CENTRALE NANTES

THESE DE DOCTORAT DE

L'ÉCOLE CENTRALE DE NANTES

ECOLE DOCTORALE N° 602
Sciences pour l'Ingénieur
Spécialité : Mécanique des Milieux Fluides

Par
Young Jun KIM

Numerical improvement and validation of a naval hydrodynamics CFD solver in view of performing fast and accurate simulation of complex ship-wave interaction

Thèse présentée et soutenue à Nantes, le 7 Juillet 2021
Unité de recherche : UMR 6598, Laboratoire de recherche en Hydrodynamique, Énergétique et Environnement Atmosphérique (LHEEA)

Rapporteurs avant soutenance :

Michel BENOIT	Enseignant-chercheur HDR, École Centrale Marseille
Bettar Ould EL MOCTAR	Professeur, Université de Duisburg-Essen (Allemagne)

Composition du Jury :

Président :	Andrea DI MASCIO	Professeur, Université de l'Aquila (Italie)
Examinatrice :	Giuseppina COLICCHIO	Docteure, CNR-INM (Italie)
Dir. de thèse :	David LE TOUZÉ	Professeur des universités, École Centrale de Nantes
Co-encadrant:	Benjamin BOUSCASSE	Chargé de recherche, École Centrale de Nantes

Invité

Sopheak SENG	Hydrodynamic Research Engineer, Bureau véritas, Paris

Doctoral Thesis

Numerical improvement and validation of a naval hydrodynamics CFD solver in view of performing fast and accurate simulation of complex ship-wave interaction

by

Young Jun Kim

École Centrale de Nantes

Nantes, 2021

Acknowledgements

First of all, I give my sincere gratitude and respect to my supervisor Prof. David Le Touzé and co-director, Dr. Benjamin Bouscasse. They were always eager to give great insights into the numerical and mathematical problems I faced during my PhD studies. I also feel thankful to my supervisors for their massive efforts in ordering my manuscripts and improving my poor grammars. I also thank Dr. Sopheak Seng and Dr. Sime Malanica from Bureau Veritas for giving me the opportunity to deeply study the OpenFOAM and CFD world.

I am deeply appreciated to Prof. Bettar Ould el Moctar at the University of Duisburg-Essen and Prof. Michel Benoit at École Centrale Marseille for being the reviewers of my thesis. I want to express gratitude to Prof. Andrea Di Mascio at Université de L'Aquila for accepting the president of the jury members. Also, I would like to thank Dr. Giuseppina Colicchio at CNR-INM and Dr. Sopheak Seng for being the jury member.

I would like to thank the researchers in École Centrale de Nantes. First, I would like to give thank Dr. Julien Michel for sharing his impact test cases. These are very helpful for the comparison with different interface treatment schemes. Also, I appreciated Maxime Canard for providing HOS data and the irregular wave experiment data. I was always pleased to work on OpenFOAM with Dr. Zhaobin Li, YoungMyung Choi, and Théo Descamps. The discussions on OpenFOAM with them was constructive and encouraged me a lot. I especially got a lot of help from Dr. Zhaobin Li on the two-phase SWENSE solver. I am thankful to my friends Dr. Chunmei Xie, Dr. Zhenrong Jing, and Gaspard Engel. Moreover, I also thank Shinwoong Kim, Seung Yoon Han, and his family for having friendly trips and fun.

I would like to give a special thanks to Prof. Sun Hong Kwon and Dr. Sime Malanica, who were my supervisors during the master course. They recommended and supported me to start the doctoral course in France. I feel very thankful for their help and also so please to finish the doctoral course.

The financial support from the Chaire Hydrodynamique et Structure Marines CENTRALE NANTES - BUREAU VERITAS should be acknowledged. Also, the financial support for wave experiments from the joint industry project *Reproducible Offshore CFD* should be acknowledged.

Finally, I would like to give thank my family in South Korea. Their warm and consistent supports were essential for a stay in France during the COVID season.

Contents

1 Introduction **1**

1.1 Numerical modelling for fast and accurate wave-structure interaction simulations 1

1.2 Previous and related researches 3

1.2.1 Finite volume method and two-phase viscous flow solvers 3

1.2.2 Interface treatment schemes 4

1.2.3 High order scheme 8

1.3 Contribution of the present work 10

1.4 Thesis outline 12

2 Governing equations **14**

2.1 Governing equations for the averaged two-phase incompressible flow model 15

2.1.1 Boundary conditions for two-phase incompressible flow model 16

2.2 SWENSE model 18

2.2.1 Functional decomposition 18

2.2.2 Conservative SWENSE formulation 19

2.2.3 Boundary conditions for SWENSE model 20

2.2.4 Extrapolation of incident flow quantities to air zone 21

2.3 Turbulence models in OpenFOAM 24

3 Wave theories **26**

3.1 Regular waves 26

3.1.1 Non-linear regular wave theories 26

3.1.2 Regular wave conditions 28

3.2 Non-linear irregular waves 28

3.2.1 Ocean waves 28

3.2.2 Wave breaking model for non-linear irregular waves 31

4 Numerical formulations and OpenFOAM **32**

4.1 Finite volume discretization 32

4.2 Discretization of averaged two-phase incompressible flow model 33

4.3 Discretization of the SWENSE model 34

4.4 Discretization of the transport equations of turbulence model 35

4.5 Wave generation and absorption 36

4.5.1 Relaxation scheme . . . 36
4.5.2 Reconstruction of the non-linear irregular waves in the CFD domain (Grid2Grid) 37
4.6 Modelling of 6 DOFs rigid body motions . . . 37
4.6.1 6 DOFs rigid body motions . . . 37
4.6.2 Mesh morphing due to rigid body motion in the computational domain . . 38
4.6.3 Arbitrary Lagrangian-Eulerian (ALE) formulation . . . 39
4.7 Solution algorithm for the two solvers . . . 41
4.7.1 Set of equations . . . 41
4.7.2 Flow chart of the algorithm . . . 42
4.8 OpenFOAM discretization schemes . . . 44
4.8.1 Discretization schemes (fvScheme) . . . 44
4.8.2 Solvers and PIMPLE options (fvSolution) . . . 47
4.9 OpenFOAM boundary conditions . . . 48
4.9.1 Body/Wall boundary . . . 49
4.9.2 Inlet & outlet wave boundary condition (Far-field condition) . . . 51
4.9.3 Open-air boundary condition . . . 52
4.9.4 Symmetry plane . . . 53

5 Interface treatment schemes 54
5.1 VOF interface convection algorithms . . . 54
5.1.1 Algebraic VOF convection algorithm with Crank-Nicolson scheme . . . 54
5.1.2 Geometric VOF convection . . . 55
5.2 Interface conditions . . . 56
5.2.1 Conditionally averaged two-phase flow model . . . 56
5.2.2 Ghost Fluid Method . . . 57
5.3 Wave propagation in periodic domain . . . 57
5.4 Impact applications . . . 64
5.4.1 Asymmetric wedge entry problem . . . 64
5.4.2 Wedge entry-exit problem . . . 76

6 High-order time integration scheme 83
6.1 Time integration methods . . . 83
6.1.1 DIRK method . . . 84
6.2 Discretization of averaged two-phase incompressible flow model with DIRK method 86
6.2.1 VOF convection algorithm with DIRK scheme . . . 87

6.2.2 Difference between OpenFOAM Crank-Nicolson scheme and DIRK-based Crank-Nicolson scheme for the averaged two-phase flow model 89
6.2.3 The PISO algorithm with DIRK scheme 90
6.3 Applications for simple test cases 91
6.3.1 2D Taylor-Green vortex 92
6.3.2 Regular wave propagation in periodic domain 98
6.4 Discretization of SWENSE model with DIRK method 103
6.5 DIRK method with moving body and grids 105

7 Non-linear wave generation and qualification 106
7.1 Regular wave propagation in a NWT 106
7.1.1 NWT with regular waves: Study on the free-surface turbulence model . . 108
7.1.2 NWT with Regular waves: Efficiency of different time integration schemes in foamStar and foamStarSWENSE 110
7.2 Irregular wave propagation in a NWT 120
7.2.1 NWT with irregular waves: Description 120
7.2.2 NWT with irregular waves: Deterministic analysis 126
7.2.3 NWT with irregular waves: Stochastic analysis 133
7.3 Conclusions and summary of the NWT performances 135

8 Naval applications: Seakeeping analysis 138
8.1 Wigley III 139
8.1.1 Wigley III: Geometry and test matrix 139
8.1.2 Wigley III: Numerical setup 140
8.1.3 Wigley III: Efficiency of DIRK methods for the *foamStar* and *foamStar-SWENSE* solvers: effect of time-step size and mesh refinement 143
8.1.4 Wigley III: RAOs comparison with experiments 152
8.2 KCS 156
8.2.1 KCS: Geometry and test matrix 156
8.2.2 KCS: Numerical setup 158
8.2.3 KCS: Results 161

9 Conclusions 169
9.1 Summary and conclusions and perspectives 170
9.1.1 Interface treatment schemes 170
9.1.2 High-order time integration schemes 171

9.1.3 Applications of the developed and updated solvers 171

A Second order upwind biased interpolation scheme with V-option 174

B Stiffly accurate DIRK methods 175

B.1 **Second-order scheme** 176

B.1.1 Implicit Euler 176

B.1.2 ESDIRK[2](2)_SA 176

B.1.3 SDIRK[2](2)_SA 176

B.1.4 ESDIRK[2](3)_SA 177

B.2 Third-order scheme 177

B.2.1 SDIRK[3](3)_SA 177

B.2.2 EDIRK[3](3)_SA_1 177

B.2.3 ESDIRK[3](3)_SA 177

B.2.4 EDIRK[3](3)_SA_2 178

B.2.5 ESDIRK[3](4)_SA 178

Bibliography 179

List of Figures

1.1 Concept and drawing of a computational domain with a body and a free surface. 2

1.2 Three interface treatments: Interface capturing, Interface convection, and Interface condition. 4

2.1 The description of the problem with the general boundaries and the relaxation zones in the computational domain. 14

2.2 SWENSE decomposition of the functional quantities of the total flow into incident and complementary parts (from [1]). 19

2.3 Incident flow extrapolation in the air zone with cubic polynomial. 22

2.4 Extrapolation of incident velocity up to wind zone using cubic polynomial. The color label represents the normalized velocity magnitude (Figure from Choi [2]). 23

3.1 Le Méhauté's diagram with three regular wave conditions [3]. 29

4.1 Finite volume cell in polyhedral shape (figure from [4]). 32

4.2 The definition of the relaxation zone and the pure computational domain 36

4.3 The Galilean referential frame R_0 $(O, \mathbf{x}_0, \mathbf{y}_0, \mathbf{z}_0)$ and the body-fixed frame R_b $(O_b, \mathbf{x}, \mathbf{y}, \mathbf{z})$ in side-view and top-view (Figure from [5]). 38

4.4 The changes of the control volume and signed swept-volume of each face. 41

4.5 Flow chart of the solvers *foamStar* and *foamStarSWENSE*. The yellow color indicates the parts solving each set of the equations. 42

5.1 Computational domain and boundary conditions 58

5.2 Free-surface profiles at times $t = 10T, t = 20T, t = 30T$ and $t = 40T$ for the solvers *interFoam* and *interFlow* 61

5.3 Free-surface profiles at times $t = 10T, t = 20T, t = 30T$ and $t = 40T$ for the solvers *interFoamGFM* and *interFlowGFM* 62

5.4 Convergence of moving window FFT first-harmonic amplitude and phase difference with respect to spatial and temporal resolutions. 63

5.5 Asymmetric Wedge Entry setup description 64

5.6 Computational mesh for the asymmetric Wedge Entry problem: Full grid (left) and the grid near the wedge (right). 66

5.7 Lightweight condition: VOF predicted free-surface and pressure field for the *interFoam* (left) and *interFoamGFM* (right) solvers. Each row is made of snapshots at times $t = 0.35$s, $t = 0.355$s, and $t = 0.36$s 67

5.8 Heave and pitch acceleration with lightweight condition and the two mesh resolutions. The four OpenFOAM solvers are compared with the experiment and the SPH result. 68

5.9 Heave and pitch acceleration with mediumweight condition and different mesh resolutions. The four OpenFOAM solvers are compared with the experiment result. 68

5.10 Lightweight condition: Pressure field (N/m^2) and free surface obtained with the four different solvers. The columns show from left to right the results from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is made of snapshots at times $t = 0.355$s, $t = 0.36$s, $t = 0.365$s, $t = 0.37$s, $t = 0.375$s, $t = 0.38$s, and $t = 0.385$s. 69

5.11 Lightweight condition: Pressure field (N/m^2) and free surface obtained with the four different solvers. The columns show from left to right the results from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is made of snapshots at times $t = 0.39$s, $t = 0.40$s, $t = 0.41$s, $t = 0.42$s, $t = 0.43$s, $t = 0.44$s, and $t = 0.45$s. 70

5.12 Mediumweight condition: Pressure field (N/m^2) and free surface obtained with the four different solvers. The columns show from left to right the results from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is made of snapshots at times $t = 0.355$s, $t = 0.36$s, $t = 0.365$s, $t = 0.37$s, $t = 0.375$s, $t = 0.38$s, and $t = 0.385$s. 71

5.13 Mediumweight condition: Pressure field (N/m^2) and free surface obtained with the four different solvers. The columns show from left to right the results from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is made of snapshots at times $t = 0.39$s, $t = 0.40$s, $t = 0.41$s, $t = 0.42$s, $t = 0.43$s, $t = 0.44$s, and $t = 0.45$s. 72

5.14 Lightweight condition: Pressure field (N/m^2) and free surface comparison between *interFoam* and SPH [6]. Each row is made of snapshots at times $t = 0.36$s, $t = 0.365$s, and $t = 0.37$s. 73

5.15 Lightweight condition: Pressure field (N/m^2) and free surface comparison between *interFoam* and SPH [6]. Each row is made of snapshots at times $t = 0.375$s, $t = 0.38$s, and $t = 0.385$s. 74

5.16 Lightweight condition: Pressure field (N/m^2) and free surface comparison between *interFoam* and SPH [6]. Each row is made of snapshots at times $t = 0.40$s, $t = 0.42$s, and $t = 0.44$s. 75

5.17 Sketch of wedge entry and exit problem 76

5.18 Vertical motion of the wedge in the four cases. Left plot is with real time, and right plot is with normalized time. 77

5.19 Case 1: Normalized Pressure field ($P/(0.5\rho V_0^2)$) and the VOF field with four different solvers. Each column shows the result from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is the snapshot at $t/t_0 = 0.5$, $t/t_0 = 1.0$, $t/t_0 = 1.5$. 79

5.20 Case 2: Normalized Pressure field ($P/(0.5\rho V_0^2)$) and the VOF field with four different solvers. Each column shows the result from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is the snapshot at $t/t_0 = 0.5$, $t/t_0 = 1.0$, $t/t_0 = 1.5$. 79

5.21 Case 3: Normalized Pressure field ($P/(0.5\rho V_0^2)$) and the VOF field with four different solvers. Each column shows the result from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is the snapshot at $t/t_0 = 0.5$, $t/t_0 = 1.0$, $t/t_0 = 1.5$. 80

5.22 Case 4: Normalized Pressure field ($P/(0.5\rho V_0^2)$) and the VOF field with four different solvers. Each column shows the result from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is the snapshot at $t/t_0 = 0.5$, $t/t_0 = 1.0$, $t/t_0 = 1.5$. 80

5.23 Comparison of normalized pressure field between the SPH (left) and *interFoam* (right) for Case 3. Each row is the snapshot at $t/t_0 = 0.5$, $t/t_0 = 1.0$, $t/t_0 = 1.5$. 81

5.24 Comparison of normalized pressure field between the SPH (left) and *interFlow* (right) for Case 3. Each row is the snapshot at $t/t_0 = 0.5$, $t/t_0 = 1.0$, $t/t_0 = 1.5$. 82

6.1 Four-stage SDIRK (top-left), SDIRK_SA (top-right), ESDIRK (bottom-left) and ESDIRK_SA (bottom-right) methods [7] 85

6.2 Targeted solver algorithm. 87

6.3 VOF convection algorithm. 88

6.4 PISO algorithm. 91

6.5 Initial condition: velocity magnitude and velocity vector (left) and vorticity and stream lines (right). 93

6.6 Convergence of velocity (left) and pressure (right) magnitude errors with respect to mesh resolution measured through the cell Reynolds number, with fixed time-step size or fixed Courant number. 94

6.7 Convergence to the analytical solution (see Eqn. (6.31)) of velocity (left plots) and pressure (right plots) magnitude errors with respect to time resolution measured through the Courant number, for three space resolutions: 128 × 128 (top plots), 256 × 256 (middle plots), and 512 × 512 grids (bottom plots). 97

6.8 Intrinsic convergence (see Eqn. (6.32)) of velocity (left plot) and pressure (right plot) magnitude errors with respect to time resolution measured through the Courant number, for the 512 × 512 space resolution. 98

6.9 Convergence of normalized wave amplitude for three different wave conditions. Left is spatial convergence and right is temporal convergence. 100

6.10 Comparison of the efficiency of the solvers, wave steepness 10% and 5%. 101

6.11 Comparison of the efficiency of the solvers, wave steepness 1%. 102

7.1 Schematic view of the NWT. 106

7.2 Three mesh refinement zones. 107

7.3 Comparison between standard $k-\omega$ SST models, free-surface $k-\omega$ SST and no turbulence model, for wave steepnesses 10%, 5% and 1% at positions $x = 0$m, and $x = 2.5$m. 109

7.4 Comparison of first harmonics of the 10%-steepness wave measured at $x = 0$m using different time integration schemes. $\Delta x = \lambda/100$. Each row corresponds to a different time step. Left: Two-phase flow solver (*foamStar*). Right: SWENSE solver (*foamStarSWENSE*). 112

7.5 Comparison of first harmonics of the 10%-steepness wave measured at $x = 0$m using different time integration schemes. $\Delta x = \lambda/50$. Each row corresponds to a different time step. Left: Two-phase flow solver (*foamStar*). Right: SWENSE solver (*foamStarSWENSE*). 113

7.6 Comparison of first harmonics of the 10%-steepness wave measured at $x = 0$m using different time integration schemes. $\Delta x = \lambda/25$. Each row corresponds to a different time step. Left: Two-phase flow solver (*foamStar*). Right: SWENSE solver (*foamStarSWENSE*). 114

7.7 Comparison of the efficiency of the different time integration schemes using the normalized error of wave amplitude averaged from $1T - 10T$ at $x = 0$ in the NWT. Left: two-phase flow solver (*foamStar*). Right: SWENSE solver (*foamStarSWENSE*). 117

7.8 Comparison of the efficiency of the different time integration schemes using the normalized error of wave amplitude averaged from $90T - 100T$ at $x = 0$ in the NWT. Left: two-phase flow solver (*foamStar*). Right: SWENSE solver (*foamStarSWENSE*). 118

7.9 Influence of time and space resolutions on the efficiency, with *RKCN099* and *SDIRK[2](2)* schemes using the normalized error of wave amplitude at $x = 0$ in the NWT. Left: Two-phase flow solver (*foamStar*). Right: SWENSE solver (*foamStarSWENSE*). 119

7.10 Target wave spectra for Case 1 and Case 3. Corresponding spectrum parameters are listed in Table 7.3. 121

7.11 Sketch of the HOS and CFD domains for irregular wave simulation. All dimensions are in real scale and the position $x_{HOS} = 700$m is identical to $x_{CFD} = 0$m. 122

7.12 Examples of wave breaking events prediction from HOS-NWT and Tian breaking model. The color of the box corresponds to its wave crest height (H_c) 124

7.13 Comparison of the last split run to the last part of the simulation without splitting for Case 1 with the two-phase flow solver *foamStar* at the target location $x_{CFD} = 0$m. 127

7.14 Comparison of the last split run to the last part of the simulation without splitting for Case 3 with the two-phase flow solver *foamStar* at the target location $x = 0$m. 128

7.15 Comparison of the time series of wave elevation at $x_{CFD} = 0$m for Case 3 seed 6. Experimental signal is compared to simulations with and without free-surface k-ω SST turbulence model. 129

7.16 The wave breaking events used for the wave profile comparison. 130

7.17 Comparison of the wave profiles from HOS-NWT solver and *foamStar* solver with three resolutions during the selected wave breaking events of Case 3 seed 3 run 7. 131

7.18 Comparison of the wave profile from HOS-NWT solver and *foamStar* solver with three resolutions during the selected wave breaking events of Case 3 seed 6 run 4. 132

7.19 Comparison of the irregular wave spectrum from HOS-NWT, experiment and *foamStar* for Case 1 and Case 3 at the target location $x_{CFD} = 0$m. 133

7.20 Difference ratio between target wave spectrum and other averaged wave spectra in Figure 7.19. Left plot: Case 1. Right plot: Case 3. 134

7.21 Distribution of wave crests at the target location $x_{CFD} = 0$m. Upper plots show the POEs of each seed, and lower plots show the POEs of the ten seeds altogether. 135

8.1 The boundary conditions for Wigley III simulation. 140

8.2 The top view of the Wigley III hull and the relaxation zones. The color contours represent the weight function intensity: red represents $w = 1$ and the blue represents $w = 0$. 141

8.3 Wigley *Normal* mesh with initialized VOF field 142

8.4 Heave and pitch motions predicted by the *foamStar* solver for the *Medium* mesh and time-step sizes $h = 0.004$s, 0.008s, 0.016s 144

8.5 Total resistance predicted by the *foamStar* solver for the *Medium* mesh and for the three time-step sizes $h = 0.004$s, 0.008s, 0.016s 145

8.6 Heave and pitch motions, and total resistance for the *Coarse* and *Medium* meshes, with time-step size $h = 0.004$s, for the *foamStar* solver 146

8.7 Heave and pitch motions, and total resistance for the *Coarse* and *Medium* meshes, with time-step size $h = 0.004$s, for the *foamStarSWENSE* solver 147

8.8 Heave and pitch motions, and total resistance for the *Medium* mesh, with time-step size $h = 0.004$s, for the *foamStar* and *foamStarSWENSE* solvers with *implicit Euler* and *RKCN099* schemes. 148

8.9 Wigley III motion RAOs in head waves for Froude numbers Fr = 0.2, Fr = 0.3 and Fr = 0.4. At each computed frequency, the *foamStar* and *foamStarSWENSE* results are shifted a little to the left and to the right for better visualization. . . . 154

8.10 Comparison of Wigley III motion RAOs in head waves for Fr = 0.2, Fr = 0.3 and Fr = 0.4 with three wave heights. The numerical results are shifted a little for better visualization. 155

8.11 Added resistance of the Wigley III hull in head waves for Froude numbers Fr = 0.2, Fr = 0.3 and Fr = 0.4. 155

8.12 Boundary patches and computational domain for the KCS simulations 158
8.13 Configuration of the relaxation zones and visualization of the exponential weight function field. The color contours represent the weight function: red represents $w = 1$ and the blue represents $w = 0$. 159
8.14 An example of computational mesh (Coarse mesh) 160
8.15 An example of computational mesh near the body (Coarse mesh) 160
8.16 Mean heave and pitch motions of KCS in head waves 163
8.17 First harmonic motions of KCS in head waves 164
8.18 Mean and first harmonic resistance of KCS in head waves 167
8.19 Added resistance of KCS in head waves 168

List of Tables

1.1 Combinations of interface capturing and convection schemes and their properties. 8

2.1 Coefficients of the k-ω SST model. 25

3.1 Three regular wave conditions which have 10%, 5%, and 1% steepnesses based on the stream function wave theory 28

4.1 Spatial discretization schemes 46

4.2 Body/Wall boundary conditions for all variables 49

4.3 Inlet & outlet boundary conditions for all variables 52

4.4 Open boundary conditions for all variables 53

5.1 Definition of the four different combinations of solvers 57

5.2 Wave condition 58

5.3 Mesh for the convergence study (Initial Courant number ≈ 0.17) 59

5.4 Summary of orders of convergence and uncertainties of harmonic wave amplitude 60

5.5 Experimental input parameters 65

5.6 Mesh resolutions used for the asymmetric Wedge Entry problem 66

5.7 Wedge entry and exit cases 77

6.1 Identification of DIRK method. Ex: ESDIRK[p](s)_SA_i. 86

6.2 Names and properties of the time schemes. 92

6.3 Test matrix for the spatial convergence on Taylor-Green benchmark. 93

6.4 Test matrix for the temporal convergence of OpenFOAM with various time schemes on the Taylor-Green vortex benchmark. 96

6.5 Four mesh resolutions and five temporal resolutions (gray colored rows are the same condition) 99

7.1 Names and properties of the DIRK schemes used for NWT simulations. 110

7.2 Test matrix for the assessment of the efficiency of each solver and time integration schemes (three mesh resolutions and four temporal resolutions). 111

7.3 Irregular wave parameters based on the JONSWAP spectrum. 121

7.4 Mesh refinement parameters in real scale for the free surface refinement region ($-H_s < z < H_s$) 123

7.5 Numbers of the seeds used for Case 1 and Case 3. 123

7.6 Time intervals of the irregular wave simulations for Case 1 and Case 3 in real scale. 124

7.7 Real scale parameters of the three resolutions used in the free-surface refinement region. 'Case 3 Dx 2m' is the suggested parameter for the irregular wave simulations. 130

7.8 Significant wave height (H_s) and peak period (T_p) of irregular waves measured from the wave elevations. 134

8.1 Main dimensions of the Wigley III model for both experiments and CFD simulations 139
8.2 The six degrees of freedom of the Wigley hull used for experiments and numerical simulations . . . 139
8.3 Test matrix for Wigley III simulations . . . 140
8.4 Boundary conditions for Wigley III simulations, the blue rows represent the variables only used in the SWENSE formulation . . . 141
8.5 Wigley III meshes with two resolutions . . . 142
8.6 Names and properties of the DIRK schemes used for Wigley III simulations. . . . 143
8.7 Comparison of the computational time of Wigley III hull simulations. . . . 149
8.8 Wigley III computations results from *foamStar* solver. . . . 150
8.9 Wigley III computations results from *foamStarSWENSE* solver. . . . 151
8.10 Detailed test matrix for Wigley III simulations . . . 152
8.11 Main particulars of KCS geometry . . . 156
8.12 Degrees of freedom during KCS experiments and numerical simulations . . . 157
8.13 Physical parameters for KCS tests in model scale, used in the experiments and numerical simulations . . . 157
8.14 Regular waves conditions for KCS tests in model scale, used in the experiments and numerical simulations . . . 157
8.15 Boundary conditions for the KCS simulations. The blue rows represent variables used only in *foamStarSWENSE* computations. . . . 159
8.16 KCS computational conditions and mesh size information inside the free-surface refinement zone . . . 160
8.17 Comparison of the computational cost (Hr) of KCS simulations. . . . 162
B.1 Identification of DIRK method. Ex: ESDIRK[p](s)_SA_i. . . . 175

Nomenclature

Latin letters

$\mathbf{S}_f$	Face surface vector
Δx	Cell length
Δy	Cell width
Δz	Cell height
$\mathbf{u}$	Fluid velocity
$\mathbf{g}$	Gravitational acceleration
$\mathbf{n}$	Normal unit vector
A	Wave amplitude
a_N	Off-diagonal coefficient of discretized matrix
a_P	Diagonal coefficient of discretized matrix
C	Courant-Friedrichs-Lewy number (CFL number, Courant number)
c_p	Wave phase velocity
C_x	Wave crest velocity
D	Water depth
F_f	VOF flux at face f
H	Wave height, volume fraction function
h	Time-step-size
H_s	Significant wave height of irregular waves
k	Wavenumber, Turbulence kinetic energy
N	Neighbor cell index
P	Owner cell index
p	Pressure
p_d	Dynamic pressure
T	Wave period
T_p	Peak period of irregular waves
U_x	Wave particle velocity under crest
V	Cell volume
w	Weight function

w^d	Dynamic weight function
CFD	Computational Fluid Dynamics
EFD	Experimental Fluid Dynamics
Fr	Froude number
Re	Reynolds number

Greek letters

α	Volume Of Fluid (VOF)
δ	Boundary layer thickness
η	Wave elevation
γ	Peak enhancement factor used for JONSWAP wave spectrum
κ_R	Reflection coefficient
λ	Wavelength
λ_f	Explicit flux limiter from MULES algorithm at face f
μ	Fluid kinematic viscosity
ν	Fluid dynamic viscosity
ν_t	Eddy viscosity
ω	Wave frequency, Specific dissipation
ϕ	Flux at face f
Φ_I	Incident wave velocity potential
ϕ_I	Incident wave velocity potential in frequency domain
Ψ_I	Incident wave stream function for regular wave
ψ_I	Incident wave stream function in frequency domain
ρ	Fluid density
$\mathbf{x} = (r, \theta, z)$	Point vector in circular cylindrical coordinates
$\mathbf{x} = (x, y, z)$	Point vector in cartesian coordinates

Superscripts

$q^{(n)}$	Time index for old time step
$q^{(n+1)}$	Time index for current computation
$q^{(n,i)}$	Time index for DIRK time index i
$q^{(n,s)}$	Time index for DIRK final time index

Subscripts

τ_i	DIRK coefficient: abscissa
a_{ij}	DIRK coefficient: stage weight
b_i	DIRK coefficient: scheme weight
q_C	Complementary wave flow
q_f	Face properties
q_I	Incident wave flow
q_N	Neighbour cell properties
q_P	Owner cell properties

1 Introduction

1.1 Numerical modelling for fast and accurate wave-structure interaction simulations

Over the past years, Computational Fluid Dynamics (CFD) has taken an important role in the practical design of ships and offshore structures because of its capability to overcome some limits of traditional methods. However, the computational cost is still high, and further numerical development and validations are still needed to evaluate the performance and advantages of using CFD solvers. To be applied in production and increase efficiency, numerical development needs to have several characteristics: accuracy, efficiency, robustness, reliability, and not complicating the user experience (meshing and simulation setup). The present study is done in this context with these requirements.

The ships or offshore platforms operating in the ocean face the environment, such as waves, winds, and current. Those will result in external forces from the ocean, and the floating bodies will continuously move during their lifetime. Therefore, it is essential to predict accurately the external forces and the body motions in the design conditions provided by regulations.

The prediction of body motions in waves is possible with various methods. Combining linearized free-surface boundary conditions and ideal flow assumption, the potential flow model can compute the excitation, diffraction, and radiation forces of the floating object. Potential flow solvers can be based on the Boundary Element Method (BEM) using Green functions [8], Rankine panel method [9], or Finite Element Method (FEM) [10]. They are typically very efficient for frequency domain computations of body motions without a forward speed. The linear hypothesis also induces the need to add some non-linear terms like Froude-Krylov or Morison ones, which might complexify the code and reduce the final efficiency. When the applications involve wave breaking, two-phase effect, vorticity, turbulence, and viscosity, linear potential flow solver became highly ineffective. Especially with a forward speed condition, the efficiency and the robustness of the potential flow solver can significantly decrease. Well-known potential flow commercial codes published are HydroStar [11] from Bureau Veritas, WAMIT [12], WADAM [13] from DNV GL, AQWA [14], which use the Green function to achieve the solution. Also, an open-source code NEMOH [15] is available.

Real ships and offshore platforms have sometimes very complex geometries, and the wave-structure interaction phenomena can be assessed accurately by conducting experiments. Performing model scale experiments is an excellent method to assess the loads and motion time history and to mimic complex natural phenomena. However, there are numerous limitations to the experiment, such as the scaling effect and overall cost. Making experimental models and using test facility is often very expensive or non-accessible. Moreover, it is challenging to add appendages or changing hull shape for design optimization. Also, constructing the measuring and data acquisition system requires a lot of manpower and is sometimes impossible for an academic research purpose.

The limitation of the linearized potential flow model and the infrastructure needed for ex-

periments make the Navier-Stokes CFD solvers more and more attractive as the computational power increases and the computational cost decreases. The Navier-Stokes equations can be discretized using the Finite Difference Method (FDM), FEM, or Finite Volume Method (FVM) and can be solved using various numerical schemes to describe the flow fields. In the marine context, CFD often refers to a viscous flow solver based on the Navier-Stokes equations. The flow's turbulence and viscosity effects are often associated with a huge Reynolds number. The Reynolds averaged Navier-Stokes equations (RANSE) with a turbulence model is the more economical way to go with. It is commonly used to predict the floating body's external forces and body motion. The FVM is the generally used methodology due to its easy application to conservation law. With FVM, the flow fields are discretized in a 3D polyhedral computational grid that can integrate the complicated body shape. The naval applications are naturally three-dimensional two-phase flow, and the numerical modelling and transportation of the free-surface is a very challenging job. Even if computational power is enhanced, 3D realistic wave simulations with floating bodies are still costly and sometimes unpractical. Well-known commercial viscous flow solvers using FVM used for naval applications are Simcenter STAR-CCM+ [16], FINE/Marine [17], and ANSYS Fluent [18]. This work used an **Open**-source **F**ield **O**peration **A**nd **M**anipulation (**OpenFOAM** [19]) with an additional in-house code package specialized for naval application such as 6DOFs motion controls and wave generation. OpenFOAM provides a C++ toolbox for the development of user-customized numerical solvers with pre/post-processing utilities, mainly for fluid mechanics solutions. Due to its flexibility, OpenFOAM is widely used for research or industrial purpose by many research institutions for various types of numerical applications. OpenFOAM has three main fork versions released under GNU General Public License: OpenFOAM developed and maintained by OpenCFD Ltd., FOAM-Extend project maintained by Wikki Ltd., and OpenFOAM-Foundation mainly developed by CFD Direct Ltd.. This thesis used the OpenFOAM-Foundation variant version OpenFOAM 5.x released in July 2017.

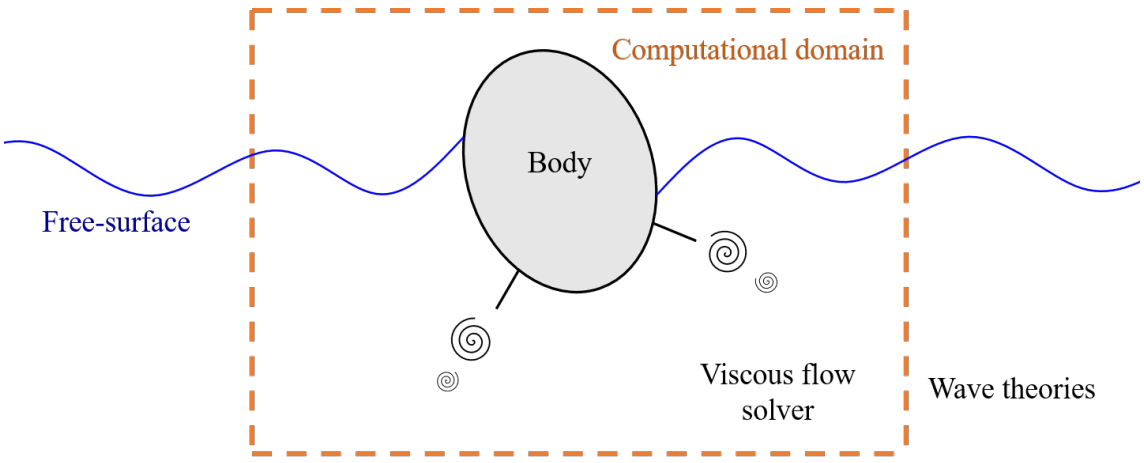


Figure 1.1. Concept and drawing of a computational domain with a body and a free surface.

Figure 1.1 presents an example of a computational domain with the floating body and the free-surface. The computational grid has to be refined more near the body to model its shape, its boundary layer and its displacement properly. The mesh far from the body can be coarse

for cases without incoming waves. However, to properly propagate waves for a few wavelengths, the numerical dissipation around the free-surface must be minimized by using more grid cells or using low dissipative schemes. Also, wave propagation is influenced by the eddy viscosity from the classical turbulence model, and specific care is needed in those conditions. For fast but accurate simulation, the computational zone must be as small as possible while it should be large enough to dissipate the disturbance from the body. For the same reason, the total number of cells should be as small as possible while maintaining accuracy.

1.2 Previous and related researches

1.2.1 Finite volume method and two-phase viscous flow solvers

The RANSE version of Navier-Stokes equations can be solved efficiently by decomposing velocities and pressure into time-averaged and fluctuating parts. These equations are solved numerically because analytical solutions are almost never available in the presence of a free-surface and floating body. The FVM satisfies the conservation laws in discretizing the fluid domain into small control volumes (CVs), where fluxes come in and out through the control surfaces. If the flow is assumed incompressible, the reconstructed governing equations allow for solving the unsteady problem using the Semi-Implicit Method for Pressure Linked Equations (SIMPLE) algorithm [20] or/and the Pressure-Implicit with Splitting of Operators (PISO) algorithm [21]. The open-source C++ library OpenFOAM provides a platform to support unstructured polyhedral FVM and other related functions for general CFD applications. The OpenFOAM solver *InterDyMFoam* solves RANSE based two-phase flows with free-surface using the SIMPLE and PISO algorithms. As a layer to OpenFOAM, this study used an additional library, `foamStar`, which supports specific functionalities for naval applications such as wave generations and floating body control.

In the thesis, an alternative set of governing equations, the Spectral Wave Explicit Navier-Stokes Equation (SWENSE) method, is also used. The SWENSE method aims to solve only the complementary flow generated from the disturbance of the incident waves. The SWENSE method was introduced for the wave-structure interaction problem using the idea that the total flow quantity can be decomposed into the incident wave and the complementary part [22]. It is expected to reduce the computation burden due to wave propagation. The SWENSE method was successfully validated in a single-phase flow solver both with regular waves and irregular waves [23, 24]. The SWENSE method has then recently been reformulated for two-phase flow using the level-set method and ghost fluid method by Vukĉević [25]. Moreover, Li [1] decomposed the averaged two-phase RANS equations into the incident component and the scattered component. He tested several SWENSE formulations with the Volume of Fluid (VOF) method for fixed cylinders and CALM buoy. Choi started from a non-conservative version of Li's SWENSE formulation and used a Level-set function to define the air/water interface [2]. However, the SWENSE study by Li and Choi did not include a validation with a moving floating body.

In the thesis, the developments are done on two numerical aspects: the interface treatment scheme and the possibility of increasing the order of the discretization schemes. The first one

largely influences the robustness and accuracy of marine simulations, and the second one has a large role in computational efficiency.

1.2.2 Interface treatment schemes

The proper modelling of the free-surface is essential for stable and accurate two-phase flow simulation. Moreover, the free-surface modelling dramatically affects the quality of the wave propagation, which is one of the essential parts of naval CFD applications. It is still challenging to generate the desired wave precisely and limit the dissipation to maintain the wave amplitude during the simulation. Further, when this is achieved, good energy conservation may lead to instabilities in the flow. Two-phase flow naval CFD solvers can be modelled as one air/water mixture or as two separated fluid connected by a set of boundary conditions across the interface.

In this thesis, the interface treatment scheme refers to the free-surface modelling and numerical treatment of the discontinuous properties at the interface. The interface treatment schemes for two-phase flow refer to the following three categories.

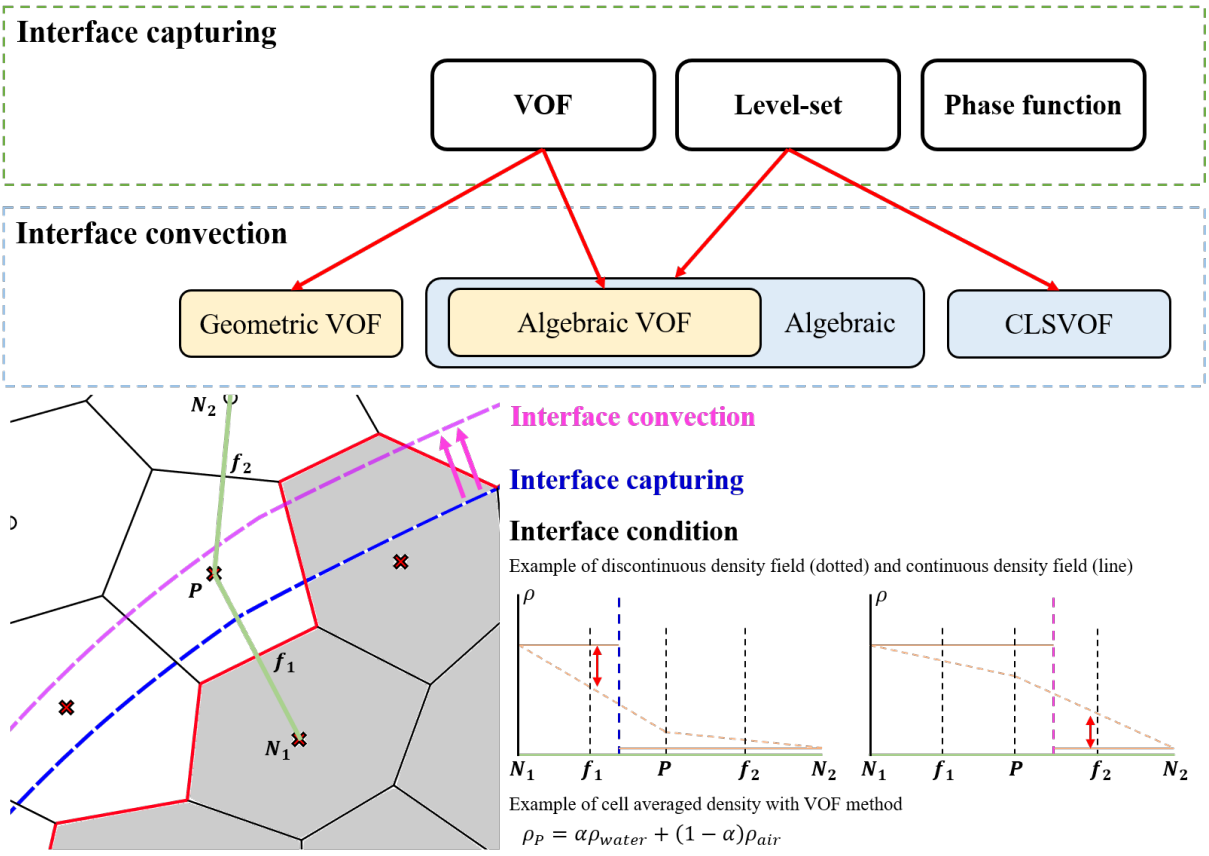


Figure 1.2. Three interface treatments: Interface capturing, Interface convection, and Interface condition.

- **Interface capturing**: Mathematical or numerical modelling of the free-surface.
- **Interface convection**: Algorithms for the transportation of the free-surface in time.

- **Interface conditions**: When the two phases are treated as two separate fluids and not as a mixture, the interface condition becomes a proper boundary condition. When the two phases are modelled with air/water mixture as in the present work, this condition is the key to evaluate the density and other properties in the interface cells (mathematical and numerical definition of the physical quantities at the free-surface cells and faces).

Figure 1.2 describes three interface treatment kinds and their example. The interface capturing and interface convection schemes are highly related. The interface boundary conditions refer to the mathematical or numerical definition of the interface quantities and boundary conditions.

Interface capturing methods

The following are popular interface capturing methods used to indicate free-surface.

- The Volume of Fluid (VOF) method by Hirt and Nichols [26]
- The Level-set method by Osher and Sethian [27]
- The Phase Function method by Boettinger et al. [28]

The Level-set method is a method that uses the signed distance function measured from the interface. The Level-set method is known as accurate with a high-order spatial scheme. However, using a Level-set method does not guarantee the boundedness of mass during the convection of the Level-set function. The Level-set method often uses re-distancing of the Level-set function to improve accuracy [29].

Similar to the Level-set method, the Phase-function method uses a smooth function varying between $[-1\ 1]$ near the interface [28]. In general, the smooth phase-function is a hyperbolic function. The phase-function method also does not guarantee mass conservation; however, the proper choice of a smooth function and interface size may improve the interface's quality during the simulation.

The VOF method uses a step-like function that varies between zero and one. The VOF indicates the volume fraction of the target fluid in a cell. Eqn. (1.1) and Eqn. (1.2) define H the volume fraction function at position $\mathbf{x}$, and α_i the volume fraction of a computational cell i (Ω_i):

$$H(\mathbf{x},t) = \frac{\rho(\mathbf{x},t) - \rho_{other}}{\rho_{target} - \rho_{other}} = \begin{cases} 1 & \text{if } \mathbf{x} \text{ is in the target fluid} \\ 0 & \text{if } \mathbf{x} \text{ is in the other fluid} \end{cases}, \tag{1.1}$$

$$\alpha_i = \frac{1}{V_i}\int_{\Omega_i} H(\mathbf{x},t)dV_i \begin{cases} 1 & \text{if cell } i \text{ is fully filled with target fluid} \\ 0 & \text{if cell } i \text{ is fully filled with other fluid} \end{cases}. \tag{1.2}$$

where V_i is the volume of the cell i, ρ is the fluid density. The material derivative of H yields Eqn. (1.3) and Eqn. (1.4). With the flow assumed incompressible, the right-hand side of Eqn. (1.4) is zero.

$$\frac{DH}{Dt} = \frac{\partial H}{\partial t} + \nabla\bullet(\mathbf{u}H) - H\nabla\bullet\mathbf{u} = 0 \tag{1.3}$$

$$\frac{\partial \alpha_i}{\partial t} + \frac{1}{V_i}\int_{\partial\Omega_i} (\mathbf{u}\bullet\mathbf{n}) H(\mathbf{x},t) dS = \frac{1}{V_i}\int_{\Omega_i} H\nabla\bullet\mathbf{u} dV_i \tag{1.4}$$

The VOF method is known to conserve mass well when the face fluxes are divergence-free. However, VOF suffers interface smearing during the convection and has a relatively large error on calculating the interface curvature (κ) for a precise surface tension calculation. This thesis used VOF interface capturing techniques to define the free-surface due to its mass conservation property among the above interface capturing schemes, neglecting surface tension which has meaningless effects on the marine engineering problem at aim.

Interface convection methods

The interface convection refers to the numerical method of transportation of the interface in time. The interface convection methods can be generally categorized into two categories; geometric convection scheme and algebraic convection scheme. As described in Figure 1.2, the interface convection schemes are dependent on the choice of interface capturing scheme. Since the VOF interface capturing method is applied for this thesis, the geometric VOF method and the algebraic VOF method are discussed here.

The geometric VOF method finds the interface and its movement geometrically, whereas the algebraic VOF method obtains the change of VOF with numerical approximations. The geometric VOF methods generally proceed in two steps. First, the interface is piecewise reconstructed in each cell using the volume fraction information. Then, the second step advects the reconstructed interface using the known flow velocity and the geometric methods. For instance, Youngs' Piecewise Linear Interface Calculation (PLIC) method uses the directional split algorithm to reconstruct the interface, which forms the underlying basis for many reconstruction algorithms in use today. The Youngs interface advection method is achieved using Eqn. (1.4) and the volume fraction advected across the cells [30]. More recently, Roenby et al. successfully implemented a geometric VOF approach named *IsoAdvector* method, which works for the unstructured polyhedral mesh [31, 32].

The algebraic VOF convection methods are more classical interface convection. With the algebraic VOF method, the volume fraction is obtained via Eqn. (1.4) with numerical approximations and numerical schemes. These methods compute the fluid fluxes of the convection term algebraically without the need for geometric reconstruction of the interface. The algebraic VOF methods conserve the mass well but suffer from interface smearing. Therefore, several algebraic interpolation (convection) schemes have been applied to prevent interface smearing and maintain the boundedness of VOF. The cells-to-face interpolation of the Eqn. (1.4) convection term is computed by blending the low-dissipative and high-resolution (high accurate and high-dissipative) fluxes with flux limiters. These types of methods are Flux Corrected Transport (FCT) schemes and techniques and commonly named compressive interface convection schemes. For example, the HRIC (High-Resolution Interface Capturing) method by Muzaferija and Perić blends the non-diffusive downwind scheme with the diffusive upwind scheme using Courant number and the angle between the VOF gradient and the vector between cell centres [33]. The CICSAM (Compressive Interface Capturing Scheme for Arbitrary Meshes) method by Ubbink and Issa blends the downwind scheme with the ULTIMATE-QUICKEST scheme [34]. After these developments,

various algebraic compressive methods such as Inter-Gamma Differencing Scheme [35], THOR [36], STACS (Switching Technique for Advection and Capturing of Surfaces scheme) [37], HiRAC (higher resolution artificial compressive) [38], M-CICSAM (modified CICSAM)[39], and THINC (Tangent of hyperbola interface capturing) [40, 41] methods were investigated for more accurate and efficient interface convection. With compressive interface treatment scheme, flux limiters are used in high-resolution schemes to avoid spurious oscillations (wiggles), which might occur with high-order spatial discretization schemes due to discontinuities in the solution domain. The use of flux limiters, together with an appropriate high-resolution scheme, makes the solutions total variation diminishing (TVD).

Interface conditions

In addition to interface capturing and interface convection methods, the interface modelling requires additional condition both in the case of a two-fluid model and in the case of air/water mixture.

In the case of the two-fluids model, the interface condition is a boundary condition similar to the free-surface boundary conditions obtained in potential flow, where the kinematic free-surface boundary condition ensures the continuous normal velocity across the free-surface and the dynamic continuity of the stress. However, this leads to discontinuous density and viscosity at the free-surface, which cause discontinuous pressure gradient and tangential velocity gradient. Vukčević developed a second-order accurate Ghost Fluid Method (GFM) for two-phase flow and arbitrary polyhedral mesh. In this GFM model, the discontinuities at the interface are treated by applying jump conditions from the free-surface boundary conditions. The details on this GFM can be found in the PhD thesis of Vukčević [25].

In the case of air/water mixture, the discontinuous properties near the free-surface can be modelled by smoothing the transition between the two phases, which ensures continuity of velocity and stress naturally, as used in interFoam and by many authors [19, 42, 43, 44, 45]. For instance, the density field is calculated by blending the densities of the two phases. The blended density field prevents the free-surface from being a sharp interface. The smeared interface often yields spurious velocity over the free-surface. To avoid having too thick smeared interface, some compressible term can be added in the convection phase.

