \tilde{\mathbf{K}}^{(i)} \quad (\text{A.29})$$

$$\Delta u_i = (u_i)_R - (u_i)_L \quad (\text{A.30})$$

Similar procedure can be carried out for finding all quantities along y -direction to complete calculating Equation A.14.

A.4 Roe with WENO scheme

The formulations explained in the above section for Roe with MUSCL scheme holds true here too, except that the extrapolations used here are WENO based [14][20] and not MUSCL based.

Explaining the WENO extrapolation in one dimension, i.e. x direction,

$$U_{i,j}^L(l) = \sum_{z=1}^3 U_{weno(z)}^L(l) \quad (A.31)$$

$$U_{i,j}^R(l) = \sum_{z=1}^3 U_{weno(z)}^R(l) \quad (A.32)$$

$$U_{weno(z)}^L(l) = \omega_z^L(l) * S_z^L(l) \quad (A.33)$$

$$U_{weno(z)}^R(l) = \omega_z^R(l) * S_z^R(l) \quad (A.34)$$

$$\begin{aligned} S_1^L(l) &= \frac{2}{6}U_{i-2,j}(l) - \frac{7}{6}U_{i-1,j}(l) + \frac{11}{6}U_{i,j}(l) \\ S_2^L(l) &= -\frac{1}{6}U_{i-1,j}(l) + \frac{5}{6}U_{i,j}(l) + \frac{2}{6}U_{i+1,j}(l) \\ S_3^L(l) &= \frac{2}{6}U_{i,j}(l) + \frac{5}{6}U_{i+1,j}(l) - \frac{1}{6}U_{i+2,j}(l) \end{aligned} \quad (A.35)$$

$$\begin{aligned} S_1^R(l) &= -\frac{1}{6}U_{i-2,j}(l) + \frac{5}{6}U_{i-1,j}(l) + \frac{2}{6}U_{i,j}(l) \\ S_2^R(l) &= \frac{2}{6}U_{i-1,j}(l) + \frac{5}{6}U_{i,j}(l) - \frac{1}{6}U_{i+1,j}(l) \\ S_3^R(l) &= \frac{11}{6}U_{i,j}(l) - \frac{7}{6}U_{i+1,j}(l) + \frac{2}{6}U_{i+2,j}(l) \end{aligned} \quad (A.36)$$

$$\omega_z^L(l) = \frac{\alpha_z^L(l)}{\sum_{z=1}^3 \alpha_z^L(l)} \quad (A.37)$$

$$\omega_z^R(l) = \frac{\alpha_z^R(l)}{\sum_{z=1}^3 \alpha_z^R(l)} \quad (A.38)$$

$$\boldsymbol{\alpha}_z^L(l) = \frac{d_z^L}{(\epsilon + \boldsymbol{\beta}_z(l))^2} \quad (\text{A.39})$$

$$\boldsymbol{\alpha}_z^R(l) = \frac{d_z^R}{(\epsilon + \boldsymbol{\beta}_z(l))^2} \quad (\text{A.40})$$

$$d_1^L = \frac{1}{10}, \quad d_2^L = \frac{6}{10}, \quad d_3^L = \frac{3}{10} \quad (\text{A.41})$$

$$d_1^R = \frac{3}{10}, \quad d_2^R = \frac{6}{10}, \quad d_3^R = \frac{1}{10} \quad (\text{A.42})$$

$$\beta_1(l) = \frac{13}{12} (\mathbf{U}_{i-2,j}(l) - 2\mathbf{U}_{i-1,j}(l) + \mathbf{U}_{i,j}(l))^2 + \frac{1}{4} (\mathbf{U}_{i-2,j}(l) - 4\mathbf{U}_{i-1,j}(l) + 3\mathbf{U}_{i,j}(l))^2 \quad (\text{A.43})$$

$$\beta_2(l) = \frac{13}{12} (\mathbf{U}_{i-1,j}(l) - 2\mathbf{U}_{i,j}(l) + \mathbf{U}_{i+1,j}(l))^2 + \frac{1}{4} (\mathbf{U}_{i-1,j}(l) - \mathbf{U}_{i+1,j}(l))^2 \quad (\text{A.44})$$

$$\beta_3(l) = \frac{13}{12} (\mathbf{U}_{i,j}(l) - 2\mathbf{U}_{i+1,j}(l) + \mathbf{U}_{i+2,j}(l))^2 + \frac{1}{4} (3\mathbf{U}_{i,j}(l) - 4\mathbf{U}_{i+1,j}(l) + \mathbf{U}_{i+2,j}(l))^2 \quad (\text{A.45})$$

A.5 Code validations

A.5.1 Sine wave propagation

In order to check the sine wave simulations, increasing grid numbers (i.e. different grid levels) are taken on a uniform computational domain and simulation is repeated with each grid number. L2 error is calculated by comparing this simulation result with the exact solution obtained on the same grid level. Figure A.1 shows the domain and the different grid numbers used. Table A.1 shows the L2 errors at two time levels $t = 80, 160$. It is observed that with increasing grid numbers (i.e. finer resolution) the error decreases, which is the expected outcome of this validation work.