Advantages and drawbacks of the different interface treatment schemes

Even if the free-surface flow problem has been studied for decades, due to the free-surface's discontinuous properties, the interface modelling of the two-phase flow solver is still a challenging research subject. For instance, none of the described well-known methods satisfies the following three properties: mass conservation, sharp interface preservation, and Courant–Friedrichs–Lewy (CFL) limitation. Table 1.1 presents a summary of these three properties for four different combinations of interface capturing and convection schemes.

$$C = \frac{\mathbf{u}h}{\Delta\mathbf{x}} \tag{1.5}$$

The CFL limitation uses the dimensionless Courant number (C) defined as Eqn. (1.5), where

$\mathbf{u}$ is a particle velocity, h is a time-step size, and $\Delta\mathbf{x}$ is a representative cell size. The Courant number represents the ratio between the moved distance of the particle and cell size where the particle exists. Therefore, if there is a Courant number limitation, a simulation needs an appropriate range of the time-step and cell size. For explicit time integration schemes, the Courant number highly influences the simulation stability, and the Courant number must stay below the threshold value 1. Simulations with an implicit time integration scheme have formally no Courant number limitation while the Courant number still influences the simulation accuracy. In most naval applications limiting the Courant number below 1 is unpractical because of the large forward velocity and the tiny cell size in boundary layers. To use the geometric VOF method, the user must then use a small time-step size or large cell size near the body, which is not commonly accepted.

Table 1.1. Combinations of interface capturing and convection schemes and their properties.

Interface treatment	Algebraic VOF	Geometric VOF	Algebraic Level-set	Algebraic Phase-function
Mass conservation	o	o	x	x
Sharp interface	x	o	o	o
Less CFL limitation	o	x	o	o

The lack of mass conservation may lead to spurious pressure oscillations and could bring, e.g., wrong mean heave motion in the case of a floating body. The algebraic and geometric VOF methods conserve the mass well. However, the Level-set and Phase-function ones do not guarantee mass conservation and require higher-order schemes to minimize the mass loss/gain. The lack of a sharp interface (smeared interface) can dissipate energy near the free-surface. The Level-set and Phase-function always have a sharp interface from their definition. The geometric VOF convection, such as in the *Isoadvector* method, ensures the free-surface sharpness, while a CFL condition strictly limits geometric convection schemes due to their explicit nature.

Various authors published many interface treatment schemes for decades, but not many of them are thoroughly tested for various naval applications. In a preliminary study, we investigated the influence of the algebraic VOF method and geometric VOF methods in two-phase flow simulations to understand the different interface treatment schemes' numerical properties [46]. Chapter 5 extends this previous study and describes the influence of different interface treatment schemes for the two-phase wave propagation test cases and 2D wedge impact cases.

1.2.3 High order scheme

Most of the phenomena studied in this field, such as a ship in waves or sloshing and slamming inside LNG (Liquid Natural Gas) tanks, are **two-phase unsteady** flows. For these simulations, second-order (in time and space) methods are frequently used, and their accuracy and efficiency for two-phase unsteady flows are validated by many authors. However with the increased re-

quirement of numerical simulation, there are needs for faster and accurate time integration and spatial discretization schemes. High-order numerical methods are believed to have the advantages of high-order of accuracy. For the single-phase Navier-Stokes equations, high-order methods can yield more accurate results in shorter computational times [47]. However, in two-phase flow applications such as wave propagation and advancing ships in waves, the accuracy and efficiency of high-order methods on unstructured grids are not fully proven yet.

High-order time integration scheme

The time integration method can be explicit or implicit. The explicit scheme is known to be fast and accurate, but the explicit scheme's problem is that the CFL condition restricts the time-step size. This restriction would make the simulations with the boundary layer flows too slow. The CFL condition does not formally restrict the time-step size for implicit time integration. Nevertheless, implicit methods often require iteration methods to solve large-scale algebraic equation system.

The OpenFOAM supported time integration schemes are Implicit Euler (Euler), Crank-Nicolson (CN), and second-order backward differential formula (BDF2). The Euler scheme is known to be stable, but the energy dissipation for every time step is significant. The CN scheme is a second-order semi-implicit scheme that includes an off-centring coefficient to allow flexibility for a fine-tuning of numerical damping [48]. The BDF2 method is a second-order scheme that uses solutions at the two previous time steps. However, the BDF2 scheme does not guarantee the VOF field's boundedness, which is problematic in two-phase flow [46].

Research on the numerical integration of ordinary differential equations (ODEs) has been a topic of interest for a long time. High-order time integration schemes have low dissipation, but they are more costly than lower-order schemes and can be less stable. One of the popular high-order time integration schemes is the third-order three-stage TVD (total-variation-diminishing) explicit Runge-Kutta (ERK) method [49]. This explicit scheme satisfies the TVD properties for a CFL number below one, and it shows good numerical stability and accuracy [50, 51]. However, it is not easy to satisfy this maximum CFL criterion in typical naval and offshore applications.

There is less limitation on the maximum usable CFL number when using an implicit or a semi-implicit time-stepping scheme. The diagonally implicit Runge-Kutta (DIRK) schemes are a category of high-order semi-implicit time integration methods [52, 53, 7, 54], used widely in single-phase CFD solvers [55, 56, 51]. Bijl et al. [57] investigated the accuracy and efficiency of the BDF2 and some DIRK schemes for the unsteady compressible Navier–Stokes equations. The authors computed the error vs computation time and concluded that fourth- and fifth-order Runge–Kutta schemes are more cost-efficient than the BDF2 scheme. D'Alessandro et al. [58] used OpenFOAM and studied the effect of second- and third-order DIRK schemes in single-phase flows.

High-order spatial integration scheme

The second-order finite volume method has become mature and is widely used in various CFD solvers. With an increased requirement of specific applications involving, e.g., vortex

shedding, the second-order schemes may not be sufficient. Over the last decades, various high-order methods for unstructured grids have been developed. Some of the main ones are the k-exact scheme [59, 60], the WENO (weighted Essentially Non-Oscillatory [61]) method, and the Discontinuous Galerkin method [62].

The k-exact reconstruction describes quantity with high-order polynomial within 2D or 3D meshes [59, 60]. The WENO (weighted Essentially Non-Oscillatory) method extended the idea of ENO for the unstructured grid and resulted in better convergence and accuracy [61]. The main objective of k-exact and WENO methods is to obtain a high-order representation of the quantity inside the cell. However, in finite volume methods, only cell-averaged values can be accessed. The construction of high-order polynomials then relies only on the averages on the neighbouring cells around the central one, which is commonly named stencil. The number of construction stencils is generally bigger than DOFs (degrees of freedom), and the high-order polynomials are solved using the least square method. Generally, higher-order (higher than second-order) schemes require data from cells beyond direct neighbour cells. Therefore, having a large stencil yields severe consequences in computational efficiency and parallel performance. There are a few faster approaches using pre-calculation of the mesh dependent properties, which could increase the computation efficiency [63]. However, those approaches are not efficient anymore in the presence of mesh morphing, which is very common in naval applications.

Instead of a large stencil, the discontinuous Galerkin method (DGM) uses internal DOFs. It directly applies the high-order polynomial approximations inside the cell and computes the polynomial coefficients using the weak form of the governing equations. DGM requires higher-order face and volume integration and requires small time-step size in the time integration, which makes the computational cost very high.

As discussed above, the computational cost of high-order schemes is expected to be very expensive for two-phase flow naval application. Moreover, some of the faster approaches seem inefficient with mesh morphing. Besides, the convergence of high-order methods can be slower or sometime up to divergence due to the low robustness of high-order discretizations in complex situations. For example, in a flow with strong shock waves or impact, the low-order schemes are already validated often using numerical techniques to mitigate the non-physical phenomena. However, similar techniques for high-order spatial schemes are naturally less stable and difficult to construct. Therefore, this thesis focused on the implementation of high-order time integration without increasing the spatial discretization order.

1.3 Contribution of the present work

The present thesis aims to study the efficiency and the accuracy of different numerical schemes for naval applications, especially for unsteady wave-structure interaction problems. The following summarizes the specific contributions made in the present work. The thesis can be divided into two parts. The first part focused more on the investigations on the numerical schemes, and the second part focused more on the applications.

In the fist part, the effects of interface treatment schemes are compared using 2D benchmark

cases. The implementation and validation of the DIRK scheme in the averaged two-phase flow solver and in the SWENSE solver is performed.

- A study on the interface treatment schemes is performed. Four combinations of the solver are proposed, which use two VOF convection methods and two interface conditions. The name given to the representative solvers are *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*. These solvers are tested for wave propagation in a periodic domain and two two-dimensional slamming cases. The result showed that our default solver *interFoam* was not the most accurate solver, but it gives stable and predictable results. The *interFlow* solver simulates very accurate wave profiles, but its pressure is less stable than *interFoam*. The GFM method showed efficiency with a coarse mesh. However, the GFM method showed numerical defects with the current computation setup and requires further numerical investments.
- The DIRK method is implemented in the two-phase flow Navier-Stokes solver and the SWENSE solver to improve the solvers accuracy and efficiency for naval applications. The latter solvers are named hereafter *foamStar* and *foamStarSWENSE* solvers. The validity of the higher-order DIRK methods is checked using the Taylor Green Vortex case with the *foamStar* solver. The efficiency of the DIRK method is then studied on a two-phase regular wave propagation in periodic domain.
- The importance of the VOF convection algorithm is also discussed in this work. The difference in the VOF convection algorithm between OpenFOAM Crank-Nicolson (*OFCN*) and the DIRK-based Crank-Nicolson (*RKCN*) is presented. The small difference in the VOF convection algorithm affects the stability and accuracy of the simulation significantly. The results of *OFCN* and *RKCN* are compared using a periodic domain wave propagation.

After these development and tests on the numerical schemes, two typical applications for marine engineering are selected, wave propagation and seakeeping.

- The generation and propagation of non-linear regular and irregular waves in a numerical wave tank (NWT) are tested. The wave qualification analysis is performed with given computational parameters and qualification criteria. For three regular wave conditions, the wave propagations with different vertical cell size are compared. The difference between the standard k-ω SST model and the free-surface k-ω SST model is also studied. A deterministic analysis and a stochastic analysis is performed for the irregular waves from the experiment and the numerical solvers to measure the quality of the irregular waves. The deterministic analysis compares the time series of the wave elevation with different computational setups. With stochastic analysis, the quality of irregular waves is compared using the averaged wave spectrum and the wave crest distribution.
- With the regular wave NWT benchmark cases, the efficiency of each DIRK scheme is compared using a 10% steepness wave condition. For both *foamStar* and *foamStarSWENSE* solvers, three grid resolutions and four temporal resolutions are utilized. The computational cost and accuracy on the different test cases are compared. For irregular wave simulation, only the capability of the DIRK method with the *foamStar* solver was examined.

- The seakeeping analysis of the Wigley III hull and the KCS with a forward speed is then performed. The motion RAOs and resistance coefficients obtained from both *foamStar* and *foamStarSWENSE* solvers are compared with the experiment or other numerical results. The Wigley III hull and the KCS seakeeping analyses have been performed using DIRK methods, and the capability of each DIRK method has been confirmed. The efficiency and accuracy of the DIRK schemes with *foamStar* and *foamStarSWENSE* solvers for seakeeping have been compared using the Wigley III hull case. The efficiency of DIRK methods is compared with two grid resolutions and three temporal resolutions for a single wave and forward speed condition.

1.4 Thesis outline

Chapter 2 presents the mathematical formulation for both the averaged two-phase flow model and the SWENSE model. The governing equations of the two two-phase flow models are reviewed, and their boundary conditions are defined. This chapter also reviews the free-surface turbulence model specialized for the multiphase flow.

Chapter 3 introduces the regular and irregular wave theories used in this thesis. The fully non-linear stream function wave theory is briefly reviewed, and the three deepwater regular wave conditions used in this thesis are given. Then two non-linear irregular wave models based on the HOS method are presented. Finally, the Tian [64] wave breaking model for the realization of the extreme event is reviewed.

Chapter 4 presents the numerical discretization and corresponding algorithm for the governing equations and boundary conditions. The discretization schemes used for naval applications are given as OpenFOAM 'fvScheme' formats in an OpenFOAM notation. Moreover, an example of a 'fvSolution' file is given, which gives control for the non-linear iterations. At the end of this chapter, the entire flow chart of the *foamStar* and *foamStarSWENSE* solvers is presented.

Chapter 5 presents a study on the interface treatment schemes which influence the definition and the transportation of the free-surface. Four combinations of the solver are proposed, which use two VOF convection methods and two interface boundary conditions. Four solvers are tested with wave propagation in periodic domain and two two-dimensional slamming cases.

Chapter 6 presents the new implementation of the DIRK method in the averaged two-phase flow solver and the SWENSE solver. The diagonally implicit Runge-Kutta (DIRK) method is introduced, and the new formulation for VOF convection and PISO algorithm is presented. The DIRK method is validated using the Taylor-Green vortex and the wave propagation test cases.

Chapter 7 describes the qualification of non-linear regular and irregular waves in the numerical wave tank (NWT). Computation parameters and qualification criteria are suggested for regular and irregular wave generation and propagation. With the regular wave NWT benchmark cases, the efficiency of each DIRK scheme is compared using three regular wave conditions. For irregular waves, deterministic analysis and stochastic analysis are performed for the experiment results and the numerical solvers' results to measure the quality of the simulated irregular waves.

Chapter 8 investigates the application of the DIRK methods for the Wigley III and the KCS

simulations with forwarding speed. The efficiency of DIRK methods is compared using two mesh resolutions and three temporal resolutions for a single test case of the Wigley III test matrix. Finally, the capability of the DIRK methods are examined for the Wigley III and the KCS, and their results are compared with the experiment or other numerical results.

Chapter 9 summarizes the contribution of the present study and draws some perspectives.

2 Governing equations

In this chapter, a general description of the problem studied is defined. Then the governing equations are derived in the context of incompressible two-phase flow assumption. Figure 2.1 illustrates an example of a two-phase CFD simulation in a numerical wave tank (NWT). The waves are propagating along the free-surface, and a body is floating in a pure computational domain. The five boundaries, which are inlet, outlet, bottom, top, and body, are defined.

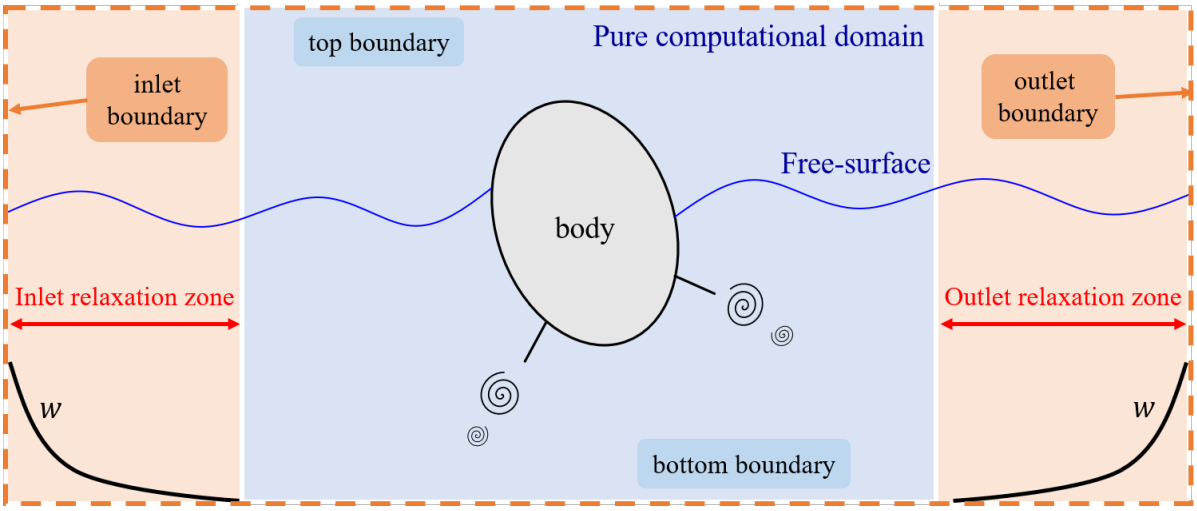


Figure 2.1. The description of the problem with the general boundaries and the relaxation zones in the computational domain.

The applications considered in the study all involve wave generation. This is done by using a zone spatially close to the outside boundaries of the numerical domain where a solution is progressively forced using a spatially varying weight function. These zones are called 'relaxation zones' and defined from the outside boundaries positions and using a prescribed length into the domain. In the implementation chosen in the present study, the solution is exactly the target solution at the boundary, and then the forcing decreases until at the end of the relaxation zone there is no forcing, the computational domain is said 'pure' and the solution is the CFD solution. To realize this, the weight (w) is one at the inlet and outlet boundary, and the weight is zero at the border of the relaxation zone and the pure computational domain. For most of the two-phase flow applications, the top boundary is commonly named open-air boundary, where the simulations define their atmosphere pressure. Depending on the relation between the water depth of the computational domain and the physical water depth, the bottom boundary condition must change.

In this study, the two-phase Navier-Stokes equations simulation is solved in two different ways. One uses a conditionally averaged two-phase flow model, and another uses a SWENSE method. The governing equations and the boundary conditions of each method are described in the following sections.

2.1 Governing equations for the averaged two-phase incompressible flow model

A typical flow model for incompressible two-phase Newtonian flow can be derived into the following form of the mixture (conditionally averaged) momentum equation (Eqn. (2.1)) and the continuity equation (Eqn. (2.2)) [65]:

$$\frac{\partial(\rho\mathbf{u})}{\partial t} + \nabla\bullet(\rho\mathbf{u}\mathbf{u}^T) - \nabla\bullet(\mu(\nabla\mathbf{u} + \nabla\mathbf{u}^T)) = -\nabla p_d - (\mathbf{g}\bullet\mathbf{x})\nabla\rho + \sigma_\kappa\nabla\alpha, \tag{2.1}$$

$$\nabla\bullet\mathbf{u} = 0, \tag{2.2}$$

where $\mathbf{u}$ is the continuous fluid velocity field; ρ is the averaged density field; $\mathbf{g}$ is the gravitational field vector ($(0, 0, -9.81)$ m/s^2); $\mathbf{x}$ is the position vector from reference to cell center; p_d is the dynamic pressure. The dynamic pressure can be derived from the gravitational force and the total pressure (p).

$$p_d = p - \rho\mathbf{g}\bullet\mathbf{x} \tag{2.3}$$

The surface tension force due to the existence of the interface between water and air is considered negligible ($\sigma_\kappa = 0$) in this thesis. The density of water and air is commonly defined as $\rho_{air} = 1.0$ kg/m^3 and $\rho_{water} = 1000$ kg/m^3. The averaged effective dynamic viscosity μ is calculated as a summation of linear viscosity (μ_l) and eddy viscosity (μ_t). The linear dynamic viscosity of water is $\mu_w = 10^{-3}$ Ns/m^2, and the linear dynamic viscosity of air is $\mu_a = 10^{-5}$ Ns/m^2. The eddy viscosity is calculated from the turbulence model. The details on the turbulence models used in this thesis are described in Section 2.3. Note that the density and the dynamic viscosity could change with the studied case.

In this thesis, VOF interface capturing technique with an interface compression method is applied due to its mass conservation property. The VOF method uses the phase indicator function (α), which varies between 0 and 1. For a two-phase water-air flow, $\alpha = 1$ if a cell is fully filled with water, and $\alpha = 0$ if a cell is full of air. Therefore, the averaged density and the averaged effective dynamic viscosity are defined as in Eqn. (2.4) and Eqn. (2.5).

$$\rho = \alpha\rho_{water} + (1 - \alpha)\rho_{air} \tag{2.4}$$

$$\mu = \alpha\mu_{water} + (1 - \alpha)\mu_{air} + \mu_t \tag{2.5}$$

The governing equation of the phase indicator function is derived from mass conservation. Where the mass in a control volume (Ω) is conserved,

$$m = \int_\Omega \rho dV, \qquad \frac{d}{dt}\int_\Omega \rho dV = 0, \tag{2.6}$$

and the application of Reynolds transport theorem to Eqn. (2.6) gives:

$$\frac{d}{dt}\int_{\Omega}\rho dV = \int_{\Omega}\left(\frac{\partial\rho}{\partial t} + \nabla\bullet(\rho\mathbf{u})\right)dV = 0. \tag{2.7}$$

The inner term of the volume integration can be transformed into Eqn. (2.8) using the vector identity.

$$\frac{\partial\rho}{\partial t} + \nabla\bullet(\rho\mathbf{u}) = \frac{\partial\rho}{\partial t} + \mathbf{u}\bullet\nabla\rho + \rho\nabla\bullet\mathbf{u} = 0 \tag{2.8}$$

The substitution of Eqn. (2.4) into Eqn. (2.8) yields the transportation equation of the phase fraction function under the incompressible assumption.

$$\frac{\partial\alpha}{\partial t} + \mathbf{u}\bullet\nabla\alpha = 0 \tag{2.9}$$

In *interFoam*, the VOF transportation equation is controlled using an artificial interface compression technique. The following equation presents the modified transportation equation:

$$\frac{\partial\alpha}{\partial t} + \nabla\bullet(\mathbf{u}\alpha) + \nabla\bullet[\mathbf{u}_r\alpha(1-\alpha)] = 0, \tag{2.10}$$

where $\mathbf{u}_r = C_\alpha(\mathbf{u}_{water} - \mathbf{u}_{air})$ is the relative velocity between air and water at the interface. The third term on the left-hand side is a compression term that artificially compresses the phase function field to minimize smearing [66, 67]. The compression term is only effective near the interface due to the term $\alpha(1-\alpha)$. The phase function must keep bounded between $[0, 1]$ to prevent non-physical behaviour. As this phase function field has strong gradients around the interface , careful treatments are required in the numerical discretization of Eqn. (2.10) to keep this field bounded and accurate [46].

2.1.1 Boundary conditions for two-phase incompressible flow model

The governing equations of the averaged two-phase incompressible flow require proper boundary conditions to obtain solutions. In general, there are three mathematical types of boundary conditions: the Dirichlet type, the Neumann type, and the mixing between the two. In a Dirichlet condition, specific values are imposed on the boundary to scalar or vector fields whereas in a Neumann condition the gradient of the field normal to the boundary is imposed. The mixed type boundary condition is given as a function of both the field values and their derivatives at the boundary. Solving Navier-stokes equations (Eqns (2.1)-(2.2)) requires boundary conditions on both velocity and dynamic pressure. Usually Dirichlet boundary condition is applied to the velocity, and Neumann boundary condition is applied to the dynamic pressure. Within the two-phase incompressible model chosen, some boundary conditions have to be considered also for the phase function field. For the typical boundaries defined in Figure 2.1, the boundary conditions of velocity, dynamic pressure and phase function are presented below.

Velocity

A no-slip boundary condition is imposed on the body surface, which sets the fluid velocity

equal to the surface velocity. With a boundary velocity $\mathbf{u}_b$, the no-slip boundary condition thus reads:

$$\mathbf{u} = \mathbf{u}_b. \tag{2.11}$$

In special occasions, free-slip boundary conditions can be applied to the body. For instance, when simulating a NWT in large or infinite depth, the bottom condition can be approximated with a free-slip boundary condition applied at a finite depth, minimizing the effect of the numerical truncation of an infinite domain. Such a free-slip condition reads:

$$\mathbf{u}\bullet\mathbf{n} = \mathbf{u}_b\bullet\mathbf{n}, \tag{2.12}$$

where $\mathbf{n}$ is the surface normal vector. For inlet or outlet boundaries, thanks to the relaxation zones and the relaxation scheme, the boundary values are explicitly known for the velocity and set to the external forcing value. The top boundary in the computational domain presents an open-air boundary condition where the total pressure is constant.

Dynamic pressure

For all boundaries where the Dirichlet velocity B.C. is applied (Eqn. (2.11)), the boundary condition for the dynamic pressure can be derived from the dot product of the momentum equation Eqn. (2.1) with the surface unit normal vector ($\mathbf{n}$):

$$\begin{aligned} \mathbf{n}\bullet\nabla p_d = \frac{\partial p_d}{\partial n} &= -\frac{D(\rho\mathbf{u}_b)}{Dt}\bullet\mathbf{n} + \mathbf{n}\bullet\nabla\bullet(\mu(\nabla\mathbf{u}_b + \nabla\mathbf{u}_b^T)) - (\mathbf{g}\bullet\mathbf{x})\nabla\rho\bullet\mathbf{n} \\ \frac{\partial p_d}{\partial n} &= -\frac{D(\rho\mathbf{u}_b)}{Dt}\bullet\mathbf{n} + \mathbf{n}\bullet\nabla\bullet(\mu(\nabla\mathbf{u}_b + \nabla\mathbf{u}_b^T)) - (\mathbf{g}\bullet\mathbf{x})\frac{\partial\rho}{\partial n} \quad \text{for} \quad \mathbf{x}\in\partial\Omega_B. \end{aligned} \tag{2.13}$$

The first term on the right-hand side is the material derivative of the momentum at a boundary point. The second term is the surface normal gradient of the shear stress, which can be considered very small. The third term has the component $\nabla\rho\bullet\mathbf{n}$, which is the surface normal gradient of density. For general incompressible two-phase flow simulation, the surface normal gradient of density is non-zero only near the free-surface.

Phase fraction

The fluid velocity passively transports the phase-function. Therefore, mathematically no specific boundary conditions are required for the body and wall boundaries. For the wave boundary conditions, at inlet and outlet, the boundary values are explicitly applied using the known wave elevation.

2.2 SWENSE model

2.2.1 Functional decomposition

A specific version of the Navier-Stokes equations named Spectral Wave Explicit Navier-Stokes Equations (SWENSE) formulation was introduced by Ferrant et al. [22]. The SWENSE formulation assumes that total flow physical quantities can be decomposed into incident and complementary parts as in Eqn. (2.14), where χ, χ_I, and χ_C are the total flow quantity, the incident flow quantity, and the complementary field, respectively.

$$\chi = \chi_I + \chi_C \tag{2.14}$$

In the SWENSE formulation, it is assumed that the incident wave flow is already available from other non-linear regular or irregular wave models. In this thesis, the fluid velocity and the pressure are decomposed as:

$$\mathbf{u} = \mathbf{u}_I + \mathbf{u}_C, \tag{2.15}$$

$$p = p_I + p_C. \tag{2.16}$$

The incident flow components are extracted from the original Navier-Stokes equations by subtracting the Euler equations from the Navier-Stokes equations in an incompressible context. Since this functional decomposition must not modify the total flow solution of the simulation, the reconstructed total flow field must still be the solution of the Navier-Stokes equations. Therefore, by using the SWENSE method, the following numerical advantages are expected:

- when the flow is incident flow dominant, SWENSE solver is expected to get advantage on the convergence speed,
- the computational mesh needs to be refined in the vicinity of the body only, which is expected to save some computational cost for wave propagation.

Figure 2.2 illustrates the SWENSE decomposition of the functional quantities of the total flow into the incident and the complementary parts. The first row in the figure represents the total flow field considering wave-structure interaction. The second row represents the single phase free-surface incident solution of the incompressible Euler equation. The third row represents the complementary wave field and the SWENS equations which describe the influence of the structure on the wave field.

In a two-phase flow context, Vukčević applied the SWENSE method to decompose the fluid velocity using the LS function [25]. He also applied a pressure jump condition with the GFM. Other detailed formulations of the SWENSE method in various form are given in Li [68]. Li performed functional decomposition on both velocity and pressure in a two-phase flow solver with non-decomposed VOF phase-function [1]. Li tested the conservative and non-conservative form of the SWENS equation with and without the ghost fluid method. Choi advanced Li's approach and used the non-conservative SWENS equations with GFM and LS phase indicator function [2].

In the next section, the conservative form of the SWENSE formulation from Li's PhD thesis is revisited as it is the formulation used in the present study.

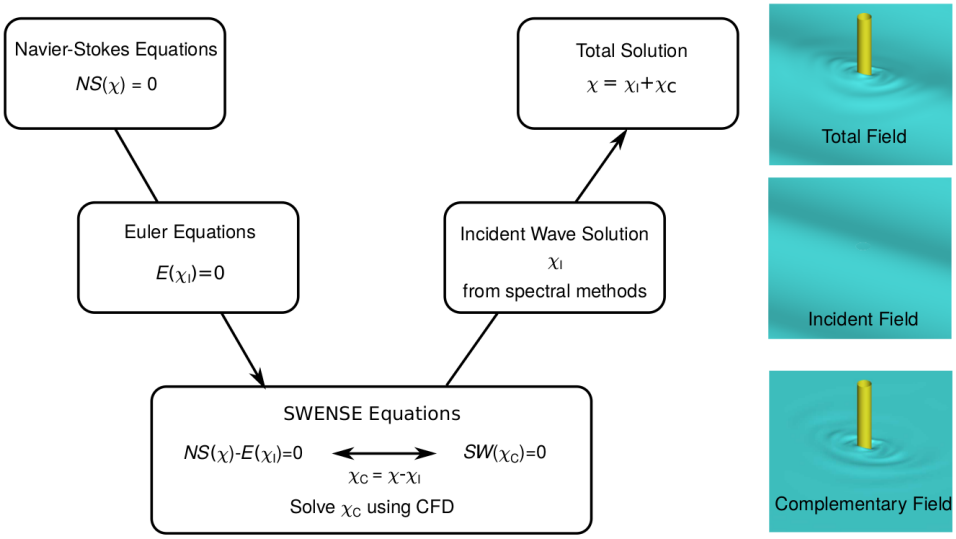


Figure 2.2. SWENSE decomposition of the functional quantities of the total flow into incident and complementary parts (from [1]).

2.2.2 Conservative SWENSE formulation

The decomposition of the continuity equation is written as:

$$\nabla\bullet\mathbf{u}_C = -\nabla\bullet\mathbf{u}_I = 0. \tag{2.17}$$

Previous studies on SWENSE remarked that the divergence of the reconstructed incident wave velocity field is not zero [1, 25], and it could affect the boundedness of the phase-function field. However, the present formulation kept $\nabla\bullet\mathbf{u}_C = 0$ since the existence of $-\nabla\bullet\mathbf{u}_I$ would yields an additional source term in the pressure equation.

A single-phase Euler equation can be written as:

$$\frac{\partial\mathbf{u}_I}{\partial t} + \nabla\bullet(\mathbf{u}_I\mathbf{u}_I) = -\frac{1}{\rho_I}\nabla p_I + \mathbf{g}. \tag{2.18}$$

The single-phase Euler equation is valid in the water domain. The solution for incident wave's fluid velocity and pressure can be anyway extrapolated in the air. However, the hyperbolic functions of the solution have huge values above the free surface, and this can lead to numerical instabilities. To overcome this, Li scaled-down the extrapolated incident pressure over the free-surface by the density ratio [1]:

$$p_I^* = \frac{\rho}{\rho_w} p_I \tag{2.19}$$

where $\rho_w = \rho_I$ is the constant water density and ρ is the averaged density of Eqn. (2.4). Substituting Eqn. (2.18) into Eqn. (2.19) one can write:

$$\rho\left(\frac{\partial \mathbf{u}_I}{\partial t}+\nabla\bullet(\mathbf{u}_I\mathbf{u}_I)\right)=-\nabla p_I^*+\frac{p_I}{\rho_w}\nabla\rho+\rho\mathbf{g}. \tag{2.20}$$

The second term of the right-hand side is zero for the fully submerged cells and the dry cells, but it is non-zero for the interface cells. The left-hand side of the Euler equation can be expanded into:

$$\rho\left(\frac{\partial \mathbf{u}_I}{\partial t}+\mathbf{u}_I\bullet\nabla\mathbf{u}_I\right)=\frac{\partial \rho\mathbf{u}_I}{\partial t}-\mathbf{u}_I\frac{\partial \rho}{\partial t}+\nabla\bullet(\rho\mathbf{u}_I\mathbf{u}_I)-\mathbf{u}_I\nabla\bullet(\rho\mathbf{u}_I)=-\nabla p_I^*+\frac{p_I}{\rho_w}\nabla\rho+\rho\mathbf{g} \tag{2.21}$$

Here, the conservative momentum Navier-Stokes equation for two-phase flow is recalled, where $p=p_d+\rho gh$ is the total pressure.

$$\frac{\partial \rho\mathbf{u}}{\partial t}+\nabla\bullet(\rho\mathbf{u}\mathbf{u})=-\nabla p+\rho\mathbf{g}+\nabla\bullet(\mu(\nabla\mathbf{u}+\nabla\mathbf{u}^T)) \tag{2.22}$$

The subtraction of the modified Euler equation from Eqn. (2.22) gives, after mathematical operations and simplifications given in [1]:

$$\frac{\partial \rho\mathbf{u}_C}{\partial t}+\nabla\bullet(\rho\mathbf{u}\mathbf{u}_C)+\rho\mathbf{u}_C\bullet\nabla\mathbf{u}_I=-\nabla p_C-\frac{p_I}{\rho_w}\nabla\rho+\nabla\bullet(\mu(\nabla\mathbf{u}_C+\nabla\mathbf{u}_C^T)) \tag{2.23}$$

The VOF convection scheme applied to the SWENSE formulation is identical to the two-phase flow model, and no functional decomposition is applied to the phase-function field.

$$\frac{\partial(\alpha)}{\partial t}+\nabla\bullet(\mathbf{u}\alpha)+\nabla\bullet\Big[\mathbf{u}_r\alpha(1-\alpha)\Big]=0 \tag{2.24}$$

As mentioned with the continuity equation, due to the interpolation error of incident velocity, the total velocity ($\mathbf{u}=\mathbf{u}_I+\mathbf{u}_C$) might not satisfy the continuity equation. This small error could bring the unboundedness of VOF in the computational domain. The discussion and more detailed formulation of SWENSE are given in the PhD thesis of Li [1].

2.2.3 Boundary conditions for SWENSE model

Similar to Section 2.1.1, boundary conditions for the phase-function (α), the complementary velocity (U_C), and the complementary pressure (p_C) are given below.

phase-function

Since no functional decomposition is applied to the phase-function field, the VOF boundary condition is identical to the two-phase flow model. For wave boundary conditions, at inlet and outlet, the boundary values are explicitly applied using the known wave elevation. For other boundaries when the boundary values are unknown, zero Neumann boundary condition is applied.

Complementary velocity

No-slip boundary condition or free-slip boundary condition is applied to the bodies and walls with the functional decomposition. Therefore, with a boundary velocity $\mathbf{u}_b$, the no-slip boundary condition reads:

$$\mathbf{u}_C = \mathbf{u}_b - \mathbf{u}_I. \tag{2.25}$$

The phase-function boundary condition reads:

$$\mathbf{u}_C \bullet \mathbf{n} = \mathbf{u}_b \bullet \mathbf{n} - \mathbf{u}_I \bullet \mathbf{n}. \tag{2.26}$$

For other boundaries, such as inlet or outlet, the boundary values are explicitly applied when the values are known.

Complementary pressure

The complementary pressure boundary condition can be derived from the pressure boundary condition and pressure functional decomposition. Eqn. (2.27) recalls the pressure boundary condition at the arbitrary boundary with the boundary velocity ($\mathbf{u} = \mathbf{u}_b$).

$$\frac{\partial p}{\partial n} = -\frac{D(\rho \mathbf{u}_b)}{Dt} \bullet \mathbf{n} + \mathbf{n} \bullet \nabla \bullet (\mu(\nabla \mathbf{u}_b + \nabla \mathbf{u}_b^T)) + \rho \mathbf{g} \bullet \mathbf{n} \quad \text{for} \quad \mathbf{x} \in \partial\Omega_B. \tag{2.27}$$

Application of pressure functional decomposition ($p = p_I^* + p_C$) yields:

$$\frac{\partial p_C}{\partial n} = -\frac{D(\rho \mathbf{u}_b)}{Dt} \bullet \mathbf{n} + \mathbf{n} \bullet \nabla \bullet (\mu(\nabla \mathbf{u}_b + \nabla \mathbf{u}_b^T)) + \rho \mathbf{g} \bullet \mathbf{n} - \frac{\partial p_I^*}{\partial n} \quad \text{for} \quad \mathbf{x} \in \partial\Omega_B. \tag{2.28}$$

The first term on the right-hand side is the material derivative of the momentum at the boundary point. The second term is the surface normal gradient of the shear stress, which can be considered very small. The fourth term is the surface normal gradient of incident pressure.

2.2.4 Extrapolation of incident flow quantities to air zone

The SWENSE method and its multi-flow formulation requires extrapolated incident flow fields in the air zone. However, these are not available in the potential flow model. The most straightforward method for the extrapolation is the Wheeler stretching, which uses the hyperbolic vertical function of the wave theory. However, the hyperbolic characteristics of incident waves in the vertical direction can generate large incident velocity and incident pressure over the free-surface. Then, it could cause undesirable numerical instability in the air zone. Li set constant incident flow field values with a limited height above the free-surface to prevent numerical errors due to large values [1]. However, induced discontinuous gradient of the extrapolated incident field at this limited height could generate the continuity error and the spurious air velocity.

Recently, Choi modified the hyperbolic extrapolation to a cubic extrapolation, preventing such a discontinuous extrapolated incident flow gradient [2]. This subsection shortly describes the boundary conditions and the solution of this cubic polynomial extrapolation.

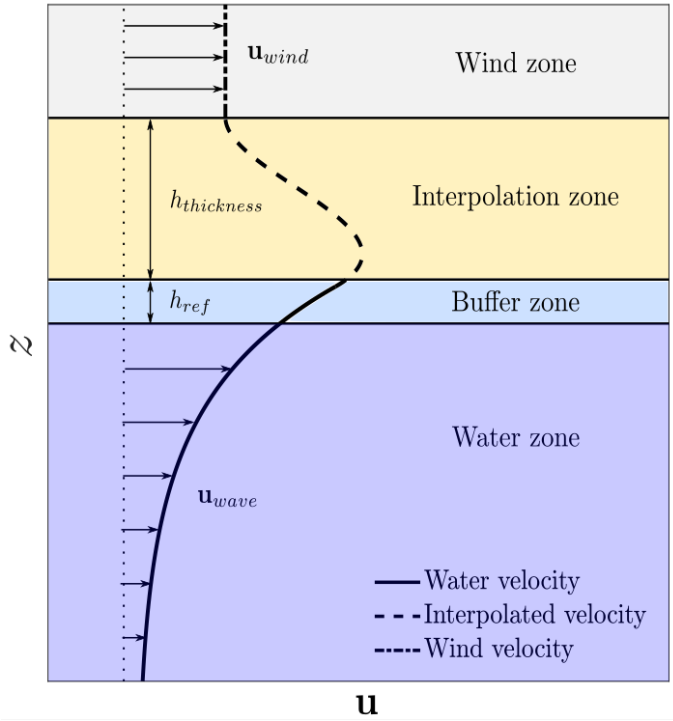


Figure 2.3. Incident flow extrapolation in the air zone with cubic polynomial.

Figure 2.3 illustrates an example of extrapolation of the incident velocity field. Two height parameters are introduced: The buffer zone height (h_{ref}) defines the height from which the extrapolation start. The interpolation zone height is $h_{thickness}$. The boundary conditions of quantity (χ) are given at $z = \eta_I + h_{ref}$ and $z = \eta_I + h_{ref} + h_{thickness}$.

$$\begin{aligned} \chi(\eta_I + h_{ref}) &= \chi_0, & \frac{d\chi(\eta_I + h_{ref})}{dz} &= \chi_0', \\ \chi(\eta_I + h_{ref} + h_{thickness}) &= \chi_{wind} & \frac{d\chi(\eta_I + h_{ref} + h_{thickness})}{dz} &= 0. \end{aligned} \tag{2.29}$$

The upper end of the computational domain is assumed as a wind zone, and the wind flow quantity (χ_{wind}) must be specified. For instance, the incident wind velocity ($\mathbf{u}_{I,wind}$) and the incident wind pressure ($p_{I,wind}$) must be defined. The gradient between the interpolation zone and the wind zone is assumed to be zero. The incident wave quantity χ_0 and its vertical gradient χ_0' between the buffer zone and the interpolation zone are available from the wave model.

To get the cubic polynomial systematically, Choi introduced a normalized coordinate $\tilde{\zeta}$ defined in the range of [0,1] [2].

$$\tilde{\zeta} = \frac{z - (\eta + h_{ref})}{h_{thickness}} \tag{2.30}$$

Then, the following equation introduces the normalized cubic polynomial with coefficients a, b, c, and d.

$$g(\tilde{\zeta}) = a\tilde{\zeta}^3 + b\tilde{\zeta}^2 + c\tilde{\zeta} + d \tag{2.31}$$

In the normalized coordinate, the boundary conditions (Eqns. (2.29)) are transformed as:

$$\begin{aligned} g(0) &= \chi_0, \qquad \frac{dg(0)}{d\tilde{\zeta}} = h_{thickness}\chi_0', \\ g(1) &= \chi_{wind}, \qquad \frac{dg(1)}{d\tilde{\zeta}} = 0. \end{aligned} \tag{2.32}$$

With these boundary conditions, the cubic polynomial coefficients are determined as:

$$\begin{aligned} a &= \left\{2(\chi_0 - \chi_{wind}) + (h_{thickness}\chi_0')\right\}, \\ b &= -\left\{3\left(\chi_0 - \chi_{wind}\right) + 2\left(h_{thickness}\chi_0'\right)\right\}, \\ c &= h_{thickness}\chi_0', \\ d &= \chi_0. \end{aligned} \tag{2.33}$$

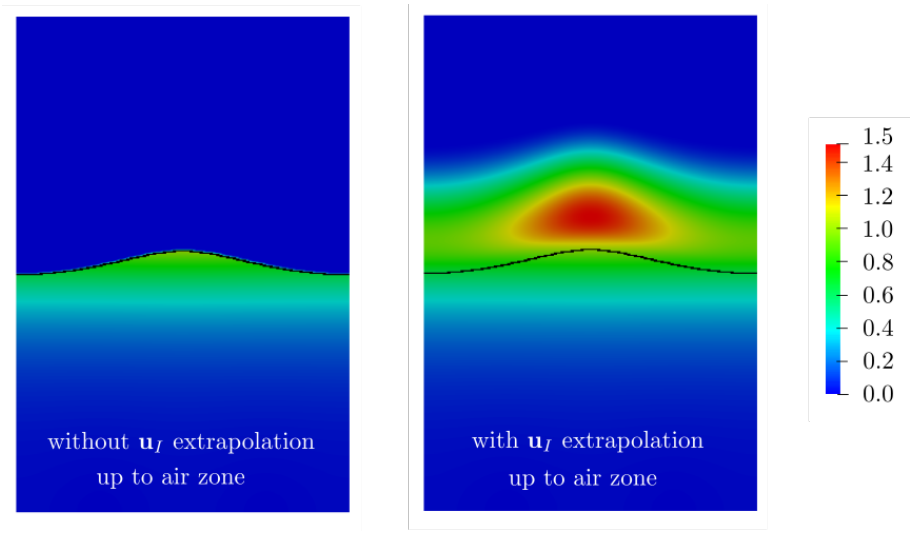


Figure 2.4. Extrapolation of incident velocity up to wind zone using cubic polynomial. The color label represents the normalized velocity magnitude (Figure from Choi [2]).

The appropriate buffer zone height (h_{ref}) and interpolation zone height ($h_{thickness}$) might be different case by case. Unless mentioned in this thesis, the default buffer zone height is half the wave height (H) and, the default interpolation zone height is twice wave height. For irregular wave simulation, the significant wave height (H_s) is used instead of the wave height.

$$\begin{aligned} h_{ref} &= 0.5H \quad \text{or} \quad h_{ref} = 0.5H_s \\ h_{thickness} &= 2H \quad \text{or} \quad h_{thickness} = 2H_s \end{aligned} \tag{2.34}$$

Figure 2.4 illustrates an example of the cubic extrapolation of the incident wave velocity with the default height parameters.

2.3 Turbulence models in OpenFOAM

Most flows in nature are turbulent. A turbulent flow can be described as an unstable state of velocity in the flow. In a turbulent flow, each of the relevant quantities can be decomposed into a 'mean' part and a 'fluctuating' part, which both vary in time and space in an unsteady computation.

There are several methods for the simulation of turbulent flow. With Direct Numerical Simulation (DNS) solvers, the flow is not decomposed and the governing equations cover the whole range of turbulence scales, which requires very high mesh resolution and temporal resolution [69]. The Large Eddy Simulation (LES) is a numerical model that controls the smallest turbulence length scale via low-pass filtering of the Navier-Stokes equation, reducing the computational cost with respect to DNS [70]. Another method, computationally cheaper, uses a statistical approach for the simulation of turbulence flow. One of the statistical approaches is named Reynolds averaging technique, leading to the Reynolds-Averaged Navier-Stokes Equations (RANSE). For FVM RANSE simulation in the naval engineering field, k-ω SST models are widely used. However, with those models, non-physical high viscosity is generated under the free-surface in two-phase flow simulations [71]. High viscosity under the free-surface dramatically damps the waves within a few wavelengths, and it gives significant limitations on the size of the computational domain.

To avoid the numerical damping of waves due to the turbulence model, Larsen and Fuhrman introduced the buoyancy production term for two-phase flow applications [71]. This study used the 'free-surface k-ω SST model' which applies the idea of buoyancy production to the OpenFOAM k-ω SST model to consider the variation of density of the two-phase flow in the vicinty of the free-surface. The standard OpenFOAM k-ω SST model is based on the model by F. Menter [72, 73]. The free-surface k-ω SST model is based, as any k-ω model, on two transportation equations: one is the transportation equation for the turbulence kinetic energy density:

$$\frac{\partial(\rho k)}{\partial t} + \nabla\bullet(\rho\mathbf{u}k) - \nabla\bullet((\mu + \rho\alpha_k\nu_T)\nabla k) = \rho p_0 - \frac{2}{3}\rho k(\nabla\bullet\mathbf{u}) - \rho\beta^* k\omega - \rho P_b, \tag{2.35}$$

another is the transportation equation for specific dissipation (ω):

$$\frac{\partial(\rho\omega)}{\partial t} + \nabla\bullet(\rho\mathbf{u}\omega) - \nabla\bullet((\mu + \rho\alpha_\omega\nu_T)\nabla\omega) = \frac{\rho\gamma p_0}{\nu} - \frac{2}{3}\rho\gamma\omega(\nabla\bullet\mathbf{u}) - \rho\beta\omega^2 - \rho(F_1 - 1)CD_{k\omega}. \tag{2.36}$$

In those equations, μ is the averaged linear dynamic viscosity. ρ is the averaged density, P_b is the buoyancy production derived as in Eqn. (2.37), and p_0 is defined in Eqn. (2.38) where 'symm' and 'skew' are the OpenFOAM tensor operator notations. The other coefficients used for the k-ω SST model are listed in Table 2.1.

$$P_b = -\frac{g_i}{\rho}\overline{\rho' u_i'} = p_b\nu_T, \quad p_b = \alpha_b^* N^2, \quad N^2 = \mathbf{g}\times\nabla\rho \tag{2.37}$$

$$p_0 = 2\mathbf{S}_{ij}\mathbf{S}_{ij} \tag{2.38}$$

$$\mathbf{S}_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) = \text{symm}\left(\nabla \mathbf{u}\right), \quad \mathbf{W}_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} - \frac{\partial u_j}{\partial x_i}\right) = \text{skew}\left(\nabla \mathbf{u}\right) \tag{2.39}$$

Table 2.1. Coefficients of the k-ω SST model.

Item	Value	Item	Value	Item	Value
α_{k1}	0.85	α_{w1}	0.5	γ_1	5/9
α_{k2}	1.0	α_{w2}	0.856	γ_2	0.44
β_1	0.075	a_1	0.31	b_1	1.0
β_2	0.0828	β^*	0.09	α_b^*	1.36

The coefficients α_k, α_k, γ, and β are the blended coefficient which are interpolated using the interpolation parameter $F1$. An example of the interpolation for α_k is given in Eqn. (2.40). The interpolation parameter $F1$ is given as in Eqn. (2.41), where y is the wall distance function.

$$\alpha_k = F_1\alpha_{k1} + (1 - F_1)\alpha_{k2}\,, \tag{2.40}$$

$$\begin{aligned} F_1 &= \tanh\left[\left(\min\left(\max\left(\frac{\sqrt{k}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right), \frac{4\alpha_{\omega 2}k}{CD_{k\omega}y^2}\right)\right)^4\right], \\ CD_{k\omega} &= \max\left(2\rho\alpha_{\omega 2}\frac{1}{\omega}\nabla k\nabla\omega, 10^{-10}\right). \end{aligned} \tag{2.41}$$

After k and ω are solved, Eqn. (2.42) and Eqn. (2.43) are used to evaluate the eddy viscosity.

$$\nu_T = \frac{a_1 k}{\max(a_1\omega, b_1 F_2 p_0)} \tag{2.42}$$

$$F_2 = \tanh\left[\left(\max\left(\frac{2\sqrt{k}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right)\right)^2\right] \tag{2.43}$$

3 Wave theories

3.1 Regular waves

Wave-structure interaction (WSI) is one of the major concerns for naval architecture and ocean engineering. Waves cause one of the most important external forces. Therefore, the quality of the generated waves is very important for accurate wave-structure interaction simulations. The waves in the ocean are random and unsteady, and these are commonly named irregular waves. Ocean waves that matter more in wave-structure interaction typically have wave periods in the range of 6s $\sim$ 25s. Generally the simulations with irregular waves are very expensive. Therefore among various alternatives of 'design waves', regular (harmonic) waves are often used. Studying WSI through regular waves uses similar approach as what is used with linear potential theory, such as computing Response Amplitude Operators (RAOs). However with CFD, as it is in experiment, the generated wave system is non-linear as it has to be a solution of the Navier-Stokes equations. Besides, as the required return period to be considered in regulations has increased over time, accounting for the non-linearity of the waves has also become more and more important in naval architecture and ocean engineering, making CFD a suitable method for modelling steep waves. The following subsections present the theoretical background of regular wave theories and specify the regular wave conditions used in this thesis.

3.1.1 Non-linear regular wave theories

The simplest regular wave theory is the Airy wave theory, also named linear wave theory which satisfies the linearized free-surface boundary conditions on the mean free-surface. Therefore, the Airy wave is accurate for small wave height only.

Additionally to the linear wave theory, the `foamStar` library supports two non-linear regular wave options: the **fifth-order Stokes wave** and the **stream function wave**. The Stokes wave theory is derived based on the perturbation series approach. The linear wave theory is expanded using Stokes' expansion (perturbation series) with wave steepness to apply non-linear free-surface boundary condition [74]. This theory is thus non-linear up to a given truncation order. The velocity, the elevation and the corresponding dispersion relation are given by J. D. Fenton [75].

The stream function theory proposed by Rienecker & Fenton [76] uses the stream function and the Fourier series expansion to solve the propagation of 2D fully non-linear regular waves. The solution given by the stream function theory is accurate for a wide range of depths, amplitudes, and wavelengths, up to the Stokes breaking limit. The stream function theory algorithm is briefly described here, but more interested readers can find information in the recent open-source stream function code [77].

In the 2D flow of a regular wave propagating at infinity, in the reference frame attached to the wave flow is steady and periodic in space. The stream function, $\psi_I(x, z)$, satisfies the Laplace equation in the fluid domain and the dynamic boundary condition at the free-surface:

$$\nabla^2 \psi_I(x,z) = 0, \tag{3.1}$$

$$g\eta_I + \frac{1}{2}\left[\left(\frac{\partial \psi_I}{\partial x}\right)^2 + \left(\frac{\partial \psi_I}{\partial z}\right)^2\right] = R, \quad \text{on} \quad z = \eta_I\,, \tag{3.2}$$

where R is the Bernoulli constant, η_I is the free-surface elevation. A free-slip boundary condition is applied at the bottom boundary and at the moving free-surface. That makes two streamlines (constant value of stream function) at the bottom and the free-surface. The stream function at the free-surface (Q) is the variation of stream function between two streamlines. Therefore, the bottom and kinematic free-surface boundary can be written as in Eqn. (3.3).

$$\psi_I(x, z = -h) = 0, \quad \psi_I(x, z = \eta_I(x)) = -Q \tag{3.3}$$

The wave elevation and the stream function are given by Eqn. (3.4) and Eqn. (3.5) in the form of the Fourier series, respectively.

$$\eta_I(x) = \frac{a_0}{2} + \sum_{n=1}^{N_2} a_n \cos\left(k_n x\right) \tag{3.4}$$

$$\psi_I(x,z) = b_0 z + \sum_{n=1}^{N_1} b_n \frac{\sinh\left(k_n(z+h)\right)}{\cosh\left(k_n h\right)} \tag{3.5}$$

The subscript $(...)_I$ denotes the incident wave; a_n and b_n are the modal amplitudes of the free-surface elevation and the stream function, respectively; k_n is the wave number of the n^{th} Fourier series component. The integers N_1 and N_2 determine the number of truncated Fourier series modes used to evaluate the stream function and the elevation. The optimal numbers of Fourier series modes are automatically selected using the algorithm suggested by Ducorzet et al. [77]. The horizontal and vertical fluid velocities are calculated from the stream function as in Eqn. (3.6). The incident wave pressure (p_I) derived from the Bernoulli equation is given in Eqn. (3.7).

$$\begin{aligned} u_I^x(x,z) &= b_0 z + \sum_{j=1}^{N_B} k_j b_j \frac{\cosh(k_j(z+h))}{\cosh(k_j h)} \cos(k_j x), \\ u_I^z(x,z) &= \sum_{j=1}^{N_B} k_j b_j \frac{\sinh(k_j(z+h))}{\cosh(k_j h)} \sin(k_j x), \end{aligned} \tag{3.6}$$

$$\frac{p_I(z)}{\rho} = R - gz - \frac{1}{2}\left[(u_I^x)^2 + (u_I^z)^2\right]. \tag{3.7}$$

The stream function theory covers regular wave height up to wave breaking limit and water depth from finite depth to deepwater condition. Due to its flexibility and accuracy, it is adopted to generate regular wave in the present study.

3.1.2 Regular wave conditions

This subsection describes the representative regular wave conditions used in this thesis, which are summarized in Table 3.1. These regular wave conditions have different wave steepnesses (H/λ), which are 10%, 5%, and 1%. Only deepwater wave conditions are considered since targeted simulations are naval applications. The three wave conditions have the same 1m wavelength (λ) and have different wave heights (H) depending on their wave steepness. The water depth (h) is also 1m; therefore, all wave conditions can be assumed deepwater conditions. The stream function wave theory determines the wave period (T) corresponding to given wavelength, wave height, and mean current (U_c).

Table 3.1. Three regular wave conditions which have 10%, 5%, and 1% steepnesses based on the stream function wave theory

Items		**Unit**	$H = 0.1$m	$H = 0.05$m	$H = 0.01$m
Wave steepness (H/λ)		-	10%	5%	1%
Wave height (H)		[m]	0.1	0.05	0.01
Wave length (λ)		[m]	1	1	1
Depth (h)		[m]	1	1	1
Wave period (T)		[s]	0.76179	0.79049	0.79991
Stream function wave theory coefficients	a_0/b_0	[m]/[m/s]	0/-1.3137	0/-1.2650	0/-1.2501
	a_1/b_1	[m]/[m^2/s]	0.0474/0.0582	0.0248/0.0308	0.0050/0.0062
	a_2/b_2	[m]/[m^2/s]	0.0083/0.0010	0.0020/6.1e-05	7.9e-05/ -
	a_3/b_3	[m]/[m^2/s]	0.0023/8.6e-05	0.0002/1.8e-06	2.0e-06/ -
	a_4/b_4	[m]/[m^2/s]	0.0007/9.2e-06	3.5e-05/ -	-

Figure 3.1 shows Le Méhauté's diagram that presents the limits of the different wave theories' applicability given wave steepness and water depth [3]. In the figure, $Ur = \frac{H\lambda^2}{h^3}$ is the Ursell number, which is important when small water depth is considered. According to Le Méhauté's diagram, the smallest wave condition considered as test case in this study ($H/\lambda = 1$ %) has a non-negligible second-order wave component, and should be solved at least with Stokes 2$^{\text{nd}}$ order.

3.2 Non-linear irregular waves

3.2.1 Ocean waves

The floating bodies operating in the ocean encounter various sea-states during their operation. In ship or offshore structure design, long-term hydro-structure analysis is commonly used to find the given design performance. A long-term analysis includes a very long simulation of the floating unit, where the vessel encounters various environmental conditions. There are two

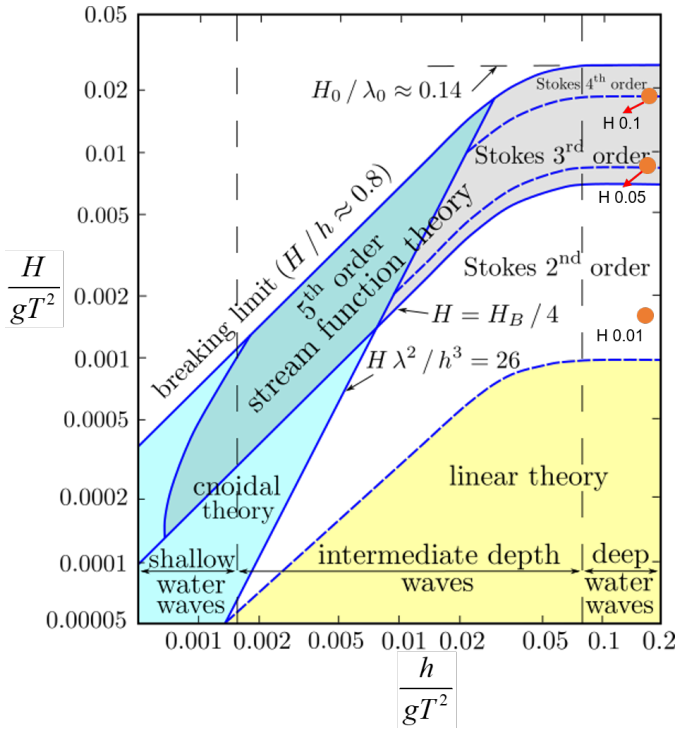


Figure 3.1. Le Méhauté's diagram with three regular wave conditions [3].

objectives of long-term analysis; one is the extreme response of the floating body during the extreme environmental conditions; another is the accumulated fatigue damage over time. The long-term analysis is obtained from the combination of short-term (3 hours) analysis results simulated with different environmental conditions.

The mentioned environmental conditions include current, wind, and waves, but only waves are considered in this thesis. Wave conditions can be described differently in the long-term and the short-term. The long-term wave conditions represent various wave conditions of a target spot, usually for several years depending on the floating unit's lifetime. The short-term wave conditions represent the wave present in a single spectrum, assuming that the sea-state is stationary during a short duration (3 hours).

The long-term wave conditions may be given mainly by wave scatter diagrams. A wave scatter diagram is constructed as a joint probability of significant wave height and wave period at a geographical area. The 104 areas of global wave statistics, including some seasonal and directional information, are provided online. These scatter diagram data are usually obtained from visual observation and hindcast data. They are extrapolated using probability density functions. The well-known scatter diagrams are the Worldwide scatter diagram and IACS scatter diagram. The Worldwide scatter diagram is a representative wave condition for worldwide trade. The IACS scatter diagram is defined based on the winter data of the North Atlantic area. The IACS scatter diagram is considered to give the most severe sea-state in the world.

The short-term wave conditions may be given in the form of wave spectra or wave statistics. The wave spectrum is often described in the form of an analytical equation. For some cases, a directional short-crested wave spectrum can be defined as a unidirectional wave spectrum ($S(\omega)$)

multiplied by a spreading function ($D(\theta)$), where θ is the spreading angle between θ_{min} and θ_{max}.

$$S_D(\omega, \theta) = S(\omega)D(\theta) \tag{3.8}$$

$$\int_{\theta_{min}}^{\theta_{max}} D(\theta)d\theta = 1 \tag{3.9}$$

The commonly used wave spectra are the Pierson-Moskowitz (P-M) spectrum [78] and the Joint North Sea Wave Project (JONSWAP) spectrum [79]. The JONSWAP spectrum is developed based on the wave recorded along a line extending over 160km on the North sea. Hasselmann et al. derived the JONSWAP spectrum using parameters of α_p, ω_p, σ, and γ [79].

$$S(\omega) = \alpha_p g^2 \omega^{-5} \exp\left[-\frac{5}{4}\left(\frac{\omega}{\omega_p}\right)^{-4}\right] \gamma^{\exp\left[\frac{-(\omega-\omega_p)^2}{2\sigma^2\omega_p^2}\right]} \tag{3.10}$$

$$\sigma = 0.07 \text{ for } \omega < \omega_p, \quad \sigma = 0.09 \text{ for } \omega > \omega_p \tag{3.11}$$

Here, ω is the angular frequency; α_p is the Phillips parameter; ω_p is the peak wave angular frequency; γ is the peak enhancement factor. Readers interested in other wave spectra and their applications to real floating units may refer to the following classification guideline books related to long-term and short-term analysis [80, 81].

Ocean waves are reconstructed in wave tanks using the target wave spectrum to perform model scale experiments or simulations. The open-ocean wave spectra in frequency domain have then to be properly transformed into time-series of irregular waves. The simplest method for the realization of irregular waves is the linear superposition of linear harmonic waves with random phases. The following equations present the 2D velocity potential and the spectral wave amplitude of the irregular waves:

$$\Phi = \sum_{i=1}^{N_{wave}} \frac{gA_i^I}{\omega_i} \frac{\cosh|\mathbf{k}_i|(z+h)}{\cosh|\mathbf{k}_i|h} e^{(-i\mathbf{k}_i \bullet \mathbf{x} - \omega_i t + \delta_i)}, \tag{3.12}$$

$$A_i^I = \sqrt{2S(\omega_i)\Delta\omega}, \tag{3.13}$$

where $\omega_i^2 = g|\mathbf{k}_i|\tanh(|\mathbf{k}_i h|)$ is the i-th wave pulsation; $\mathbf{k}_i$ is the directional wave number; N_{wave} is the number of wave frequencies considered; δ_i is the random phase; A_i^I is the i-th frequency wave amplitude that is given from the wave spectrum S.

The superposition of linear waves assumes that each wave component is independent. However, in reality, interaction exists between wave components. To overcome the limitation of linear superposition, many researchers use direct simulations keeping the non-linear free surface boundary conditions in the model. Two Higher-Order Spectral (HOS) methods are developed in LHEEA Lab (ECN/CNRS), HOS-Ocean and HOS-NWT. The HOS-Ocean solver applies a

pseudo-spectral method to simulate 3D irregular waves with fully non-linear free surface boundary conditions on a periodic domain [82, 83]. The HOS-NWT solver extends the original methodology to the numerical wave tank problem (wavemaker + numerical beach) by adding extra velocity potential in the same principle of pseudo-spectral expansion [84]. This study focuses on the simulation of NWT, and consequently the HOS-NWT solver is used for the generation of irregular waves.

3.2.2 Wave breaking model for non-linear irregular waves

The non-linear potential flow models, such as the HOS-NWT model, naturally neglect vorticity and viscosity effects. These irrotational and inviscid assumptions dramatically facilitate the wave modelling. However, these assumptions prevent the use of potential flow solvers for relatively high vorticity phenomena such as breaking waves as it is typically found in design sea-states. To overcome this limitation, a numerical breaking model, which could take into account the effect of breaking wave events on the potential model, must be considered.

In nature, breaking waves occur when the water particle velocity under the wave crest exceeds or approaches the wave crest velocity. When the waves break, energy from the breaking waves is dissipated in free-surface fragmentation, vorticity and turbulence. Some numerical methods can mimic the effect of wave breaking in a potential flow model. In the in-house version of open-source HOS solvers of LHEEA lab, the Tian-Barthelemy wave breaking model is applied, which uses Tian's eddy viscosity model and Barthelemy's breaking criterion [64, 85, 86]. It uses the breaking onset criteria to predict the breaking event before the wave breaking and estimates the amount of dissipation by using the wave profile.

Following Barthelemy's approach, the breaking onset criterion is the ratio between water particle velocity under the wave crest U_x and the wave crest velocity C_x. Eqn. (3.14) presents Barthelemy's breaking onset criterion. The crest velocities of the random waves are computed using Hilbert transform, and the water particle velocities are given from the potential flow solver. For the present study, the threshold value is 0.85, which was suggested from the previous studies [86, 87].

$$B_x = \frac{U_x}{C_x} > \text{threshold} \tag{3.14}$$

Eddy viscosity (ν_{eddy}) is estimated from the Tian-Barthelemy wave breaking model and included as a diffusion term in the kinematic and dynamic free surface boundary conditions:

$$\begin{aligned}
\frac{\partial \eta_I}{\partial t} &= \left(1 + |\nabla \eta_I|^2\right) \frac{\partial \Phi}{\partial z} - \nabla \tilde{\Phi}_I \bullet \nabla \eta_I + 2\nu_{\text{eddy}} \nabla \bullet \nabla \eta_I \\
\frac{\partial \tilde{\Phi}_I}{\partial t} &= -g\eta_I - \frac{1}{2}\left|\nabla \tilde{\Phi}_I\right|^2 + \frac{1}{2}\left(1 + |\nabla \eta_I|^2\right)\left(\frac{\partial \Phi}{\partial z}\right)^2 + 2\nu_{\text{eddy}} \nabla \bullet \nabla \tilde{\Phi}_I
\end{aligned} \tag{3.15}$$

where the surface potential are defined as $\tilde{\Phi}_I(\mathbf{x}) = \Phi_I(x, y, z = \eta(x, y, t), t)$.

4 Numerical formulations and OpenFOAM

The open-source C++ library OpenFOAM [88] is used together with an additional naval application library `foamStar`. The OpenFOAM library provides a platform to support unstructured polyhedral FVM and other related functions for general CFD applications. Some of the commonly used OpenFOAM discretization schemes are shortly explained in the OpenFOAM user guide [19]. The additional library `foamStar` is an in-house code co-developed by Bureau Veritas and Ecole Centrale de Nantes to support specific functionalities for naval applications as wave generation & absorption and hydroelastic model [89, 90]. The descriptions of wave generation methods and 6DOFs motion solver implemented in the `foamStar` library are given in this chapter. Also, the following sections describe some basics of OpenFOAM and the finite volume discretization of the governing equations and boundary conditions used in this study. Finally, an example of `fvScheme` and `fvSolution` files for naval applications is given.

4.1 Finite volume discretization

The numerical discretization of the governing equations for averaged two-phase flows is presented in this section. OpenFOAM divides their computational space into a finite number of arbitrary convex polyhedral control volumes (cells). Each cell is treated as a control volume, having averaged quantities within the cell. Cells must not overlap each other and must fill the entire computational domain. A geometrical description of a computational cell is illustrated in Figure 4.1, where P is an owner cell of the face f and a index of the other cell N indicates the neighbour cell.

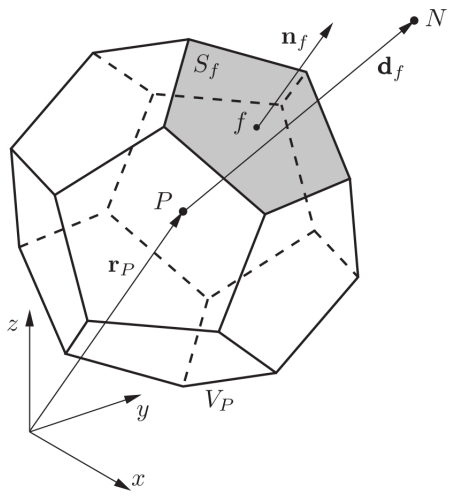


Figure 4.1. Finite volume cell in polyhedral shape (figure from [4]).

A computational cell has a volume (V_P) and has its faces connected with the neighbour cells or boundary surfaces. The surface area vector ($\mathbf{S}_f$) is defined as a vector normal to the corresponding face with its magnitude equal to the surface area. The distance vector ($\mathbf{d}_f = \mathbf{x}_N - \mathbf{x}_P$)

is defined as the distance from owner cell P to neighbour cell N. General FV discretization on collocated grids is well described in the thesis of Jasak [65].

4.2 Discretization of averaged two-phase incompressible flow model

This section presents the numerical discretization of the governing equations for the averaged two-phase incompressible flow model. The finite volume representation of its momentum equation is given in Eqn. (4.1). Eqn. (4.2) presents the finite volume representation of the algebraic VOF convection equation with the artificial compression term. The subscript $(...)_P$ refers to the owner cell averaged properties, $(...)_N$ refers to the neighbour cells averaged properties, and $(...)_f$ refers to the face averaged properties usually obtained by a cells-to-face interpolation. $\phi_f = \mathbf{u}_f \bullet \mathbf{S}_f$ represents the velocity face flux at face f; $\phi_{r,f}$ is the flux of the relative velocity.

$$\frac{\partial (V\rho\mathbf{u})_P}{\partial t} + \sum_f^N ((\rho\phi)_f \mathbf{u}_f) - \sum_f^N (\mu_f \mathbf{S}_f \bullet (\nabla \mathbf{u}_f)) - \nabla \mathbf{u} \bullet \nabla \mu V_P = -\nabla p_d V_P - (\mathbf{g} \bullet \mathbf{x}) \nabla \rho V_P \tag{4.1}$$

$$\frac{\partial (V\alpha)_P}{\partial t} + \sum_f^N (\phi\alpha)_f + \sum_f^N (\phi_{r,f} \alpha_f (1 - \alpha_f)) = 0 \tag{4.2}$$

$$(\rho\phi)_f = (\phi\alpha)_f (\rho_{water} - \rho_{air}) + \phi_f \rho_{air} \tag{4.3}$$

The mass flux $((\rho\phi)_f)$ can be computed from the VOF flux $((\phi\alpha)_f)$ using Eqn. (4.3), and this mass flux formulation couples Eqn. (4.1) and Eqn. (4.2). The time derivative term can have many forms depending on the time integration scheme. For unstructured mesh, the face property (f) is calculated from interpolation between the owner cell (P) and the neighbour cell (N) sharing the same face.

$$q_f = w_f q_P + (1 - w_f) q_N \tag{4.4}$$

Eqn. (4.4) shows the cells-to-face interpolation of face value from owner cell and neighbour cell using the interpolation weight (w_f). The interpolation weight is determined from the grid geometry and the discretization scheme. Therefore, the choice of the spatial discretization scheme determines the order (bounded by 2 because only one neighbour is used) and the accuracy of the solver.

Assuming an arbitrary time integration scheme is applied to Eqn. (4.1), the equation can be transformed into Eqn. (4.5) using the notations $\mathbf{a}_P$ and $\mathbf{H}$, which are commonly used in OpenFOAM to explain the PISO algorithm [65]. $\mathbf{a}_P$ indicates the diagonal components of the momentum equation (UEqn), and $\mathbf{H}$ includes all off-diagonal and source components of the momentum equation.

$$\text{UEqn} : \mathbf{a}_P \mathbf{u}_P - \mathbf{H} = \nabla p_d V_P + (\mathbf{g} \bullet \mathbf{x}) \nabla \rho_P V_P \tag{4.5}$$

The velocity $\mathbf{u}_P$ from the LHS of Eqn. (4.5) must satisfy the continuity condition (Eqn. (2.2)). Appying the continuity equation to $\mathbf{u}_P$ yields the following pressure equation.

$$\nabla\bullet\mathbf{u}_P = \nabla\bullet\Big[\frac{\mathbf{H}}{\mathbf{a}_P} - \frac{1}{\mathbf{a}_P}\big[-\nabla p_{d,P}V_P - (\mathbf{g}\bullet\mathbf{x})\nabla\rho_P V_P\big]\Big] = 0 \tag{4.6}$$

$$\text{pEqn}: \sum_f \Big[\Big(\frac{1}{\mathbf{a}_P}\nabla p_d\Big)_f\Big]\bullet\mathbf{S}_f = \sum_f \Big[\frac{\mathbf{H}}{\mathbf{a}_P} - \frac{1}{\mathbf{a}_P}(\mathbf{g}\bullet\mathbf{x})\nabla\rho_P V_P\Big]_f\bullet\mathbf{S}_f \tag{4.7}$$

The pressure equation (pEqn) is a Poisson equation used to obtain the dynamic pressure. Using this computed dynamic pressure, the velocity and flux are corrected using Eqns. (4.8). With UEqn, pEqn and the correction equation (Eqns. (4.8)), the dynamic pressure and the velocity are strongly coupled inside the non-linear iteration.

$$\begin{aligned}\mathbf{u}_P &= \Big[\frac{\mathbf{H}}{\mathbf{a}_P} - \frac{1}{\mathbf{a}_P}(\mathbf{g}\bullet\mathbf{x})\nabla\rho_P V_P\Big] - \Big[\frac{V_P}{\mathbf{a}_P}\nabla p_{d,P}\Big]\\ \phi_f &= \Big[\frac{\mathbf{H}}{\mathbf{a}_P} - \frac{1}{\mathbf{a}_P}(\mathbf{g}\bullet\mathbf{x})\nabla\rho_P V_P\Big]_f\bullet\mathbf{S}_f - \Big[\frac{V_P}{\mathbf{a}_P}\nabla p_{d,P}\Big]_f\bullet\mathbf{S}_f\end{aligned} \tag{4.8}$$

4.3 Discretization of the SWENSE model

This section presents the finite volume (FV) representation of the SWENSE formulation. The VOF convection equation does not change from the one in the averaged two-phase flow model, and it is not presented in this section. Eqn. (4.9) recalls the momentum equation of the SWENSE method.

$$\frac{\partial\rho\mathbf{u}_C}{\partial t} + \nabla\bullet(\rho\mathbf{u}\mathbf{u}_C) + \rho\mathbf{u}_C\bullet\nabla\mathbf{u}_I = -\nabla p_C - \frac{p_I}{\rho_w}\nabla\rho + \nabla\bullet(\mu(\nabla\mathbf{u}_C + \nabla\mathbf{u}_C^T)) \tag{4.9}$$

The finite volume representation of the SWENSE momentum equation reads:

$$\begin{aligned}\frac{\partial(V\rho\mathbf{u}_C)_P}{\partial t} + \sum_f^N((\rho\phi)_f\mathbf{u}_{C,f}) - \sum_f^N(\mu_f\mathbf{S}_f\bullet(\nabla\mathbf{u}_{C,f})) - \nabla\mathbf{u}_{C,P}\bullet\nabla\mu_P V_P\\ +\rho\mathbf{u}_{C,P}\bullet\nabla\mathbf{u}_{I,P}V_P = -\nabla p_{C,P}V_P - \frac{p_{I,P}}{\rho_w}\nabla\rho_P V_P.\end{aligned} \tag{4.10}$$

The subscripts and variables are identical to those defined for the two-phase flow formulation. Similar to Section 4.2, Eqn. (4.10) can be rewritten as Eqn. (4.11). $\mathbf{a}_{C,P}$ indicates the diagonal components of the SWENSE momentum equation (U_CEqn), and $\mathbf{H}_C$ includes all off-diagonal and source components of the system.

$$\text{U}_C\text{Eqn}: \mathbf{a}_{C,P}\mathbf{u}_{C,P} - \mathbf{H}_C = -\big[\nabla p_{C,P}V_P + \frac{p_{I,P}}{\rho_w}\nabla\rho_P V_P\big] \tag{4.11}$$

$$\nabla\bullet\mathbf{u}_{C,P} = \nabla\bullet\Big[\frac{\mathbf{H}_C}{\mathbf{a}_{C,P}} - \frac{1}{\mathbf{a}_{C,P}}\big[\nabla p_{C,P}V_P + \frac{p_{I,P}}{\rho_w}\nabla\rho_P V_P\big]\Big] = 0 \tag{4.12}$$

The velocity $\mathbf{u}_{C,P}$ from the LHS of Eqn. (4.11) must satisfy the continuity equation. The substitution of this velocity into the continuity equation yields the following complementary pressure equation:

$$\text{pcEqn}: \sum_f \left[\left(\frac{1}{\mathbf{a}_{C,P}} \nabla p_{C,P} \right)_f \right] \bullet \mathbf{S}_f = \sum_f \left[\frac{\mathbf{H}_C}{\mathbf{a}_{C,P}} - \frac{1}{\mathbf{a}_{C,P}} \frac{p_{I,P}}{\rho_w} \nabla \rho_P V_P \right]_f \bullet \mathbf{S}_f. \tag{4.13}$$

The complementary pressure equation is a Poisson equation which permits to obtain the complementary pressure. After the complementary pressure is computed, the complementary velocity and the fluid flux are corrected as in Eqns. (4.14). Therefore, the complementary pressure and the complementary velocity are strongly coupled within the non-linear iterations between U_CEqn, pcEqn, and Eqns. (4.14).

$$\begin{aligned} \mathbf{u}_C &= \left[\frac{\mathbf{H}_C}{\mathbf{a}_{C,P}} - \frac{1}{\mathbf{a}_{C,P}} \frac{p_{I,P}}{\rho_w} \nabla \rho_P V_P \right] - \left[\frac{V_P}{\mathbf{a}_{C,P}} \nabla p_{C,P} \right] \\ \phi_f &= \phi_{I,f} + \left[\frac{\mathbf{H}_C}{\mathbf{a}_{C,P}} - \frac{1}{\mathbf{a}_{C,P}} \frac{p_{I,P}}{\rho_w} \nabla \rho_P V_P \right]_f \bullet \mathbf{S}_f - \left[\frac{V_P}{\mathbf{a}_{C,P}} \nabla p_{C,P} \right]_f \bullet \mathbf{S}_f \end{aligned} \tag{4.14}$$

4.4 Discretization of the transport equations of turbulence model

The following equations recall the two transportation equations of the free-surface k-ω SST model. The coefficients and the definitions of each term can be found in Section 2.3.

$$\begin{aligned} \frac{\partial(\rho k)}{\partial t} + \nabla\bullet(\rho \mathbf{u} k) - \nabla\bullet((\mu + \rho\alpha_k \nu_T)\nabla k) &= \rho p_0 - \frac{2}{3}\rho k(\nabla\bullet\mathbf{u}) - \rho\beta^* k\omega - \rho P_b, \\ \frac{\partial(\rho\omega)}{\partial t} + \nabla\bullet(\rho \mathbf{u}\omega) - \nabla\bullet((\mu + \rho\alpha_\omega \nu_T)\nabla\omega) &= \frac{\rho\gamma p_0}{\nu} - \frac{2}{3}\rho\gamma\omega(\nabla\bullet\mathbf{u}) - \rho\beta\omega^2 - \rho(F_1 - 1)CD_{k\omega}. \end{aligned}$$

The finite volume expressions of these two turbulence transportation equations are derived as follows. The second term of the LHS of these equations is the convection term, and the third term is the diffusion term. From author's experience, the free-surface k-ω SST model was much more unstable than the standard k-ω SST model due to the mass flux term in the convection term since the standard k-ω SST model did not consider the discontinuous density field. Therefore, the free-surface k-ω SST model was solved using only the *implicit Euler* time integration scheme since other choices made the solution diverged. In case of unstable simulation, the two turbulence equations can be solved after the velocity-pressure correction to enhance the stability of the simulation, giving better convergence of k and ω.

$$\begin{aligned} \frac{\partial(V\rho k)_P}{\partial t} + \sum_f^N \left((\rho\phi)_f k_f \right) - \sum_f^N \left((\mu + \rho\alpha_k \nu_T)_f \nabla k_f \right) \bullet \mathbf{S}_f \\ = \left(\rho p_0 - \frac{2}{3}\rho k(\nabla\bullet\mathbf{u}) - \rho\beta^* k\omega - \rho P_b \right)_P V_P \end{aligned} \tag{4.15}$$

$$\begin{aligned} \frac{\partial(V\rho\omega)_P}{\partial t} + \sum_f^N \left((\rho\phi)_f \omega_f \right) - \sum_f^N \left((\mu + \rho\alpha_\omega \nu_T)_f \nabla\omega_f \right) \bullet \mathbf{S}_f \\ = \left(\frac{\rho\gamma p_0}{\nu} - \frac{2}{3}\rho\gamma\omega(\nabla\bullet\mathbf{u}) - \rho\beta\omega^2 - \rho(F_1 - 1)CD_{k\omega} \right)_P V_P \end{aligned} \tag{4.16}$$

4.5 Wave generation and absorption

Accurate wave generation and absorption are essential for the construction of a NWT. The most common methodology is named relaxation scheme [91, 92, 93, 42, 5]. It smoothly blends the computed solution with the target solution in a relaxation zone. In general, two types of relaxation method are available. One is the implicit blending, which inserts the target equations into the governing equations. The implicit blending has been applied and validated by many researchers [94, 25]. Perić et al. numerically found the optimal strength coefficient for implicit relaxation and checked the effect of the strength coefficient [95]. Another relaxation method is the explicit blending method, which explicitly blends the computed solution with the target solution after solving the governing equation. The explicit blending is used in this thesis, and is described below.

4.5.1 Relaxation scheme

In the `foamStar` library, the generation and absorption of waves are based on an explicit relaxation scheme [42, 5]. Figure 4.2 illustrates the definition of the relaxation zones and the pure computational domain. The explicit relaxation scheme uses a smooth weight function (w) between 0 and 1 in the relaxation zone. After solving the governing equations, the computed solution (χ) is relaxed with the target (incident wave) solution (χ_I) as:

$$\chi = (1 - w)\chi + w\chi_I. \tag{4.17}$$

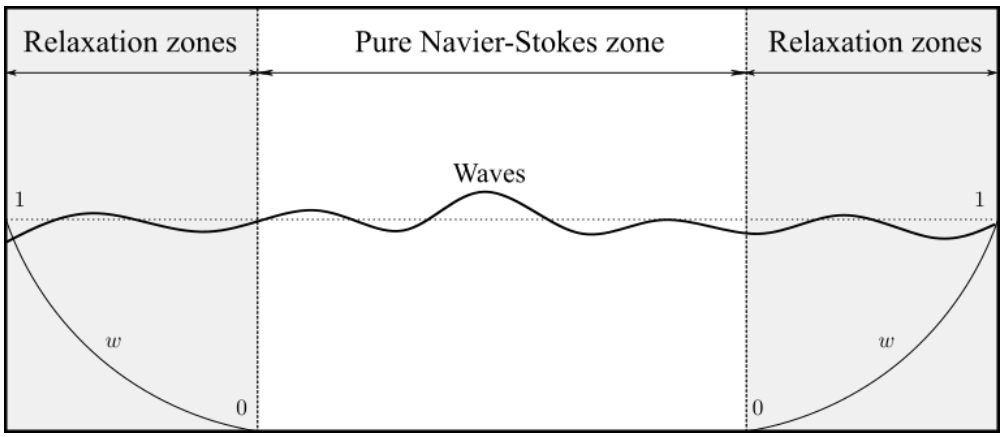


Figure 4.2. The definition of the relaxation zone and the pure computational domain

The size of the relaxation zone is a critical factor in terms of numerical cost against accuracy and in particular to minimize wave reflection. The choice of weight function is also important to that matter. In Choi et al. [96], the wave propagation was studied in 2D NWT with different relaxation zone lengths, absorption methods, and various weight functions. Different absorption methods such as linear damping, quadratic damping, and additional viscosity in the outlet are compared to the standard wave relaxation method. The results showed that the wave relaxation

scheme gives the minimum wave reflection. The polynomial weight function [91, 93], the exponential weight function [42], and the dynamic weight function [5] are described in Eqn. (4.18).

$$w(\xi_{relax}) = \begin{cases} \dfrac{e^{\xi_{relax}^{3.5}} - 1}{e - 1} & \text{exponential weight} \\ -2\xi_{relax}^3 + 3\xi_{relax}^2 & \text{polynomial weight} \\ w^d = 1 - (1 - w)^{|\mathbf{u} - \mathbf{u}^{Target}|\Delta t/\Delta x} & \text{dynamic weight} \end{cases} \tag{4.18}$$

$\xi_{relax} \in [0, 1]$ is the normalized coordinate in relaxation zones; $\mathbf{u}$ is the computed fluid velocity; $\mathbf{u}^{Target}$ is the target fluid velocity; Δx is the size of the cell. The dynamic weight function calculates the relaxation weight based on the difference between target and computed solution. Choi et al. [96] concluded that the static exponential weight function gave the best preservation of wave amplitude, while the dynamic weight function gave minimum wave reflection.

4.5.2 Reconstruction of the non-linear irregular waves in the CFD domain (Grid2Grid)

A wrapper program named `Grid2Grid` has been developed to reconstruct the wavefields from other wave solvers' results, such as HOS computation [97]. `Grid2Grid` is an open-source C++ package published at https://github.com/LHEEA/Grid2Grid. `Grid2Grid` applies inverse FFTs [98] and a quick B-spline module [99] to reconstruct the non-linear wave fields for an arbitrary simulation time and location. For instance, HOS solvers provide the time histories of mode amplitudes. By applying inverse FFTs with vertical functions, $f(z) = \frac{\cosh k_{mn}(z+h)}{\cosh k_{mn}h}$ and $f^{add}(z') = \cos k_p^{add}(z'+1)$, the exact non-linear wavefields are reconstructed in a rectilinear grid at HOS simulation time $t = t^{HOS}$. The rectilinear grid is not the CFD grid but a grid used for the interpolation. The rectilinear grid does not change with respect to time, and the wavefield data are then interpolated to the movable CFD grid using a B-spline interpolation scheme in space and time. Further details on the `Grid2Grid` interpolation scheme and its application method are presented in the `Grid2Grid` manual [97].

4.6 Modelling of 6 DOFs rigid body motions

4.6.1 6 DOFs rigid body motions

Before introducing the 6 DOFs motions, it is crucial to define the global reference frame in which the governing equations are valid. The global reference frame's origin is positioned at the intersection between the waterline and the Aft Perpendicular (AP) line at $t = 0$. The body-fixed frame is introduced to define the 6 DOFs motions and equations easily. Both frames are Cartesian systems and defined according to the right-hand rule. The three translation motions, x_g, y_g, and z_g, are a surge, sway, and heave motions and defined as the motion of the Center Of Gravity (COG) in the global reference frame R_0 $(O, \mathbf{x}_0, \mathbf{y}_0, \mathbf{z}_0)$. A positive surge (x_g) is defined forward, and a positive heave (z_g) is defined upwards.

The rotation angles need to be defined carefully since the transformation between the reference frame and the body-fixed frame is not unique. Different methods exist for represent-

ing rotation motions in 3D. Among these methods, `foamStar` library supports Euler angle and the quaternion representations. Current work follows the definition of Euler angles with zyx-convection. The three angles (ϕ, θ, ψ) are roll, pitch, and yaw, respectively. The right-hand rule is applied to determine the positive direction of each angle.

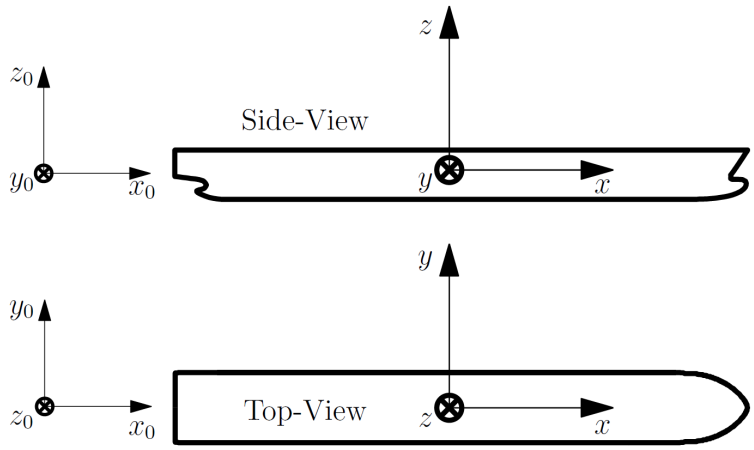


Figure 4.3. The Galilean referential frame R_0 $(O, \mathbf{x}_0, \mathbf{y}_0, \mathbf{z}_0)$ and the body-fixed frame R_b $(O_b, \mathbf{x}, \mathbf{y}, \mathbf{z})$ in side-view and top-view (Figure from [5]).

The equations of motion are given by Newton's second law expressed in the body-fixed frame R_b in Eqn. (4.19):

$$[M]\ddot{\mathbf{q}} = \mathbf{Q}_e + \mathbf{Q}_v + \mathbf{Q}_c - [K]\mathbf{q} - [C]\dot{\mathbf{q}} \tag{4.19}$$

where $[M]$ is the mass matrix; $[C]$ is the damping matrix; $[K]$ is the stiffness matrix; $\mathbf{q}$ are the motions; $\mathbf{Q}_e$ are the external forces acting on the body; $\mathbf{Q}_v$ are the quadratic velocity forces (Coriolis/centrifugal forces); $\mathbf{Q}_c$ is the constraint forces. The external forces can be the gravity force, the mooring force, the propulsion force, and the fluid force. The fluid force is evaluated as the surface integration of pressure and viscous stress over the body surface. Here, $\mathbf{n}$ is a unit surface normal vector pointing inside the body and σ indicates the viscous stress tensor.

$$\mathbf{f}_{o,fluid} = \int_{body} (p\mathbf{n} - \sigma_{\bullet}\mathbf{n})\, dS \tag{4.20}$$

4.6.2 Mesh morphing due to rigid body motion in the computational domain

After the 6 DOFs body motions are solved, the CFD computational domain needs to move and deform using a mesh morphing technique. The mesh morphing changes only the topology of the mesh without changing the connectivity of the mesh. When the topology of the mesh changes, the flow field must be interpolated and transferred to the updated mesh.

Several mesh morphing methodologies can be applied to finite volume meshes. Jasak and Tukovič summarized the commonly used mesh morphing strategies [100]. Here, only the mesh

morphing strategies used in this thesis are summarized: `Laplacian smoothing` and `Solid body motion` technique. The `Laplacian smoothing` technique solves an equation for the mesh motion which is put in the form of the Poisson equation. The movement of internal mesh points is determined with the displacement boundary conditions and the controllable diffusivity coefficient. If $\mathbf{x}$ is the displacement of a grid point, γ is the diffusive coefficient, and r is the distance from the moving boundary, the displacement can be obtained by solving the following Poisson equation.

$$\nabla\bullet(\gamma(r)\nabla\mathbf{x}) = 0 \quad , \quad \gamma(r) = \frac{1}{r^2} \tag{4.21}$$

The `Laplacian smoothing` technique requires considerably high CPU time compared to other mesh deformation schemes since it has to solve a Poisson equation. The `Solid body motion` method moves the whole mesh as if it were a solid body. There is thus no need to deform the internal mesh. However, this technique add some difficulties in specifying some boundary conditions, such as the open-air boundary condition at the top boundary.

In this thesis, the motions that do not have large restoring forces (surge, sway, and yaw) use the `Solid body motion` method not to make a highly non-orthogonal and distorted mesh from the morphing. The heave, roll, and pitch motions morph the mesh using the `Laplacian smoothing` technique. The Laplacian mesh smoothing technique exists in the OpenFOAM package, and an additional non-deformable inner region can be predefined to avoid mesh deformation close to the body. For all applications in this thesis, the quadratic inverse distance function is used as diffusive coefficient. Note that OpenFOAM also supports other diffusivity coefficient options, such as the exponential function.

The implementation of the forward speed of the ship also uses the `Solid body motion` technique, as it considers that the whole numerical domain moves with the ship forward speed u_{target}. The forward speed is gradually increased in time from zero to the target speed using a sine-shaped weight function to stabilize the numerical simulations. Eqn. (4.22) shows the surge position and the surge velocity at the time t due to the forward speed, where t_{R} is the ramp duration.

$$u_g(t) = (0.5 - 0.5\cos((\pi/t_{\text{R}})t))u_{\text{target}} \quad \text{for } t < t_{\text{R}} \tag{4.22}$$

4.6.3 Arbitrary Lagrangian-Eulerian (ALE) formulation

The governing equations described in this section are based on the classical Eulerian formulation which describes the flow quantities on a fixed spatial coordinate. The Arbitrary Lagrangian-Eulerian (ALE) formulation allows the change of computational grids in time. The general ALE formulation for a generic flow quantity f is written as:

$$\left.\frac{df_L}{dt}\right|_X = \left.\frac{df_{ALE}}{dt}\right|_\chi + \mathbf{u}_c\bullet\nabla(f), \quad \mathbf{u}_c = \mathbf{u} - \mathbf{u}_m. \tag{4.23}$$

$$\left.\frac{df_{ALE}}{dt}\right|_\chi = \left.\frac{df_E}{dt}\right|_\chi + \mathbf{u}_c\bullet\nabla(f), \quad \mathbf{u}_c = \mathbf{u} - \mathbf{u}_m. \tag{4.24}$$

where the subscript $\frac{d}{dt}\big|_L$ indicates the Lagrangian derivative and $\frac{d}{dt}\big|_{ALE}$ indicates the time derivative in an arbitrary motion of the domain. $\mathbf{u}_c$ is the flow convective velocity and $\mathbf{u}_m$ is the mesh velocity. If the mesh moves in time, for a generic flow quantity f and the position vector $\mathbf{r}$, the following equation gives the linear interpolation of the old time value ($t^{(n)}$) to the new cell ($P^{(n+1)}$) center [101].

$$f^{(n)}_{P^{(n+1)}} = f^{(n)}_{P^{(n)}} + (\nabla f^{(n)}_{P^{(n)}}) \bullet (\mathbf{r}_{P^{(n+1)}} - \mathbf{r}_{P^{(n)}}) \tag{4.25}$$

Replacing the generic flow quantity f to the V_P/h yields following space conservation law:

$$\begin{aligned} \frac{V^{(n)}_{P^{(n+1)}}}{h} &= \frac{V^{(n)}_{P^{(n)}}}{h} + \frac{\nabla V^{(n)}_{P^{(n)}} \bullet (\mathbf{C}^{(n+1)} - \mathbf{C}^{(n)})}{h} \\ &= \frac{V^{(n)}_{P^{(n)}}}{h} + \sum_{f^{(n)}} \frac{\mathbf{S}^{(n)}_f \bullet (\mathbf{f}^{(n+1)}_c - \mathbf{f}^{(n)}_c)}{h} \\ &= \frac{V^{(n)}_{P^{(n)}}}{h} + \sum_{f^{(n)}} \mathbf{S}^{(n)}_f \bullet \mathbf{u}_{m,f} \end{aligned} \tag{4.26}$$

where h is the time-step size, $\mathbf{C}$ is a cell center point vector, $\mathbf{f}_c$ is a face center point vector, and $\mathbf{u}_{m,f}$ is the mesh velocity at the face. The solver needs to accurately calculate the mesh velocity $\mathbf{u}_{m,f}$ at each face to minimize the error from the ALE formulation. OpenFOAM measures the mesh velocity using 'signed swept-volume' between $V^{(n)}_{P^{(n+1)}}$ and $V^{(n)}_{P^{(n)}}$ by computing the volume covered by each face during the moving mesh step in a grid level. Figure 4.4 illustrates the concept of the signed swept-volume of a cell. The rate of signed swept-volume at each face is named mesh flux ($\phi^{(n+1)}_{mesh}$). OpenFOAM commonly uses the mesh flux instead of the mesh velocity, and the mesh flux term is combined with the convection term of the governing equations.

$$\frac{V^{(n)}_{P^{(n+1)}}}{h} = \frac{V^{(n)}_{P^{(n)}}}{h} + \sum_{f^{(n+1)}} \phi^{(n+1)}_{mesh} \tag{4.27}$$

For example, the application of ALE formulation to the VOF convection equation is shown below. An *implicit Euler* time integration scheme is applied.

$$\frac{(V\alpha)^{(n+1)}_{P^{(n+1)}}}{h} - \frac{(V\alpha)^{(n)}_{P^{(n+1)}}}{h} + \sum_{f^{(n+1)}} (\phi\alpha)^{(n+1)}_f + \sum_{f^{(n+1)}} (\phi_{r,f}\alpha_f(1-\alpha_f))^{(n+1)} = 0 \tag{4.28}$$

Since it is an interpolation, the VOF could be both $\alpha^{(n)}$ and $\alpha^{(n+1)}$.

$$\frac{(V\alpha)^{(n)}_{P^{(n+1)}}}{h} = \frac{(V\alpha)^{(n)}_{P^{(n)}}}{h} + \sum_{f^{(n+1)}} \alpha^{(n,n+1)}_f \phi^{(n+1)}_{mesh} \tag{4.29}$$

$$\frac{(V\alpha)_{P^{(n+1)}}^{(n+1)}}{h} - \frac{(V\alpha)_{P^{(n)}}^{(n)}}{h} + \sum_{f^{(n+1)}} \left(\left(\phi_f^{(n+1)} - \phi_{mesh}^{(n+1)} \right) \alpha_f^{(n+1)} \right) + \sum_{f^{(n+1)}} \left(\phi_{r,f} \alpha_f (1 - \alpha_f) \right)^{(n+1)} = 0 \tag{4.30}$$

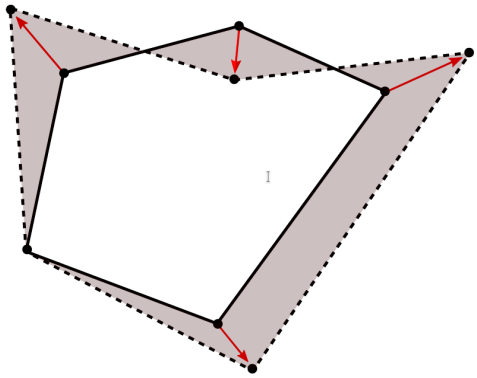


Figure 4.4. The changes of the control volume and signed swept-volume of each face.

4.7 Solution algorithm for the two solvers

4.7.1 Set of equations

This section summarizes the sets of equations used in the two solvers, which are named *foamStar* and *foamStarSWENSE*. Each set of equations corresponds to a yellow box in the flow chart presented in the next section.

VOF convection equation (see equation (4.2))

$$\frac{\partial (V\alpha)_P}{\partial t} + \sum_f^N (\phi\alpha)_f + \sum_f^N \left(\phi_{r,f} \alpha_f (1 - \alpha_f) \right) = 0$$

Averaged two-phase flow model (see equations (4.5), (4.7), and (4.8))

$$\text{UEqn} : \mathbf{a}_P \mathbf{u}_P - \mathbf{H} = \nabla p_d V_P + (\mathbf{g}_\bullet \mathbf{x}) \nabla \rho_P V_P$$

$$\text{pEqn} : \sum_f \left[\left(\frac{1}{\mathbf{a}_P} \nabla p_d \right)_f \right]_\bullet \mathbf{S}_f = \sum_f \left[\frac{\mathbf{H}}{\mathbf{a}_P} - \frac{1}{\mathbf{a}_P} (\mathbf{g}_\bullet \mathbf{x}) \nabla \rho_P V_P \right]_f {}_\bullet \mathbf{S}_f$$

$$\mathbf{u}_P = \left[\frac{\mathbf{H}}{\mathbf{a}_P} - \frac{1}{\mathbf{a}_P} (\mathbf{g}_\bullet \mathbf{x}) \nabla \rho_P V_P \right] - \left[\frac{V_P}{\mathbf{a}_P} \nabla p_{d,P} \right]$$

$$\phi_f = \left[\frac{\mathbf{H}}{\mathbf{a}_P} - \frac{1}{\mathbf{a}_P} (\mathbf{g}_\bullet \mathbf{x}) \nabla \rho_P V_P \right]_f {}_\bullet \mathbf{S}_f - \left[\frac{V_P}{\mathbf{a}_P} \nabla p_{d,P} \right]_f {}_\bullet \mathbf{S}_f$$

SWENSE model (see equations (4.11), (4.13), and (4.14))

$$\text{U}_C\text{Eqn} : \mathbf{a}_{C,P} \mathbf{u}_{C,P} - \mathbf{H}_C = -\left[\nabla p_{C,P} V_P + \frac{p_{I,P}}{\rho_w} \nabla \rho_P V_P \right]$$

$$\text{pcEqn}: \sum_f \left[\left(\frac{1}{\mathbf{a}_{C,P}}\nabla p_{C,P}\right)_f\right]\bullet\mathbf{S}_f = \sum_f \left[\frac{\mathbf{H}_C}{\mathbf{a}_{C,P}} - \frac{1}{\mathbf{a}_{C,P}}\frac{p_{I,P}}{\rho_w}\nabla\rho_P V_P\right]_f \bullet\mathbf{S}_f.$$

$$\mathbf{u}_C = \left[\frac{\mathbf{H}_C}{\mathbf{a}_{C,P}} - \frac{1}{\mathbf{a}_{C,P}}\frac{p_{I,P}}{\rho_w}\nabla\rho_P V_P\right] - \left[\frac{V_P}{\mathbf{a}_{C,P}}\nabla p_{C,P}\right]$$

$$\phi_f = \phi_{I,f} + \left[\frac{\mathbf{H}_C}{\mathbf{a}_{C,P}} - \frac{1}{\mathbf{a}_{C,P}}\frac{p_{I,P}}{\rho_w}\nabla\rho_P V_P\right]_f \bullet\mathbf{S}_f - \left[\frac{V_P}{\mathbf{a}_{C,P}}\nabla p_{C,P}\right]_f \bullet\mathbf{S}_f$$

Turbulence transportation equation (free-surface k-ω SST model) (see equations (4.15) and (4.16))

$$\frac{\partial(V\rho k)_P}{\partial t} + \sum_f^N((\rho\phi)_f k_f) - \sum_f^N((\mu + \rho\alpha_k\nu_T)_f\nabla k_f)\bullet\mathbf{S}_f = (\rho p_0 - \frac{2}{3}\rho k(\nabla\bullet\mathbf{u}) - \rho\beta^* k\omega - \rho P_b)_P V_P$$

$$\frac{\partial(V\rho\omega)_P}{\partial t} + \sum_f^N((\rho\phi)_f\omega_f) - \sum_f^N((\mu + \rho\alpha_\omega\nu_T)_f\nabla\omega_f)\bullet\mathbf{S}_f = (\frac{\rho\gamma p_0}{\nu} - \frac{2}{3}\rho\gamma\omega(\nabla\bullet\mathbf{u}) - \rho\beta\omega^2 - \rho(F_1 - 1)CD_{k\omega})_P V_P$$

4.7.2 Flow chart of the algorithm

Figure 4.5 presents a flow chart of the segregated coupling algorithm to solve the wave-structure interaction problem for both the averaged two-phase flow model and the SWENSE model. The two solvers are named *foamStar* and *foamStarSWENSE*.