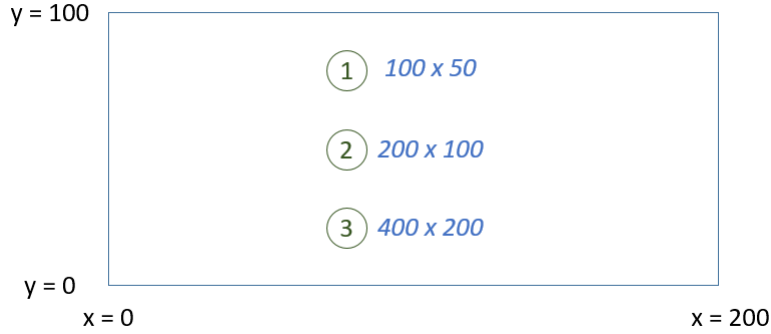


Figure A.1: Grid specifications in the domain

Time	$t = 80$	$t = 160$
Grid 1	0.125256	0.123001
Grid 2	0.078520	0.078520
Grid 3	0.039267	0.039267

Table A.1: L2 error for different grid levels in uniform domain sine wave propagation simulation

A.5.2 Vortex convection

In order to check the 2-D vortex convection simulations, increasing grid numbers (i.e. different grid levels) are taken on a uniform computational domain and simulation is repeated with each grid number. L2 error is calculated by comparing this simulation result with the exact solution obtained on the same grid level.

Figure A.2 shows the domain and the different grid numbers used. Table A.2 shows the L2 errors at two time levels $t = 6, 16$. We observe that with increasing grid numbers (i.e. finer resolution) the error decreases, which is the expected outcome of this validation work.



Figure A.2: Grid specifications in all the regions of the entire domain

Time	$t = 6$	$t = 16$
Grid 1	3.840022 e-04	9.830458 e-04
Grid 2	1.906245 e-04	4.997540 e-04
Grid 3	9.958738 e-05	4.238349 e-04

Table A.2: L2 error for different grid levels in uniform domain vortex convection simulation

A.5.3 Shock wave propagation

In order to check the shock wave simulations, increasing grid numbers (i.e. different grid levels) are taken on a uniform computational domain and simulation is repeated with each grid number. L2 error is calculated by comparing this simulation result with the exact solution obtained on the same grid level. Figure A.3 shows the domain and the different grid numbers used. Table A.3 shows the L2 errors at two time levels $t = 0.60, 0.85$. We observe that with increasing grid numbers (i.e. finer resolution) the error decreases, which is the expected outcome of this validation work.

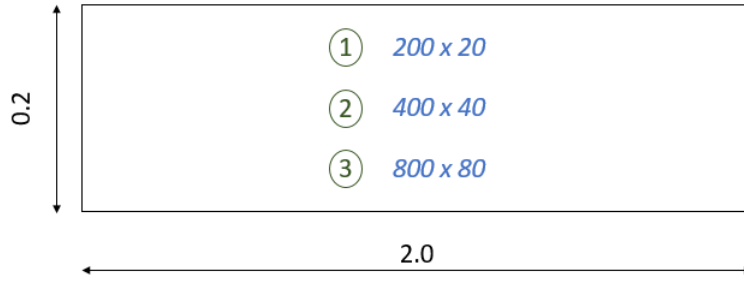


Figure A.3: Grid specifications in all the regions of the entire domain

Time	$t = 0.60$	$t = 0.85$
Grid 1	0.025538	0.019505
Grid 2	0.017985	0.014836
Grid 3	0.009200	0.009499

Table A.3: L2 error for different grid levels in uniform domain shock wave propagation simulation

Appendix B

More simulations

B.1 Shock wave propagation with 1-D hanging node interface

As shown in Figure B.1, a 2-D shock tube with the mentioned dimensions are used in simulation. The initial states in the shock-tube are also mentioned. The left and right states in the shock tube are separated at $x = 0.5$.

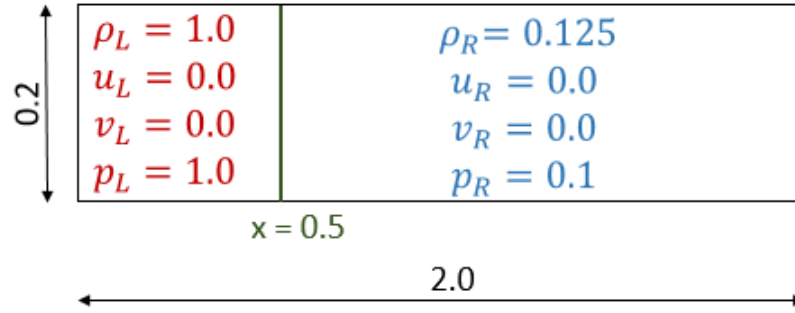


Figure B.1: Initial States in the shock tube

Here, in order to replicate the BCM domain with jump only along one axis i.e. x -direction, the computational domain for shock tube is divided into 2 regions as shown in Figure B.2. The region 1 is the fine one which means it has smaller cells (smaller cell spacing) while region 2 has larger cells (with double cell spacing as smaller cells).