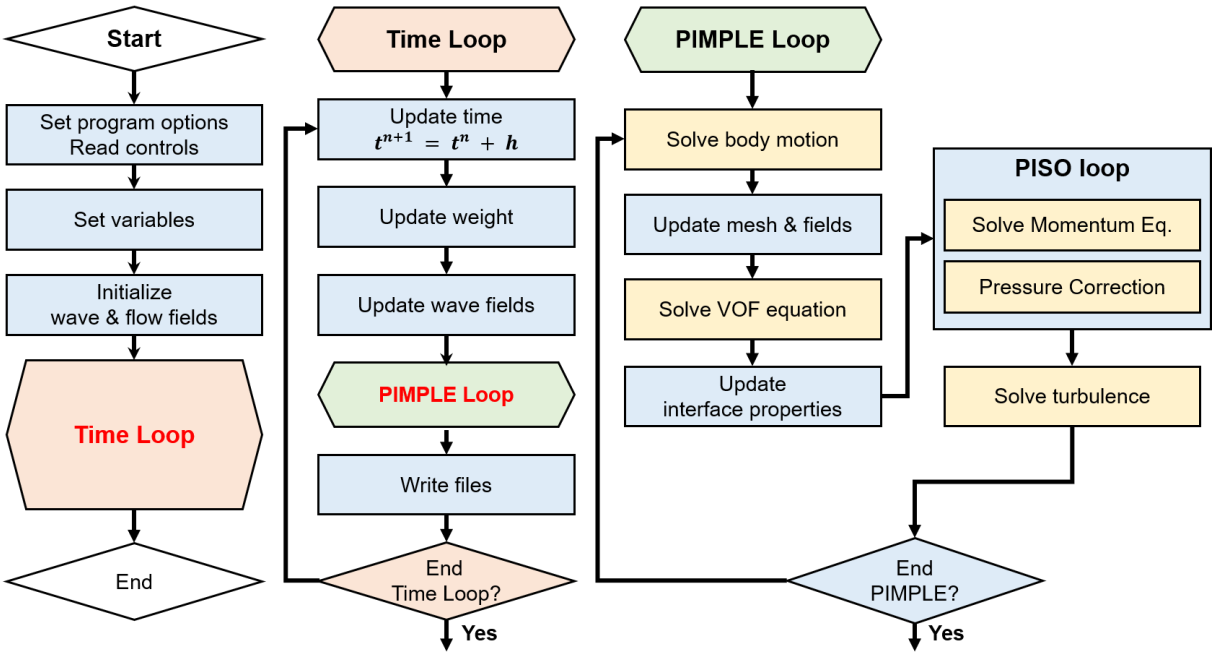


Figure 4.5. Flow chart of the solvers *foamStar* and *foamStarSWENSE*. The yellow color indicates the parts solving each set of the equations.

The flow chart is mainly constructed with three loops. These are the time loop, the PIMPLE loop, and the PISO loop. The time loop includes the PIMPLE loop and updates all fields to the new time step. The PIMPLE loop aims to couple body/mesh motion, VOF field, velocity, pressure, and turbulent quantities within the PIMPLE iteration [20]. The PIMPLE loop includes the inner PISO loop, which is used for pressure–velocity coupling [21].

The following description of on the flow chart is listed sequentially.

Flow chart of the solvers

1. Start the program and set all variables and constants, such as velocity, pressure, and density.
2. Initialize all variables and initialize the wavefield using the wave theory or `Grid2Grid`.
3. Start time loop. When the time is updated to the new time $(t + h)$, re-initialize the wave data and update the relaxation zone.
4. Start PIMPLE iteration. The PIMPLE loop iterates the following procedure.
   - If the body exists, it solves the body motion and updates the mesh using pre-calculated fluid forces. The ALE formulation is applied to the governing equations for the updated mesh.
   - The VOF convection equation is solved with the bounded discretization schemes. It updates the transportation and interface properties based on the updated VOF field and interface.
   - **PISO loop:** The PISO loop solves the averaged two-phase flow model or the SWENSE model described in the previous section. After the momentum equation is solved, it solves the pressure equation derived from the continuity equation and gives the pressure and the velocity. It solves the complementary pressure if the solver uses the SWENSE method. With the updated fields, it repeats PISO iterations.
   - The turbulence of the flow is solved using the turbulence transportation equations in the previous section or using any other turbulence model. The turbulence model can be solved every PIMPLE iteration or only at the last PIMPLE iteration. This option is controlled using a user-selectable option `turbOnFinalIterOnly`.
   - The PIMPLE loop repeats the PIMPLE iteration until the solver satisfies the tolerance criteria or the solver iterates a certain number of iterations.
5. Before the simulation jumps to the next time step, the solver writes files and runs the OpenFOAM post processing function objects, such as calculating surface elevation and body force calculation. The time loop iterates until the current time is less than the end time.
6. End the simulation.

4.8 OpenFOAM discretization schemes

This section presents the OpenFOAM discretization schemes used for general two-phase flow simulations. Moreover, an example of a set of typical discretization schemes and control settings is given for OpenFOAM solvers. These setting files are `fvScheme` and `fvSolution` and placed in the system folder. The `fvScheme` gives control for the discretization scheme of each term of the governing equations. The `fvSolution` gives control to the solution algorithms, the PIMPLE iterations, the tolerances, and the relaxation factors.

4.8.1 Discretization schemes (fvScheme)

The `fvScheme` file gives control of the discretization scheme for each term of the governing equations. For example, Eqn. (4.1) has a divergence term $\sum_f^N ((\rho\phi)_f \mathbf{u}_f)$, and the corresponding OpenFOAM name of this divergence term is `div(rhoPhi,U)`. The appropriate discretization scheme for each term is specified in `fvScheme`. An example of the `fvScheme` used in this thesis is provided below.

```
// ************************************ fvSchemes ************************************ //
ddtSchemes
{
    default             Euler; or CrankNicolson 0.95; or backward;

    // Free-surface $k-\omega$ SST model
    ddt(rho,k)          Euler;
    ddt(rho,omega)      Euler;
}

gradSchemes
{
    default             cellLimited Gauss linear 1.0;
}

divSchemes
{
    default             Gauss linear;
    div(rhoPhi,U)       Gauss linearUpwindV Grad(U);
    div(phi,alpha)      Gauss vanLeer;
    div(phirb,alpha)    Gauss linear;                   (VOF Compression)

    div(rhoPhi,Uc)      Gauss linearUpwindV Grad(Uc);   (SWENSE)
    div(phiComp,UInc)   Gauss linear;                   (SWENSE)

    // Standard $k-\omega$ SST model
    div(phi,k)          Gauss linearUpwind limitedGrad;
    div(phi,omega)      Gauss linearUpwind limitedGrad;

    // Free-surface $k-\omega$ SST model
    div(rhoPhi,k)       Gauss upwind;
```

```
    div(rhoPhi,omega)   Gauss upwind;
}

laplacianSchemes
{
    default             Gauss linear corrected 0.5;
}

interpolationSchemes
{
    default             linear;
}

snGradSchemes
{
    default             corrected;
}

fluxRequired
{
    default             no;
    p_rgh;
    pc;                 (SWENSE)
    pcorr;
    alpha.water;
}

wallDist
{
    method             meshWave;  (Related to turbulence wall functions)
}
// ************************************ fvSchemes ************************************ //
```

Time integration schemes (ddtSchemes) for an unsteady problem in OpenFOAM can be selected among Implicit Euler (Euler), Crank-Nicolson (CN), and second-order backward differential formula (BDF2). The Euler scheme is stable, but the energy dissipation for each time step is large. The CN scheme is a second-order implicit scheme that includes an off-centering coefficient to allow flexibility for a fine-tuning between numerical damping and stability [48]. In most cases, $0.9 \sim 0.95$ off-centering is applied. The BDF2 method is a second-order convergent scheme; however, BDF2 can not guarantee the VOF field's boundedness in two-phase flow [46].

Various spatial discretization schemes are available in OpenFOAM, but only a few of them are thoroughly validated. In this thesis, most of the spatial discretization schemes are those commonly used by the OpenFOAM community. In addition to the `fvScheme` file, Table 4.1 summarizes the spatial discretization schemes for each term of the governing equations generally used in the simulations. The variable q is an arbitrary cell averaged quantity, and q_f is a cells-to-face interpolated quantity.

Table 4.1. Spatial discretization schemes

Item	Equation	fvScheme	Scheme
Gradient	∇q	`grad(q)`	cellLimited Gauss linear 1.0
S. N. gradient	$\mathbf{n}\bullet(\nabla q)_f$	snGrad	corrected 0.5
VOF convection	$\nabla\bullet(\mathbf{u}\alpha)$	`div(phi,alpha)`	Gauss vanLeer + MULES limiter
VOF relative flux	$\nabla\bullet(\mathbf{u}_r\alpha(1-\alpha))$	`div(phirb,alpha)`	Gauss linear
Mom. convection	$\nabla\bullet(\rho\mathbf{u}\mathbf{u})$	`div(rhoPhi,U)`	Gauss linearUpwindV grad(U)
Mom. convection	$\nabla\bullet(\rho\mathbf{u}\mathbf{u}_C)$	`div(rhoPhi,Uc)`	Gauss linearUpwindV grad(Uc)
Mom. Laplacian	$\nabla\bullet(\mu\nabla(\mathbf{u}))$	Laplacian	Gauss linear corrected 0.5
Pressure Laplacian	$\nabla\bullet(\frac{1}{q_p}\nabla(p))$	Laplacian	Gauss linear corrected 0.5
k convection	$\nabla\bullet(\rho\mathbf{u}k)$	`div(rhoPhi,k)`	Gauss upwind
ω convection	$\nabla\bullet(\rho\mathbf{u}\omega)$	`div(rhoPhi,omega)`	Gauss upwind
k convection	$\nabla\bullet(\mathbf{u}k)$	`div(phi,k)`	Gauss linearUpwind limitedGrad
ω convection	$\nabla\bullet(\mathbf{u}\omega)$	`div(phi,omega)`	Gauss linearUpwind limitedGrad
Under-relaxation	q	-	1.0

For 'gradient' computation, the `cellLimited Gauss linear` scheme is used for most of cases. The `cellLimited` scheme bounds the cells-to-face interpolated value based on the owner cell and the neighbour cell information. The 'S. N. gradient' is the surface normal gradient, and the coefficient after 'corrected' is related to the non-orthogonal correction. The divergence (convection) schemes critically affect the stability of the simulation. The divergence scheme for VOF transportation uses Van-Leer scheme [102] and the additional OpenFOAM internal limiter MULES (Multidimensional Universal Limiter for Explicit Solver). MULES employs the Flux-Corrected Transport scheme based on the neighbour cells information and controls the scheme's accuracy and VOF boundedness [103, 104]. Further details on each step of the MULES algorithm can be found in the following thesis [105].

The convection terms with mass flux ($\rho\phi$) also significantly affect the solver numerical dissipation and stability. In this thesis, the momentum equation convection term is discretized with a second-order upwind biased scheme with V-option named `linearUpwindV`. The V-option is an additional option available for the discretization of a vector field. The flow limiters are calculated separately for each vector component [19]. The V-option results in a more stable but less accurate simulation than the simulation case without V-scheme [96]. Further details on the linearUpwindV scheme are provided in Appendix A.

When using the free-surface $k-\omega$ SST model, the choice of the convection scheme critically affects the stability and the computation time of the simulation. The reason the occurrence of the instability is due again to the mass flux term in the convection term. The `Gauss upwind` scheme is applied to both convection terms of the two turbulence transportation equations. Also, the *implicit Euler* time integration scheme is applied to the `ddt(rho,k)` and `ddt(rho,omega)` terms for stability reason. The validity of the free-surface $k-\omega$ SST model is tested using a regular wave test case in Section 7.1.1.

4.8.2 Solvers and PIMPLE options (fvSolution)

The `fvSolution` file controls the linear systems solver, the solution algorithms, the PIMPLE iterations, the tolerances, and the relaxation factors. The selection of linear systems solver for each variable and its tolerance is skipped here. The average level of the tolerance used in this thesis is around 10^{-7}.

```
// ************************************ fvSolution ************************************ //
Tolerance for each variables are skipped.

PIMPLE
{
    momentumPredictor         on;     // Solve momentum equation before pressure equation
    nOuterCorrectors          8;      // n PIMPLE iteration
    nCorrectors               2;      // n PISO iteration
    // n Non-orthogonal correction (With non-orthogonal mesh, use 1)
    nNonOrthogonalCorrectors  0;
    correctPhi                no;

    // When false, Update body motion only at the first PIMPLE loop
    moveMeshOuterCorrectors   true;
    // Work with moveMeshOuterCorrectors, update motion after n PIMPLE iterations
    skipIter                  2;
    fsiTol                    1e-4;   // Body motion tolerance
    fsiMaxIter                20;     // Maximum body motion solver iteration

    // The turbulence model is solved only for the final PIMPLE iteration when it is true
    turbOnFinalIterOnly       false;
}

relaxationFactors
{
    equations
    {
        U           1.0;
        UFinal      1.0;
    }

    fields
    {
        p_rgh       1.0;
        p_rghFinal  1.0;
    }
}
// ************************************ fvSolution ************************************ //
```

The PIMPLE options in `fvSolution` define the number of non-linear iterations used for the PIMPLE algorithm. `nOuterCorrectors` is the number of outer iterations (PIMPLE), and `nCorrectors` is the number of inner iterations (PISO). `nNonOrthogonalCorrectors` defines the

number of non-orthogonal corrector iterations. It is zero for the fully-orthogonal mesh, and use one or more iterations for non-orthogonal meshes. The option `moveMeshOuterCorrectors` controls the body motion solver, and when it is 'false', the solver updates mesh only at the first PIMPLE iteration for the simulation's stability. When it is 'true', another option named `skipIter` can skip mesh update for a certain number of PIMPLE iterations to enhance the stability. The option `turbOnFinalIterOnly` controls the solver to solve the turbulence model every PIMPLE iteration or only at the final PIMPLE iteration. The default is true, but the free-surface k-ω SST has been found to be more stable when it is false.

The relaxation factors (not linked with relaxation zones) are commonly used for improving the stability of the simulations. However, all relaxation factors are set to one to minimize the effect of the computational coefficients on the simulations and focus more on the numerical schemes themselves.

4.9 OpenFOAM boundary conditions

In order to obtain the unique numerical solution from the governing equations, boundary conditions of the CFD domain need to be specified. The mathematical formulation of the boundary conditions of the two-phase flow model and SWENSE formulation can be found in Section 2.1.1 and Section 2.2.3, respectively. This section describes the numerical and physical properties of each variable at the boundary. Before describing the physical properties of boundary conditions, the numerical implementations of two basic types of numerical boundary conditions (Dirichlet and Neumann) are discussed.

Dirichlet boundary condition

The Dirichlet boundary condition defines the flow quantity q at the boundary face b to be q_b. This boundary condition affects the convection and the Laplacian (diffusion) terms in the discretization procedure. If a owner cell (P) includes the boundary face (b), the general convection term can be discretized as:

$$\int_{V_P} \nabla\bullet(\rho\mathbf{u}q)dV = \sum_f (\rho\phi)_f q_f = \sum_{f\neq b} (\rho\phi)_f q_f + (\rho\phi)_b q_b \,. \tag{4.31}$$

The general diffusion term can be discretized as:

$$\int_{V_P} \nabla\bullet(\mu\nabla q)dV = \sum_f \left(\mu_f \mathbf{S}_f\bullet(\nabla q)_f\right) = \sum_{f\neq b} \left(\mu_f \mathbf{S}_f\bullet(\nabla q)_f\right) + \left(\mu_b \mathbf{S}_b\bullet(\nabla q)_b\right), \tag{4.32}$$

$$\mathbf{S}_b\bullet(\nabla q)_b = |\mathbf{S}|\,\frac{q_b - q_P}{|\mathbf{d}_n|}\,. \tag{4.33}$$

where $\mathbf{d}_n$ is the normal vector from face to the cell center.

Neumann boundary condition

The Neumann boundary condition defines the surface normal gradient at the boundary face as:

$$\left(\frac{\mathbf{S}}{|\mathbf{S}|}\bullet\nabla q\right)_b = g_b\,. \tag{4.34}$$

This boundary condition also affects the convection and the Laplacian terms within the discretization. With given cell center value and the gradient, the boundary quantity can be evaluated using an interpolation and can be directly applied to the convection term in Eqn. (4.31).

$$q_b = q_P + \mathbf{d}_n\bullet g_b\,. \tag{4.35}$$

Similarly, the Neumann boundary condition can be applied to the diffusion term as:

$$\int_{V_P}\nabla\bullet(\mu\nabla q)dV = \sum_{f\neq b}\left(\mu_f\mathbf{S}_f\bullet(\nabla q)_f\right) + \left(\mu_b\,|\mathbf{S}|\,g_b\right), \tag{4.36}$$

In the following, the boundary conditions for the two-phase flow model and SWENSE formulation are given. Each variable's boundary condition is presented with its physical representation, such as body/wall, far-field (inlet/outlet), symmetry plane, and open air. The following subsections describe each boundary condition in detail. Also, for each boundary condition, a table of boundary conditions for each field is given.

4.9.1 Body/Wall boundary

This subsection presents the body and wall boundary conditions for all variables, and Table 4.2 summarizes the body/wall boundary conditions.

Table 4.2. Body/Wall boundary conditions for all variables

	Boundary condition		**Boundary condition**
$\mathbf{u}$	movingWallVelocity / slip	p_d	fixedFluxPressure
$\mathbf{u}_C$	$\mathbf{u}_{body} - \mathbf{u}_I$	p_C	fixedFluxPressure
α	zeroGradient	ν_t	nutUSpaldingWallFunction
k	kqRWallFunction	ω	omegaWallFunction

- The gradient of phase fraction at wall/body boundary is assumed zero, approximating that the surface normal gradient of ρ is zero everywhere.

$$\frac{\partial\alpha}{\partial t} = 0 \tag{4.37}$$

- The three turbulence variables, turbulent kinetic energy (k), specific dissipation (ω), and turbulent viscosity (ν_t), follow the OpenFOAM embedded wall functions: `kqRWallFunction`,

`omegaWallFunction`, and `nutUSpaldingWallFunction` [106, 107]. These boundary conditions were selected by testing various combinations of turbulence boundary conditions using Wigley III hull calm resistance analysis.

Velocity & Pressure body boundary conditions with averaged two-phase flow model

- Eqn. (4.38) and Eqn. (4.39) recall the boundary condition for the velocity and the dynamic pressure, where $\mathbf{u}_b$ is the boundary velocity. A Dirichlet type boundary condition is applied to the fluid velocity, and the Neumann type boundary condition is applied to the dynamic pressure.

$$\mathbf{u} = \mathbf{u}_b, \quad \mathbf{u}\bullet\mathbf{n} = \mathbf{u}_b\bullet\mathbf{n}. \tag{4.38}$$

$$\frac{\partial p_d}{\partial n} = -\frac{D(\rho\mathbf{u}_b)}{Dt}\bullet\mathbf{n} + \mathbf{n}\bullet\nabla\bullet(\mu_E(\nabla\mathbf{u}_b + \nabla\mathbf{u}_b^T)) - (\mathbf{g}\bullet\mathbf{x})\frac{\partial\rho}{\partial n} \tag{4.39}$$

- The surface of the solid body is considered impermeable. When the no-slip condition is assumed, the fluid velocity is set equal to the body surface velocity. The `movingWallVelocity` boundary condition covers no-slip body boundary condition with/without body motion.
- For some specific cases, a free-slip boundary condition can be applied instead of a no-slip condition. For example, when simulating the NWT on infinitely deep water, the sea bed boundary may be more appropriately defined using a free-slip condition to minimize the effect of a numerical truncation of an infinite domain. The `slip` boundary condition in OpenFOAM applies the free-slip boundary condition.
- The dynamic pressure boundary condition at an arbitrary boundary should follow Eqn. (4.39) since the right-hand side terms of the equation are non-zero at the boundary. However, homogeneous Neumann boundary condition (`zeroGradient`) is applied due to its simplicity.
- OpenFOAM recommends using the `fixedFluxPressure` pressure boundary condition when the boundary does not move. With `fixedFluxPressure` boundary condition, the dynamic pressure gradient is adjusted by the fluid flux, which is specified with the velocity boundary condition. Fluid velocity on the boundary face is written as follows:

$$(\mathbf{u})_b = \overline{\left(\frac{1}{a_P}\mathbf{H}(\mathbf{u})\right)}_b - \overline{\left(\frac{1}{a_P}\right)}_b (\nabla p_d)_b \,. \tag{4.40}$$

Applying the inner product with the surface vector $\mathbf{S}_b = |\mathbf{S}_b|\,\mathbf{n}_b$ gives:

$$(\mathbf{u})_b\bullet\mathbf{S}_b = \overline{\left(\frac{1}{a_P}\mathbf{H}(\mathbf{u})\right)}_b\bullet\mathbf{S}_b - \overline{\left(\frac{1}{a_P}\right)}_b (\nabla p_d)_b\bullet\mathbf{S}_b. \tag{4.41}$$

Therefore, the pressure boundary condition is given as

$$
\begin{aligned}
(\nabla p_d)_b \bullet \mathbf{n}_b &= \frac{\overline{(a_P)}_b}{|\mathbf{S}_b|}\left(\overline{\left(\frac{1}{a_P}\mathbf{H}(\mathbf{u})\right)}_b \bullet \mathbf{S}_b - (\mathbf{u})_b \bullet \mathbf{S}_b\right) \\
&= \frac{\overline{(a_P)}_b}{|\mathbf{S}_b|}\left(F_b^{H/A} - F_b\right),
\end{aligned}
\tag{4.42}
$$

$$
F_b^{H/A} = \overline{\left(\frac{1}{a_P}\mathbf{H}(\mathbf{u})\right)}_b \bullet \mathbf{S}_b, \qquad F_b = (\mathbf{u})_b \bullet \mathbf{S}_b. \tag{4.43}
$$

where $F_b^{H/A}$ is the flux predicted by the momentum equation, and F_b is the flux calculated by the velocity boundary condition.

Velocity & Pressure body boundary conditions with SWENSE model

- Eqn. (4.44) and Eqn. (4.45) recall the boundary condition for the complementary velocity and complementary pressure.

$$
\mathbf{u}_C = \mathbf{u}_b - \mathbf{u}_I, \quad \mathbf{u}_C \bullet \mathbf{n} = \mathbf{u}_b \bullet \mathbf{n} - \mathbf{u}_I \bullet \mathbf{n} \tag{4.44}
$$

$$
\frac{\partial p_C}{\partial n} = -\frac{D(\rho \mathbf{u}_b)}{Dt} \bullet \mathbf{n} + \mathbf{n} \bullet \nabla \bullet (\mu(\nabla \mathbf{u}_b + \nabla \mathbf{u}_b^T)) + \rho \mathbf{g} \bullet \mathbf{n} - \frac{\partial p_I}{\partial n} \tag{4.45}
$$

- Similar to the two-phase flow model's boundary condition, the complementary velocity boundary condition follows Eqn. (4.44). A new boundary condition named `movingWallSwnese Velocity` is introduced to deal with the SWENSE body boundary condition.
- For the complementary pressure, `fixedFluxPressure` boundary condition is applied when the boundary does not move.
- When the boundary moves in time, the boundary condition of the complementary pressure has to follow Eqn. (4.45). However, homogeneous Neumann boundary condition (`zeroGradient`) is applied due to its simplicity.

4.9.2 Inlet & outlet wave boundary condition (Far-field condition)

For most naval applications in this thesis, the inlet and outlet boundaries are inside the relaxation zone. The inlet and outlet boundaries are assumed to be an undisturbed wave field due to the relaxation scheme. Therefore, the application of inlet & outlet boundary conditions can be considered an explicit relaxation with a weight of 1.0. Table 4.3 summarizes the inlet and the outlet boundary conditions in OpenFOAM notations for all variables.

Table 4.3. Inlet & outlet boundary conditions for all variables

	Boundary condition		**Boundary condition**
$\mathbf{u}$	waveVelocity	p_d	zeroGradient
$\mathbf{u}_C$	zero vector	p_C	zeroGradient
α	waveAlpha	ν_t	calculated
k	zeroGradient	ω	zeroGradient

- At the inlet and outlet boundaries, the total velocity is identical to the incident wave velocity. It is named `waveVelocity` condition. Similar to the total velocity, the incident wave elevation defines the VOF field at the boundary. It is `waveAlpha` condition. Even though the incident dynamic pressure is known, `zeroGradient` boundary condition is applied to the dynamic pressure.
- For the SWENSE model, the complementary flow is gradually diminished in the relaxation zone, and the complementary flow must disappear at the inlet and outlet boundaries. Therefore, zero velocity is applied for the complementary velocity and `zeroGradient` is applied for the complementary pressure. Since the VOF field is not decomposed, the same `waveAlpha` boundary condition is applied.
- The kinetic energy (k) and the specific dissipation rate (ω) use `zeroGradient` boundary condition not to block any turbulence energy at the inlet and outlet boundaries. The `calculated` boundary condition applies the eddy viscosity calculated from solved k and ω.

4.9.3 Open-air boundary condition

For most naval applications, the top boundary of the computational domain is commonly presented as an open-air boundary condition. Table 4.4 summarizes the open boundary conditions applied to each variable.

- Since the incident flow components are diminished at the top boundary, the boundary conditions for the total velocity and the complementary velocity are the same.
- For pressure, an approximation introduced at this boundary is that the total pressure p is constant and equal to the atmospheric pressure. The reference pressure p_0 is set to zero.

$$p = p_0 - 0.5\rho \left|\mathbf{u}\right|^2 \tag{4.46}$$

- A mixed boundary condition is applied for the velocity. When the flow at the boundary is inflow, zero normal gradient is imposed. When it comes to outflow, cell-center velocity is imposed on the boundary. In OpenFOAM notation, this boundary condition is named `pressureInletOutletVelocity`.
- The VOF uses the `inletOutlet` boundary condition, which normally works the same as `zeroGradient`. However, it switches to `fixedValue` if the velocity vector next to the boundary aims inside the domain.

- For turbulent kinetic energy (k) and specific dissipation rate (ω), `zeroGradient` boundary condition is used since no specific physical boundary condition is assumed for the open-air boundary.

Table 4.4. Open boundary conditions for all variables

	Boundary condition		**Boundary condition**
$\mathbf{u}$	pressureInletOutletVelocity	p_d	totalPressure
$\mathbf{u}_C$	pressureInletOutletVelocity	p_C	totalPressure
α	inletOutlet	ν_t	calculated
k	zeroGradient	ω	zeroGradient

4.9.4 Symmetry plane

When the flow fields are symmetric about a plane, the computational effort can be reduced by introducing a smaller domain accounting for only one side of the symmetry plane. This boundary condition is named `symmetryPlane`. The fluid velocity normal to the symmetry plane is zero, and the pressure gradient is also zero.

5 Interface treatment schemes

As introduced in Section 1.2.2, a careful choice of interface treatment schemes is essential for quality two-phase flow simulations. Previous studies investigated the interface treatment schemes and their interactions with existing time schemes on simple two-phase flow simulations such as wave propagation and slamming [46, 108, 109]. In this chapter, two VOF convection methods are considered: the algebraic VOF convection method with artificial compression and the geometric VOF convection method. Moreover, two interface conditions are compared. One uses averaged interface properties, and the other uses interface jump conditions, which is named Ghost Fluid Method (GFM) [110]. The following sections describe each interface treatment scheme and applications.

5.1 VOF interface convection algorithms

5.1.1 Algebraic VOF convection algorithm with Crank-Nicolson scheme

This section presents the algebraic VOF convection algorithm, which is embedded in the OpenFOAM solver interFoam (incompressible, Newtonian two-phase flow solver). Eqn. (5.1) expresses the FV representation of the VOF convection equation with the off-centered Crank Nicolson (CN) temporal discretization, where $C_{CN} = 1/(C_{oc}+1)$ and C_{oc} is the off-centering coefficient. Crank Nicolson coefficient $C_{CN} = 0.5$ yields the classical Crank Nicolson scheme, and $C_{CN} = 1$ yields the first-order Backward Euler scheme.

$$\begin{aligned}\frac{V_P^{(n+1)}\alpha_P^{(n+1)} - V_P^{(n)}\alpha_P^{(n)}}{h} + C_{CN}\sum_f (\phi_f\alpha_f + (\phi_{r,f}\alpha_f(1-\alpha_f))^{(n+1)} \\ +(1-C_{CN})\sum_f (\phi_f\alpha_f + (\phi_{r,f}\alpha_f(1-\alpha_f))^{(n)} = 0\end{aligned} \tag{5.1}$$

Eqn. (5.1) is separated into two equations to get a bounded and non-smeared VOF field, using the *Predictor-Corrector* algorithm: *Predictor* equation (Eqn. (5.2)) and *Corrector* equation (Eqn. (5.4)). The *Predictor* equation reads, where α^* is an intermediate phase fraction:

$$\frac{V_P^{(n+1)}\alpha_P^{*(n+1)} - V_P^{(n)}\alpha_P^{(n)}}{h} + \sum_f F_{BD}^{CN} = 0 \tag{5.2}$$

$$F_{BD}^{CN} = (\phi_f^{CN}\alpha_f^{*(n+1)})_{BD}, \qquad \phi_f^{CN} = C_{CN}\phi_f^{n+1} + (1-C_{CN})\phi_f^n \tag{5.3}$$

This *Predictor* equation uses a time-blended VOF flux, and it is discretized with the bounded (linear upwind) scheme. The time-blended flux (ϕ_f^{CN}) and the time-blended VOF flux (F_{BD}^{CN}) are shown in Eqn. (5.3). The subtraction of Eqn. (5.2) from Eqn. (5.1) yields the *Corrector* equation Eqn. (5.4), with the correction flux of Eqn. (5.5). α_P^{**} is the corrected phase fraction, and λ_f is the explicit limiter field evaluated from the MULES (Multidimensional Universal Limiter for Explicit Solver [105]) algorithm.

$$\frac{V_P^{(n+1)}\alpha_P^{**(n+1)} - V_P^{(n)}\alpha_P^{*(n+1)}}{h} + \lambda_f F_{Corr}^{CN} = 0 \tag{5.4}$$

$$\begin{aligned} F_{Corr}^{CN} = \Big[& C_{CN}(\phi_f \alpha_f^* + (\phi_{r,f}\alpha_f^*(1-\alpha_f^*))_{UB}^{(n+1)} \\ & +(1-C_{CN})(\phi_f\alpha_f + (\phi_{r,f}\alpha_f(1-\alpha_f))_{UB}^{(n)}\Big] - F_{BD}^{CN} \end{aligned} \tag{5.5}$$

After the correction of the VOF field with MULES iteration, Eqn. (5.6) updates the bounded flux and intermediate VOF. After the MULES iteration, the final VOF flux and VOF field are updated with Eqn. (5.7).

$$F_{BD}^{CN} = F_{BD}^{CN} + \lambda_f F_{Corr}^{CN} \qquad \alpha_P^* = \alpha_P^{**} \tag{5.6}$$

$$(\phi\alpha)_f^{CN} = F_{BD}^{CN} \qquad \alpha_P^{(n+1)} = \alpha_P^{**} \tag{5.7}$$

The VOF flux at time index $(n+1)$ is required to evaluate the mass flux at time $t^{(n+1)} = t^{(n)} + h$. The time-blended VOF flux ($(\phi\alpha)_f^{CN}$) has to be extrapolated from $t^{(CN)}$ to $t^{(n+1)}$ using Eqn. (5.8). After the extrapolation, the mass flux for the momentum equation is evaluated using Eqn. (5.9).

$$(\phi\alpha)_f^{t^{(n)}+h} = (1+C_{oc})(\phi\alpha)_f^{CN} - C_{oc}(\phi\alpha)_f^{t^{(n)}} \tag{5.8}$$

$$(\rho\phi)_f^{t^{(n)}+h} = (\rho_{water} - \rho_{air})(\phi\alpha)_f^{t^{(n)}+h} + \rho_{air}\phi_f^{t^{(n)}+h} \tag{5.9}$$

5.1.2 Geometric VOF convection

The algorithm of a geometric VOF scheme named *IsoAdvector* method is shortly presented in this subsection. Instead of solving the VOF convection equation in an algebraic way, Roenby et al. developed a new VOF convection method that constructs the iso-surface and moves it in a cell [31]. From the mass conservation equation, Eqn. (5.10) is valid for each cell i.

$$\alpha_i(t+h) = \alpha_i(t) - \frac{1}{V_i}\sum_{faces=F_j}\int_t^{t+h}\int_{F_j} H(\mathbf{x},\tau)\mathbf{u}(\mathbf{x},\tau)d\mathbf{S}d\tau\,. \tag{5.10}$$

where $H(\mathbf{x},t)$ is a continuous phase indicator function for the position vector $\mathbf{x}$, and it satisfies:

$$H(\mathbf{x},\tau) = \frac{\rho(\mathbf{x},\tau) - \rho_{air}}{\rho_{water} - \rho_{air}}\,, \qquad \alpha_i = \frac{1}{V_i}\int_{V_i} H(\mathbf{x},\tau)dV\,. \tag{5.11}$$

The physical meaning of the integration on the RHS of Eqn. (5.10) is that the volume of a phase moves across the face j of cell i during the time interval $[t, t+h]$.

$$\Delta V_{i,j}(t,h) = \int_t^{t+h}\int_{F_{i,j}} H(\mathbf{x},\tau)\mathbf{u}(\mathbf{x},\tau)d\mathbf{S}d\tau \tag{5.12}$$

It is assumed that the face velocity during the time interval is constant. The constant velocity is computed using linear interpolation between t and $t+h$. The constant velocity assumption simplifies Eqn. (5.12) into Eqn. (5.13).

$$\Delta V_{i,j}(t,h) \approx \frac{0.5(\phi_{i,j}(t)+\phi_{i,j}(t+h))}{|\mathbf{S}_{i,j}|}\int_t^{t+h} A_{i,j}(\tau)d\tau \tag{5.13}$$

$A_{i,j}(\tau)$ is the time series of the submerged area of each face j of cell i. The main computational burden of the *IsoAdvector* scheme is to calculate the time integration of $A_{i,j}(\tau)$ at a mesh level.

$$A_{i,j}(\tau) = \int_{F_{i,j}} H(\mathbf{x},\tau)d\mathbf{S}\,. \tag{5.14}$$

The *IsoAdvector* scheme constructs the initial iso-surface in cell i to calculate the time variation of submerged area. The constructed iso-surface moves with a constant velocity which permits to evaluate the change of submerged area at the mesh level. The detailed algorithms of iso-surface construction, iso-surface advection, and bounding procedures are well described in the *IsoAdvector* papers [31, 109]. The *IsoAdvector* scheme is published in GitHub [32] and also included in the solver OpenFOAM v2006 [19].

5.2 Interface conditions

In this section, the numerical treatment of the free-surface quantities is discussed.

5.2.1 Conditionally averaged two-phase flow model

The following equation recalls the averaged two-phase momentum equation.

$$\frac{\partial(\rho\mathbf{u})}{\partial t} + \nabla\bullet(\rho\mathbf{u}\mathbf{u}^T) - \nabla\bullet(\mu_E(\nabla\mathbf{u}+\nabla\mathbf{u}^T)) = -\nabla p_d - (\mathbf{g}\bullet\mathbf{x})\nabla\rho$$

In an averaged two-phase flow model, the velocity and the stress are continuous across the free-surface, though no specific condition is applied. This is because the quantities depending on the phase such as the density and the effective viscosity are averaged using the VOF field. However, the air tangential velocity just above the free-surface and the water tangential velocity just under the free surface are opposite in presence of propagating waves. The continuity should be ensured by a free-surface boundary layer. In practical naval hydrodynamics problems, the free-surface zone is never discretized enough to properly model this boundary layer. As a result, the averaged two-phase flow model gives a smoothed field representing the almost discontinuous free-surface properties, and smoothed fields can bring significant energy dissipation.

$$\rho = \alpha\rho_{water} + (1-\alpha)\rho_{air} \tag{5.15}$$

$$\mu_E = \alpha\mu_{water} + (1-\alpha)\mu_{air} + \mu_t \tag{5.16}$$

5.2.2 Ghost Fluid Method

With the Ghost Fluid Method (GFM), the two-phase flow is modeled as two single-phase flows, by considering discontinuous density across the interface. This model allows for discontinuous dynamic pressure at the free-surface by using boundary conditions. Those conditions reduce to three jump conditions of quantity at the free-surface. The operator [...] stands for the jump at the free-surface.

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla_{\bullet}(\mathbf{u}\mathbf{u}^T) - \nabla_{\bullet}(\nu_E(\nabla \mathbf{u} + \nabla \mathbf{u}^T)) = -\frac{1}{\rho}\nabla p_d \tag{5.17}$$

- Density jump condition.

$$[\rho] = \rho_{air} - \rho_{water} \tag{5.18}$$

- Pressure jump and dynamic pressure jump condition.

$$[p] = 0, \qquad [p_d] = -[\rho](\mathbf{g}_{\bullet}\mathbf{x}) \tag{5.19}$$

- Dynamic Pressure gradient jump condition.

$$\left[\frac{1}{\rho}\nabla p_d\right] = 0 \tag{5.20}$$

These three jump conditions are applied to the interface cells using the discretization scheme, applying extrapolation of the discontinuous density and dynamic pressure. The interface cells are picked selecting the cells with non-zero or -one VOF values. The discretization of momentum equation and jump conditions is not further described in this thesis and the interested reader is referred to the PhD thesis of Vukĉević [25].

5.3 Wave propagation in periodic domain

Table 5.1 presents four different combinations of solvers based on the mathematical models presented in the previous sections. For convenience, each combination of the solver is given a name: *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively.

Table 5.1. Definition of the four different combinations of solvers

Solvers	VOF convection	Interface condition
interFoam	Compressive VOF	Averaged two-phase
interFlow	*IsoAdvector*	Averaged two-phase
interFoamGFM	Compressive VOF	GFM
interFlowGFM	*IsoAdvector*	GFM

A non-linear wave propagating towards positive x-direction in a two-dimensional domain is considered to investigate the ability of each solver in view of naval applications. The initial wave

velocity field and the free-surface positions are evaluated with the stream function wave theory [76]. Table 5.2 tabulates the non-linear wave condition, the same also used in the previous studies [89, 96].

The computational domain is precisely one wavelength in the x-direction. Figure 5.1 shows the computational domain, the boundary conditions, and the initial VOF field. Periodic boundary condition is applied at the 'Inlet' and 'Outlet' boundaries. Free-slip condition is applied at the 'Bottom' boundary and open-air boundary condition is applied at the 'Air' boundary. One hundred wave probes are installed uniformly from the 'Inlet' to 'Outlet' boundaries to measure the waves.

Table 5.2. Wave condition

Item	Unit	Value
Depth (D)	[m]	0.6
Period (T)	[s]	0.7018
Height (H)	[m]	0.05753
Wave length (λ)	[m]	0.8082
Wave steepness (kA)	-	0.24

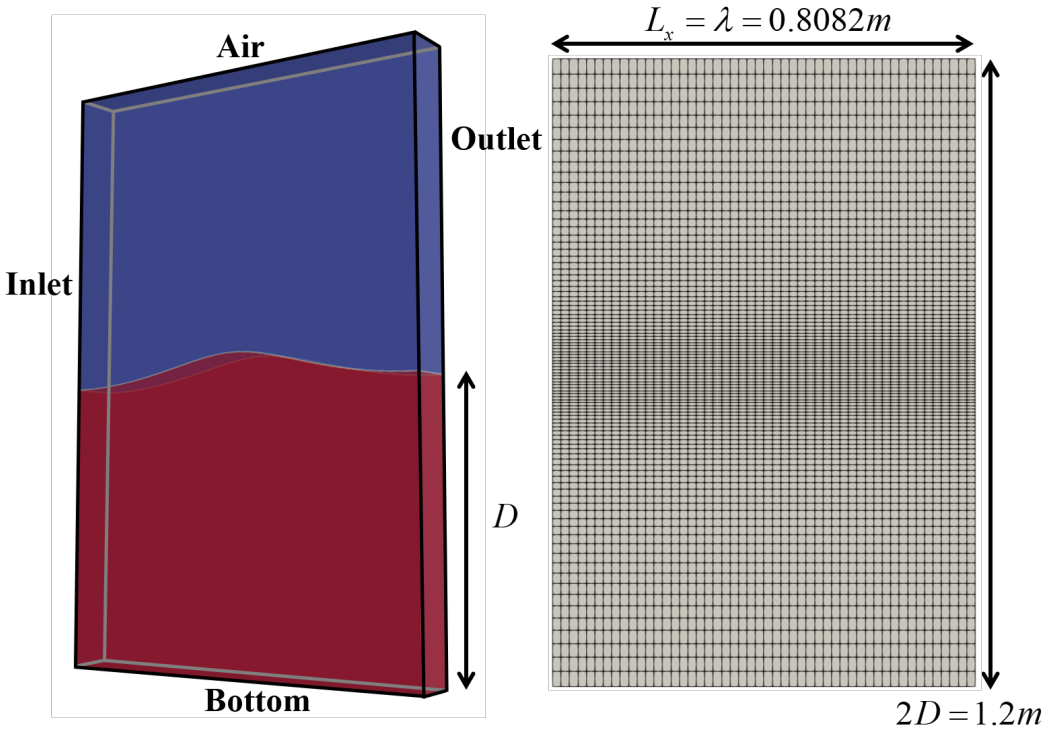


Figure 5.1. Computational domain and boundary conditions

The second-order Crank-Nicolson temporal discretization scheme is used for all four solvers.

The solvers *interFoam* and *interFlow* are used with Crank-Nicolson off-centering parameter $C_{oc} = 0.95$, for a better stability of the simulation. In contrast, solvers with GFM are run without off-centering ($C_{oc} = 1.0$).

The spatial discretization schemes applied for *interFoam* are listed in Table 4.1. The GFM uses a specialized second-order spatial discretization scheme to consider the jump conditions at the free-surface. *InterFlow* (*IsoAdvector*) does not require any spatial discretization scheme, while the assumption of constant velocity field between $[t, t+h]$ affects the accuracy and temporal order of convergence. As presented in Eqn. (5.13), the average of the velocities at t and at $t + h$ is used.

For all simulations, eight outer iterations (PIMPLE) and two inner iterations (PISO) are used to minimize the temporal error. With given number of non-linear iterations, the initial residuals of the velocity and the pressure at the final non-linear iteration were lower than 10^{-7}.

Table 5.3 tabulates a simulation campaign with five different grids with similar Courant numbers to check the convergence order of each solver. Five grid resolutions from *Grid 1* to *Grid 5* have the same cell aspect ratio, and the ratio between time-step size (h) and cell size is identical for all cases. The refinement ratio is $r = 2$ for each level of the grid except for the coarsest grid. The uncertainty assessment has been performed with the open-access tool developed by Eça and Hoekstra [111], providing the order of convergence P and the uncertainty U.

Table 5.3. Mesh for the convergence study (Initial Courant number ≈ 0.17)

Case	$\lambda/\Delta x$	$H/\Delta z$	T/h	$\Delta z/\Delta x$
Grid 1	15	3	50	0.356
Grid 2	25	5	100	0.356
Grid 3	50	10	200	0.356
Grid 4	100	20	400	0.356
Grid 5	200	40	800	0.356

Figure 5.2 and 5.3 show the wave profiles captured at times $t = 10T, t = 20T, t = 30T$, and $t = 40T$ with four different types of solvers and four computational resolutions from *Grid 2* to *Grid 5*. The solid black line is the wave profile from the stream function wave theory. For *Grid 2*, the *interFoam* solver shows large damping and small wiggling of the free-surface. The *interFoam* solver shows apparent convergence of the wave profile from *Grid 2* to *Grid 5*. The solver *interFlow* showed instability for resolutions *Grid 2* and *Grid 3*. For *Grid 4* and *Grid 5*, the wave profile is accurate until 10 wave periods, but the phase velocity has increased. Both solvers *interFoamGFM* and *interFlowGFM* showed very similar wave profiles for *Grid 3*, *Grid 4*, and *Grid 5*. The use of the GFM scheme led to a significant and continuous decrease of wave amplitude in time, even with *Grid 5*.

Figure 5.4 presents the convergence of time-averaged first harmonic wave amplitude and time-averaged phase difference with respect to the refinement levels. The entire computation is

separated into four intervals, and each interval has a duration of ten wave periods.

interFlow gives better conservation of amplitude than *interFoam* for *Grid 3*, *Grid 4*, and *Grid 5*. However, for those cases, *interFlow* generates oscillation of the free-surface and larger phase differences in time than *interFoam*. For the high refinement levels (*Grid 4* and *Grid 5*), the phase differences of *interFlow* solver is very small for first ten periods and gradually increase in time. *interFoam* solver shows the best preservation of phase difference for the entire range of refinement levels.

The solvers using the GFM algorithm (*interFoamGFM* and *interFlowGFM*) show a large and continuous dissipation of wave amplitude in time even with *Grid 4* and *Grid 5* refinement levels. Both *interFoamGFM* and *interFlowGFM* solvers also show the continuous decrease of phase differences.

The orders of convergence (P) and uncertainties (U) calculated from the uncertainty assessment tool are listed in Table 5.4. The uncertainty assessment was performed for the wave amplitude only. The averaged first-harmonic amplitude from the solvers *interFoam* and *interFlow* has an order of convergence larger than one, and during the first ten periods, second-order convergence is achieved. The solvers using the GFM algorithm (*interFoamGFM* and *interFlowGFM*) have a lower order of convergence than the solvers *interFoam* and *interFlow*.

In conclusion, considering that the refinement level *Grid 4* is the refinement usually acceptable for practical naval applications in view of computational cost, *interFoam* was not the most accurate solver but the most stable and predictable solver. *interFlow* is a very accurate solver for a short duration simulation, but it suffers large instability for a longer simulation, which requires further investigations. Also, further research is required for the *interFoamGFM* and *interFlowGFM* solvers.