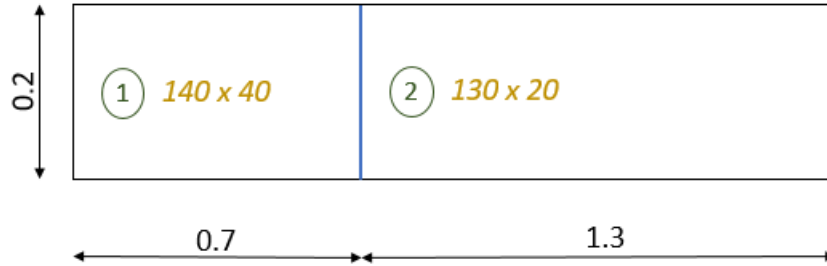


Figure B.2: Grid distribution

The simulation is conducted till time $t = 0.85$ when the shock wave reaches approximately $x = 1.9925$. The shock wave reaches the interface of fine and coarse grids at $t = 0.11$. The simulations are carried out by using Roe scheme with MUSCL.

B.1.1 Results and discussion

Comparing standard and upwind interpolation for Roe with MUSCL

In this section, the results of simulation using standard and upwind interpolation for calculating ghost cell values are compared. Roe scheme with MUSCL is used for simulations. Pressure (p) contour in the domain at a random time $t = 0.75$ when the shock wave has crossed the hanging node interface (in y direction) is used for comparison.

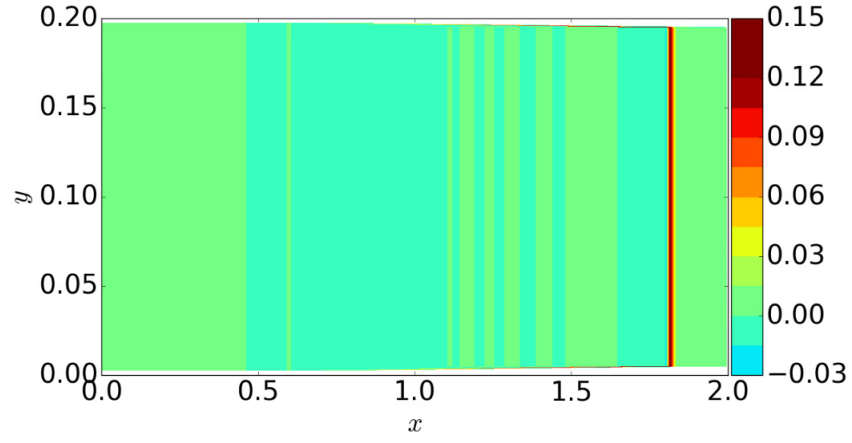


Figure B.3: Error in pressure throughout the computational domain with standard interpolation at $t = 0.75$

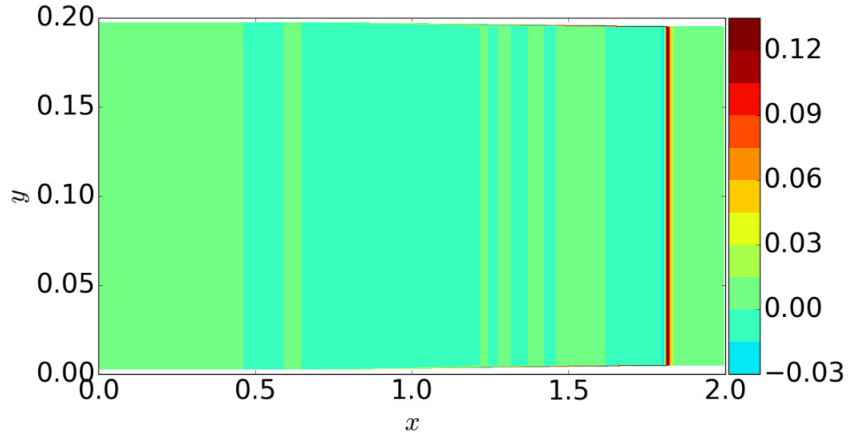


Figure B.4: Error in pressure throughout the computational domain with upwind interpolation at $t = 0.75$

Figures B.3 and B.4 compares the error contours in Pressure at time $t = 0.75$ for Roe scheme with MUSCL. Difference in pressure contours is not clearly visible from these plots, but the higher limit of error values are observed while using standard interpolation.

L2 error

L2 error in p is defined as

$$\frac{\sqrt{\sum_{i,j} \left[(p_{fine-coarse} - p_{analytic})^2 \right]_{i,j}}}{\sqrt{\sum_{i,j} \left[(p_{analytic})^2 \right]_{i,j}}} \quad (B.1)$$

L2 error for simulations using Roe scheme with MUSCL

Figure B.5 shows the L2 error in pressure throughout the domain for Roe scheme with MUSCL from time $t = 0$ to $t = 0.85$ when the shock wave has almost reached the end of the shock tube while using standard and upwind interpolation. It can be observed that the error values decrease while using upwind interpolation.

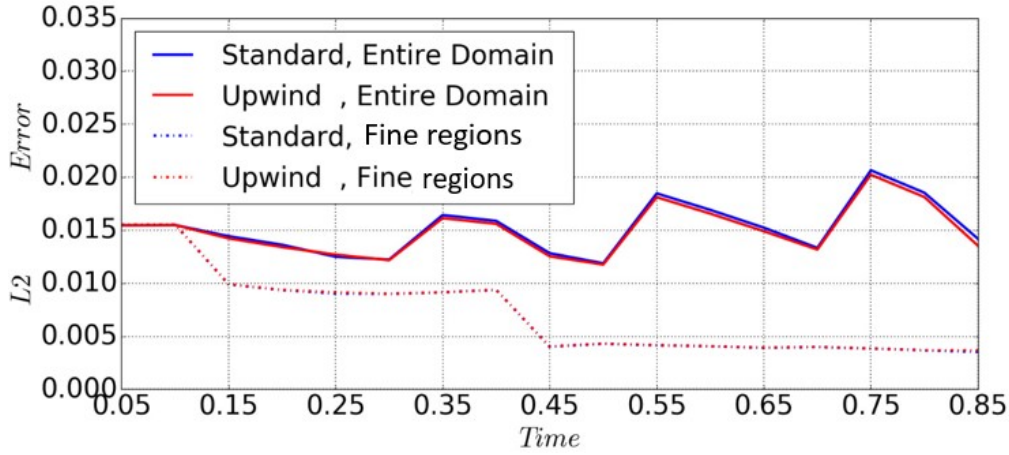


Figure B.5: L2 error of pressure in the domain at different times for Roe scheme with MUSCL

Table B.1, shows the time-averaged L2 errors of Pressure throughout the entire domain, which helps us to understand that the errors have reduced while using the new upwind interpolation.

	Standard Interpolation	Upwind Interpolation
Entire domain	0.015186	0.014951
Fine region	0.007191	0.007218

Table B.1: Time averaged L2 error of pressure in the domain for Roe scheme with MUSCL

B.1.2 Summary

A 2-D shock tube simulation is carried out using standard and upwind interpolation using Roe scheme with MUSCL for grid size jump in only one direction, i.e. for 1-D hanging node interface in the simulation domain. From the error in pressure plots, L2 error histories and the time-averaged L2 error tables, it is observed that upwind interpolation reduces the error in the entire domain when compared to using standard interpolation.

B.2 1-D sine wave propagation

A sine wave is propagated through a 1-D domain which has both fine and coarse grids to study how the wave propagates across the interface between these two regions in the domain.

A single sine wave is propagated with wave speed $c = 1$ and wavelength $\Lambda = 40$ units. To mimic the BCM kind of simulation wherein the hanging node interface occurs, the entire domain is divided into two halves at $x = 100$ by allocating grid resolution as shown in Figure B.6. Simulation conditions are shown in Table B.2. The wave propagates from fine to coarse domain, since the wave speed is a positive quantity.

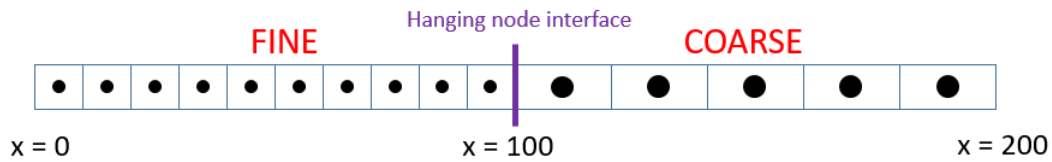


Figure B.6: Simulation domain for sine wave propagation

Domain	Length - 200
Mesh	Fine grid - 100, $x=0$ to 100 Coarse grid - 50, $x=100$ to 200
CFL	Fine grid - 1.0 Coarse grid - 0.5
Simulation end time	180
Governing equation	1D Wave equation
Scheme	2^{nd} order Central differencing scheme

Table B.2: Simulation conditions for sine wave propagation

B.2.1 Results and discussion

Analytic solution

To compare the simulation results while using standard and upwind S2L interpolation, the analytic solution is used and is calculated from sine wave equation 3.4 for all y values on the fine-coarse domain.

$$U(x, t) = U(x - ct) = \sin(\omega(x - ct)) \quad (\text{B.2})$$

The wave at times $t = 80$ (a randomly chosen time when the wave still has not reached the hanging node interface and is completely in fine region) and $t = 160$ (a randomly chosen time when the wave has completely crossed the hanging node interface and is completely in coarse region) are plotted.