Table 5.4. Summary of orders of convergence and uncertainties of harmonic wave amplitude

Time interval	*interFoam*	*interFlow*	*interFoamGFM*	*interFlowGFM*
Periods $1-10$	$P = 2.0$ $U = 0.4\%$	$P = 2.0$ $U = 0.7\%$	$P = 1.2$ $U = 1.0\%$	$P = 1.1$ $U = 1.2\%$
Periods $11-20$	$P = 2.0$ $U = 1.3\%$	$P = 1.8$ $U = 3.9\%$	$P = 0.9$ $U = 6.1\%$	$P = 0.9$ $U = 5.5\%$
Periods $21-30$	$P = 1.5$ $U = 2.6\%$	$P = 1.8$ $U = 4.8\%$	$P = 0.6$ $U = 22.8\%$	$P = 0.5$ $U = 22.5\%$
Periods $31-40$	$P = 1.2$ $U = 6.4\%$	$P = 1.8$ $U = 11.1\%$	–	$P = 0.5$ $U = 30.6\%$

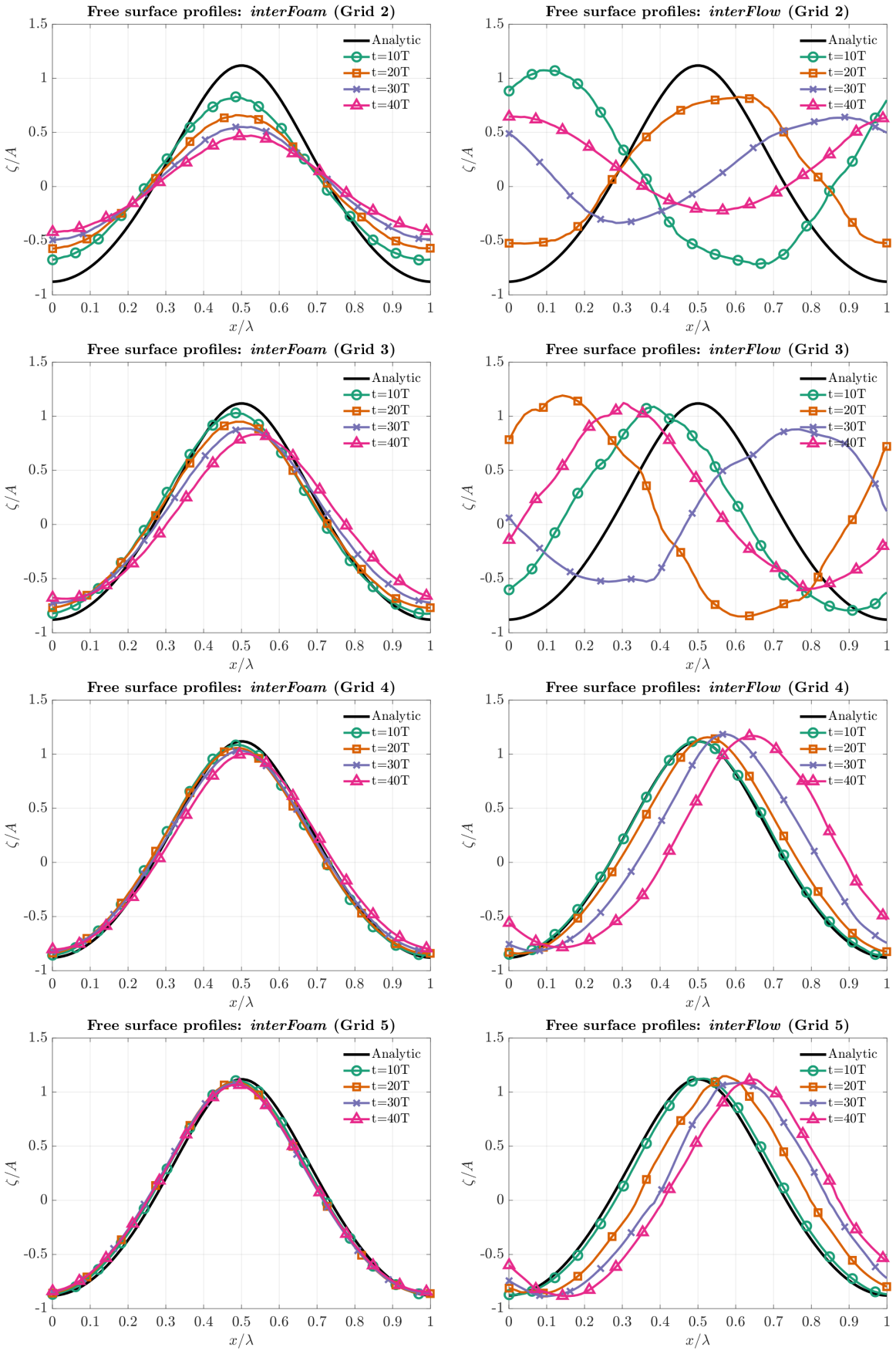


Figure 5.2. Free-surface profiles at times $t = 10T, t = 20T, t = 30T$ and $t = 40T$ for the solvers *interFoam* and *interFlow*

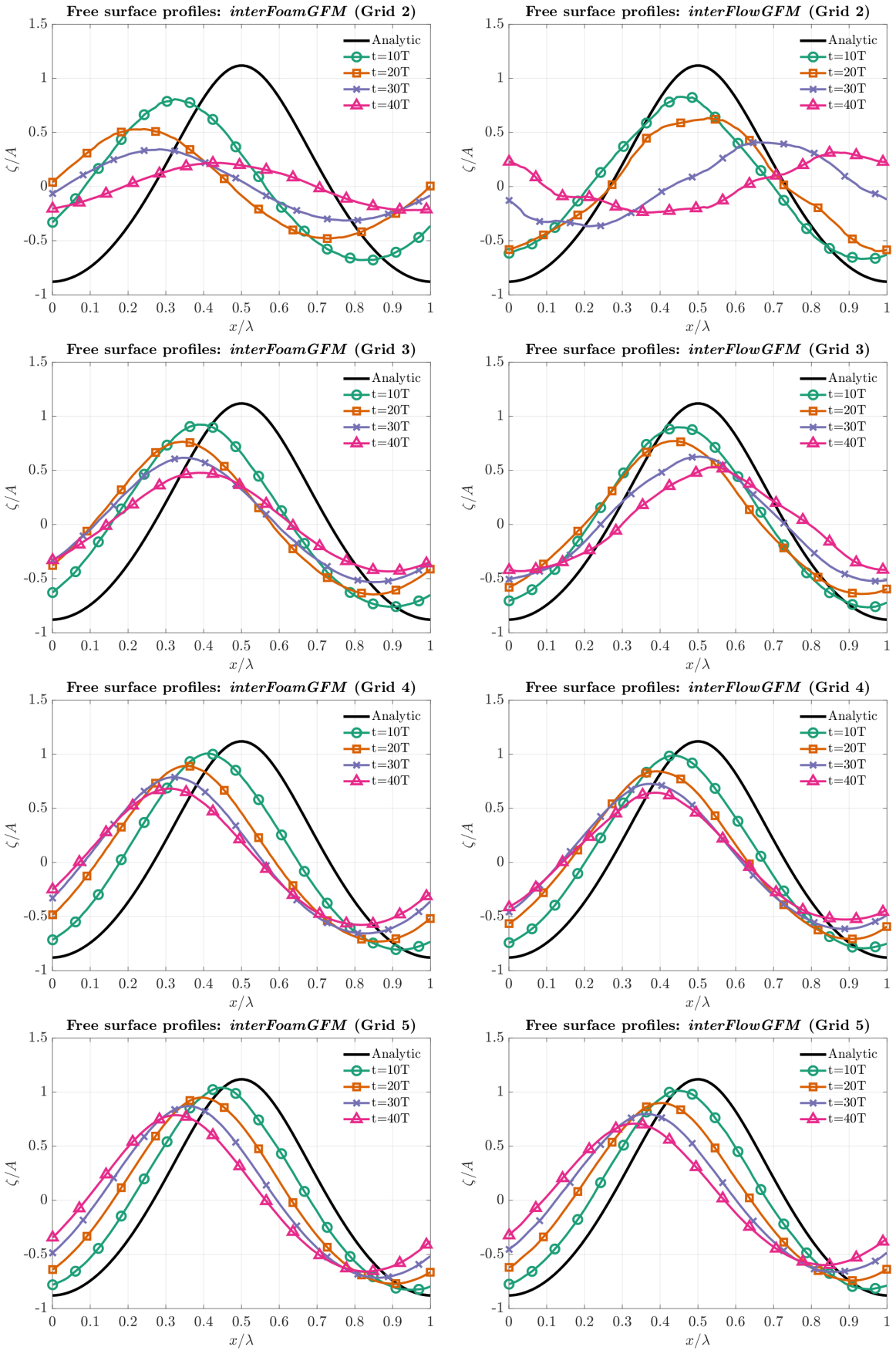


Figure 5.3. Free-surface profiles at times $t = 10T, t = 20T, t = 30T$ and $t = 40T$ for the solvers *interFoamGFM* and *interFlowGFM*

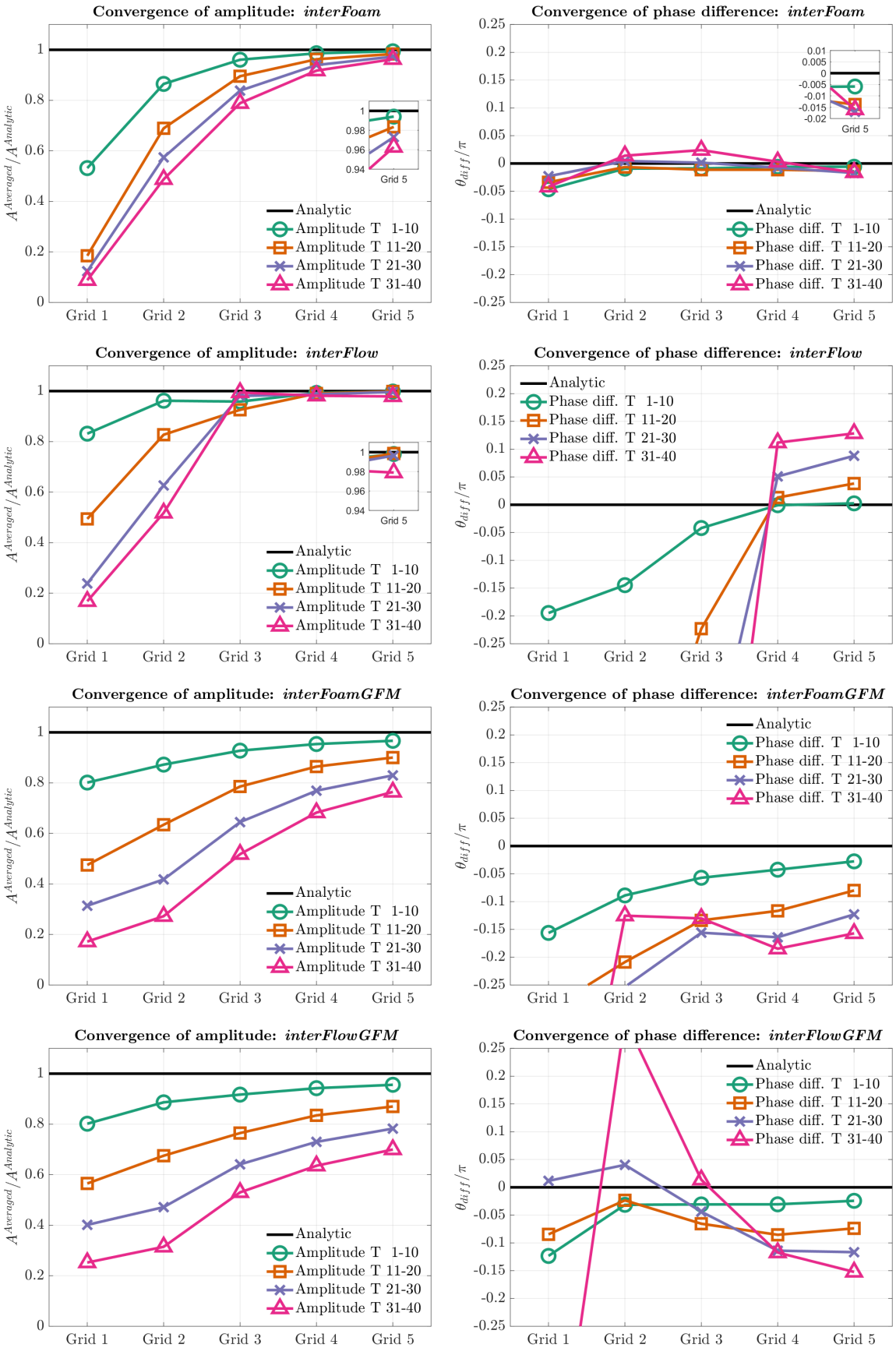


Figure 5.4. Convergence of moving window FFT first-harmonic amplitude and phase difference with respect to spatial and temporal resolutions.

5.4 Impact applications

The vessels operating in the ocean face various types of loads, and the slamming is one of the extreme loads that can damage the hull. Slamming is a water impact between the bottom of the ship and the sea surface, which typically occurs when large pitch motion is induced by steep wave conditions. The stern and the bow flare of the ship are the most influenced sections by slamming. To study this problem the complicated 3D ship sections are often simplified to 2D sections, and numerous experimental and numerical works on 2D slamming problems have been performed. Especially in the CFD field, 2D slamming applications are widely used to check the capability of codes and to validate them, see, e.g., [5, 112, 113]. During slamming, the dynamics of the body is highly dependent on the jet flow near the body and the jet detachment. Therefore, accurate estimation of these flow features is important. In the present section, two 2D slamming test cases are studied, which present jet flow and jet detachment near the body. The first one is an asymmetric wedge entry problem which requires accurate angular momentum from the jet flow. The second application is a wedge entry-exit problem which includes jet detachment during the simulation. Again, the four different solvers listed in Table 5.1 are tested and their capabilities are compared.

5.4.1 Asymmetric wedge entry problem

The first studied impact application is an asymmetric wedge water entry problem. The 5° inclined wedge is dropped 0.61m above the free-surface, and the wedge is free in heave and pitch.

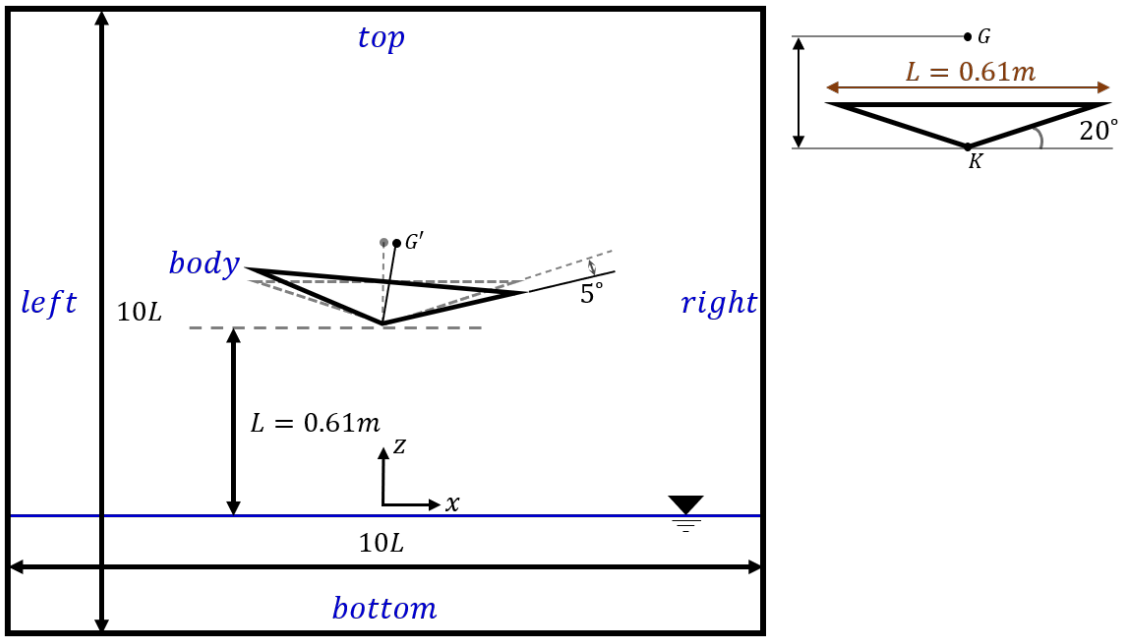


Figure 5.5. Asymmetric Wedge Entry setup description

The asymmetric wedge dynamics is purely dependent on the wedge pressure distribution and the

body gravity force. Therefore, this simulation campaign is suitable to estimate the accuracy of the four considered solvers in impact cases.

The results of heave and pitch acceleration of the wedge are compared to experimental data given by Xu et al. [114] and to the SPH simulation result by Oger et al. [6]. The experimental test section description and the inertia properties are provided in Figure 5.5 and Table 5.5. Two weight conditions, lightweight and mediumweight, are considered. For both weight conditions, the x length of the wedge is 0.61m, and the y length is 2.44m. At $t = 0$s, the 20° wedge is dropped from 0.61m above the free surface. The wedge is initially inclined 5° clockwise, where the keel of the wedge is a temporal center of rotation. The position of the center of gravity (G) changes to G' with this initial angle, and the G' is thus the center of rotation during the simulation.

Table 5.5. Experimental input parameters

Conditions	Unit	lightweight	mediumweight
y-direction length	m	2.44	
Angle	-	5°	
Weight	kg	124	293
Inertia	kg·m^2	8.85	10.95
KG	m	0.216	0.165

Figure 5.5 also provides the dimensions of the computational domain and the name of each boundary. The entire mesh moves with the body motions, including vertical motion and rotation, for better computational efficiency, without any grid morphing. Moving wall velocity boundary condition is applied to the 'body' boundary, and zero velocity boundary condition is applied to the 'left', 'right', and 'bottom' boundaries. Open-air boundary condition is applied to the 'top' boundary. Other variables, such as phase fraction and dynamic pressure, follow the boundary conditions described in Section 4.9.

Table 5.6 tabulates the simulation parameters, and Figure 5.6 illustrates the computational 'Fine' mesh. The mesh is refined more near the wedge. The smallest cell near the wedge has a boundary layer thickness of 0.001m for the Fine mesh. These simulations used the fixed time-steps listed in Table 5.6. The first-order implicit Euler scheme is applied for all solvers. The order of time scheme is different from the wave propagation test for the better stability of the simulations. This is accepted since the targeting simulation is very short and the time-step size is very small. The spatial discretization schemes for the four solvers are identical to the ones used for the wave propagation campaign. For both mesh refinement, eight outer iterations (PIMPLE) and two inner iterations (PISO) are used to minimize the temporal error. Even if turbulent effects are expected to be negligible for this impact phenomenon, the standard k-ω SST model is applied, this in view of naval applications where slamming impacts and ship produced turbulence will occur at the same time.

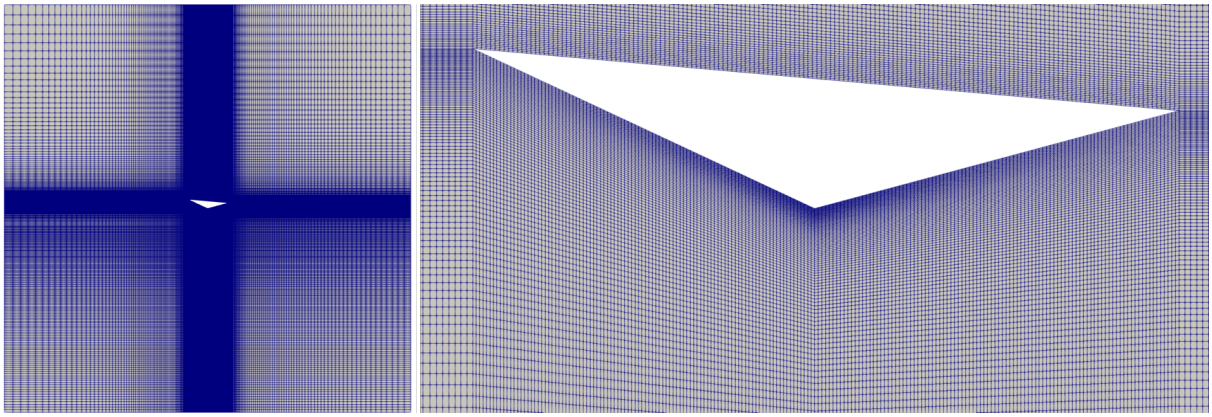


Figure 5.6. Computational mesh for the asymmetric Wedge Entry problem: Full grid (left) and the grid near the wedge (right).

Table 5.6. Mesh resolutions used for the asymmetric Wedge Entry problem

Refinement	Medium	Fine
Number of cells	42038	118878
time-step size (sec)	0.0001	0.00005

The following describes the results of the asymmetric Wedge Entry simulations with two weight conditions, two mesh resolutions, and four solvers with different interface treatment schemes. Figure 5.8 and Figure 5.9 compare the heave and pitch accelerations obtained numerically with the experiment for the lightweight condition and the mediumweight condition, respectively. The *interFoam* solver shows best correspondence with the experiment, but the general trend of the accelerations simulated with all solvers is similar to that of the experiment. The oscillation of the experimental data is due to the structural oscillation caused by the impact. The *interFlow* solver shows a small high-frequency oscillation of the accelerations; however, this is not big enough to change the global motion of the wedge. This instability requires further investigation. The impact time of GFM solvers is slightly delayed with respect to the other two.

Looking at a zoom of the free-surface predicted by the VOF function and the pressure field, pictured around the impact time in Figure 5.7 for two solvers, with lightweight condition, with and without GFM, it appears that the free-surface is modified already before the wedge touches it with the GFM method, explaining the time shift observed for the GFM results in Figures 5.8-5.9.

For a more detailed investigation of the whole impact phase, the VOF predicted free-surface and the pressure field near the free-surface simulated with the Fine mesh are presented for several time instants, in Figures 5.10 to 5.11 for the lightweight condition and in Figures 5.12 to 5.13 for the mediumweight one. In these figures each column shows the results from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Comparing the results of the various solvers, the pressure fields are not very different apart for the time lag already commented. However,

there is a significant difference in the jet shape computed by the four solvers.

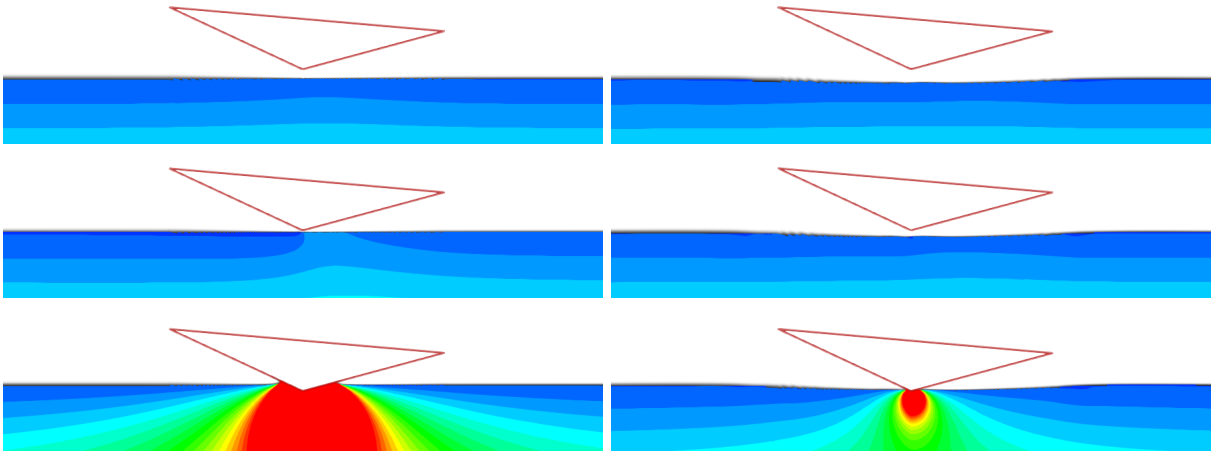


Figure 5.7. Lightweight condition: VOF predicted free-surface and pressure field for the *interFoam* (left) and *interFoamGFM* (right) solvers. Each row is made of snapshots at times $t = 0.35$s, $t = 0.355$s, and $t = 0.36$s

Figures 5.14 to 5.16 compare the VOF predicted free surface and the pressure field near the wedge from *interFoam* to the (rather old) SPH result from Oger et al. [6]. Only the lightweight condition is compared. The figures share the same pressure range from -1000Pa to 10000Pa. The *interFoam* pressure field is similar to the SPH one, though it is more continuous due to the incompressible flow assumption, the acoustic waves present in the SPH simulation being neglected in *interFoam*.

Besides, in Figure 5.16 especially, one can observe that the use of VOF leads to an expected smearing of the water jets compared to SPH, even though their shapes remain very comparable at these late times of the impact. Moreover, comparing back the jet shapes predicted by the four solvers in Figures 5.10 to 5.11, it can be concluded that on this test case, *interFoam* and *interFlow* clearly outperform than the two other solvers in terms of free-surface prediction. Considering the high frequency oscillation of body acceleration by *interFlow*, *interFoam* is the most promising solver for now.

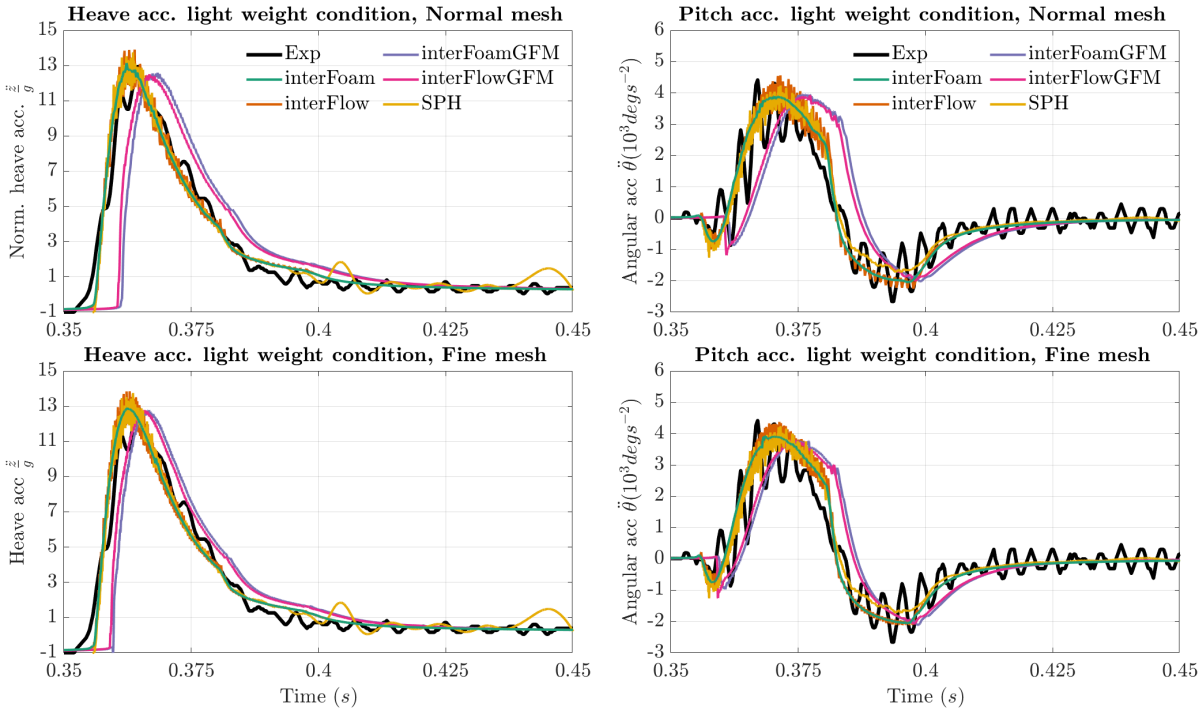


Figure 5.8. Heave and pitch acceleration with lightweight condition and the two mesh resolutions. The four OpenFOAM solvers are compared with the experiment and the SPH result.

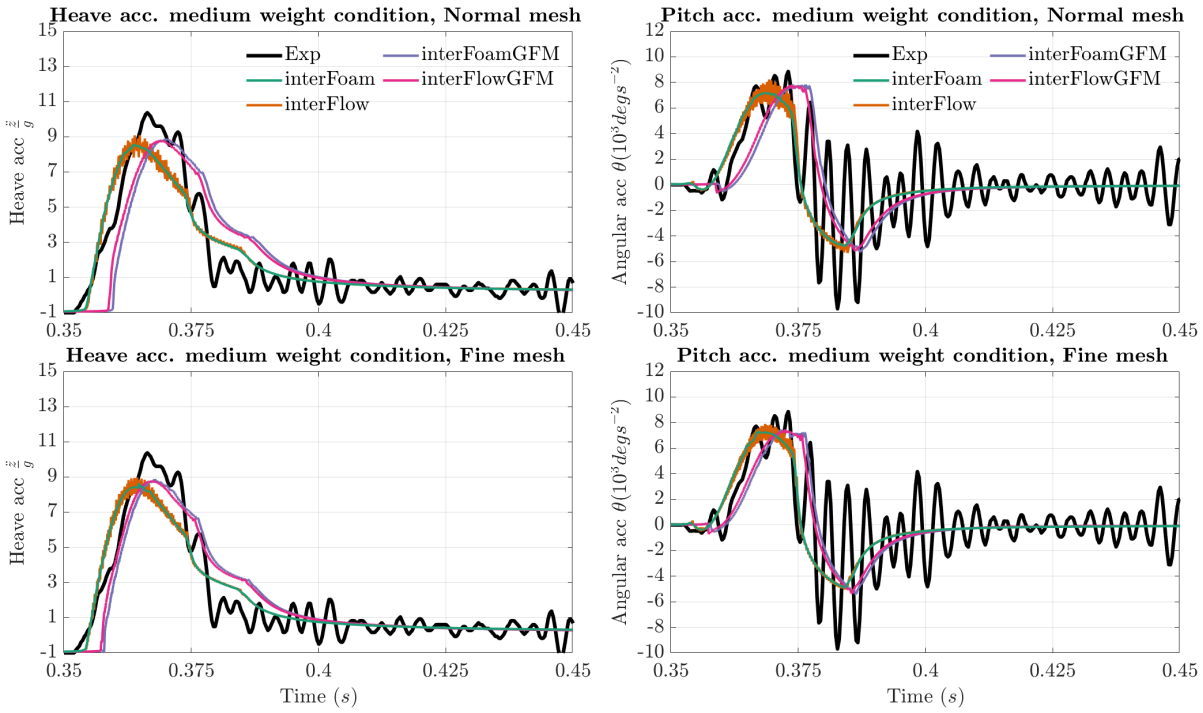


Figure 5.9. Heave and pitch acceleration with mediumweight condition and different mesh resolutions. The four OpenFOAM solvers are compared with the experiment result.

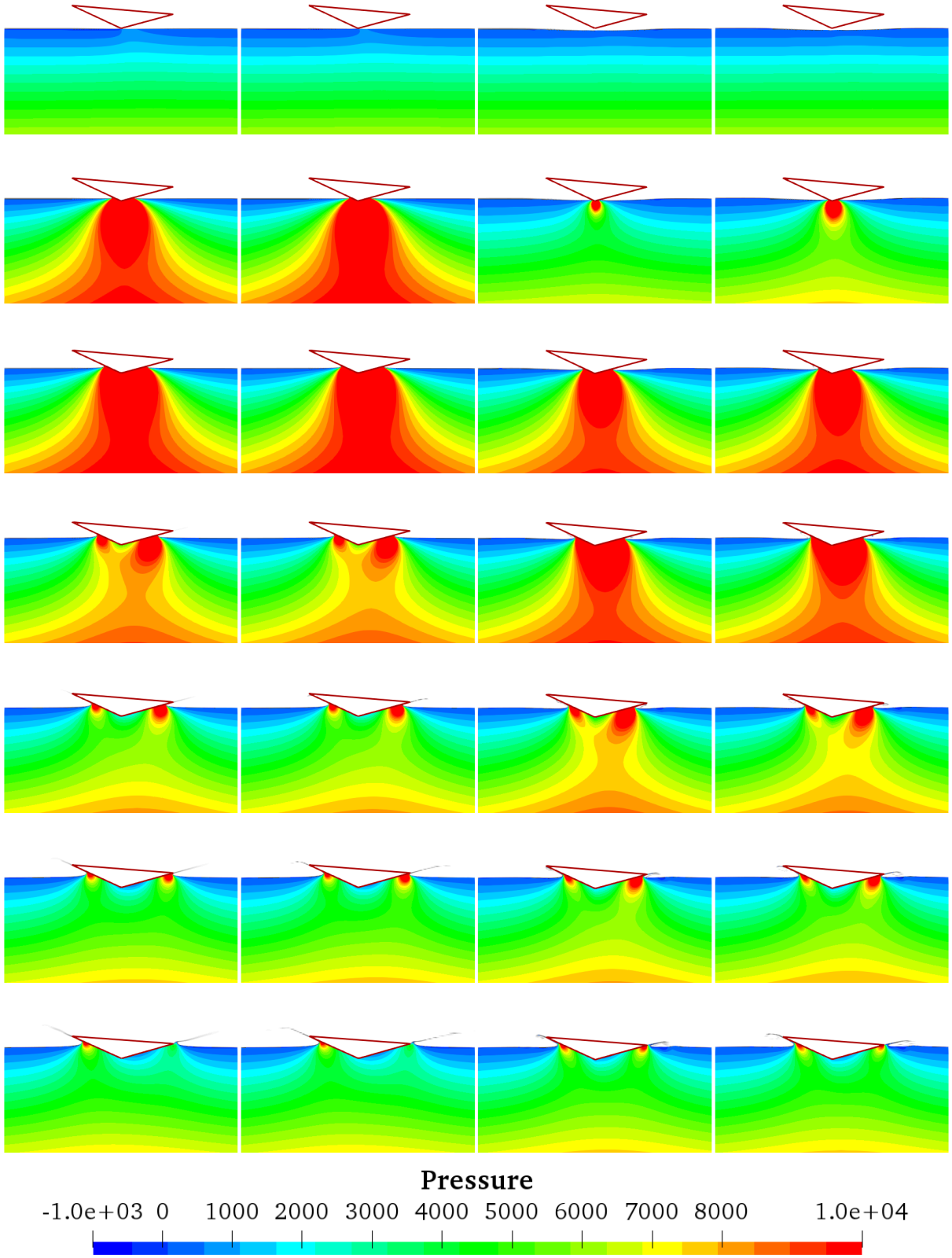


Figure 5.10. Lightweight condition: Pressure field (N/m^2) and free surface obtained with the four different solvers. The columns show from left to right the results from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is made of snapshots at times $t = 0.355$s, $t = 0.36$s, $t = 0.365$s, $t = 0.37$s, $t = 0.375$s, $t = 0.38$s, and $t = 0.385$s.

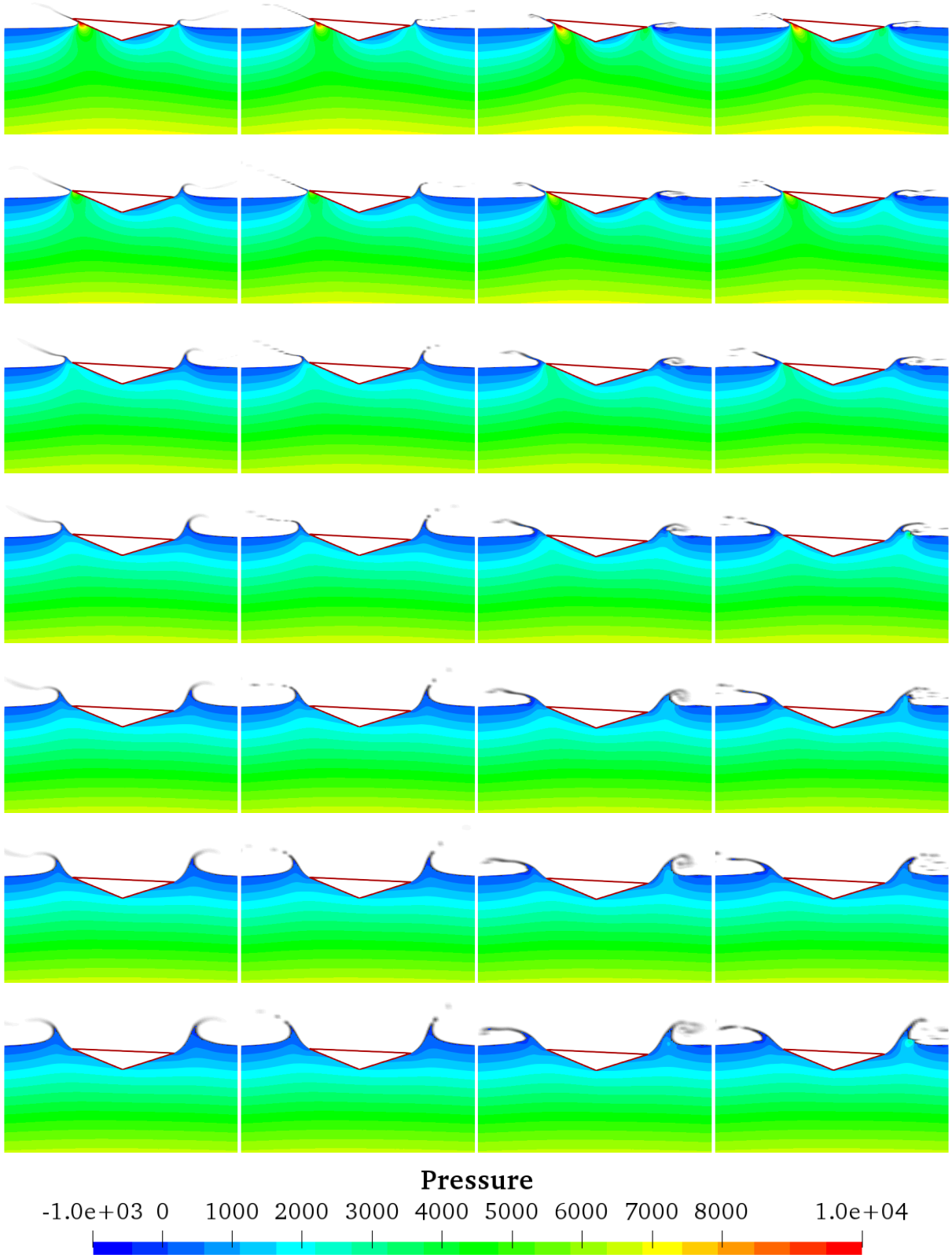


Figure 5.11. Lightweight condition: Pressure field (N/m^2) and free surface obtained with the four different solvers. The columns show from left to right the results from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is made of snapshots at times $t = 0.39$s, $t = 0.40$s, $t = 0.41$s, $t = 0.42$s, $t = 0.43$s, $t = 0.44$s, and $t = 0.45$s.

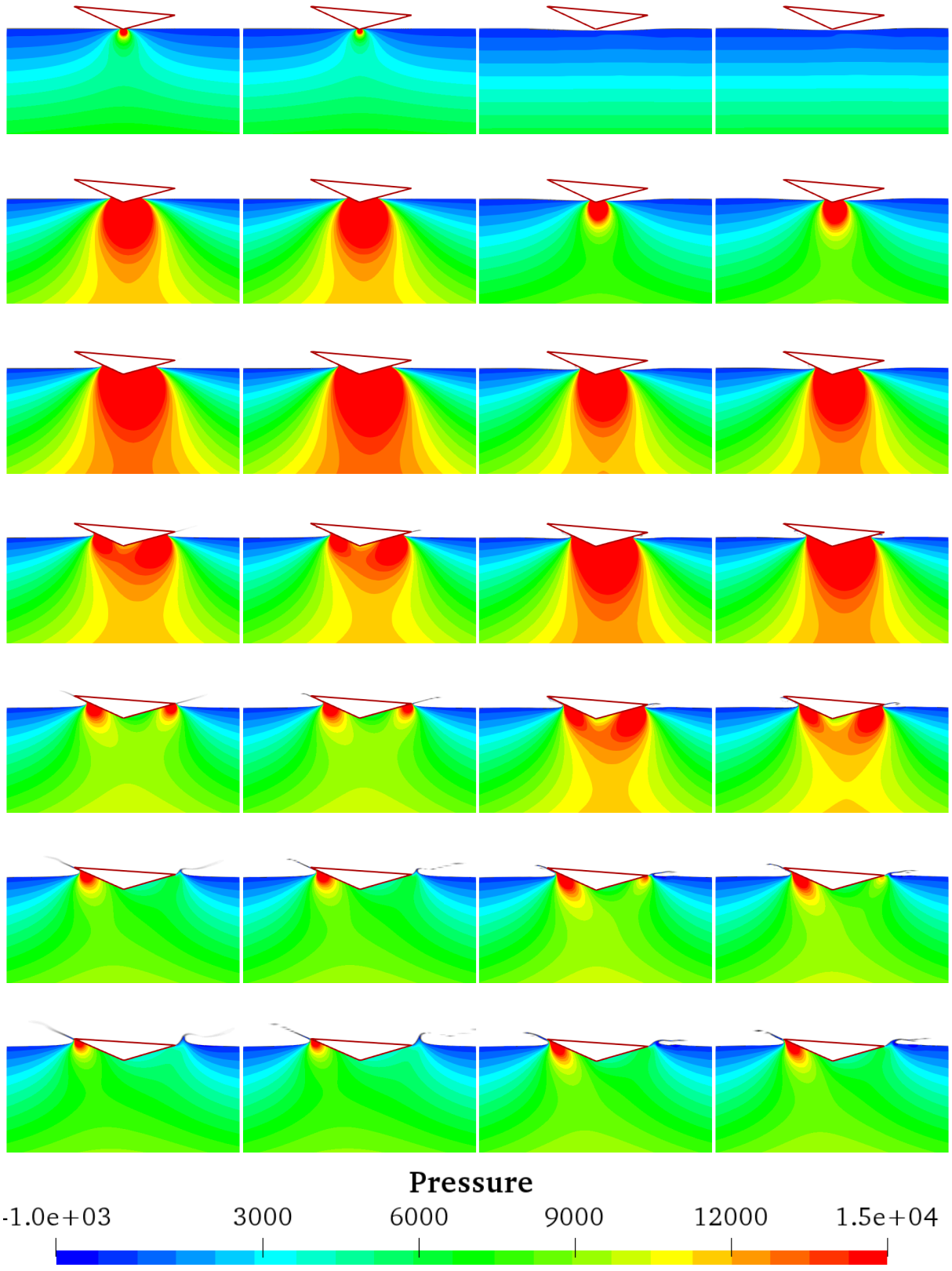


Figure 5.12. Mediumweight condition: Pressure field (N/m^2) and free surface obtained with the four different solvers. The columns show from left to right the results from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is made of snapshots at times $t = 0.355$s, $t = 0.36$s, $t = 0.365$s, $t = 0.37$s, $t = 0.375$s, $t = 0.38$s, and $t = 0.385$s.

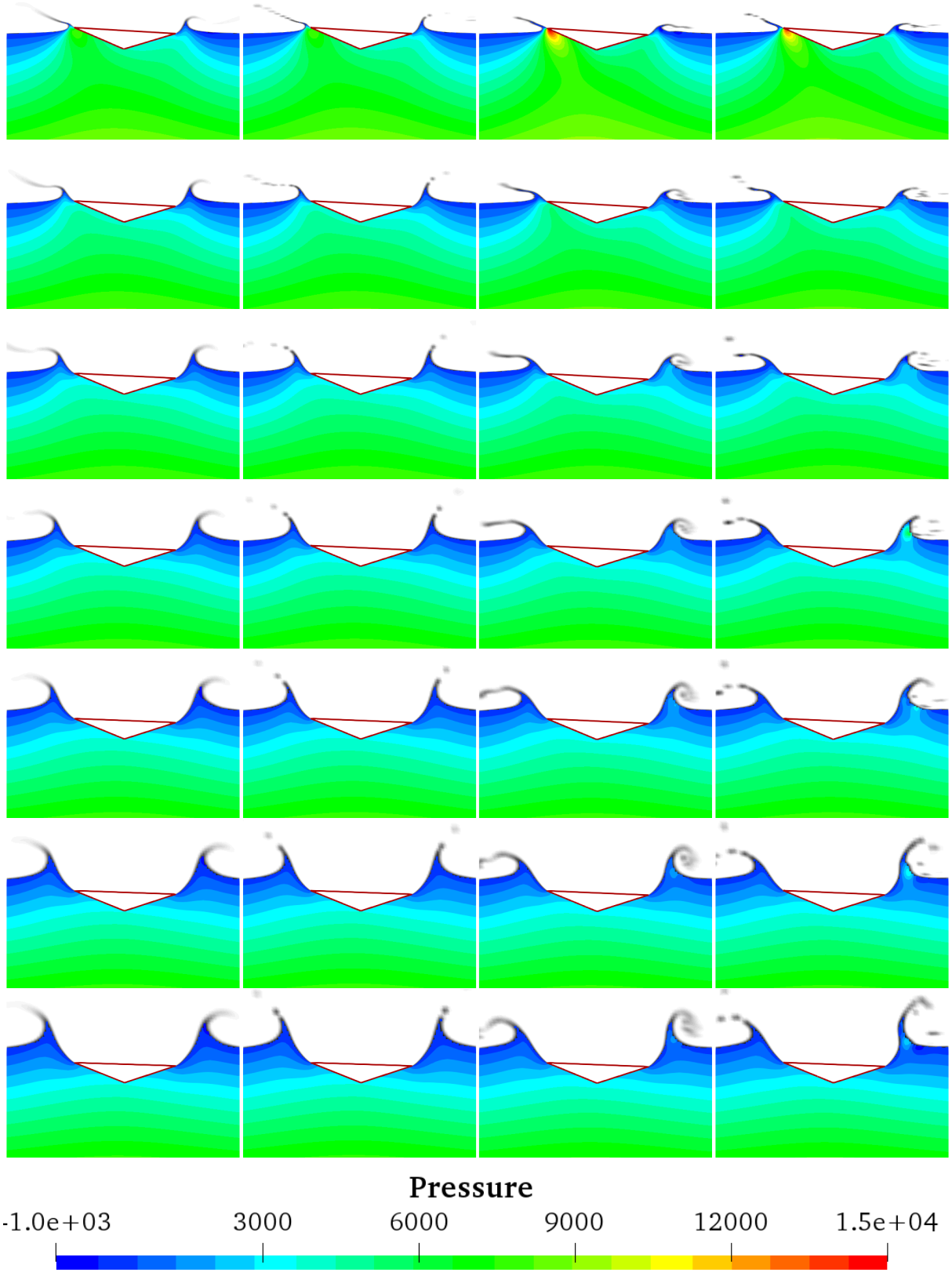


Figure 5.13. Mediumweight condition: Pressure field (N/m^2) and free surface obtained with the four different solvers. The columns show from left to right the results from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is made of snapshots at times $t = 0.39$s, $t = 0.40$s, $t = 0.41$s, $t = 0.42$s, $t = 0.43$s, $t = 0.44$s, and $t = 0.45$s.

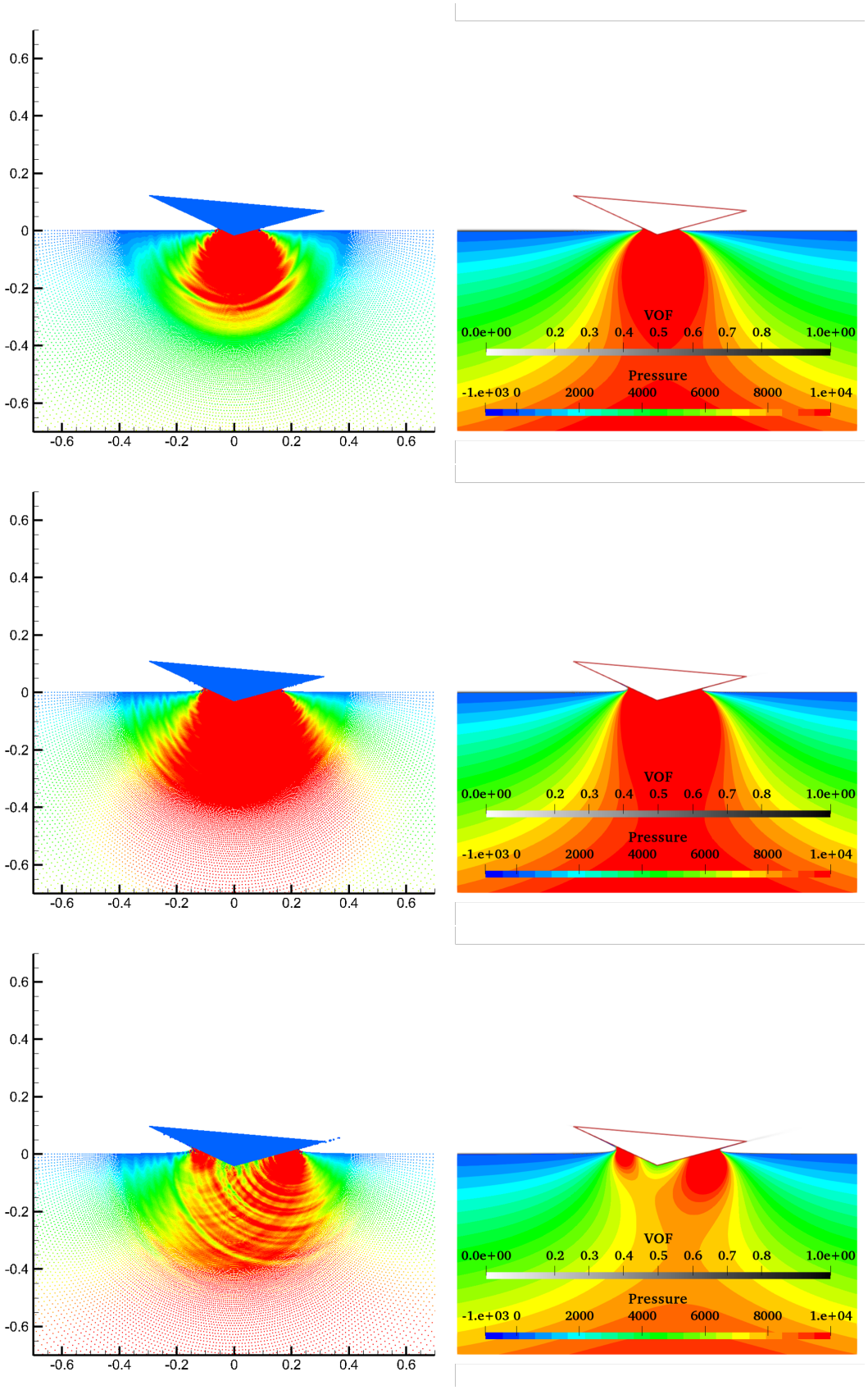


Figure 5.14. Lightweight condition: Pressure field (N/m^2) and free surface comparison between *interFoam* and SPH [6]. Each row is made of snapshots at times $t = 0.36$s, $t = 0.365$s, and $t = 0.37$s.

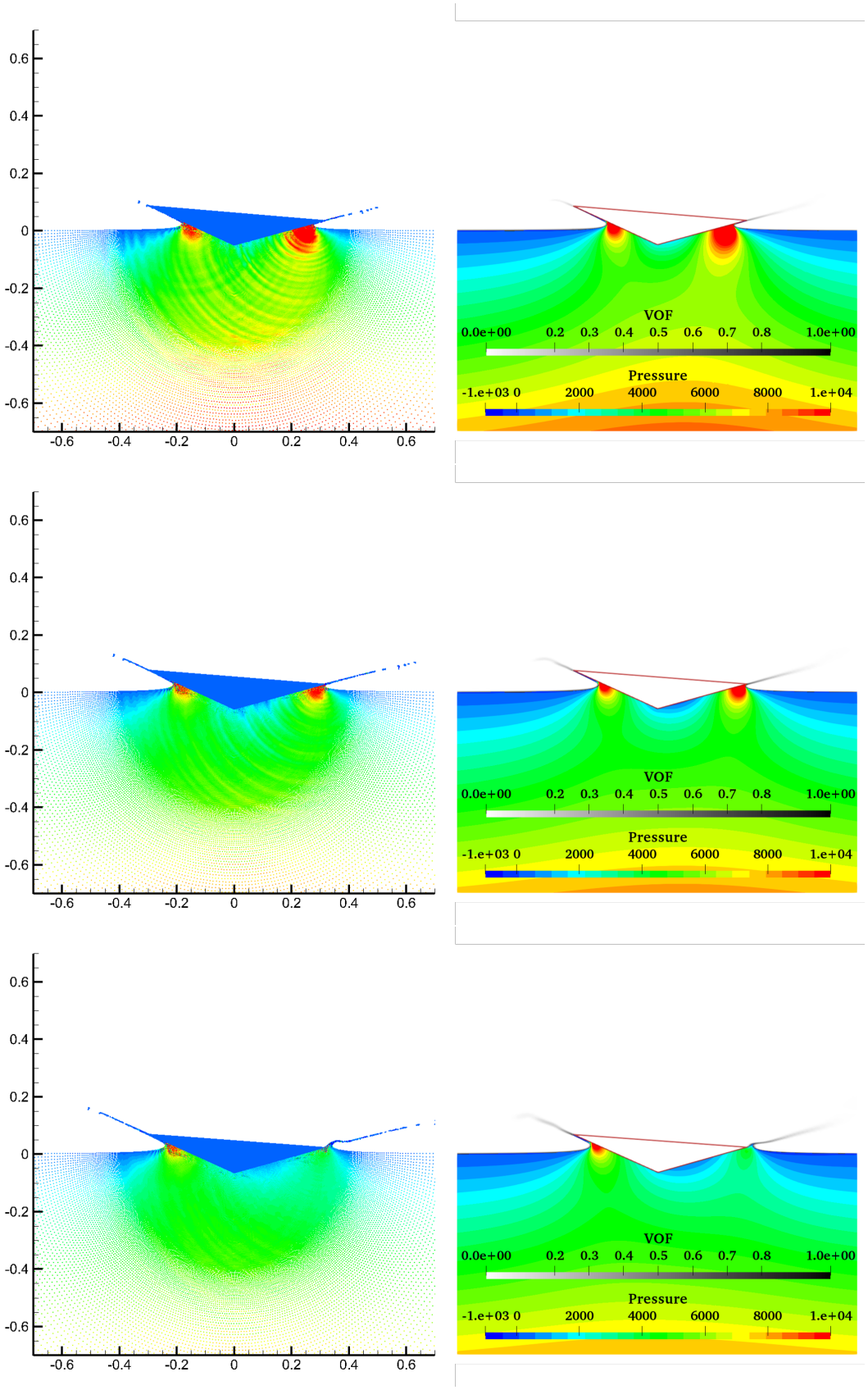


Figure 5.15. Lightweight condition: Pressure field (N/m^2) and free surface comparison between *interFoam* and SPH [6]. Each row is made of snapshots at times $t = 0.375$s, $t = 0.38$s, and $t = 0.385$s.

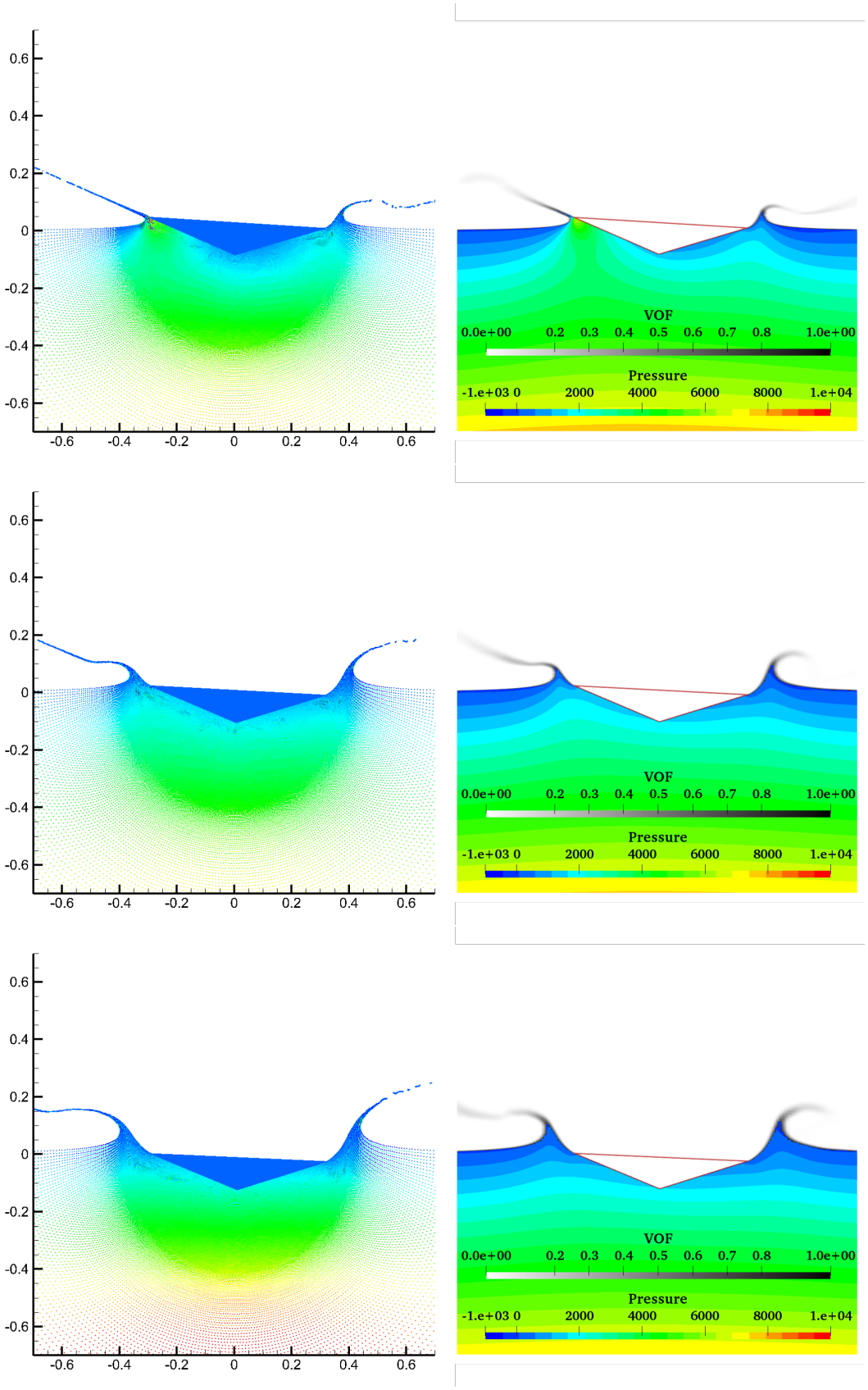


Figure 5.16. Lightweight condition: Pressure field (N/m^2) and free surface comparison between *interFoam* and SPH [6]. Each row is made of snapshots at times $t = 0.40$s, $t = 0.42$s, and $t = 0.44$s.

5.4.2 Wedge entry-exit problem

The second impact application is a wedge water entry and exit problem. Figure 5.17 gives a general description of the problem. The length of the slope is $L = 0.5$m and the length of the upper box is $B = 2L/\sin(10°)$. The test section is a rigid two-dimensional 10° wedge, and it follows the numerical simulation done by Piro and Maki [115]. At $t = 0$s the keel of the wedge is touching the free-surface, and the wedge moves with a prescribed vertical motion given in Eqn. (5.21) where A is a constant positive vertical acceleration and $-V_0$ is the negative vertical impact velocity at $t = 0$s. With this law, in a first stage, the wedge gets down with decreasing speed, up to a time noted t_0 in Eqn. (5.21) when it reaches maximal immersion and zero speed. In a second stage, after this time, the wedge gets up with increasing speed. The computational domain is 5m ($10L$) high in z-direction and 2.5m ($5L$) wide in x-direction. The boundary conditions are similar to those of the asymmetric wedge entry problem. However, in the current test case, only half of the physical domain is simulated by using a symmetric boundary condition.

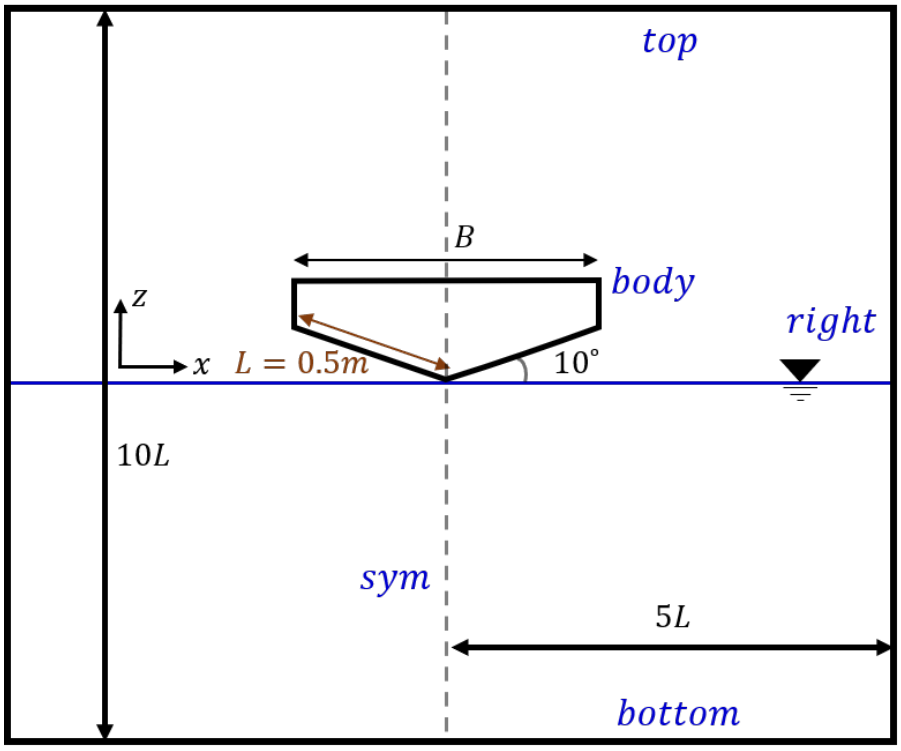


Figure 5.17. Sketch of wedge entry and exit problem

$$z(t) = \frac{1}{2}At^2 - V_0 t \quad , \quad t_0 = \frac{V_0}{A} \tag{5.21}$$

Table 5.7 gives the summary of the cases. Four cases are considered with two values of impact velocity V_0 and two values of acceleration A. The latter are chosen so that the wedge has two different values of maximal immersion ratio $z/(L\ \sin\beta)$ at t_0. Figure 5.18 plots the evolution of

the vertical motion with respect to time (left) and normalized time (right).

Table 5.7. Wedge entry and exit cases

Case	$z/(L\ \sin\beta)$	V_0 (m/s)
Case 1	0.5	4
Case 2	0.5	8
Case 3	1.0	4
Case 4	1.0	8

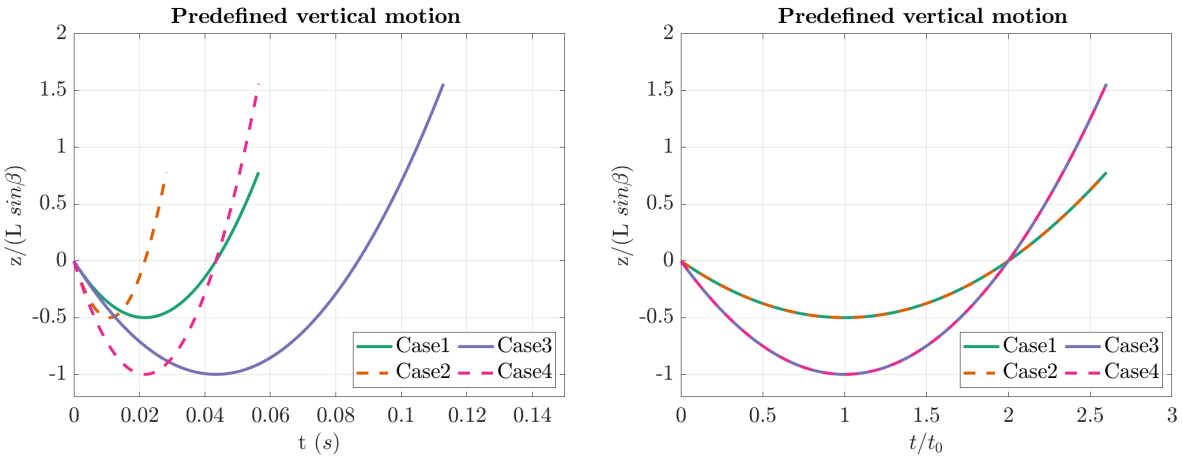


Figure 5.18. Vertical motion of the wedge in the four cases. Left plot is with real time, and right plot is with normalized time.

The same four solvers as previously (the four combinations of the different interface schemes) are tested as well on this problem. The computational mesh is composed of 62994 cells. The time-step size is $h = t_0/400$ when the maximal immersion ratio is 0.5 and $h = t_0/800$ when the immersion ratio is 1.0.

Figures 5.19 to 5.22 illustrate the normalized pressure distribution and the VOF predicted free surface during the wedge water entry and exit, for the four cases. Each column of the figures shows the results from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is made of snapshots at $t/t_0 = 0.5$, $t/t_0 = 1.0$, and $t/t_0 = 1.5$. These three times correspond to mid-time of water entry, maximal immersion, and time when the wedge reaches the same position as initially during its vertical ascension.

Figures 5.19 and 5.20 presents the result of maximal immersion ratio of $z/(L\ \sin\beta) = 0.5$. The free-surface profiles during the water entry phase are similar, but the pressure fields of *interFlow* was larger than others. For both Case 1 and Case 3, the free-surface and the pressure fields obtained from *interFoam* and *interFlow* at $t/t_0 = 1.0$ was similar. However, solvers using GFM method show different wave profiles and non-physical high pressure field near the free-

surface.

Figures 5.21 and 5.22 presents the result of maximal immersion ratio of $z/(L\ \sin\beta) = 1.0$. Similar to the Case 1 and Case 2, the free-surface profile and non-dimensionalized pressure between Case 3 and Case 4 are similar. The difference between *interFoam* and *interFlow* was small for at $t/t_0 = 0.5$ and at $t/t_0 = 1.0$. However, the length of detachment of water on the wedge slope is differnet, and this affects to the entire flow fields. The difference of the length of detachment is larger between the solvers with and without GFM scheme.

Figures 5.23 and 5.24 compare the pressure fields and the free-surface profiles from SPH to the best two of the four OpenFOAM solvers, *interFoam* and *interFlow*. The SPH free-surface profiles and pressure fields are from the Ph.D. thesis of Michel [113]. Similar to the previous figures, each row presents snapshot at $t/t_0 = 0.5$, $t/t_0 = 1.0$, and $t/t_0 = 1.5$. The SPH and OpenFOAM results share the same color contour levels and the identical size of wedge in figures. In Figures 5.23, large pressure difference is captured at $t/t_0 = 0.5$, and significant differences on the detachment are captured at $t/t_0 = 1.0$ and $t/t_0 = 1.5$ with *interFoam* solver. In Figures 5.24, similar large pressure difference is captured. However with *interFlow*, the prediction of water detachment and free-surface profile were better than *interFoam*. As a result, the free-surface profiles and pressure fields obtained from the *interFlow* solver match better with the SPH results than the *interFoam* ones.

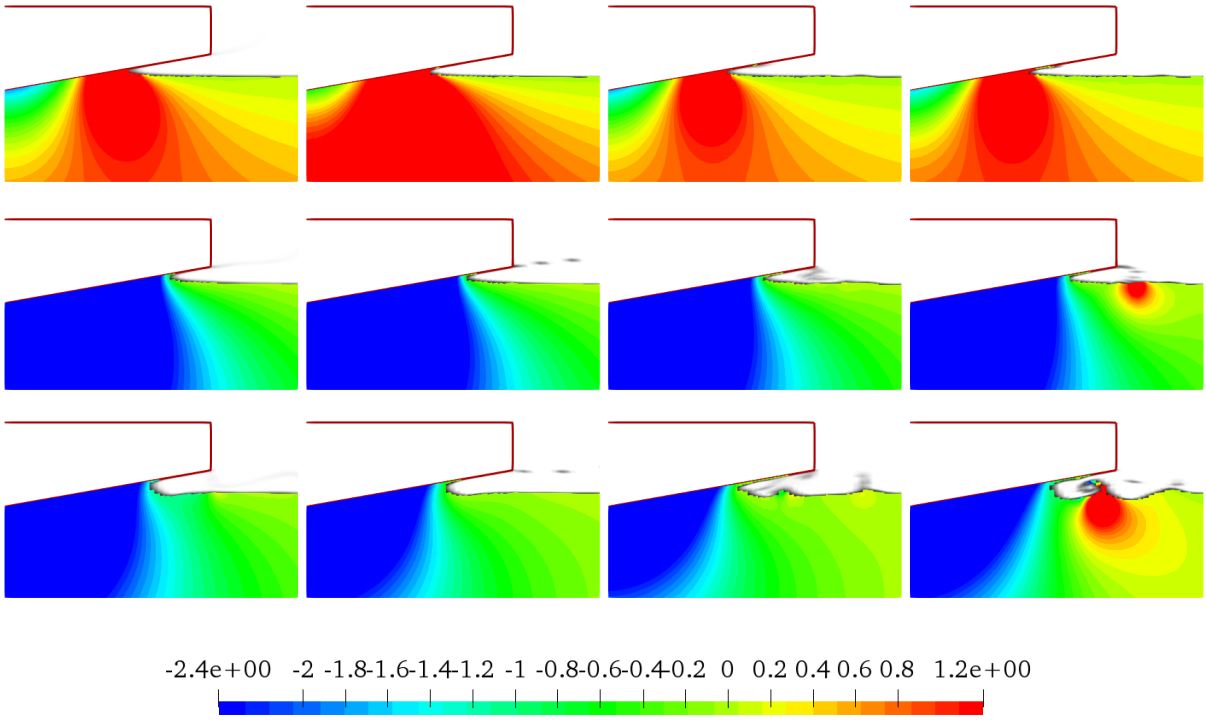


Figure 5.19. Case 1: Normalized Pressure field ($P/(0.5\rho V_0^2)$) and the VOF field with four different solvers. Each column shows the result from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is the snapshot at $t/t_0 = 0.5$, $t/t_0 = 1.0$, $t/t_0 = 1.5$.

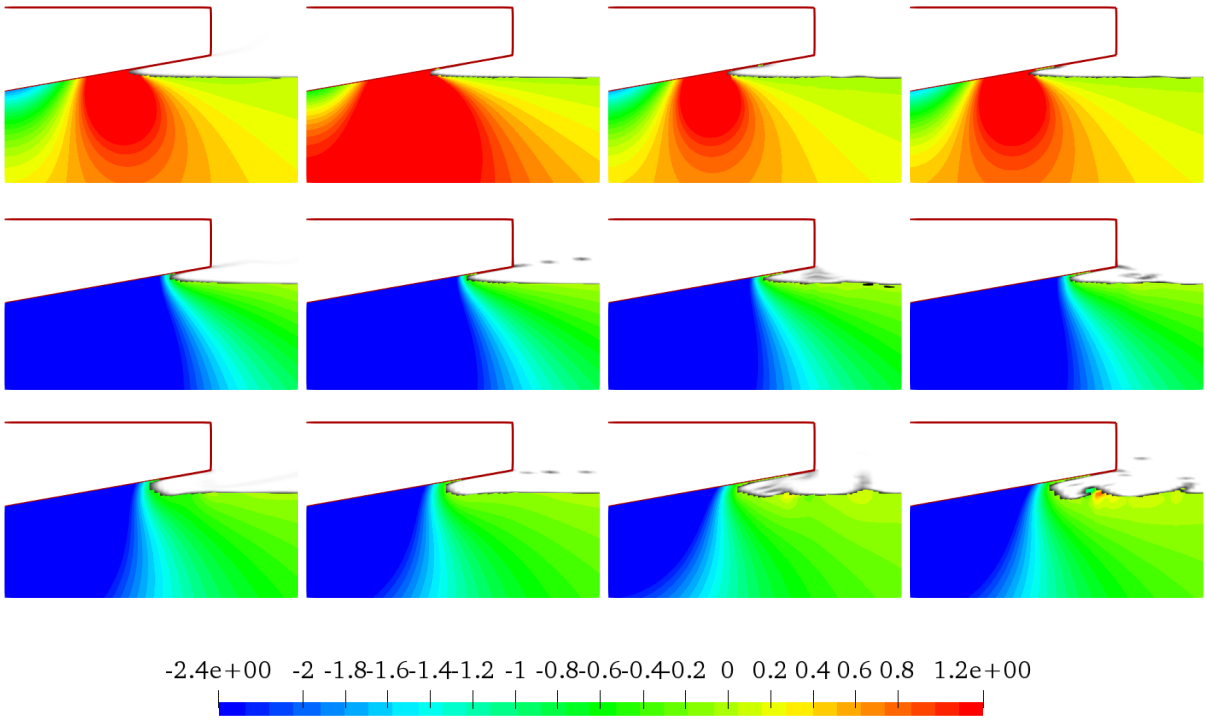


Figure 5.20. Case 2: Normalized Pressure field ($P/(0.5\rho V_0^2)$) and the VOF field with four different solvers. Each column shows the result from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is the snapshot at $t/t_0 = 0.5$, $t/t_0 = 1.0$, $t/t_0 = 1.5$.

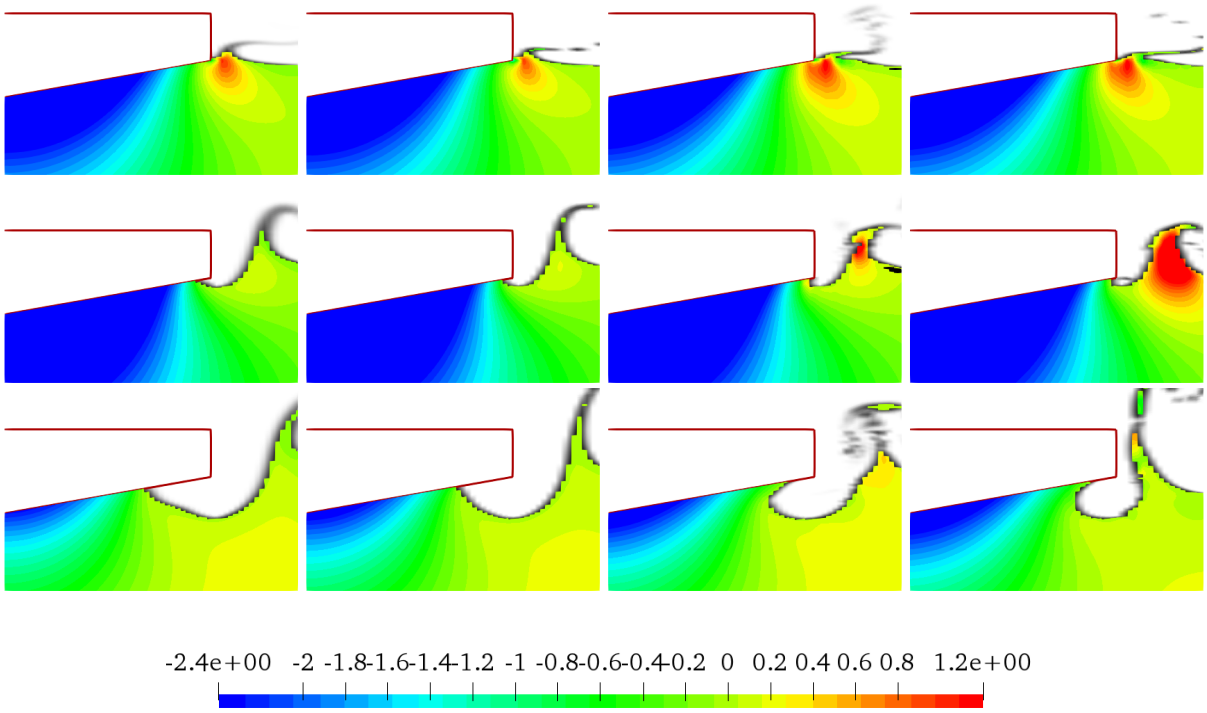


Figure 5.21. Case 3: Normalized Pressure field ($P/(0.5\rho V_0^2)$) and the VOF field with four different solvers. Each column shows the result from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is the snapshot at $t/t_0 = 0.5$, $t/t_0 = 1.0$, $t/t_0 = 1.5$.

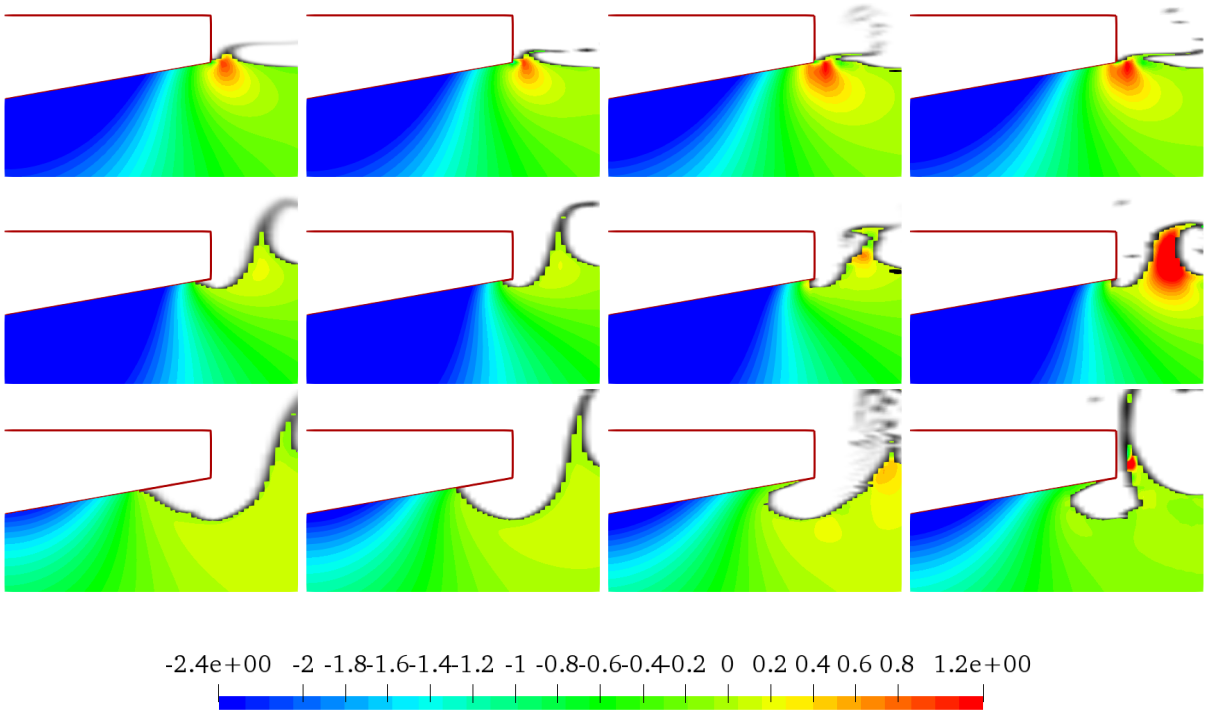


Figure 5.22. Case 4: Normalized Pressure field ($P/(0.5\rho V_0^2)$) and the VOF field with four different solvers. Each column shows the result from *interFoam*, *interFlow*, *interFoamGFM*, and *interFlowGFM*, respectively. Each row is the snapshot at $t/t_0 = 0.5$, $t/t_0 = 1.0$, $t/t_0 = 1.5$.

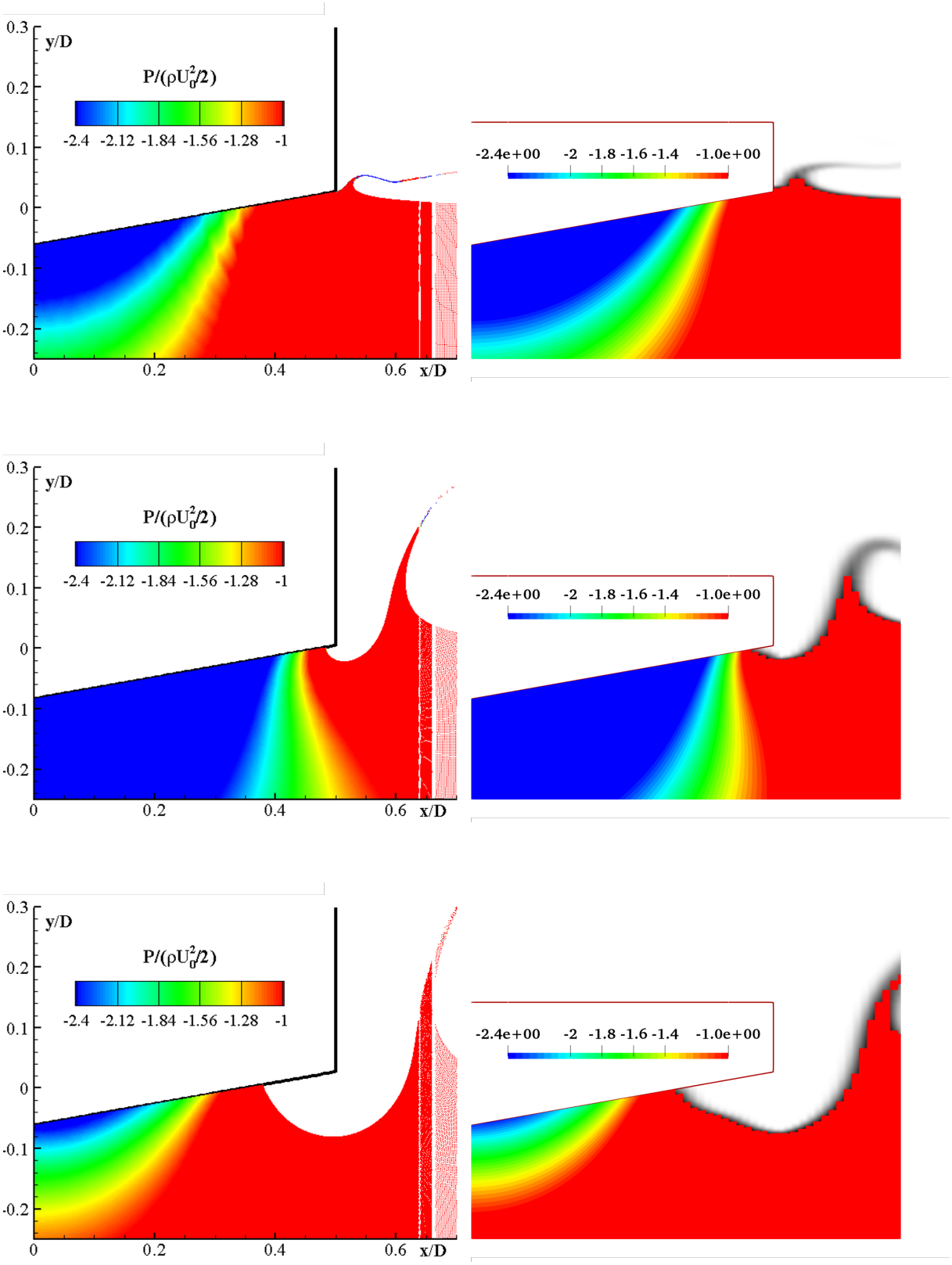


Figure 5.23. Comparison of normalized pressure field between the SPH (left) and *interFoam* (right) for Case 3. Each row is the snapshot at $t/t_0 = 0.5$, $t/t_0 = 1.0$, $t/t_0 = 1.5$.

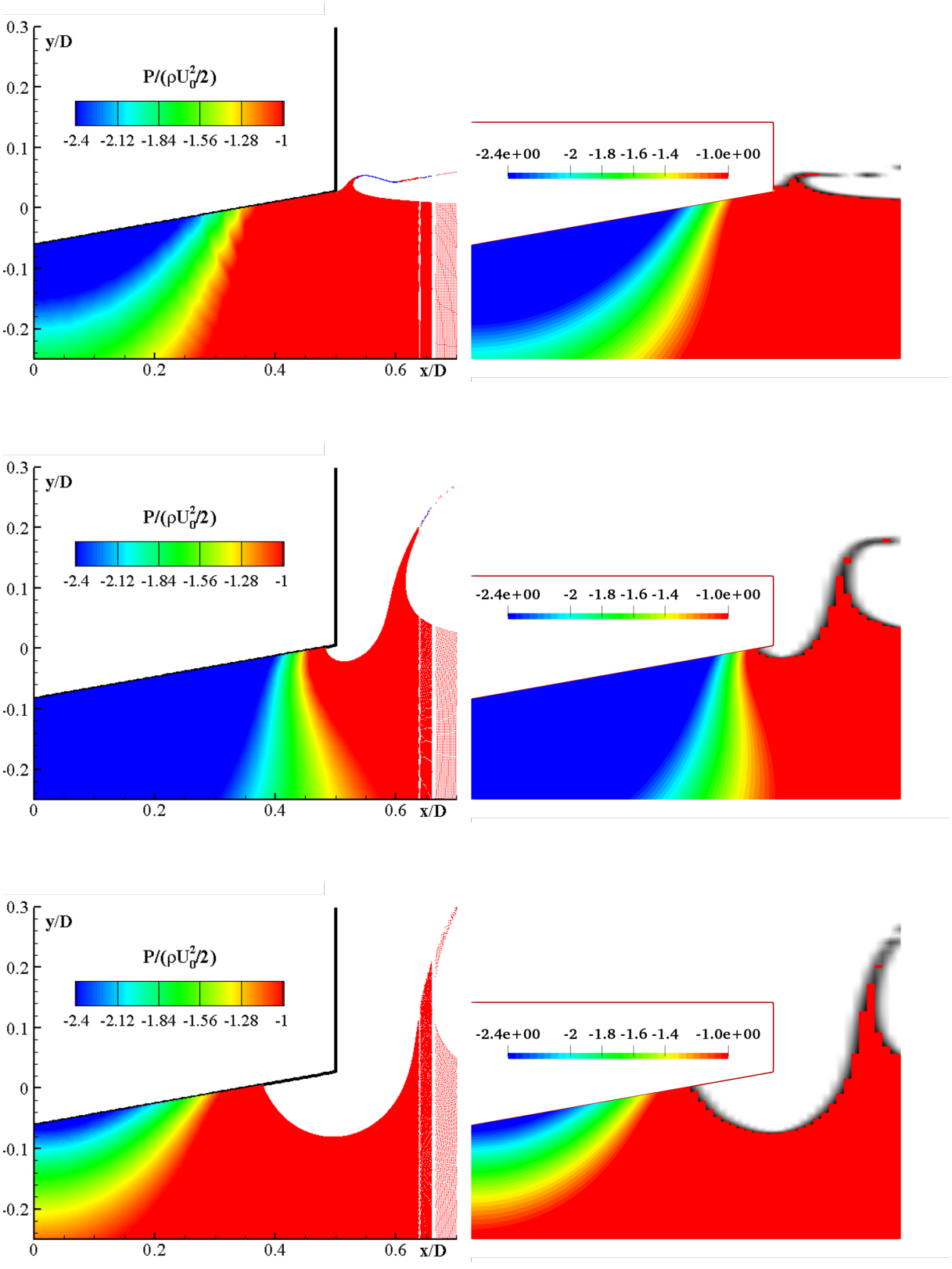


Figure 5.24. Comparison of normalized pressure field between the SPH (left) and *interFlow* (right) for Case 3. Each row is the snapshot at $t/t_0 = 0.5$, $t/t_0 = 1.0$, $t/t_0 = 1.5$.

6 High-order time integration scheme

6.1 Time integration methods

Many important phenomena studied in the naval hydrodynamic field, such as sea-keeping, sloshing, and slamming inside the LNG tank, are two-phase unsteady flows. To properly simulate these phenomena, fast and accurate numerical methods are often required. In this chapter, one of the main objectives is to develop a CFD solver for two-phase incompressible flow with a less diffusive time integration scheme.

Research on the numerical integration of ordinary differential equations (ODEs) has been a topic of interest for a long time. High-order time integration schemes have low dissipation, but they are more costly than lower-order schemes and can be less stable. One of the popular high-order time integration schemes is the third-order three-stage TVD (total-variation-diminishing) explicit Runge-Kutta (ERK) method [49]. This explicit scheme satisfies the TVD properties for a CFL number below one, and it shows good numerical stability and accuracy [50, 51]. However, it is practically challenging to satisfy this maximum CFL criterion in typical naval and offshore applications.

There is less limitation on the maximum usable CFL number when using an implicit or a semi-implicit time-stepping scheme. The diagonally implicit Runge-Kutta (DIRK) schemes are a category of high-order semi-implicit time integration methods [52, 53], used widely in single-phase CFD solvers [55, 56, 51]. Bijl et al. [57] investigated the accuracy and efficiency of the BDF2 and some DIRK schemes for the unsteady compressible Navier–Stokes equations. The authors computed the error vs. computation time and concluded that fourth- and fifth-order Runge–Kutta schemes are more cost-efficient than the BDF2 scheme. D'Alessandro et al. [58] used OpenFOAM and studied the effect of second-order and third-order DIRK schemes in single-phase flows.

The present chapter aims to investigate the efficiency of the DIRK method in a two-phase flow solver. Its implementation includes specific treatments for the time integration of the VOF convection equation. These treatments are essential to keep sharp the free-surface. A second-order DIRK method and three third-order DIRK methods are compared with existing OpenFOAM time integration schemes. In total, the efficiency of six different time integration schemes is compared.

This section is organized as follows. First, the general theory on the DIRK methods and each DIRK method's properties are presented. Second, the detailed implementations of DIRK methods on the VOF convection equation, the Navier-Stokes equation, and the SWENSE method are given. Finally, the DIRK methods and the conventional time integration methods are compared using benchmark cases, such as 2D Taylor-Green vortex, wave propagation, and Dam break. The order of convergence, stability, and computational costs are compared and discussed.

6.1.1 DIRK method

An arbitrary ODE, Eqn. (6.1), can be discretized in $t^{(n)}$, with $n \in N$ being the time index. At each time $t^{(n)}$, the quantity y is discretized in $y^{(n)} = y(t^{(n)})$.

$$\dot{y} = f(t, y(t)) \tag{6.1}$$

The DIRK temporal discretization uses sub-stages i with $i = 1..s$, where s is the number of Runge-Kutta sub-stages [53]. At each sub-stage, the quantity $y^{(n,i)} = y(t^{(n,i)})$ is the result valid at $t^{(n,i)} = t^{(n)} + h\tau_i$ where h is the time-step size, and τ_i is a real coefficient called the abscissa. The DIRK methods discretize Eqn. (6.1) into Eqn. (6.2) and Eqn. (6.3), where a_{ij} is the stage weight; b_i is the scheme weight.

$$y^{(n,i)} = y^{(n)} + ha_{ii}f(t^{(n,i)}, y^{(n,i)}) + h\sum_{j=1}^{i-1} a_{ij}f(t^{(n,j)}, y^{(n,j)}) \tag{6.2}$$

$$y^{(n+1)} = y^{(n,s)} = y^{(n)} + h\sum_{j=1}^{s} b_j f(t^{(n,j)}, y^{(n,j)}) \tag{6.3}$$

The DIRK method described in Eqn. (6.2) and Eqn. (6.3) is a semi-implicit time integration method with implicit (i) and explicit (j) terms. The Runge-Kutta method depends on the coefficients a_{ij}, b_i, and τ_i. When presented in the form of the Butcher matrix in Eqn. (6.4), $\mathbf{A}$ is a lower triangular matrix [52]. If the upper triangular matrix of the butcher matrix were non-zero, the method would be called the implicit Runge-Kutta method. However, this is not considered in this thesis because of the heavy additional iterative computation.

$$\begin{array}{c|cccc} \tau_1 & a_{11} & 0 & 0 & 0 \\ \tau_2 & a_{21} & a_{22} & 0 & 0 \\ \tau_3 & a_{31} & a_{32} & a_{33} & 0 \\ \tau_4 & a_{41} & a_{42} & a_{43} & a_{44} \\ \hline & b_1 & b_2 & b_3 & b_4 \end{array} \quad \Rightarrow \quad \begin{array}{c|c} \tau & \mathbf{A} \\ \hline & \mathbf{b}^T \end{array} \tag{6.4}$$

By using a lower triangular matrix, the equations may then be solved row by row in a successive manner. The diagonal coefficients of the matrix are weights for the implicit term of Eqn. (6.2). Thus, it is named a 'diagonally implicit' method. If the diagonal coefficients are zero and the matrix $\mathbf{A}$ is a strictly lower triangular matrix, it is then an explicit RK method. The DIRK method's sub-classes are presented under Butcher matrix form in Fig Figure 6.1. They are named using the notations S for 'Singly diagonal', E for 'Explicit first stage', and SA for 'Stiffly Accurate'.

τ_1	γ	0	0	0
τ_2	a_{21}	γ	0	0
τ_3	a_{31}	a_{32}	γ	0
τ_4	a_{41}	a_{42}	a_{43}	γ
	b_1	b_2	b_3	b_4

τ_1	γ	0	0	0
τ_2	a_{21}	γ	0	0
τ_3	a_{31}	a_{32}	γ	0
1	a_{41}	a_{42}	a_{43}	γ
	a_{41}	a_{42}	a_{43}	γ

0	0	0	0	0
τ_2	a_{21}	γ	0	0
τ_3	a_{31}	a_{32}	γ	0
τ_4	a_{41}	a_{42}	a_{43}	γ
	b_1	b_2	b_3	b_4

0	0	0	0	0
τ_2	a_{21}	γ	0	0
τ_3	a_{31}	a_{32}	γ	0
1	a_{41}	a_{42}	a_{43}	γ
	a_{41}	a_{42}	a_{43}	γ

Figure 6.1. Four-stage SDIRK (top-left), SDIRK_SA (top-right), ESDIRK (bottom-left) and ESDIRK_SA (bottom-right) methods [7]

.

In the singly diagonal 'SDIRK' method, each diagonal element has the same value $a_{ii} = \gamma$. It facilitates designing an A- and L-Stable method by simplifying the stability function. For some ODEs, this property could be beneficial for the matrix solver by repeatedly using the stored LU-factorization [116, 117].

In the 'EDIRK' methods, the explicit first stage is obtained by having $a_{11} = \tau_1 = 0$ to make the stage order two. This method uses the solution from the previous time step as a first RK sub-stage ($y^{(n,1)} = y^{(n)}$). The computation of the explicit first stage method is almost inexpensive compared to the other implicit stages' computation. Therefore, when approximating the computation cost of the DIRK methods, only the number of implicit stages (s_I) must be considered. The explicit first stage is also known to benefit the convergence rate of the solution of very stiff ODEs. The simplest EDIRK (ESDIRK) scheme possible to imagine is the Crank-Nicolson method. It is a second-order two-step ($s = 2$) explicit first stage ($s_I = s - 1 = 1$) DIRK method (Eqn. (B.4)).

Another constraint on the coefficients, which is both beneficial and straightforward, is the stiffly-accurate (SA) assumption which imposes $a_{sj} = b_j$ and $\tau_s = 1$ [118]. The substitution of this SA assumption to Eqn. (6.2) makes the final Runge-Kutta stage ($i = s$) of Eqn. (6.2) equal to Eqn. (6.3). This assumption induces $y^{(n+1)}$ implicitly without explicitly solving Eqn. (6.3). Also, the SA assumption increases the stability and the convergence rate of numerical solutions of stiff ODEs. In two-phase flow CFD, it helps not to use the additional explicit limiter in the VOF convection equation when solving Eqn. (6.3). Therefore, in this thesis, only stiffly-accurate methods are considered.

Appendix B summarizes the associated numerical properties and the list of DIRK-type methods used in this study. This thesis applied maximum third-order DIRK methods. The third-order time integration scheme is considered enough, recognizing that only second-order spatial schemes are applied. Moreover, the number of implicit stages is higher than the convergence order for

higher-order methods. For instance, the third-order SDIRK_SA method requires three implicit stages, but the fourth-order SDIRK_SA method requires five implicit stages. Therefore, from the fourth-order DIRK scheme, it is difficult to be efficient unless the time integration scheme shows significant improvement in the solution. In Appendix B, the DIRK coefficients are listed using the Butcher matrix, and the properties of each DIRK method are shortly described. The schemes are labeled using the convention given in Eqn. (6.5), where each letter's significance is described in Table 6.1. For example, the Crank-Nicolson scheme corresponds to the ESDIRK[2](2)_SA.

$$(\mathrm{E})(\mathrm{S})\mathbf{DIRK[p](s)_SA_}\mathrm{i} \tag{6.5}$$

Table 6.1. Identification of DIRK method. Ex: ESDIRK[p](s)_SA_i.

E	Explicit first stage	S	Singly
DI	Diagonally-implicit	RK	Runge-Kutta
p	Global order	s	Number of stage
SA	Stiffly accurate	i	identifier

6.2 Discretization of averaged two-phase incompressible flow model with DIRK method

This section presents the formulation and the implementation of DIRK time integration schemes in an incompressible two-phase Navier-Stokes solver. The finite volume (FV) representation of the VOF convection equation and the momentum equation of the averaged two-phase flow are recalled in Eqn. (6.6) and Eqn. (6.7). The description of variables and FV notations are described in Section 2.1 and Section 4.2.

$$\frac{\partial(V_P\alpha)}{\partial t} + \sum_f (\phi\alpha)_f + \sum_f (\phi_{r,f}\alpha_f(1-\alpha_f)) = 0 \tag{6.6}$$

$$\frac{\partial(V_P\rho\mathbf{u})}{\partial t} + \sum_f ((\rho\phi)_f\mathbf{u}_f) - \sum_f (\mu_{E,f}\mathbf{S}_f\bullet(\nabla\mathbf{u}_f)) - \nabla\mathbf{u}\bullet\nabla\mu V_P = -\nabla p_d V_P - (\mathbf{g}\bullet\mathbf{x})\nabla\rho V_P\,. \tag{6.7}$$

The entire algorithm of the solver system with DIRK schemes is illustrated in Figure 6.2. It is very similar to the original algorithm presented in Section 4.7, but the additional Runge-Kutta loop is added over the PIMPLE iteration. The DIRK integration scheme needs several sub-stages, and the PIMPLE algorithm described above is applied to each of them. Each DIRK sub-stage explicitly uses the previous DIRK sub-stages results ($1 \sim i-1$) to increase the order of the scheme, as shown in Eqn. (6.2). Therefore, the VOF convection equation and the PISO iteration must be modified due to the additional explicit terms from the DIRK method. The following subsections describe the detailed formulation of the DIRK algorithm to the VOF convection equation and momentum equation.

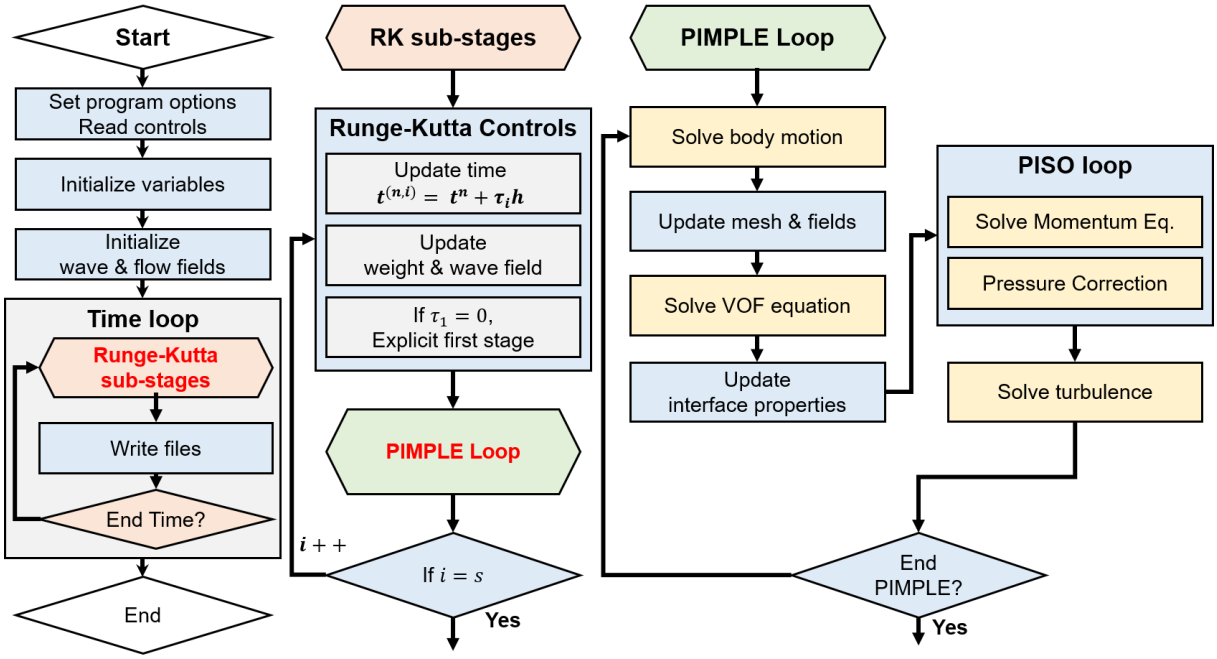


Figure 6.2. Targeted solver algorithm.

6.2.1 VOF convection algorithm with DIRK scheme

This subsection describes the formulation of the DIRK method to the VOF convection equation. Some arguments are defined to facilitate the formulation. The subscript $(...)_{BD}$ refers to the bounded spatial discretization scheme applied to the VOF flux ($\phi\alpha$). The subscript $(...)_{UB}$ refers to an unbounded spatial discretization scheme. The subtraction of bounded flux from unbounded flux is named correction flux, and the subscript $(...)_{Corr}$ is used. The corrected flux is limited by the MULES (Multidimensional Universal Limiter for Explicit Scheme) limiter to increase accuracy and ensure the boundedness. Assuming computational domain and grids do not move in time, applying the DIRK scheme to the VOF convection equation yields to Eqn. (6.8).

$$\begin{aligned}\frac{V_P\alpha_P^{(n,i)} - V_P\alpha_P^{(n)}}{h} + a_{ii}\sum_f (\phi_f\alpha_f)^{(n,i)} + a_{ii}\sum_f (\phi_{r,f}\alpha_f(1-\alpha_f))^{(n,i)} \\ + \sum_{j=1}^{i-1} a_{ij}\sum_f (\phi_f\alpha_f)^{(n,j)} + \sum_{j=1}^{i-1} a_{ij}\sum_f (\phi_{r,f}\alpha_f(1-\alpha_f))^{(n,j)} = 0\end{aligned} \tag{6.8}$$

The explicit limiter ensures the boundedness of the VOF during the convection of the VOF. The details of the MULES limiter are presented in the Ph.D. thesis of Li [68], and Fig. 6.3 illustrates the algorithm for bounded VOF convection.