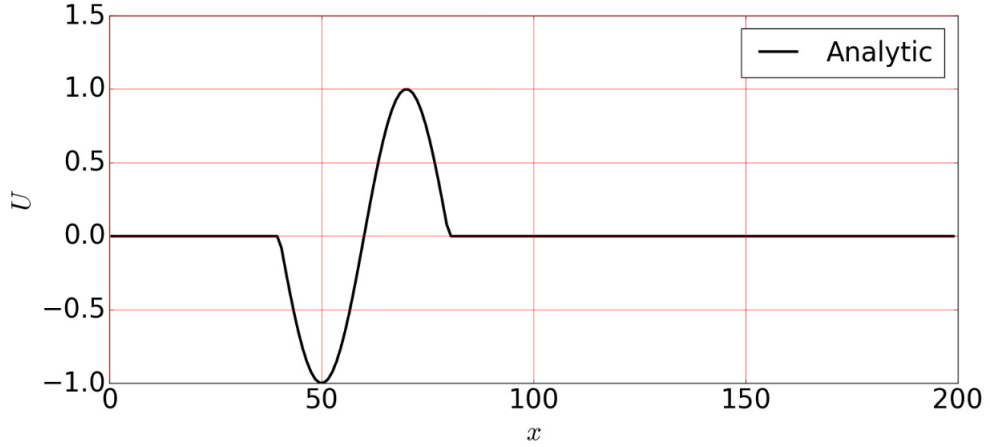
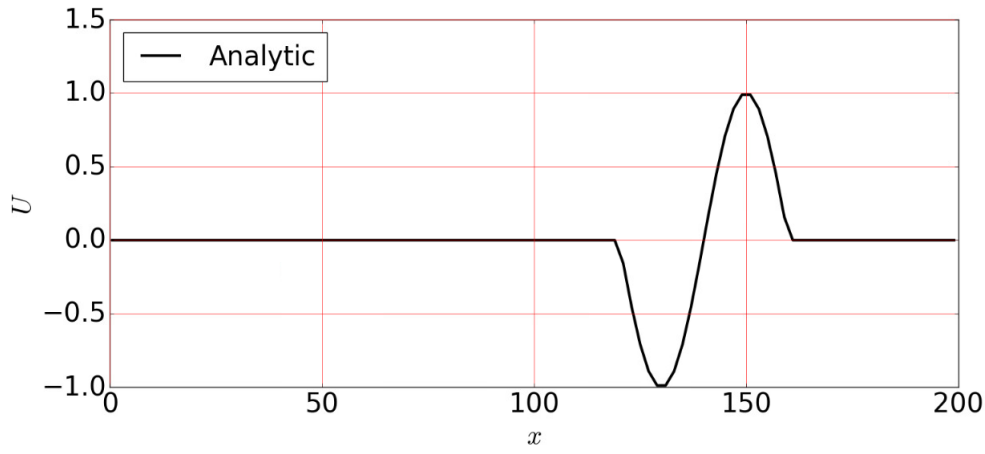