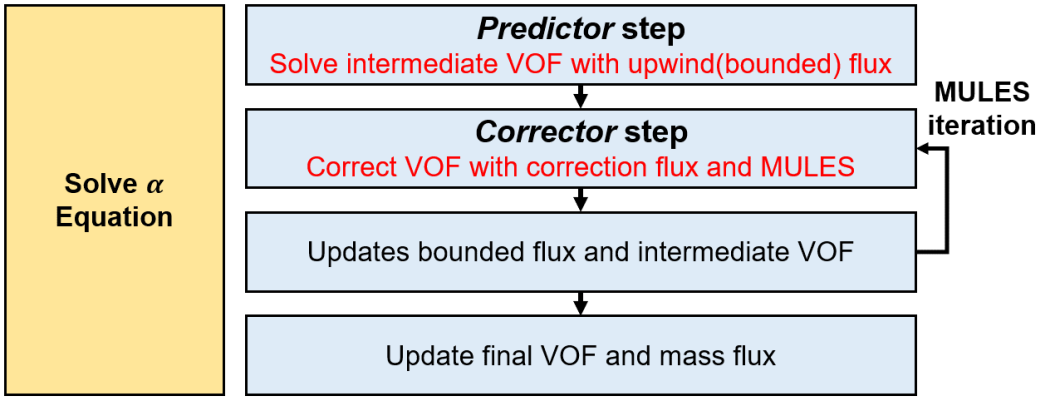


Figure 6.3. VOF convection algorithm.

The VOF convection at each sub-stage (Eqn. (6.8)) is solved through a *Predictor-Corrector* approach. The *Predictor* equation is given in Eqn. (6.9). α_P^* is an intermediate VOF, and it is the solution of Eqn. (6.9). The bounded (upwind) flux is used to evaluate the intermediate VOF, and the artificial compression is not included. It is important to note that, the solver has to store the previous RK sub-stage fluxes and VOF ($\phi^{(n,j)}$ and $\alpha^{(n,j)}$) to solve Eqn. (6.9).

$$\frac{V_P\alpha_P^{*(n,i)} - V_P\alpha_P^{(n)}}{h} + a_{ii}\sum_f (\phi_f\alpha_f^*)_{BD}^{(n,i)} + \sum_{j=1}^{i-1} a_{ij}\sum_f (\phi_f\alpha_f)_{BD}^{(n,j)} = 0 \tag{6.9}$$

The correction step is presented in Eqn. (6.10). The subscript $(...)_{UB}$ is the spatial discretization scheme selected by the user.

$$\begin{aligned}\frac{V_P\alpha_P^{(n,i)} - V_P\alpha_P^{*(n,i)}}{h} &+ a_{ii}\sum_f [(\phi_f\alpha_f)_{UB}^{(n,i)} - (\phi_f\alpha_f^*)_{BD}^{(n,i)}] \\ &+ \sum_{j=1}^{i-1} a_{ij}\sum_f [(\phi_f\alpha_f)_{UB}^{(n,j)} - (\phi_f\alpha_f)_{BD}^{(n,j)}] \\ + a_{ii}\sum_f (\phi_{r,f}\alpha_f(1-\alpha_f))_{UB}^{(n,i)} &+ \sum_{j=1}^{i-1} a_{ij}\sum_f (\phi_{r,f}\alpha_f(1-\alpha_f))_{UB}^{(n,j)} = 0\end{aligned} \tag{6.10}$$

F_{BD} describes bounded fluxes. The definitions are different for sub-stages i and j. With these definitions, the correction fluxes are defined by Eqn. (6.12).

$$F_{BD}^{(n,i)} = a_{ii}\left[(\phi_f\alpha_f^*)_{BD}^{(n,i)}\right] \qquad F_{BD}^{(n,j)} = a_{ij}\left[(\phi_f\alpha_f)_{BD}^{(n,j)}\right] \tag{6.11}$$

$$\begin{aligned}F(\phi,\alpha^*)_{Corr}^{(n,i)} &= a_{ii}\left[(\phi_f\alpha_f^*)_{UB}^{(n,i)} + (\phi_{r,f}\alpha_f^*(1-\alpha_f^*))_{UB}^{(n,i)}\right] - F_{BD}^{(n,i)} \\ F(\phi,\alpha)_{Corr}^{(n,j)} &= a_{ij}\left[(\phi_f\alpha_f)_{UB}^{(n,j)} + (\phi_{r,f}\alpha_f(1-\alpha_f))_{UB}^{(n,j)}\right] - F_{BD}^{(n,j)}\end{aligned} \tag{6.12}$$

The correction flux is limited by the MULES limiter and assures the boundedness of the VOF. α_P^{**} is a corrected VOF field, and λ_f is an explicit limiter field evaluated from the MULES algorithm.

$$\frac{V_P\alpha_P^{**(n,i)} - V_P\alpha_P^{*(n,i)}}{h} + \lambda_f \left[\sum_f F(\phi,\alpha^*)_{Corr}^{(n,i)} + \sum_f \sum_{j=1}^{i-1} F(\phi,\alpha)_{Corr}^{(n,j)} \right] = 0 \tag{6.13}$$

After the correction of the VOF, Eqn. (6.14) updates the bounded flux and indeterminate VOF, respectively. The corrected VOF and the corrected bounded flux are iteratively used to correct the VOF in Eqn. (6.12). This iteration is named MULES iteration, and this procedure ensures boundedness while having maximum accuracy.

$$\begin{aligned} F_{BD}^{(n,i)} &= F_{BD}^{(n,i)} + \lambda_f F(\phi,\alpha^*)_{Corr}^{(n,i)} \\ F_{BD}^{(n,j)} &= F_{BD}^{(n,j)} + \lambda_f F(\phi,\alpha)_{Corr}^{(n,j)} \\ \alpha_P^{*(n,i)} &= \alpha_P^{**(n,i)} \end{aligned} \tag{6.14}$$

After the MULES iteration, Eqn. (6.15) evaluates the final updated VOF flux and the VOF field at RK sub-step i. Finally, the Eqn. (6.16) shows the mass flux for RK sub-step i.

$$(\phi\alpha)_f^{(n,i)} = F_{BD}^{(n,i)}/a_{ii} \qquad \alpha_P^{(n,i)} = \alpha_P^{**(n,i)} \tag{6.15}$$

$$(\rho\phi)_f^{(n,i)} = (\phi\alpha)_f^{(n,i)}(\rho_{water} - \rho_{air}) + \phi^{(n,i)}\rho_{air} \tag{6.16}$$

6.2.2 Difference between OpenFOAM Crank-Nicolson scheme and DIRK-based Crank-Nicolson scheme for the averaged two-phase flow model

The ESDIRK[2](2)_SA method is a second-order accurate, two-stage, explicit first, stiffly-accurate, and A-stable scheme, and its Butcher matrix is presented at Eqn. (B.4). The ESDIRK[2](2)_SA method is also named the Crank-Nicolson scheme, and it is mathematically identical. The main difference between ESDIRK[2](2)_SA (*RKCN*) and OpenFOAM Crank-Nicolson (*OFCN*) scheme is on the definition of the bounded VOF flux and its correction method. Therefore, this section describes the formulation of the *OFCN* scheme and compared it with the Eqn. (B.4) method.

Eqn. (6.17) shows the VOF convection equation with CN scheme, where $C_{CN} = 1/(C_{oc}+1)$ and C_{oc} is the off-centering coefficient. Note that the off-centering can also be applied to the ESDIRK[2](2)_SA, and it is described in Eqn. (B.5) using the same coefficient.

$$\begin{aligned} \frac{(V\alpha)^{t+h} - (V\alpha)^t}{h} + C_{CN}\sum_f [F_f]^{t+h} + (1-C_{CN})\sum_f [F_f]^t = 0\,, \\ F_f = \phi_f \alpha_f \end{aligned} \tag{6.17}$$

The *OFCN* scheme mixes two summation terms in Eqn. (6.17), and uses time-blended flux defined at 6.18. Then, the *Predictor* and the *Corrector* algorithm is applied to the time-blended flux.

$$F_{CN} = C_{CN}[F_f]^{t+h} + (1-C_{CN})[F_f]^t \tag{6.18}$$

However, the momentum equation requires the mass fluxes at time $t+h$. Therefore, the time-blended flux is extrapolated from blended time to $t+h$ using extrapolation at Eqn. (6.20).

$$\rho\phi^{t+h} = (\rho_{water} - \rho_{air})F^{t+h} + \rho_{air}\phi \tag{6.19}$$

$$F^{t+h} = (1 + C_{oc})F_{f,CN} - C_{oc}F^{t} \tag{6.20}$$

The main difference between *OFCN* and Eqn. (B.4) methods is the definition of the VOF flux. *OFCN* used the time-blended VOF flux, and the *Predictor-Corrector* algorithm is applied to the time-blended flux. While *RKCN* applied the *Predictor-Corrector* algorithm to the implicit term and the explicit term separately.

6.2.3 The PISO algorithm with DIRK scheme

This subsection presents the algorithm of PISO iteration, including the DIRK method. The difference between the original PISO algorithm is the additional source terms and coefficients in the equation. If a mesh does not move in time, the substitution of the DIRK method to momentum equation yields to Eqn. (6.21), while $\mathbf{R}$ includes all components except the time discretization.

$$\frac{(V\rho\mathbf{u})_P^{(n,i)} - (V\rho\mathbf{u})_P^{(n)}}{h} + a_{ii}\mathbf{R}^{(n,i)} + \sum_{j=1}^{i-1} a_{ij}\mathbf{R}^{(n,j)} = 0 \tag{6.21}$$

$$\begin{aligned}
\mathbf{R}^{(n,i)} &= \sum_f \left[(\rho\phi)_f^{(n,i)}\mathbf{u}_f^{(n,i)}\right] - \sum_f \left[\mu_f^{(n,i)}\mathbf{S}_f\bullet(\nabla\mathbf{u}_f^{(n,i)})\right] \\
&\quad - \nabla\mathbf{u}_P^{(n,i)}\bullet\nabla\mu^{(n,i)}V_P + \nabla p_d^{(n,i)}V_P + (\mathbf{g}\bullet\mathbf{x}^{(n,i)})\nabla\rho_P^{(n,i)}V_P \\
\mathbf{R}^{(n,j)} &= \sum_f \left[(\rho\phi)_f^{(n,j)}\mathbf{u}_f^{(n,j)}\right] - \sum_f \left[\mu_f^{(n,j)}\mathbf{S}_f\bullet(\nabla\mathbf{u}_f^{(n,j)})\right] \\
&\quad - \nabla\mathbf{u}_P^{(n,j)}\bullet\nabla\mu^{(n,j)}V_P + \nabla p_d^{(n,j)}V_P + (\mathbf{g}\bullet\mathbf{x}^{(n,j)})\nabla\rho_P^{(n,j)}V_P
\end{aligned} \tag{6.22}$$

The mass flux $(\rho\phi)_f^{(n,i)}$ is constant in the PISO iteration, and it is calculated from the VOF convection equation. The transposition of dynamic pressure and hydro-static terms inside the $\mathbf{R}^{(n,i)}$ yields to Eqn. (6.23).

$$\begin{aligned}
&\frac{(V\rho\mathbf{u})_P^{(n,i)} - (V\rho\mathbf{u})_P^{(n)}}{h} + \sum_{j=1}^{i-1} a_{ij}\mathbf{R}^{(n,j)} + a_{ii}\Big[\sum_f \left[(\rho\phi)_f^{(n,i)}\mathbf{u}_f^{(n,i)}\right] \\
&- \sum_f \left[\mu_f^{(n,i)}\mathbf{S}_f\bullet(\nabla\mathbf{u}_f^{(n,i)})\right] - \nabla\mathbf{u}_P^{(n,i)}\bullet\nabla\mu^{(n,i)}V_P\Big] = -a_{ii}\left[\nabla p_d^{(n,i)}V_P + (\mathbf{g}\bullet\mathbf{x}^{(n,i)})\nabla\rho_P^{(n,i)}V_P\right]
\end{aligned} \tag{6.23}$$

$$\mathrm{UEqn}^{(n,i)} : \mathbf{a}_P^{(n,i)}\mathbf{u}_P^{(n,i)} - \mathbf{H}^{(n,i)} = -a_{ii}\left[\nabla p_d^{(n,i)}V_P + (\mathbf{g}\bullet\mathbf{x}^{(n,i)})\nabla\rho_P^{(n,i)}V_P\right] \tag{6.24}$$

The notations $\mathbf{a}_P$ and $\mathbf{H}$ are commonly used in OpenFOAM to explain the PISO algorithm [65]. $\mathbf{a}_P^{(n,i)}$ indicates the diagonal components of $\mathrm{UEqn}^{(n,i)}$, and $\mathbf{H}^{(n,i)}$ includes the all off-diagonal and source components.

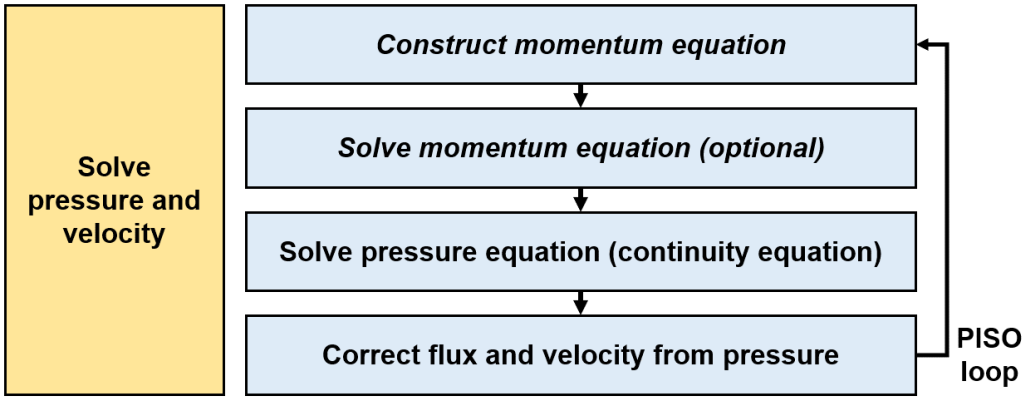


Figure 6.4. PISO algorithm.

The velocity $\mathbf{u}_P^{(n,i)}$ from the l.h.s of Eqn. (6.24) must satisfy the continuity condition. The substitution of velocity to the continuity equation yields to Eqn. (6.25).

$$\nabla \bullet \mathbf{u}_P^{(n,i)} = \nabla \bullet \Big[\frac{\mathbf{H}^{(n,i)}}{\mathbf{a}_P^{(n,i)}} - \frac{a_{ii}}{\mathbf{a}_P^{(n,i)}} \big[\nabla p_d^{(n,i)} V_P + (\mathbf{g} \bullet \mathbf{x}^{(n,i)}) \nabla \rho_P^{(n,i)} V_P \big] \Big] = 0 \tag{6.25}$$

$$\mathrm{pEqn}^{(n,i)} : \sum_f \Big[\Big(\frac{a_{ii} V_P}{\mathbf{a}_P^{(n,i)}} \nabla p_d^{(n,i)} \Big)_f \Big] \bullet \mathbf{S}_f = \sum_f \Big[\frac{\mathbf{H}^{(n,i)}}{\mathbf{a}_P^{(n,i)}} - \frac{a_{ii}}{\mathbf{a}_P^{(n,i)}} (\mathbf{g} \bullet \mathbf{x}^{(n,i)}) \nabla \rho_P^{(n,i)} V_P \Big]_f \bullet \mathbf{S}_f \tag{6.26}$$

Eqn. (6.26) is commonly named the pressure equation in the OpenFOAM community. It solves the Laplacian equation to obtain the dynamic pressure, where the subscript $(...)_f$ represents the cell center to face center interpolation. The velocity and flux are corrected using dynamic pressure, and the pressure and velocity are strongly coupled inside the non-linear iteration of the PISO algorithm.

$$\begin{aligned} \mathbf{u}^{(n,i)} &= \Big[\frac{\mathbf{H}^{(n,i)}}{\mathbf{a}_P^{(n,i)}} - \frac{a_{ii}}{\mathbf{a}_P^{(n,i)}} (\mathbf{g} \bullet \mathbf{x}^{(n,i)}) \nabla \rho_P^{(n,i)} V_P \Big] - \Big[\frac{a_{ii}}{\mathbf{a}_P^{(n,i)}} \nabla p_d^{(n,i)} V_P \Big] \\ \phi^{(n,i)} &= \Big[\frac{\mathbf{H}^{(n,i)}}{\mathbf{a}_P^{(n,i)}} - \frac{a_{ii}}{\mathbf{a}_P^{(n,i)}} (\mathbf{g} \bullet \mathbf{x}^{(n,i)}) \nabla \rho_P^{(n,i)} V_P \Big]_f \bullet \mathbf{S}_f - \Big[\frac{a_{ii}}{\mathbf{a}_P^{(n,i)}} \nabla p_d^{(n,i)} V_P \Big]_f \bullet \mathbf{S}_f \end{aligned} \tag{6.27}$$

6.3 Applications for simple test cases

The commonly used OpenFOAM Euler and CN method are compared to various newly implemented DIRK-type methods using several test cases. Table 6.2 summarizes the time integration schemes presented in this paper. Only Stiffly Accurate (SA) DIRK methods are used up to maximum third-order. In total, the efficiency of the two existing schemes, two DIRK second-order schemes, and three DIRK third-order schemes are compared. For the validation of each time integration scheme, the 2D Taylor-Green vortex benchmark is used to check the order of each scheme carefully. Then, two-phase flow test cases are studied, a periodic wave propagation and a dam break flow. Each of the test cases is described in detail in the following subsections.

Table 6.2. Names and properties of the time schemes.

Code name	Identification name	Order	Implicit stage
Euler	Implicit Euler	1	1
OFCN	OpenFOAM CN	2	1
RKCN	ESDIRK[2](2)_SA	2	1
SDIRK[2](2)	SDIRK[2](2)_SA	2	2
ESDIRK[2](3)_1	ESDIRK[2](3)_SA	2	2
SDIRK[3](3)	SDIRK[3](3)_SA	3	3
ESDIRK[3](3)	ESDIRK[3](3)_SA	3	2
ESDIRK[3](4)	ESDIRK[3](4)_SA	3	3

6.3.1 2D Taylor-Green vortex

2D Taylor-Green vortices are a simple and well-known benchmark to measure the errors of numerical methods quantitatively. The exact solution of the incompressible Navier-Stokes equations exists for the given initial condition. Eqn. (6.28) presents the 2D governing equations, where u and v are the velocity components in x and y direction, respectively; ν is the kinematic viscosity coefficient; ρ is the constant density field.

$$\begin{aligned}
&\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \\
&\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \\
&\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial y} + \nu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right)
\end{aligned} \tag{6.28}$$

The 2D computational domain lies within $-\pi \leq x, y \leq \pi$, and periodic boundary conditions are imposed on all the four boundaries. The initial condition is $u_0 = -\sin(x)\cos(y)$ and $v_0 = \cos(x)\sin(y)$, as illustrated in Figure 6.5. With this initial condition, the analytic velocity and pressure fields satisfying the 2D governing equation are given in Eqn. (6.29) and Eqn. (6.30), respectively. $\mathrm{Re} = LU/\nu$ is the Reynolds number of the flow. The reference length is the domain length $L = 2\pi$, and the reference velocity is the maximum magnitude of the velocity of the initial condition. All computations are made fixing the Reynolds number to $\mathrm{Re} = 10$ with $\rho = 1\mathrm{kg.m}^{-3}$. The meshes used are Cartesian grids, and the second-order central diffirencing (Gauss linear in OpenFOAM notation) spatial discretization method is applied.

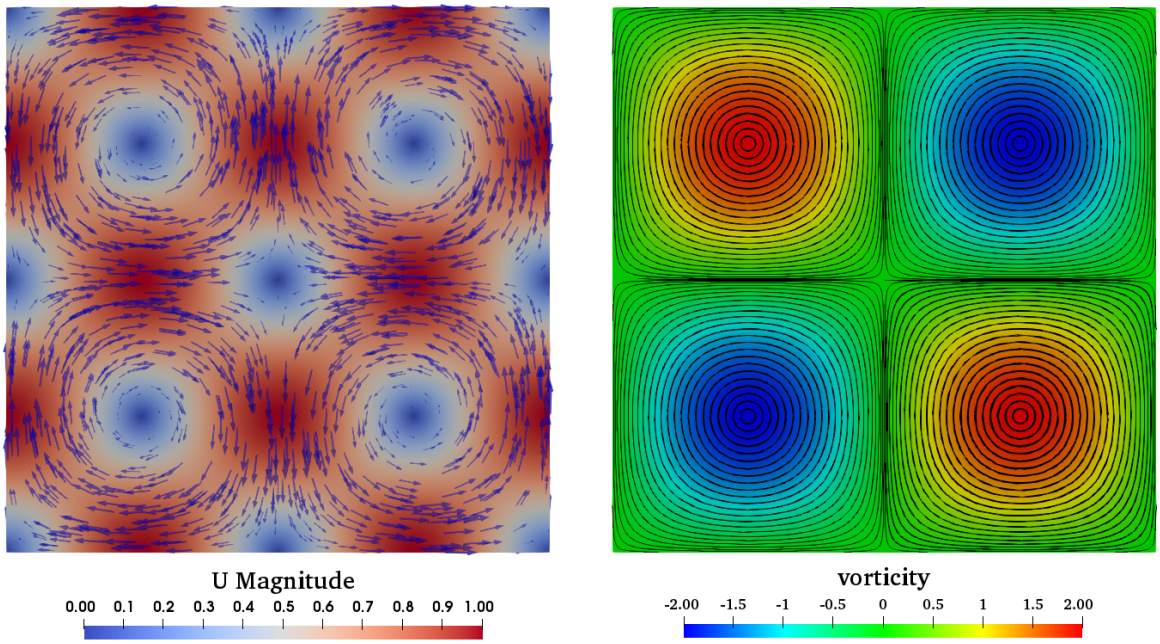


Figure 6.5. Initial condition: velocity magnitude and velocity vector (left) and vorticity and stream lines (right).

$$\mathbf{u}_a = \mathbf{i}u_a + \mathbf{j}v_a \quad , u_a(x,y,t) = -\sin(x)\cos(y)e^{-2\nu t} \quad , v_a(x,y,t) = \cos(x)\sin(y)e^{-2\nu t} \tag{6.29}$$

$$p_a(x,y,t) = \frac{\rho}{4}[\cos(2x) + \cos(2y)]e^{-4\nu t} \tag{6.30}$$

Table 6.3. Test matrix for the spatial convergence on Taylor-Green benchmark.

Number of cells $Nx \times Ny$	$\mathrm{Re}_{\Delta x,y}$	$h = 0.004$s	$C \approx 0.2$	$C \approx 0.02$
4×4	2.50	$C \approx 0.001$	$h = 0.25$s	$h = 0.025$s
8×8	1.25	$C \approx 0.004$	$h = 0.2$s	$h = 0.02$s
16×16	0.63	$C \approx 0.01$	$h = 0.08$s	$h = 0.008$s
32×32	0.31	$C \approx 0.02$	$h = 0.04$s	$h = 0.004$s
64×64	0.16	$C \approx 0.04$	$h = 0.02$s	$h = 0.002$s
128×128	0.08	$C \approx 0.08$	$h = 0.01$s	$h = 0.001$s
256×256	0.04	$C \approx 0.16$	$h = 0.005$s	$h = 0.0005$s
512×512	0.02	$C \approx 0.33$	$h = 0.0025$s	$h = 0.00025$s

The errors and order of convergence are assessed by monitoring field averaged magnitude of

the relative difference between simulation and analytic solution in the entire domain (L_2 norm) at $t = 2s$ as:

$$L_{2,q}(t=2) = \left[\sum_{i=1}^{Ncells} \left(\frac{(q_{CFD}(\mathbf{x}_i) - q_a(\mathbf{x}_i))^2}{\max(|q_a(\mathbf{x}_i)|^2)(Ncells)} \right) \right]_{t=2}^{(1/2)} \tag{6.31}$$

To first check that we correctly recover the second-order convergence from the spatial schemes, several grid refinement levels are used (from 4×4 cells to 512×512). The grid refinement ratio between two successive grids is 2. Table 6.3 summarizes the test conditions, giving the corresponding cell-Reynolds number $\mathrm{Re}_{\Delta x,y} = (\Delta(x,y))U/\nu$, and the correspondence between time-step size h, and Courant number C for each spatial resolution. The convergence of the spatial schemes is first conducted by fixing the time step to $h = 0.004$s, see third column of Table 6.3. Standard OpenFOAM time schemes (implicit Euler and 0.95 off-centered OpenFOAM Crank-Nicolson) and two SDIRK schemes (*SDIRK[2](2)* and *SDIRK[3](3)*) are used for now.

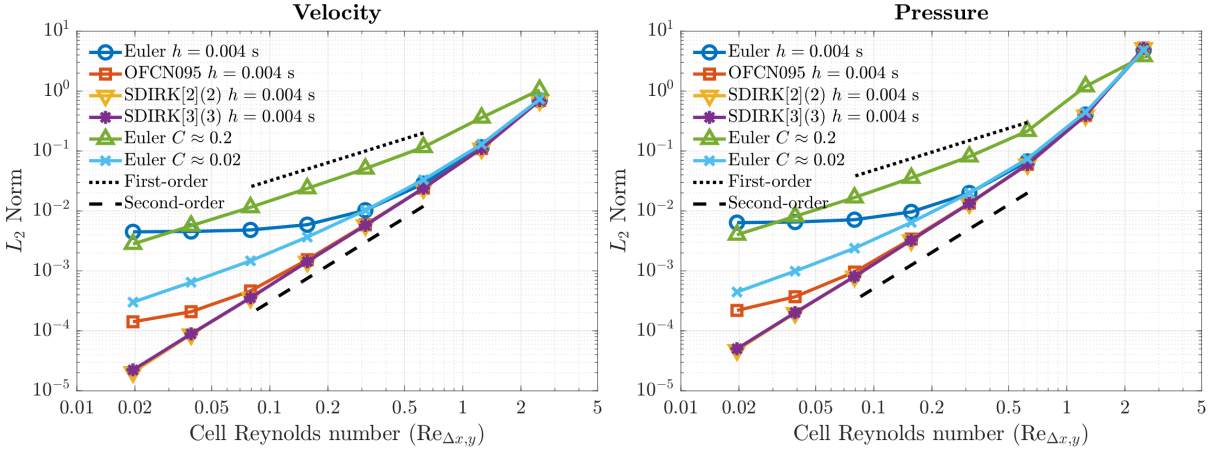


Figure 6.6. Convergence of velocity (left) and pressure (right) magnitude errors with respect to mesh resolution measured through the cell Reynolds number, with fixed time-step size or fixed Courant number.

Figure 6.6 plots the velocity and pressure error evolution with respect to grid refinement for the two time schemes: blue curves with circle symbols for implicit Euler, red curves with square symbols for OpenFOAM CN *OFCN095*, yellow curves with lower triangle symbols for *SDIRK[2](2)*, and purple curves with star symbol for *SDIRK[3](3)*. The black-dashed lines in the figure indicate first- and second-order convergence slopes, respectively. The trends are similar for velocity (left plot) and pressure (right plot). At this constant time step, for the first-order time scheme (implicit Euler) second-order space convergence is observed for the coarsest meshes (up to a cell-Re of the order $O(1)$), and then the space error gets low enough to be dominated by the low-order time-scheme error so that saturation occurs and the curve gets flat for finer meshes (N.B.: this saturation occurs because we are looking at a convergence towards the consistent (analytic) solution, and not at the intrinsic convergence where we would have compared the error between successive pairs of meshes. In the latter case, a monotonic convergence would be

observed till the finest meshes since the time error would not matter).

For the second and third-order time schemes (*OFCN095*, *SDIRK[2](2)*, and *SDIRK[3](3)*) the time error is much lower, so that second-order space convergence is observed up to the finest meshes used. The slope of the *OFCN095* curve slightly decreases for the finest space resolutions showing that we are getting close to the saturation there, while *SDIRK[2](2)* and *SDIRK[3](3)* show clear second-order space convergence. This means that with *OFCN095* time scheme, with the chosen fixed time step, saturation occurs for cell-Re of the order $O(10^{-2})$ instead of $O(1)$ for the first-order time scheme. This is an interesting conclusion since, in view of an optimal (most efficient) use of the schemes, CFD practitioners should seek space-time resolutions, i.e. $(h, \Delta x)$ couples, such that errors are of the same order of magnitude in both time and space. Comparisons of CPU-to-accuracy ratio should then be made in these regions to check whether the use of high-order schemes has actually a benefit for the user. Note that despite its evident lower convergence the implicit Euler scheme, which is still used in some naval CFD applications for its stability, yields similar accuracy when the global error is dominated by the space error (i.e. for coarse space resolutions).

Another interesting way of looking at things is to use Courant numbers. Actually, CFD practitioners usually set in a first step their mesh resolution from complex considerations (free-surface region refinement, boundary-layer refinement, y+ value at the top of first boundary cell, flow features capturing, turbulence model application range, etc.) with, in the end, small variations of mesh resolutions when checking for convergence. Then, with this given mesh, they try to use as high as possible Courant number to get an accurate and kept stable simulation. To this end, we now fix the Courant number, at two different values, 0.2 and 0.02, respectively, see columns 4 and 5 of Table 6.3, which means that the time and space steps will now evolve together being both refined with same ratio. Only the first-order time-scheme results at fixed Courant number are given in Figure 6.6 which permit to better exhibit the different possible behaviors. The corresponding curves are the green one with upper triangle symbols (implicit Euler with $C \approx 0.2$) and the cyan one with cross symbols (implicit Euler with $C \approx 0.02$). It can thus be observed that with the higher Courant number (larger time step for a given grid), error is always dominated by the low-order time-scheme, and a first-order convergence in then observed. While with the lower Courant number (smaller time step for a given grid), the error is first dominated by the space scheme, so that the curve has a second-order slope for the coarsest meshes (high cell-Re) while it decreases to first order for the finest meshes, the error in space decreasing faster than the one in time due to the difference in order of the two schemes. Finally, in terms of error magnitude, dividing the time step by 10 (from $C \approx 0.2$ to $C \approx 0.02$) yields a gain of about one order.

Now that we have checked the space convergence and discussed the links between space and time convergence using as examples the classic OpenFOAM time schemes, we will compare the latter to the DIRK schemes we implemented in terms of time errors and convergence. Table 6.4 lists the ten different time-step sizes used and corresponding Courant numbers. The three former finest grids are used to study the space resolution influence on the results.

Figure 6.7 shows the L_2 norm of velocity and pressure magnitude with respect to the Courant

Table 6.4. Test matrix for the temporal convergence of OpenFOAM with various time schemes on the Taylor-Green vortex benchmark.

time-step size	Number of cells $Nx \times Ny$		
h	128 × 128	256 × 256	512 × 512
0.004	$C = 0.08$	$C = 0.16$	$C = 0.32$
0.01	$C = 0.20$	$C = 0.41$	$C = 0.81$
0.02	$C = 0.41$	$C = 0.81$	$C = 1.63$
0.04	$C = 0.81$	$C = 1.63$	$C = 3.24$
0.08	$C = 1.62$	$C = 3.24$	$C = 6.48$
0.1	$C = 2.04$	$C = 4.07$	$C = 8.15$
0.125	$C = 2.54$	$C = 5.09$	$C = 10.2$
0.2	$C = 4.07$	$C = 8.15$	$C = 16.3$
0.25	$C = 5.09$	$C = 10.2$	$C = 20.4$
0.4	$C = 8.14$	$C = 16.3$	$C = 32.6$

number. Actually, the horizontal scale is reported in terms of Courant number rather than in terms of time-step sizes for the reason explained above regarding the practice of CFD users. Each row corresponds to the results for a different space resolution. The black-dashed lines in the figure indicate first-, second- and third-order convergence slopes, respectively. Looking at this figure the following statements can be made:

- The different schemes globally behave as expected, exhibiting especially convergence slopes close to the theoretical ones for the largest time steps (Courant numbers).

- For the small time-step sizes, saturation of the the L_2 norm is observed for the time schemes of highest orders. This is a similar behavior as the one discussed before, the saturation being here due to a space error higher than the time one for a given fixed-resolution grid. It can be further observed by comparing different space resolution results (different rows of the figure) that, as expected from the space convergence, the levels of the saturation plateaus are lower and lower with the refinement of the grid.

- In terms of time error magnitude, there is a clear advantage to use higher order schemes, even for very large Courant numbers. Besides, within the groups of second-order time schemes, respectively third-order time schemes, some yield errors of about half an order lower than others (before saturation, of course).

- A peculiar behavior is that the convergence curves for the higher-order DIRK time schemes are not monotonous. Actually an over-convergence is found in a limited range of Courant number (for C in $[1-10]$) before saturation occurs due to the space error. As noted before, this is linked to the fact that we are looking at the convergence towards the consistent (analytic) solution and not at the intrinsic convergence. It thus happens that, by chance, the CFD solutions "cross" the analytic one (on average) for some meshes whereas it is still not converged.

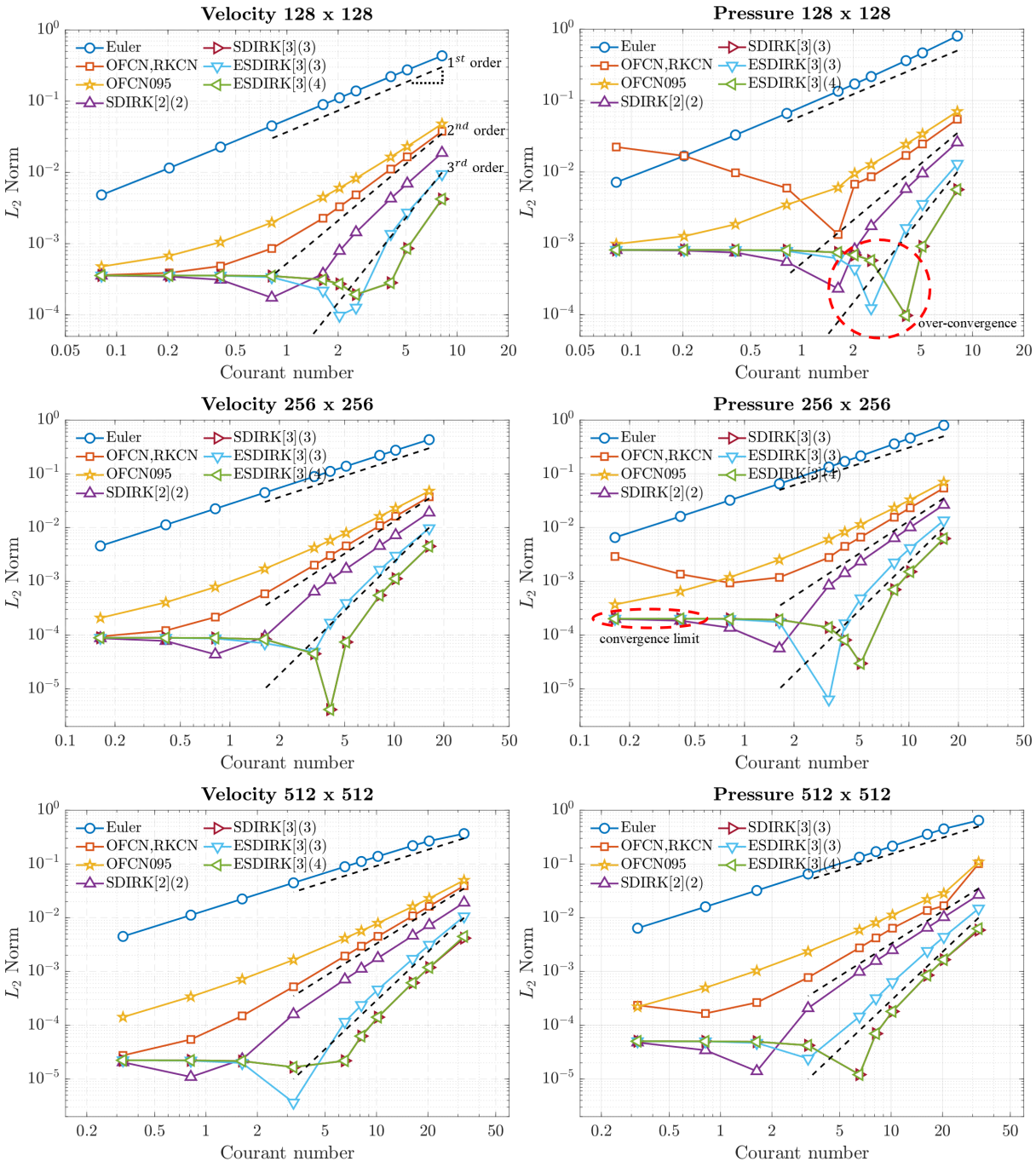


Figure 6.7. Convergence to the analytical solution (see Eqn. (6.31)) of velocity (left plots) and pressure (right plots) magnitude errors with respect to time resolution measured through the Courant number, for three space resolutions: 128 × 128 (top plots), 256 × 256 (middle plots), and 512 × 512 grids (bottom plots).

$$L_{2,q}(t=2\text{s}) = \left[\sum_{i=1}^{Ncells} \left(\frac{\left(q_{\text{CFD}}(\mathbf{x}_i) - q_{\text{CFD}}^{h=0.004\text{s}}(\mathbf{x}_i)\right)^2}{\max(\left|q_{\text{CFD}}^{h=0.004\text{s}}(\mathbf{x}_i)\right|^2) N_{\text{cells}}} \right) \right]_{t=2\text{s}}^{(1/2)} \tag{6.32}$$

To check this, we plot in Figure 6.8 the intrinsic convergence, that is a convergence ac-

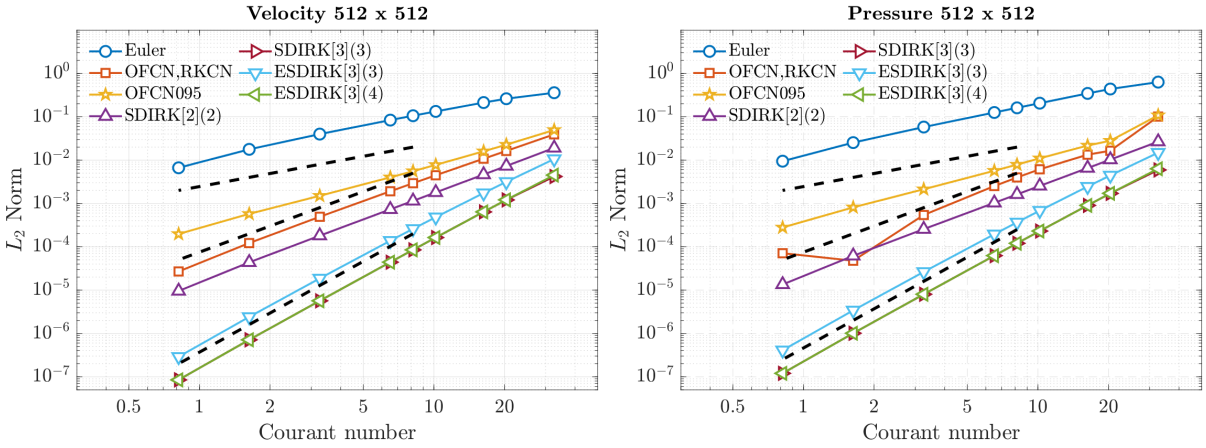


Figure 6.8. Intrinsic convergence (see Eqn. (6.32)) of velocity (left plot) and pressure (right plot) magnitude errors with respect to time resolution measured through the Courant number, for the 512 × 512 space resolution.

cording to Eqn. (6.32) where q_a is replaced by $q_{\mathrm{CFD}}^{h=0.004\mathrm{s}}$ in Eqn. (6.31) with $h = 0.004$s indicating the time-step size listed in Table 6.4. The expected orders are then actually found, with no saturation.

- Finally, it is noticeable that similar trends are again observed in terms of both velocity and pressure. However, one scheme has a different behavior in terms of pressure, the *OFCN_RKCN* one. We remind that the existing *OFCN* and the proposed *RKCN* are strictly identical in this single-phase flow situation, so that they are noted *OFCN_RKCN*. Actually, with the latter scheme instead of reaching a saturation plateau for the smallest time steps (Courant numbers), the error increases dramatically. This instability of the *OFCN_RKCN* scheme can be remedied using the off-centering of the CN coefficient, which however induces a loss of accuracy of the solution. Eqn. (6.33) describes an arbitrary ODE ($\dot{y} = f(t, y(t))$) using the off-centered CN scheme, where C_{oc} is an off-centering coefficient, and $C_{CN} = 1/(1 + C_{oc})$ is the Crank-Nicolson coefficient. Using this off-centering the *OFCN095* (or identically, *RKCN095*) scheme actually shows regular convergence, but with a smaller order of convergence (around 1.5) compared to the classic CN scheme.

$$\frac{y^{(n+1)} - y^{(n)}}{h} = C_{CN} f(t^{(n+1)}, y(t^{(n+1)})) + (1 - C_{CN}) f(t^{(n)}, y(t^{(n)})) \tag{6.33}$$

- The two third-order schemes *SDIRK[3](3)* and *ESDIRK[3](4)* show almost identical behavior, while these two schemes use different Butcher matrix coefficients.

6.3.2 Regular wave propagation in periodic domain

This section presents the application of DIRK schemes to the regular periodic wave propagation test case. Three wave conditions used for wave propagation are described in Section 3.1 with its wave theory. Table 6.5 shows the information on the grids and temporal resolutions for the periodic wave propagation. The vertical mesh size is defined based on wave height (H).

Thus, the mesh aspect ratio $\Delta x/\Delta z$ changes with respect to the wave condition. The first four cases have the same Courant number to provide the spatial convergence. The last five cases are expected to show the effect of the temporal discretization scheme to wave propagation. The reference case of the simulation campaign is 'Grid3 T/h t400', and it is listed twice in the table as it is used in both convergence studies. This reference case is recognized as having sufficient conditions for accurate wave propagation [89]. The total simulation time is 40 wave periods. The initial maximum Courant number for the reference case is around $C = 0.2 \sim 0.3$, with respect to its wave condition.

Table 6.5. Four mesh resolutions and five temporal resolutions (gray colored rows are the same condition)

Item	$\lambda/\Delta x$	$H/\Delta z$	T/h
Grid 1 T/h 100	25	5	100
Grid 2 T/h 200	50	10	200
Grid 3 T/h 400	100	20	400
Grid 4 T/h 800	200	40	800
Grid 3 T/h 100	100	20	100
Grid 3 T/h 200	100	20	200
Grid 3 T/h 400	100	20	400
Grid 3 T/h 800	100	20	800
Grid 3 T/h 1600	100	20	1600

In order to minimize the non-linear iteration uncertainty, the wave propagation simulation imposed 8 PIMPLE (outer iteration) and 2 PISO (inner iteration) loops. The initial residual of velocity components and dynamic pressure reduced to 10^{-6} within these iterations. The summary of spatial discretization schemes used for wave propagation is listed in Table 4.1. A good choice of discretization scheme could be essential for achieving accurate and stable simulations. No non-orthogonal corrector is applied since the computation grids are fully orthogonal.

Figure 6.9 presents the error of normalized wave amplitude averaged from the first ten wave periods. Each line is the result obtained with one temporal scheme. The left plots represent the spatial convergence of wave amplitude with respect to the cell size, taken as the number of cells' square root. The right plots represent the temporal convergence for the maximum Courant number measured at the initial state. The black dotted line is a guideline for the comparison of the rate of convergence. The order of each dashed guideline is given in the legend of each figure.

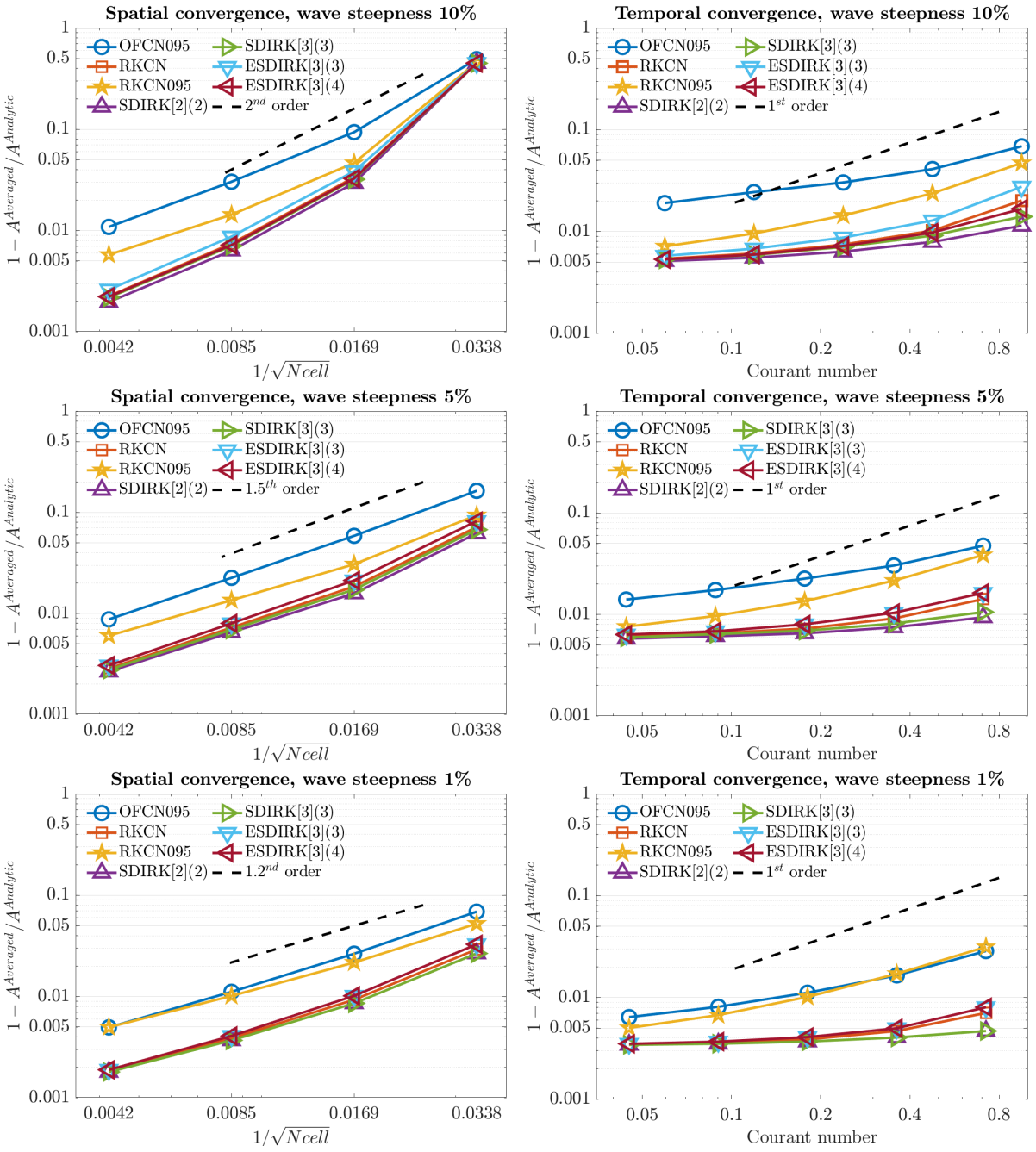


Figure 6.9. Convergence of normalized wave amplitude for three different wave conditions. Left is spatial convergence and right is temporal convergence.

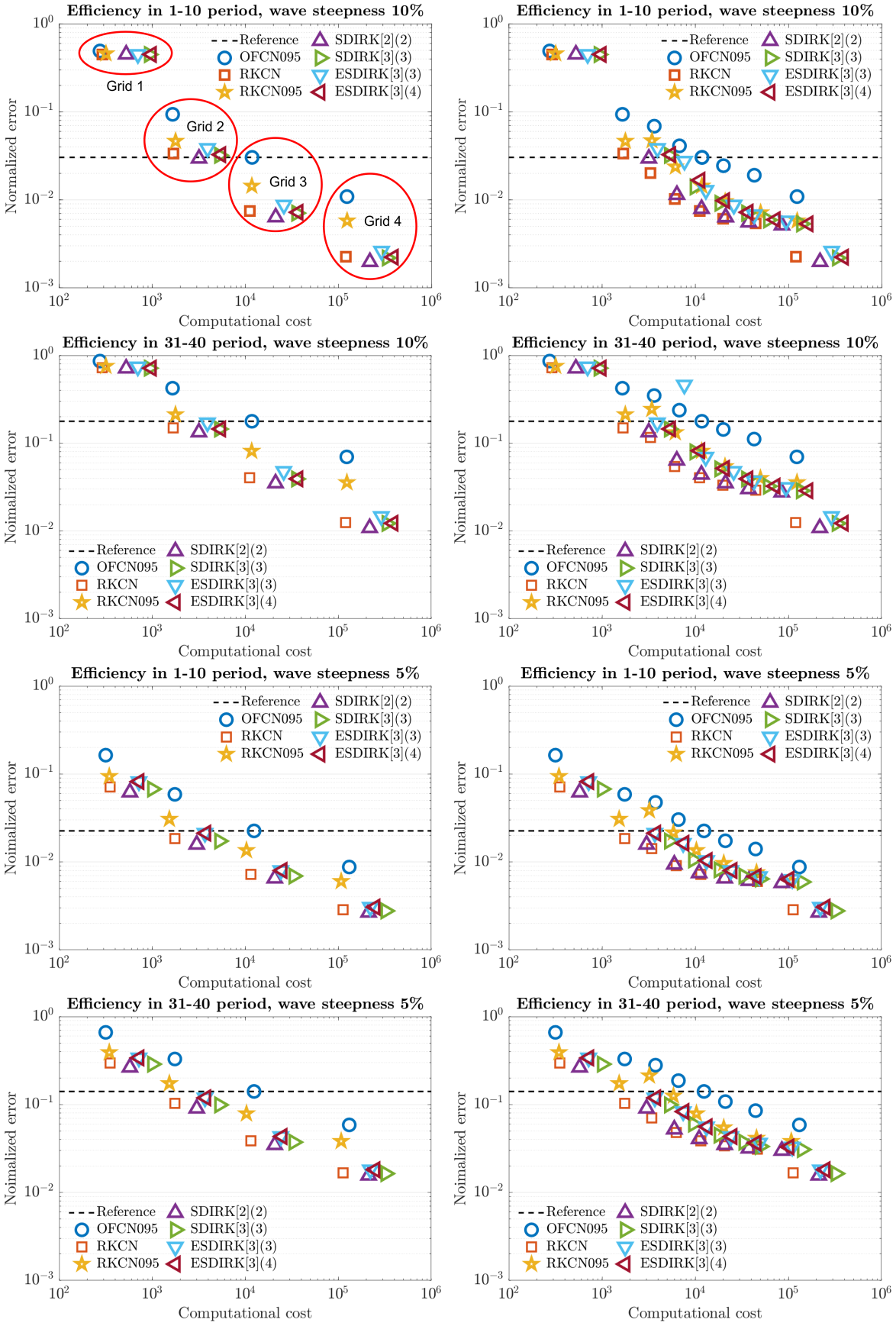


Figure 6.10. Comparison of the efficiency of the solvers, wave steepness 10% and 5%.