Figure B.7: Analytic solution at $t = 80$

Figure B.8: Analytic solution at $t = 160$

Standard interpolation

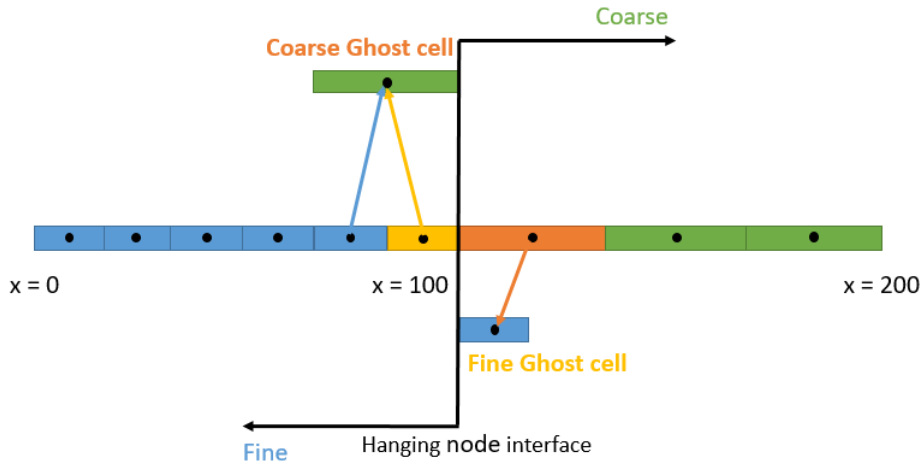
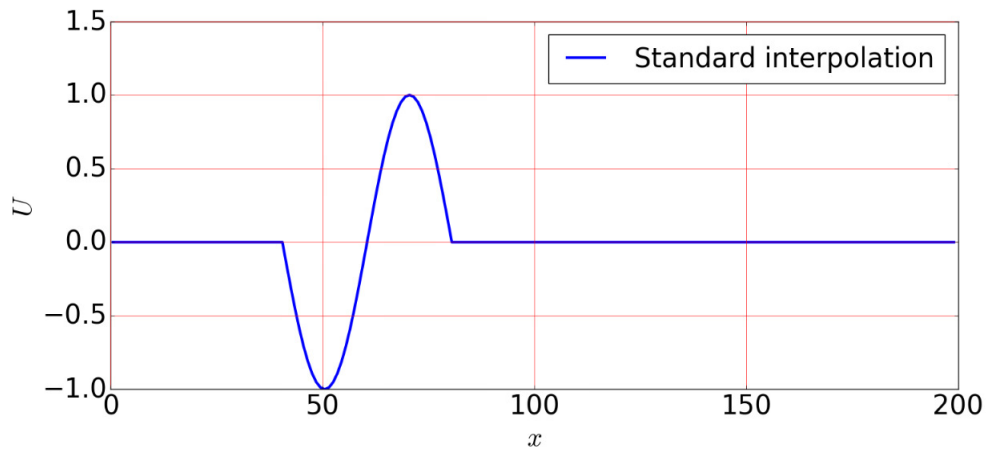
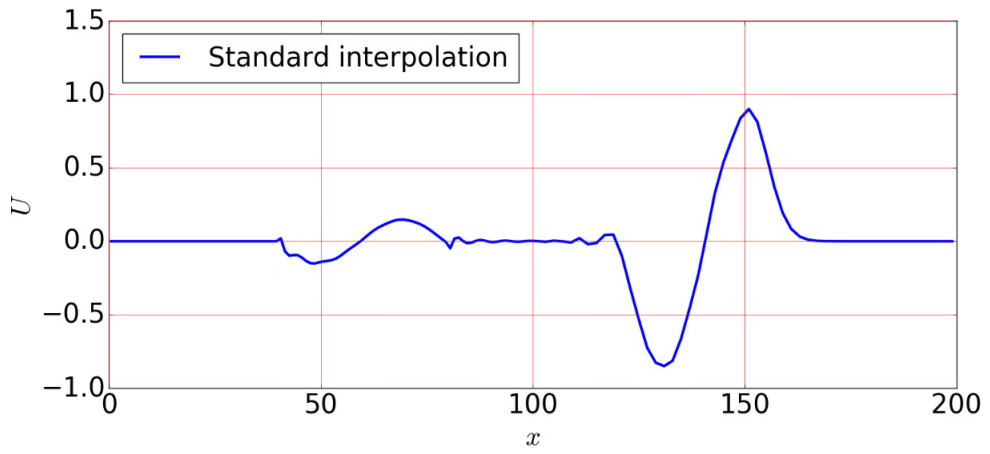


Figure B.9: Standard interpolation

While using standard interpolation as in Figure B.9 at the hanging node interface located at the boundary between fine and coarse region, the wave at times $t = 80$ and $t = 160$ are plotted in Figure B.10 and in Figure B.11. It is observed that standard interpolation produces a reflected wave from the hanging node interface. This interpolation also reduces the amplitude of the sine wave after the interface.

Figure B.10: Sine wave at time $t = 80$ using standard interpolationFigure B.11: Sine wave at time $t = 160$ using standard interpolation

Upwind interpolation

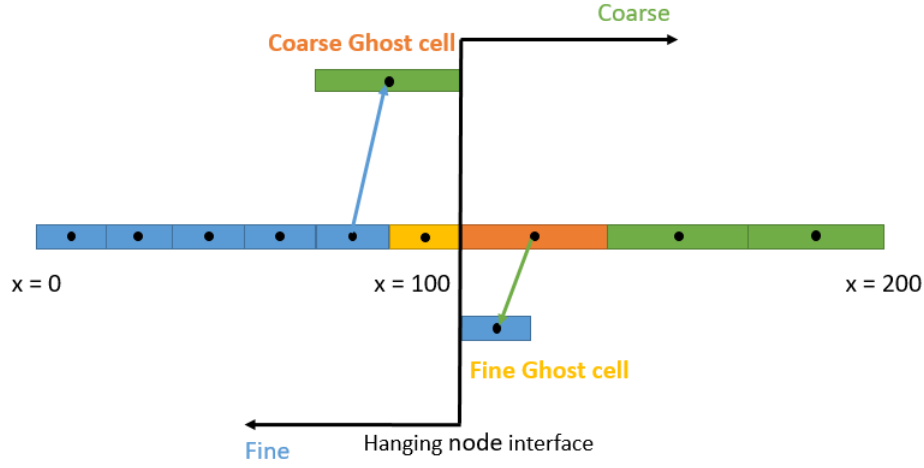


Figure B.12: Upwind S2L interpolation

While using upwind S2L interpolation shown in Figure B.12 at the hanging node interface located at the boundary between fine and coarse region, the wave at times $t = 80$ and $t = 160$ are plotted in Figure B.13 and in Figure B.14.

It is observed that upwind S2L interpolation reduces the reflected wave from the hanging node interface. This interpolation also maintains the amplitude of the sine wave after the hanging node interface.

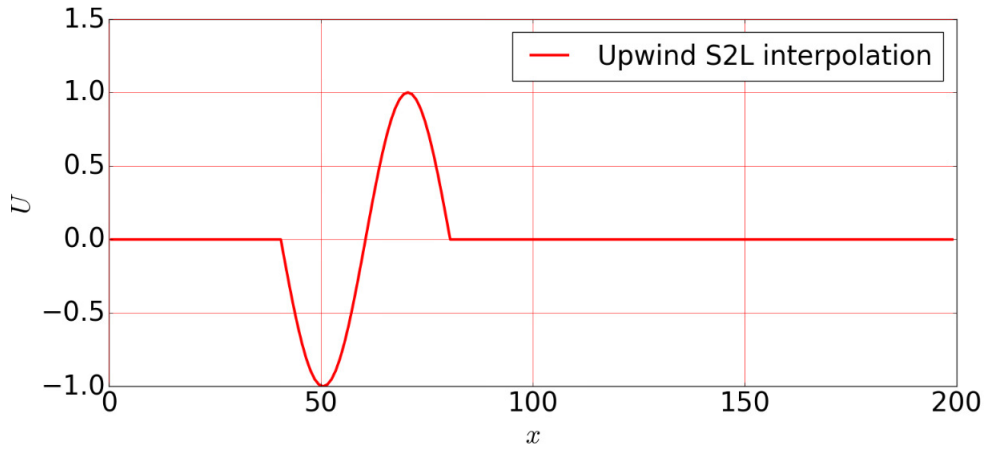


Figure B.13: Sine wave at time $t = 80$ using upwind S2L interpolation

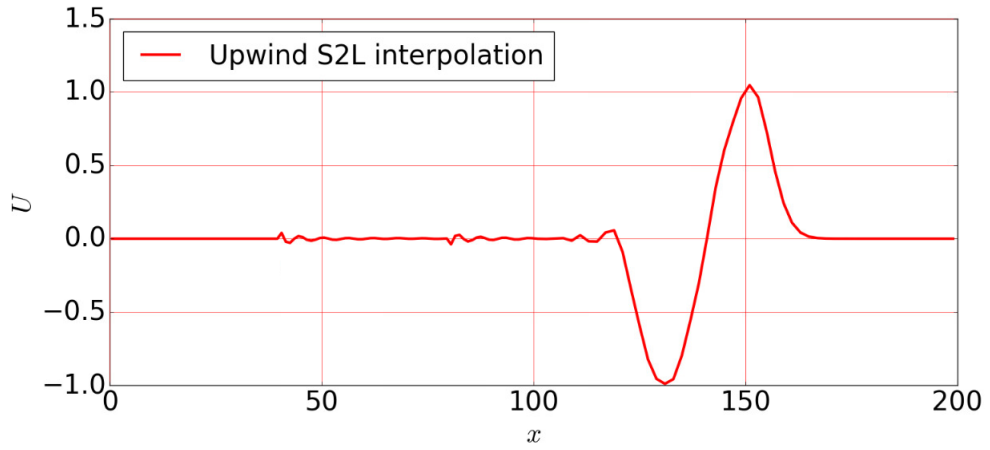


Figure B.14: Sine wave at time $t = 160$ using upwind S2L interpolation

Comparison of standard and upwind interpolation

In this section, the wave simulation results from using standard interpolation, upwind S2L interpolation, and the analytical solution are compared.

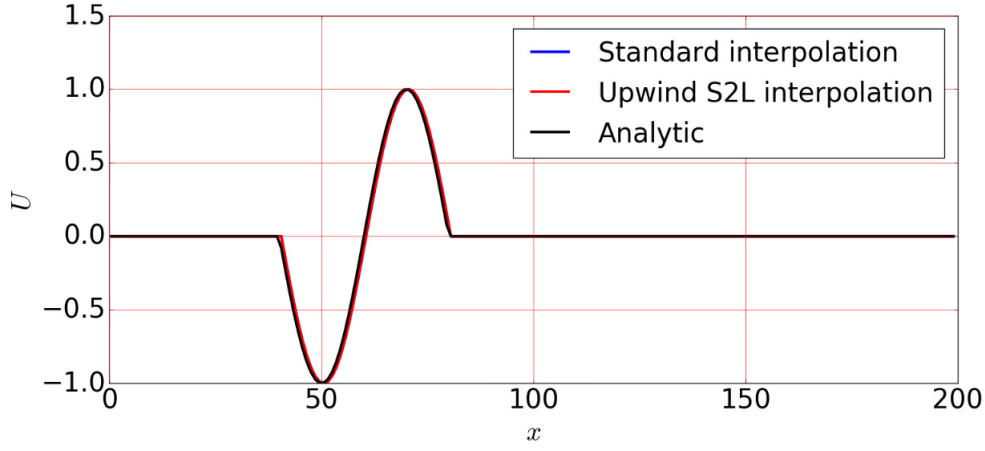
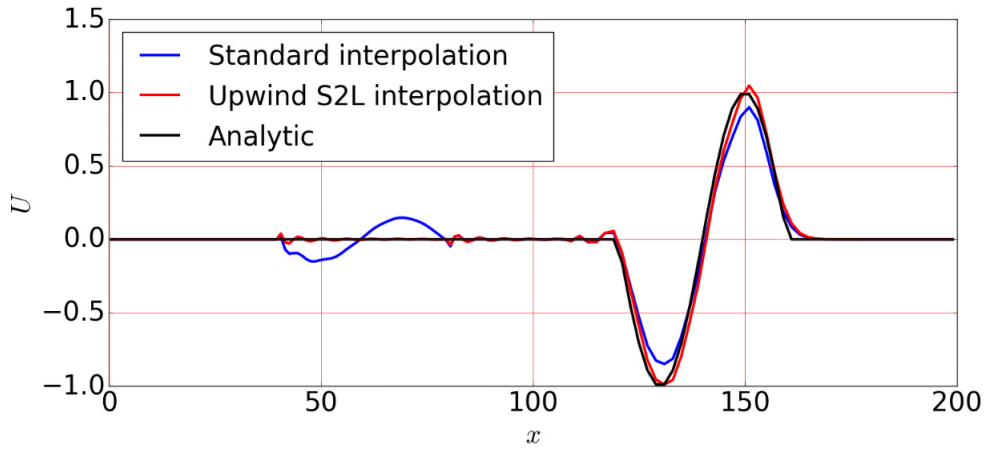
Figure B.15: Comparing sine waves at time $t = 80$ Figure B.16: Comparing sine waves at time $t = 160$

Figure B.15 shows the wave simulations using standard, upwind S2L interpolations and analytic wave simulation at time $t = 80$ before reaching hanging node interface. Here, all the waves are travelling in the same manner.

Figure B.16 shows waves at time $t = 160$ after the waves have moved past the hanging node interface at $x = 100$ and is in the coarse region. Here, while using standard interpolation, a reflected wave can be observed to the left of the hanging node. Also, the amplitude of the wave moving to the right reduces. This problem is solved by using upwind S2L interpolation, where the reflected wave is minimized and the wave moving to the right almost maintains its amplitude.

L2 error

L2 error in U is defined as

$$\frac{\sqrt{\sum_{i,j} \left[(U_{fine-coarse} - U_{analytic})^2 \right]_{i,j}}}{\sqrt{\sum_{i,j} \left[(U_{analytic})^2 \right]_{i,j}}} \quad (\text{B.3})$$

L2 error in U is calculated for the entire domain while employing both standard and upwind S2L interpolations according to the Equation B.3.

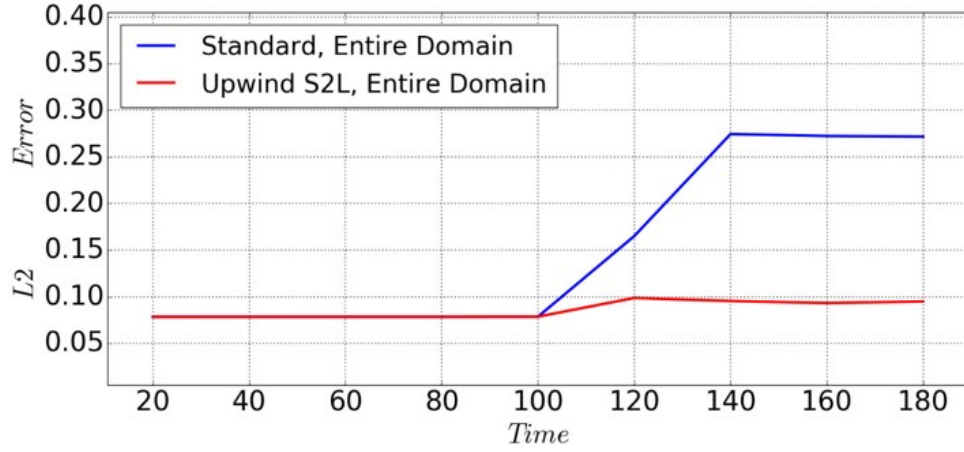


Figure B.17: L2 error plot

The L2 error values are plotted in Figure B.17 for better visualization of error variation with time.

It can be observed that L2 error values (after the hanging node interface in the domain) while using upwind S2L interpolation are less than half (about one-third) the error value obtained while using standard interpolation. This reduction in error might have been possible because of considering the upwind direction in calculating values at ghost cell in the coarse region.

Summary

A sine wave is propagated in a 2-D domain from fine to coarse region. A reflected wave is observed from the boundary between fine and coarse grid section, i.e. from the hanging node interface in the domain and travelling back into the finer region. The amplitude of the actual sine wave going into

the coarse region reduces due to this reflected wave. This clarifies the effect of employing different grid spacing on the simulation accuracy. The new interpolation is found to be successful in reducing the reflected wave and in almost maintaining the amplitude of the propagating sine wave.