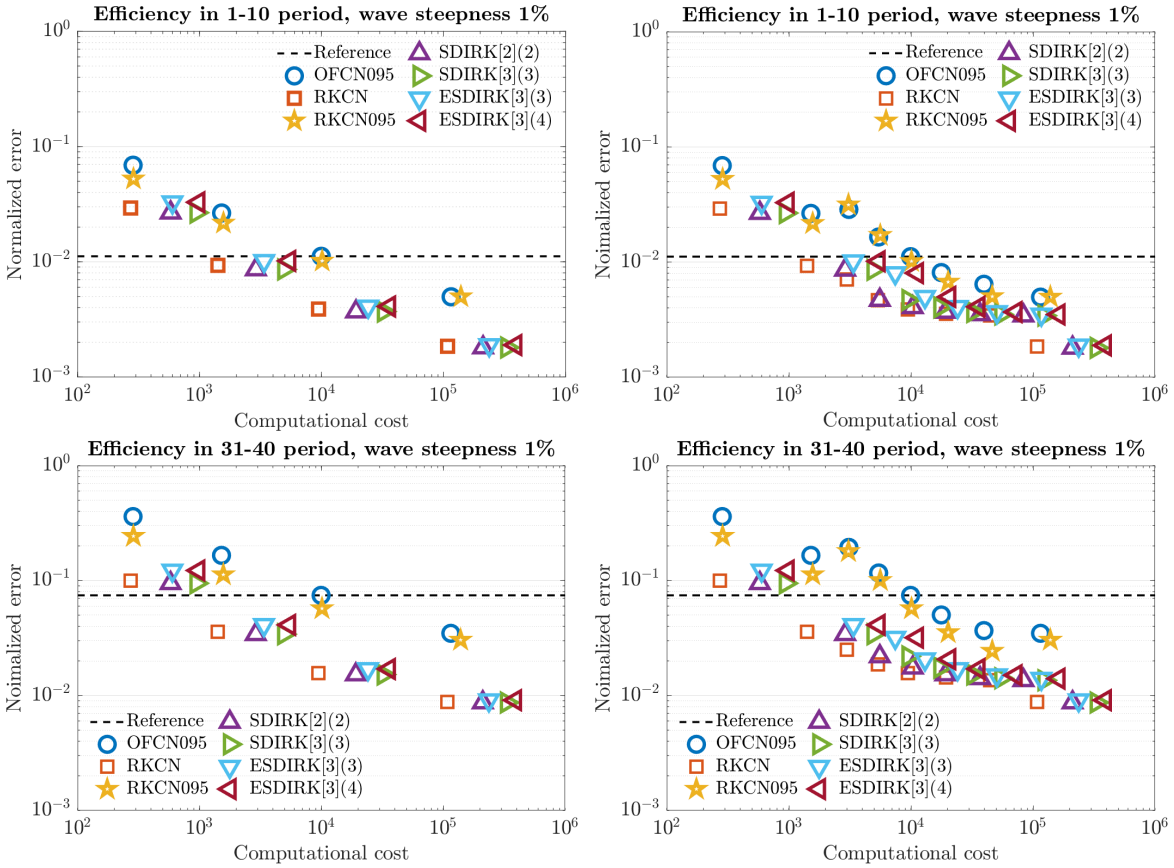


Figure 6.11. Comparison of the efficiency of the solvers, wave steepness 1%.

The temporal schemes referred to as *OFCN095* and *RKCN095* are 0.95 off-centered CN schemes also used in the Taylor-Green vortex benchmark. For the two-phase flow simulation, the time schemes *OFCN095* and *RKCN095* are different on the implementation of VOF flux, as discussed in the previous section. Comparing two time schemes *OFCN095* and *RKCN095*, *RKCN095* scheme was more accurate than *OFCN095* scheme for the most of the computations. It was not possible to run wave propagation with *OFCN* scheme without off-centering. For all cases, the *RKCN095* scheme was less accurate and lower convergence rate than the original *RKCN* scheme.

With 10% wave steepness, the second-order spatial convergence is obtained; however, the order of convergence decreases with the wave steepness. It means that the second-order spatial schemes do not guarantee the second-order convergence for a two-phase flow simulation. For all time integration schemes, the rate of temporal convergence is lower than first-order. While no order of convergence has been found, the accuracy is different between the schemes. The higher-order schemes provide better accuracy than the lower-order schemes, especially for the high Courant number.

What matters practically is not the order of the schemes but the efficiency of each temporal discretization scheme, which is defined here as the lowest CPU time for a given normalized error criterion. Figure 6.10 shows the normalized wave amplitude error for the wave steepness 10%

and 5%. Y-axis is the normalized error averaged in time, and X-axis is the total computation time. Both axes are in log scale. For the multi-core computations, the computation time is linearly multiplied by the number of the cores. On the 40 periods simulated in each case, the averaged error is computed for the first ten and the last ten periods. The black dotted line represents the reference test case error, which uses the *OFCN095* scheme with the computation condition 'Grid3 T/h 400'. The figures on the left plot result from Grid 1 $\sim$ 4, and the right figures include all test cases. These cases are divided to measure the error from mesh resolution and temporal resolution separately. The spatial convergence result shows that the resolution 'Grid1 T/h 100' is not appropriate for steep wave simulation. When comparing time integration schemes considering the level of error of the reference data, *RKCN* and *SDIRK[2](2)* schemes show the best efficiency. This conclusion also holds for higher and lower levels of error. The third-order schemes are as accurate as a second-order scheme but less efficient than second-order DIRK schemes.

Similarly, Figure 6.11 summarized the efficiency of each time integration schemes for wave steepness 1%. Only data for the first ten periods and the last ten periods are illustrated. Similar to what was obtained with the 10% wave steepness, *RKCN* and *SDIRK[2](2)* schemes are the most efficient.

6.4 Discretization of SWENSE model with DIRK method

This section describes the formulation of the DIRK method with the SWENS equation. The VOF field's convection with the DIRK method is identical to the original one and described in Section 6.2.1. Similar to the PISO algorithm described in Section 6.2.3, it couples the complementary pressure and the complementary velocity using the momentum equation and the continuity equation. Eqn. (6.34) recalls the SWENSE momentum equation derived in Section 2.2.2.

$$\frac{\partial \rho \mathbf{u}_C}{\partial t} + \nabla_\bullet(\rho \mathbf{u}\mathbf{u}_C) + \rho \mathbf{u}_C \bullet \nabla \mathbf{u}_I = -\nabla p_C - \frac{p_I}{\rho_w}\nabla \rho + \nabla_\bullet(\mu_E(\nabla \mathbf{u}_C + \nabla \mathbf{u}_C^T)) \tag{6.34}$$

If a grid does not move in time, the finite volume representation of the SWENSE momentum equation reads:

$$\frac{(V\rho \mathbf{u}_C)_P^{(n,i)} - (V\rho \mathbf{u}_C)_P^{(n)}}{h} + a_{ii}\mathbf{R}_C^{(n,i)} + \sum_{j=1}^{i-1} a_{ij}\mathbf{R}_C^{(n,j)} = 0 \tag{6.35}$$

where $\mathbf{R}_C$ includes all components except the time discretization.

$$
\begin{aligned}
\mathbf{R}_C^{(n,i)} &= \sum_f \left[(\rho\phi)_f^{(n,i)}\mathbf{u}_{C,f}^{(n,i)}\right] - \sum_f \left[\mu_f^{(n,i)}\mathbf{S}_f\bullet(\nabla\mathbf{u}_{C,f}^{(n,i)})\right] \\
&\quad - \nabla\mathbf{u}_{C,P}^{(n,i)}\bullet\nabla\mu^{(n,i)}V_P + \rho\mathbf{u}_{C,P}^{(n,i)}\bullet\nabla\mathbf{u}_{I,P}^{(n,i)}V_P + \nabla p_C^{(n,i)}V_P + \frac{p_I^{(n,i)}}{\rho_w}\nabla\rho_P^{(n,i)}V_P \\
\mathbf{R}_C^{(n,j)} &= \sum_f \left[(\rho\phi)_f^{(n,j)}\mathbf{u}_{C,f}^{(n,j)}\right] - \sum_f \left[\mu_f^{(n,j)}\mathbf{S}_f\bullet(\nabla\mathbf{u}_{C,f}^{(n,j)})\right] \\
&\quad - \nabla\mathbf{u}_{C,P}^{(n,j)}\bullet\nabla\mu^{(n,j)}V_P + \rho\mathbf{u}_{C,P}^{(n,j)}\bullet\nabla\mathbf{u}_{I,P}^{(n,j)}V_P + \nabla p_C^{(n,j)}V_P + \frac{p_I^{(n,j)}}{\rho_w}\nabla\rho_P^{(n,j)}V_P
\end{aligned}
\tag{6.36}
$$

The mass flux $(\rho\phi)_f^{(i)}$ is calculated from the VOF convection equation, and it is constant in the PISO iteration.

$$
\begin{aligned}
&\frac{(V\rho\mathbf{u}_C)_P^{(n,i)} - (V\rho\mathbf{u}_C)_P^{(n)}}{h} + \sum_{j=1}^{i-1} a_{ij}\mathbf{R}_C^{(n,j)} + a_{ii}\Big[\sum_f \left[(\rho\phi)_f^{(n,i)}\mathbf{u}_{C,f}^{(n,i)}\right] \\
&- \sum_f \left[\mu_f^{(n,i)}\mathbf{S}_f\bullet(\nabla\mathbf{u}_{C,f}^{(n,i)})\right] - \nabla\mathbf{u}_{C,P}^{(n,i)}\bullet\nabla\mu^{(n,i)}V_P + \rho\mathbf{u}_{C,P}^{(n,j)}\bullet\nabla\mathbf{u}_{I,P}^{(n,j)}V_P\Big] \\
&\qquad = -a_{ii}\big[\nabla p_C^{(n,i)}V_P + \frac{p_I^{(n,i)}}{\rho_w}\nabla\rho_P^{(n,i)}V_P\big]
\end{aligned}
\tag{6.37}
$$

$$
\mathrm{U_C Eqn}^{(n,i)} : \mathbf{a}_{C,P}^{(n,i)}\mathbf{u}_{C,P}^{(n,i)} - \mathbf{H}_C^{(n,i)} = -a_{ii}\big[\nabla p_C^{(n,i)}V_P + \frac{p_I^{(n,i)}}{\rho_w}\nabla\rho_P^{(n,i)}V_P\big] \tag{6.38}
$$

The notations $\mathbf{a}_P^{(n,i)}$ indicates the diagonal components of $\mathrm{U_C Eqn}^{(n,i)}$, and $\mathbf{H}^{(n,i)}$ includes the all off-diagonal and source components. The velocity $\mathbf{u}_{C,P}^{(n,i)}$ from the LHS of Eqn. (6.38) must satisfy the continuity condition. The substitution of velocity to the continuity equation yields to Eqn. (6.39).

$$
\nabla\bullet\mathbf{u}_{C,P}^{(n,i)} = \nabla\bullet\Big[\frac{\mathbf{H}_C^{(n,i)}}{\mathbf{a}_{C,P}^{(n,i)}} - \frac{a_{ii}}{\mathbf{a}_{C,P}^{(n,i)}}\big[\nabla p_C^{(n,i)}V_P + \frac{p_I^{(n,i)}}{\rho_w}\nabla\rho_P^{(n,i)}V_P\big]\Big] = 0 \tag{6.39}
$$

$$
\mathrm{p_C Eqn}^{(n,i)} : \sum_f \Big[\Big(\frac{a_{ii}V_P}{\mathbf{a}_{C,P}^{(n,i)}}\nabla p_C^{(n,i)}\Big)_f\Big]\bullet\mathbf{S}_f = \sum_f \Big[\frac{\mathbf{H}_C^{(n,i)}}{\mathbf{a}_{C,P}^{(n,i)}} - \frac{a_{ii}}{\mathbf{a}_{C,P}^{(n,i)}}\frac{p_I^{(n,i)}}{\rho_w}\nabla\rho_P^{(n,i)}V_P\Big]_f\bullet\mathbf{S}_f \tag{6.40}
$$

Eqn. (6.40) solves the Laplacian equation to obtain the complementary pressure, where the subscript $_f$ represents the cell center to face center interpolation. The velocity and flux are corrected using complementary pressure, and the pressure and complementary velocity are strongly coupled inside the non-linear iteration.

$$
\begin{aligned}
\mathbf{u}_C^{(n,i)} &= \Big[\frac{\mathbf{H}_C^{(n,i)}}{\mathbf{a}_{C,P}^{(n,i)}} - \frac{a_{ii}}{\mathbf{a}_{C,P}^{(n,i)}}\frac{p_I^{(n,i)}}{\rho_w}\nabla\rho_P^{(n,i)}V_P\Big] - \Big[\frac{a_{ii}}{\mathbf{a}_{C,P}^{(n,i)}}\nabla p_C^{(n,i)}V_P\Big] \\
\phi^{(n,i)} &= \Big[\frac{\mathbf{H}_C^{(n,i)}}{\mathbf{a}_{C,P}^{(n,i)}} - \frac{a_{ii}}{\mathbf{a}_{C,P}^{(n,i)}}\frac{p_I^{(n,i)}}{\rho_w}\nabla\rho_P^{(n,i)}V_P\Big]_f\bullet\mathbf{S}_f - \Big[\frac{a_{ii}}{\mathbf{a}_{C,P}^{(n,i)}}\nabla p_C^{(n,i)}V_P\Big]_f\bullet\mathbf{S}_f
\end{aligned}
\tag{6.41}
$$

6.5 DIRK method with moving body and grids

If the mesh moves during the simulation, the change of quantities in a cell due to the mesh movement must be considered with a method called Arbitrary Lagrangian-Eulerian (ALE) formulation. As discussed in Section 4.6.3, the ALE formulation can be applied using the the mesh flux (ϕ_{mesh}).

In this section, the ALE formulation is applied to both averaged two-phase flow model and the SWENSE model with the DIRK method. The application of ALE formulation to VOF convection equation writes:

$$\begin{aligned}\frac{V^{(n,i)}_{P^{(n+1)}}\alpha^{(n,i)}_{P^{(n+1)}} - V^{(n)}_{P^{(n)}}\alpha^{(n)}_{P^{(n)}}}{h} - \sum_{f^{(n,i)}}(\phi_{mesh}\alpha_f)^{(n,i)} + a_{ii}\sum_{f^{(n,i)}}(\phi_f\alpha_f)^{(n,i)} \\ + a_{ii}\sum_f(\phi_{r,f}\alpha_f(1-\alpha_f))^{(n,i)} + \sum_{j=1}^{i-1}a_{ij}\sum_{f^{(n,j)}}(\phi_f\alpha_f)^{(n,j)} \\ + \sum_{j=1}^{i-1}a_{ij}\sum_f(\phi_{r,f}\alpha_f(1-\alpha_f))^{(n,j)} = 0.\end{aligned} \tag{6.42}$$

The additional term, mesh flux (ϕ_{mesh}), is also adapted to *Predictor* & *Corrector* algorithm.

Similar to the VOF convection equation, the application of the ALE method to conditionally averaged momentum equation yields Eqn. (6.43), where the $\mathbf{R}$ is defined to Eqn. (6.22).

$$\frac{(V\rho\mathbf{u})^{(i)}_P - (V\rho\mathbf{u})^{(n)}_P}{h} - \sum_f\left[(\rho\phi_{mesh})^{(i)}\mathbf{u}^{(i)}_f\right] + a_{ii}\mathbf{R}^{(i)} + \sum_{j=1}^{i-1}a_{ij}\mathbf{R}^{(j)} \tag{6.43}$$

Similarly, the SWENSE momentum equation with the ALE method yields Eqn. (6.44).

$$\frac{(V\rho\mathbf{u}_C)^{(n,i)}_P - (V\rho\mathbf{u}_C)^{(n)}_P}{h} - \sum_f\left[(\rho\phi_{mesh})^{(i)}\mathbf{u}^{(i)}_{C,f}\right] + a_{ii}\mathbf{R}^{(n,i)}_C + \sum_{j=1}^{i-1}a_{ij}\mathbf{R}^{(n,j)}_C = 0 \tag{6.44}$$

The discretization method of the convection terms with the mesh flux in both VOF transportation equation and momentum equations are identical to the discretization method of their convection term.

7 Non-linear wave generation and qualification

For naval applications, an accurate and robust solver to generate and propagate waves is essential. This chapter aims to present the quality assessment of regular wave and irregular wave propagation in a Numerical Wave Tank (NWT) with various simulation parameters and numerical schemes. For all simulations, the wave generation and the wave absorption in the NWT are based on an explicit relaxation scheme. There are a large number of possible numerical setups, considering both possible variations in the discretization parameters and in the numerical schemes. The results obtained with different configurations are compared. In an attempt to maximize the cost-error efficiency, some simulation parameters for regular waves and irregular waves are recommended, such as mesh resolution and time-step size. Additionally, the effects of varying some numerical methods or schemes, such as turbulence models, boundary conditions, and high-order time integration schemes, are discussed. This chapter is separated into two parts. The first part presents the regular waves qualification, and the second part presents the irregular waves qualification.

7.1 Regular wave propagation in a NWT

In CFD for marine engineering, the simulations with regular waves in a NWT allow for obtaining relevant information about the behaviour of ships or offshore structures, keeping the computational cost acceptable compared to open-sea simulations. This section presents the qualification of regular wave propagation in a NWT. Differently to what would be done typically, the total duration of the numerical runs for this test case is 100 periods. Checking the accuracy and stability of the simulations with this long duration gives information in view of the final goal of simulating 3 hours of irregular waves, which is a typical standard for engineering design. The newly developed temporal discretization schemes presented in Chapter 6 are applied and compared with the original OpenFOAM schemes.

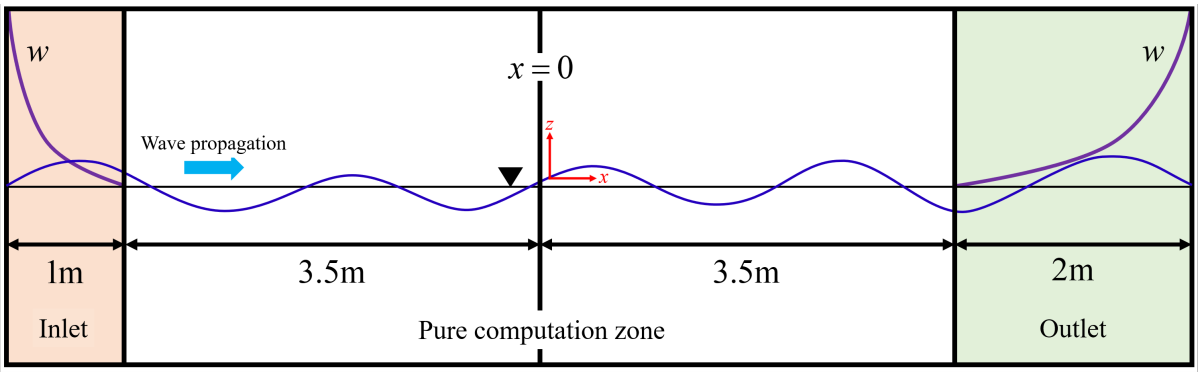


Figure 7.1. Schematic view of the NWT.

The wave qualification procedure we defined uses three deepwater regular wave conditions. These wave conditions have for steepnesses $H/\lambda = 10\%$, 5%, and 1%. Fully non-linear stream function theory is used as reference for these waves, and to define the incoming wave conditions

in the inlet zone of the NWT. Note that in the three wave conditions the wavelength $\lambda = 1$m is kept. Detailed information on the wave conditions can be found in Table 3.1. Figure 7.1 shows a schematic view of the NWT for the regular wave propagation. The total length of the computational domain is 10λ (10m). The wave is propagating from the inlet to the outlet. Two relaxation zones are introduced for the generation and the absorption of waves in the inlet and outlet zones. The static exponential weight function of Eqn. (4.18) is applied in these two zones since a previous study showed it to yield the best preservation of the wave amplitude. Choi et al. found that at least 0.5λ-long relaxation zone is required for wave generation, and a longer relaxation zone is required for optimal wave absorption [96]. Therefore, the inlet length (L^{inlet}) is taken to be 1λ, and the outlet length (L^{outlet}) is set to 2λ. More details on the relaxation zone and the weight functions can be found in Section 4.5.

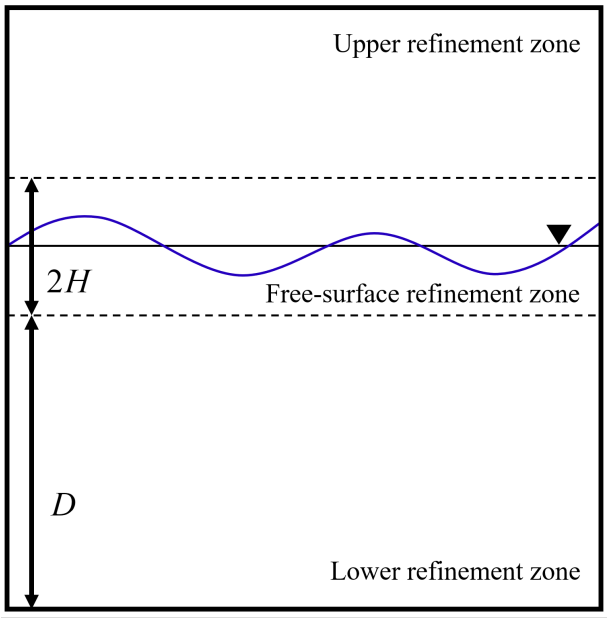


Figure 7.2. Three mesh refinement zones.

The total length of the domain is 10m, and the pure computational domain is thus 7m. The origin of the Cartesian coordinate system is at the center of the pure computational zone. Therefore, the whole computational domain is from $x = -4.5$m to $x = 5.5$m. In the computational domain, 500 wave probes are uniformly distributed from inlet to outlet to monitor the wave amplitude in time. Time series and moving window FFT analyses are performed mainly on the wave elevation measured at two positions: $x = 0$m, and $x = 2.5$m.

In wave simulations, the numerical domain is usually divided in several zones, because most of the kinetic energy is around the free surface, and it would be very costly to keep the same refinement for the whole domain. Figure 7.2 displays the three different refinement zones used in this study. The free-surface refinement zone is from $z = -0.1$m to $z = 0.1$m, which is twice the biggest wave height. The upper refinement zone and lower refinement zone have similar mesh sizes near the free-surface refinement zone and are then linearly stretched when moving away

from the free surface. The upper refinement zone has 25 cells in the vertical direction, the lower refinement zone has 75 cells, and the free-surface refinement zone has 40 cells. The corresponding cell aspect ratio in the free-surface refinement zone is $\Delta z/\Delta x = 0.5$.

The following presents a summary of the regular wave propagation test campaign and analysis performed in this section.

- Three different wave steepnesses, 1%, 5%, and 10%, are used for the quality assessment of the regular waves.
- A comparison between the standard $k-\omega$ SST turbulence model and the free-surface $k-\omega$ SST turbulence model is investigated. Also, the numerical stability of the free-surface $k-\omega$ SST model is discussed.
- A comparison of the efficiency of the DIRK methods both with the standard averaged two-phase flow solver and the SWENSE solver is discussed.

7.1.1 NWT with regular waves: Study on the free-surface turbulence model

Generally, the waves with small steepness are expected to be little affected by turbulence. However, when the targeted wave is the breaking wave condition, the effect of turbulence viscosity might be high. Moreover, for naval applications, such as maneuvering and estimating added-resistance, it is important to simulate wave structure interaction with an adequate turbulence model.

The standard $k-\omega$ SST turbulence model for the two-phase flow wave propagation is known to excessively damp the waves during the propagation, and Larson introduced a modified 'free-surface $k-\omega$ SST' model to avoid the overdamping of the wave energy [119]. The free-surface $k-\omega$ SST turbulence model is described in Section 2.3. In the present section, the regular wave propagation results obtained with both models are compared. Namely, the amount of wave dissipation obtained with the two turbulence models is compared looking at the first harmonic computed using a moving window FFT. The mesh used has a free-surface refinement of $\Delta z/\Delta x = 0.5$ and $\lambda/\Delta x = 100$. The boundary conditions for each boundary are also identical to the previous section. The other spatial schemes are identical to the NWT simulations without the turbulence model. Eight PIMPLE iterations and two PISO iterations are applied for the non-linear iterations. The two transportation equations for the free-surface $k-\omega$ SST model are solved only at the last PIMPLE loop. The discretizations of the convection terms of the k and ω transportation equations use the `Gauss upwind` scheme. As described in Section 4.9, only the `zeroGradient` boundary condition is applied for variables k and ω since there is no body boundary or symmetric boundary in this test case. The initial conditions are given for the turbulence variable fields, which are $k = 1.0 \times 10^{-6}$ m^{2}s^{-2} and $\omega = 100$ s^{-1}. This initial condition corresponds to turbulence viscosity ratio $\nu/\nu_t = 0.01$.

The DIRK-based Crank-Nicolson scheme with 0.99 off-centering ($RKCN$) is used here, following the conclusion of the previous section. However, regardless of the temporal scheme used for the other governing equations, the two $k-\omega$ SST turbulence equations are solved using only the Euler time integration scheme since other choices were not successful (divergence of k and

ω).

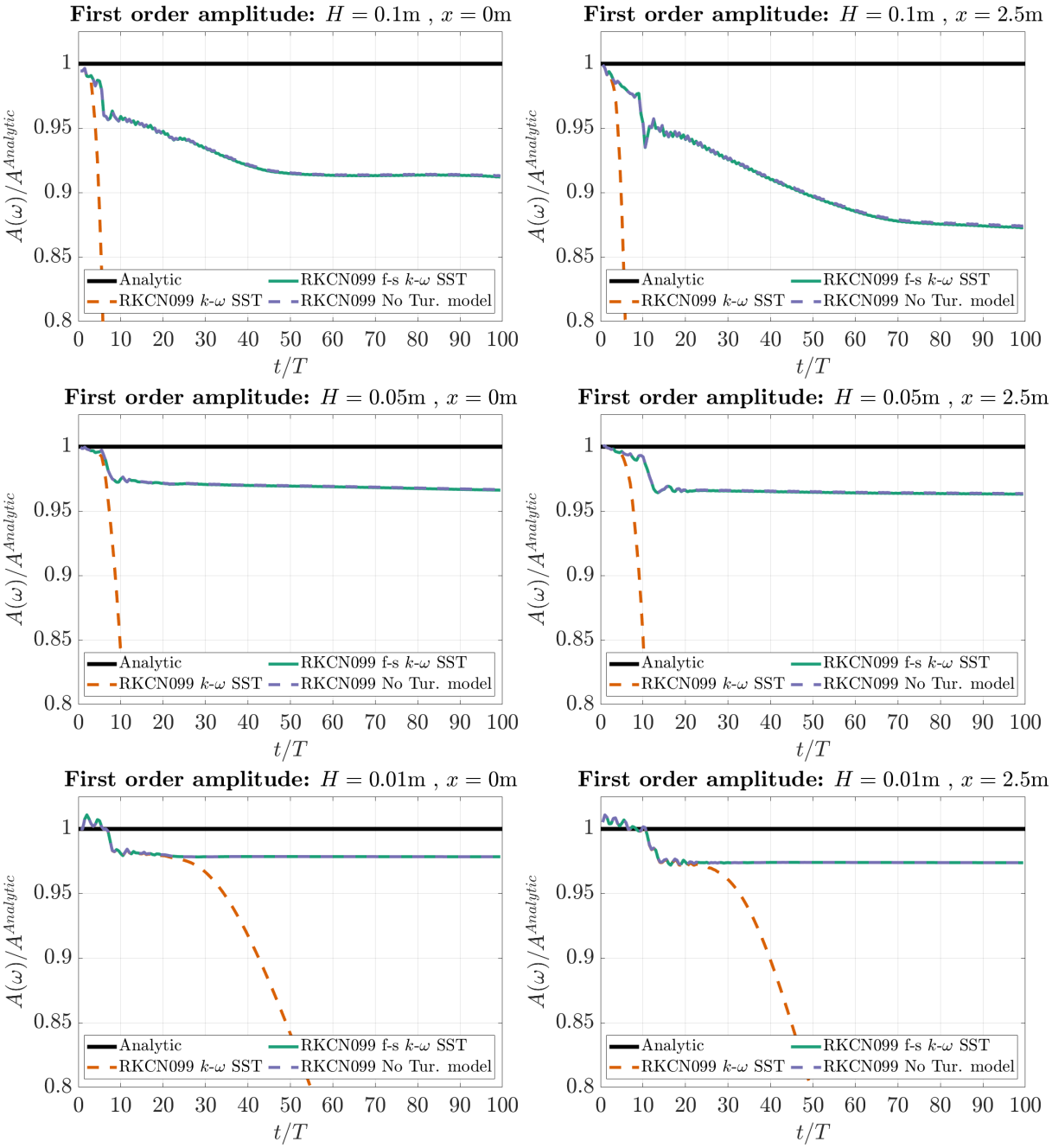


Figure 7.3. Comparison between standard $k-\omega$ SST models, free-surface $k-\omega$ SST and no turbulence model, for wave steepnesses 10%, 5% and 1% at positions $x = 0$m, and $x = 2.5$m.

Figure 7.3 shows the first-harmonic wave amplitudes at $x = 0$m and $x = 2.5$m for wave steepnesses 10%, 5%, and 1% in time. Three turbulence models are compared: without turbulence model (the laminar model in OpenFOAM notation), the standard $k-\omega$ SST model, and the free-surface $k-\omega$ SST model. The standard $k-\omega$ SST model dramatically increases non-physical turbulence viscosity under the free-surface within a few wave periods. The increase

of turbulence viscosity causes a dramatic decrease in wave amplitude. As a result an amplitude drop is observed in all plots of Figure 7.3. The increase of non-physical turbulence viscosity gets larger with higher wave steepness. The difference of wave amplitude between simulations with and without the free-surface $k-\omega$ SST model is almost inexistent for these three wave conditions. This is expected and desired from the free-surface $k-\omega$ SST model because the wave propagation itself should not generate large turbulence eddy viscosity near the free-surface without a floating body or a wave breaking event.

7.1.2 NWT with Regular waves: Efficiency of different time integration schemes in foamStar and foamStarSWENSE

The efficiency of the DIRK time integration schemes described in Chapter 6 with both the averaged two-phase flow solver (*foamStar*) and the SWENSE solver (*foamStarSWENSE*) is studied in this section. Only the 10%-steepness wave is considered to estimate the efficiency of each combination of solver and time scheme because the previous periodic wave propagation test case in Section 6.3 verified that the efficiency of the scheme is similar regardless of the wave steepness.

Table 7.1. Names and properties of the DIRK schemes used for NWT simulations.

Code name	Identification name	Order	Implicit stages
Euler	Implicit Euler	1	1
RKCN	ESDIRK[2](2)_SA	2	1
SDIRK[2](2)	SDIRK[2](2)_SA	2	2
ESDIRK[3](2)	ESDIRK[3](2)_SA	2	2
SDIRK[3](3)	SDIRK[3](3)_SA	3	3
EDIRK[3](3)_1	EDIRK[3](3)_SA_1	3	2
ESDIRK[3](3)	ESDIRK[3](3)_SA	3	2
EDIRK[3](3)_2	EDIRK[3](3)_SA_2	3	2

The appropriate turbulence model (free-surface $k-\omega$ SST) and the aspect ratio $\Delta z/\Delta x = 0.5$ are selected from the analysis done in previous sections. The spatial discretization schemes and the boundary conditions are identical to previous NWT simulations and are listed in Table 4.1. As previously, 8 PIMPLE and 2 PISO iterations are used.

Table 7.1 recalls some DIRK schemes and basic time integration schemes applied to the NWT test case. In total eight time integration schemes are compared, both with the two-phase flow solver and the SWENSE solver, including the implicit Euler scheme. The OpenFOAM default Crank-Nicolson (*OFCN*) scheme is not tested again because its accuracy was previously compared with the *RKCN* scheme.

All solvers are tested for a set of 12 combinations of spatial and time resolutions, and Table 7.2 tabulates the test matrix. Therefore, the simulation campaign includes 12 combinations of resolutions, two solvers, and eight DIRK schemes, so that in total 192 cases are simulated. The

set of resolutions $\Delta x = \lambda/25$ and $h = T/50$ is too low for proper wave simulation for the averaged two-phase flow solver. However, the SWENSE solver is expected to show already efficiency for such low-resolution cases.

Table 7.2. Test matrix for the assessment of the efficiency of each solver and time integration schemes (three mesh resolutions and four temporal resolutions).

Horizontal cell size	$h = T/400$	$h = T/200$	$h = T/100$	$h = T/50$
$\Delta x = \lambda/100$	o	o	o	o
$\Delta x = \lambda/50$	o	o	o	o
$\Delta x = \lambda/25$	o	o	o	o

First harmonic amplitudes of wave elevation from the moving FFT

Figures 7.4, 7.5 and 7.6 show the first harmonic of wave elevation obtained by the moving window FFT for all the 192 cases simulated. The black solid line is the 1st harmonic wave amplitude from the stream function wave theory. Waves are measured at $x = 0$m for 100 wave periods. The plots on the left show the results of the *foamStar* solver, and the plots on the right show the results of the *foamStarSWENSE* solver. A strong decrease in the first ten periods can be observed for all runs. For the *foamStar* solver, the amplitude does not increase again afterwards, except for the cases with large time steps (high Courant number), and a more moderate decay is observed from approximately $t = 10T$ to $t = 50T$, before stabilizing to a certain value. For the *foamStarSWENSE* solver, the wave amplitude generally increases and decreases from approximately $t = 10T$ to $t = 20T$ and then stabilizes to a certain value.

Generally, two time schemes behave differently than the others. The *Euler* implicit scheme induces a significant decrease of the wave amplitude for both solvers and all resolutions, as expected from the previous results, and the *EDIRK[3](3)_1* scheme shows less accurate and more unstable results than the other schemes. The author suspects that one reason for this instability is the A-stable property of the *EDIRK[3](3)_1* scheme. Among the tested schemes, the *EDIRK[3](3)_1* was the only A-stable method, while the others are L-stable. The L-stability is a special case of A-stability, which is generally better at integrating stiff equations (see Appendix B).

For the rest, the differences between DIRK schemes are more and more remarkable when the time step increases. The explicit-first DIRK schemes, such as *ESDIRK[3](2)*, *EDIRK[3](3)_2*, *ESDIRK[3](3)*, *RKCN099*, are less accurate then the second and third-order SDIRK schemes. Especially, the explicit first, third-order, and three-stages schemes (*EDIRK[3](3)_2* and *ESDIRK[3](3)*), are less accurate than the others.

Another interesting conclusion is that the curves of the left (*foamStar*) plots are always below the corresponding curves of the right (*foamStarSWENSE*) plots, showing that the SWENSE method is, as expected, more accurate at computing regular waves with the same resolution, with an error typically 2 to 3 times lower than with the standard two-phase averaged solver.

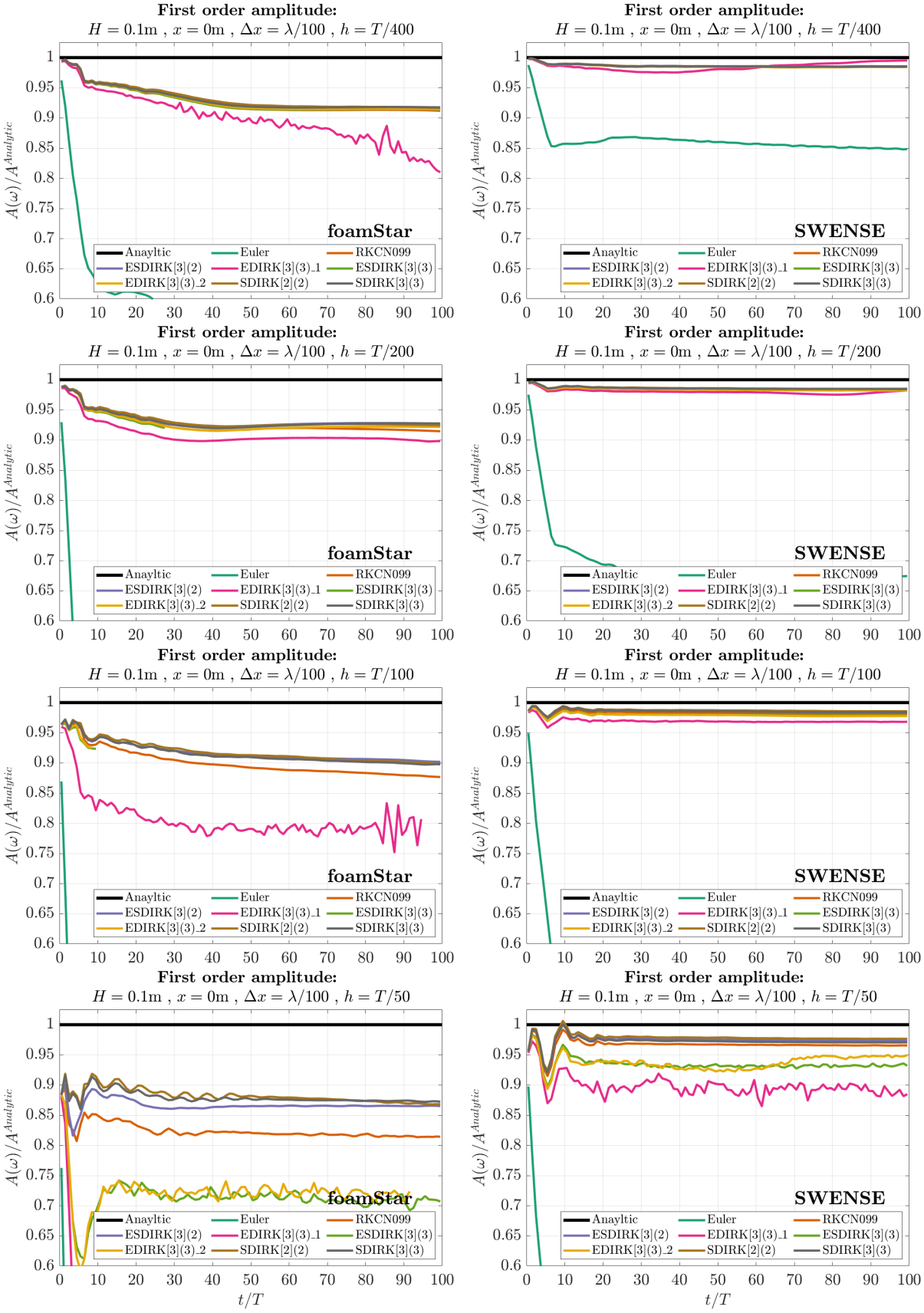


Figure 7.4. Comparison of first harmonics of the 10%-steepness wave measured at $x = 0$m using different time integration schemes.

$\Delta x = \lambda/100$. Each row corresponds to a different time step. Left: Two-phase flow solver (*foamStar*). Right: SWENSE solver (*foamStarSWENSE*).

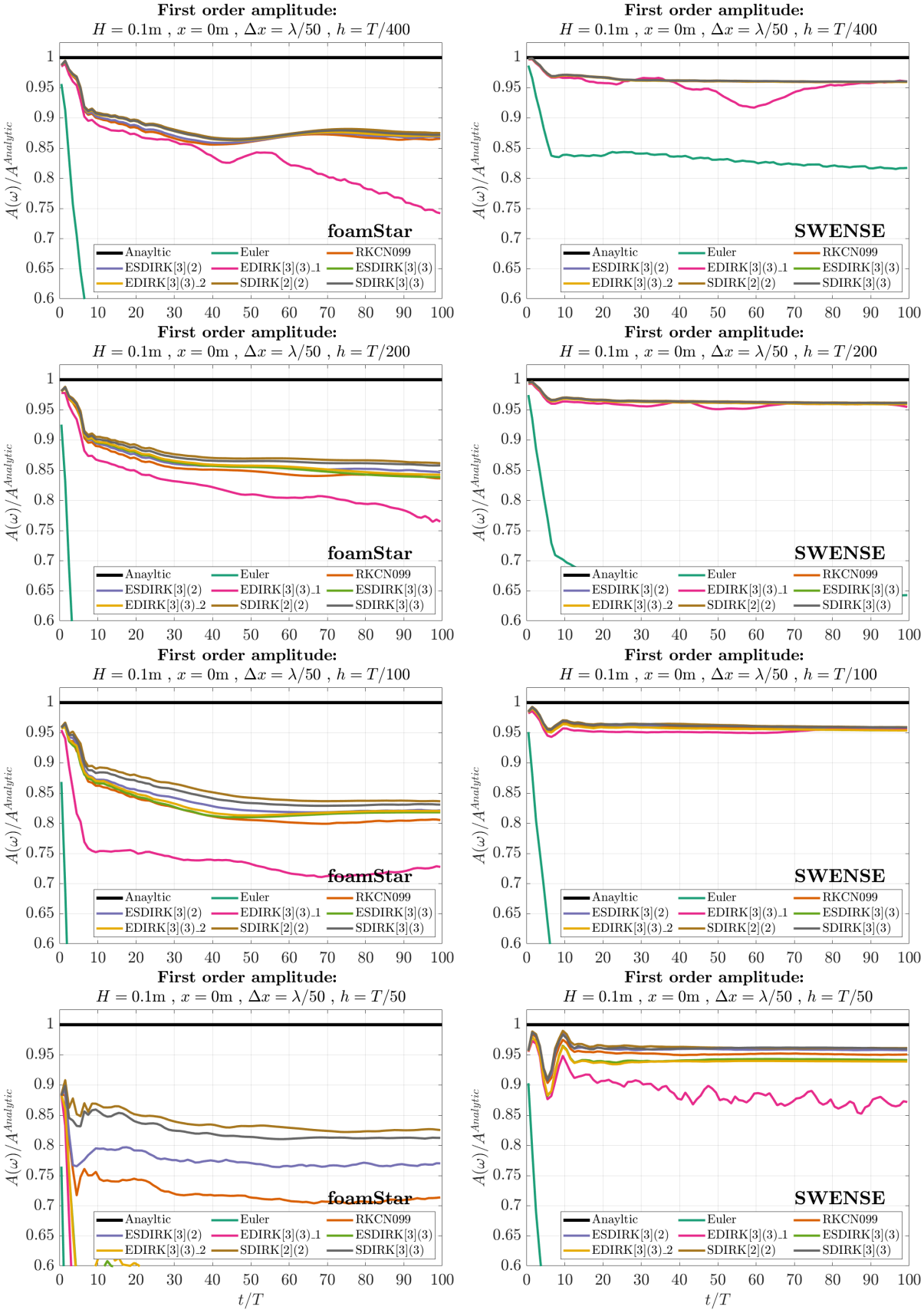


Figure 7.5. Comparison of first harmonics of the 10%-steepness wave measured at $x = 0$m using different time integration schemes.
$\Delta x = \lambda/50$. Each row corresponds to a different time step. Left: Two-phase flow solver (*foamStar*). Right: SWENSE solver (*foamStarSWENSE*).

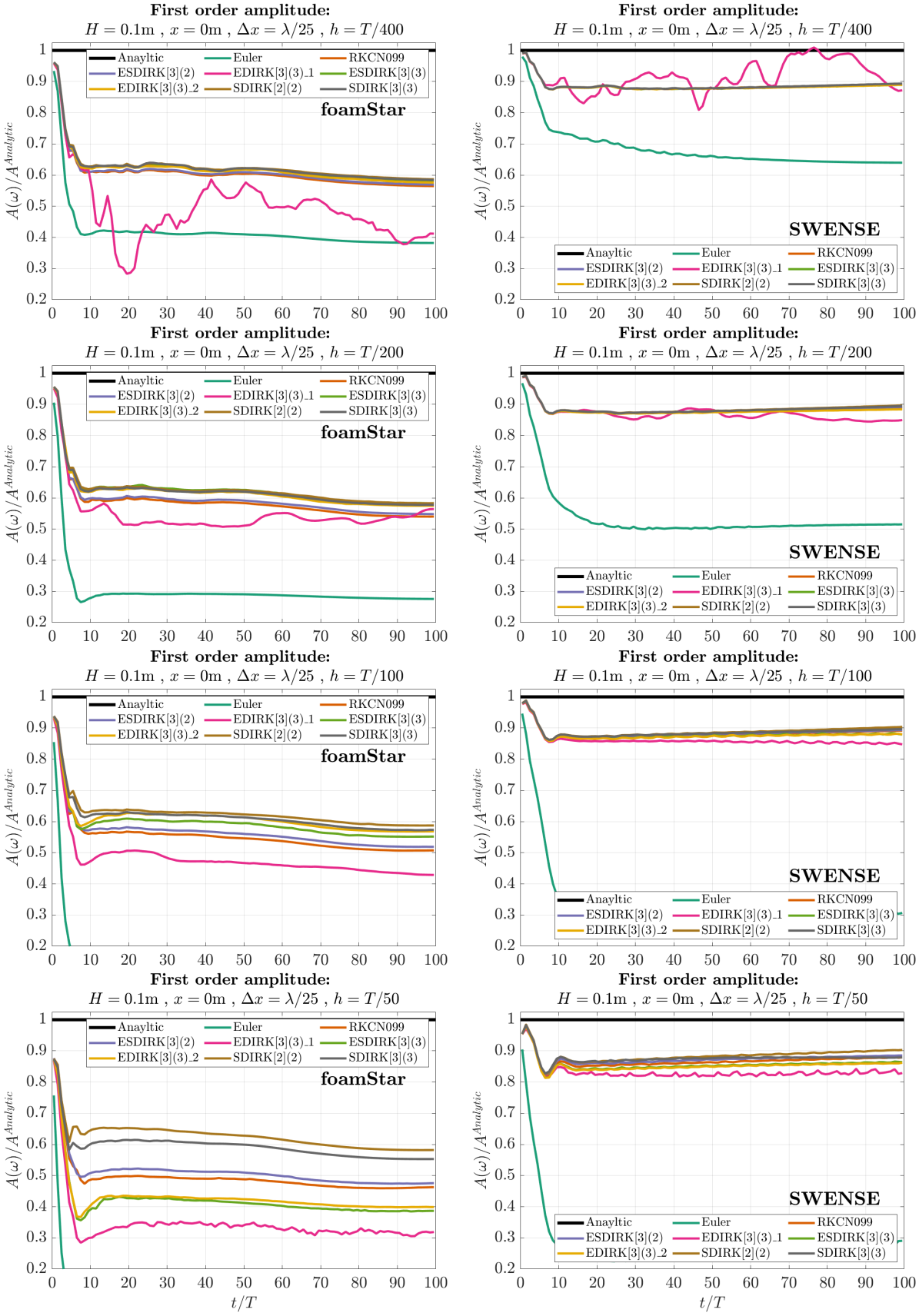


Figure 7.6. Comparison of first harmonics of the 10%-steepness wave measured at $x = 0$m using different time integration schemes.
$\Delta x = \lambda/25$. Each row corresponds to a different time step. Left: Two-phase flow solver (*foamStar*). Right: SWENSE solver (*foamStarSWENSE*).

Definition of efficiency

The efficiency of each scheme is the main point of interest; as the error (defined as the difference of the measured 1st-harmonic wave amplitude to the reference) evolves in time, it is useful to define a time interval where the error will be computed as a scalar. So, two time intervals were defined to make this efficiency comparison: the first ten periods $1T - 10T$ and the last ten $90T - 100T$. Figures 7.7 and 7.8 compare the efficiency of each time integration scheme for both the *foamStar* and the *foamStarSWENSE* solvers, for these two time intervals. The y-axis presents the averaged normalized error and the x-axis is the computational cost which is calculated as a multiplication of the computational time by the number of used CPUs. For a given error (horizontal line at a certain y-value), the scheme most on the left is the most efficient.

Identically to the previous figures, the left plots display the *foamStar* solver results, and the right plots the *foamStarSWENSE* solver ones. The upper six plots illustrate the efficiency of schemes for each mesh resolution, and the lower two plots show the entire data. The analysis of these 192 runs is done below, discussing first the effect of DIRK schemes, second the effect of SWENSE, and lastly, the influence of time and space resolution.

Efficiency of DIRK schemes

The DIRK schemes have similar effects for *foamStar* and *foamStarSWENSE*, though the accuracy is not the same between the two solvers. As expected the first-order implicit *Euler* integration scheme is clearly not as efficient as the higher-order schemes. The explicit-first DIRK schemes, except for the *RKCN099* scheme, are generally less efficient then the SDIRK schemes. Especially, the *EDIRK[3](3)_1* scheme gives inefficient and unpredictable results. The two most efficient time integration schemes are *SDIRK[2](2)* and *RKCN099*.

Efficiency of the SWENSE model

As mentioned earlier, the SWENSE solver better preserves the wave amplitude for a given resolution, so that all symbols are lower in the graphs. However, the computational burden is higher for the SWENSE model. Nonetheless, if one takes a given CPU time (a vertical line), comparing left and right plots yields the conclusion that SWENSE outperforms the standard solver in all the situations, yielding a lower error for this given CPU amount. Actually, the *foamStarSWENSE* results are globally shifted only slightly to the right compared to the *foamStar* results, but are significantly shifted downwards (note that the vertical scale is logarithmic and not the horizontal one).

Efficiency of time and space resolutions

Previous analysis showed that the *RKCN099* and *SDIRK[2](2)* time schemes were the most efficient schemes among the eight time integration schemes in Table 7.2. Therefore, Figure 7.9 compares more finely the influence of time and space resolutions on the efficiency with these two schemes. Each marker represents each resolution combination, and the markers with same color share the same time-step size.

With the *foamStar* solver, the resolution $\Delta x = \lambda/100 \,\&\, \Delta t = T/50$ is less efficient for the

first ten periods than the resolutions $\Delta x = \lambda/50 \,\&\, \Delta t = T/100$ and $\Delta x = \lambda/50 \,\&\, \Delta t = T/200$. To get a target 5% error with this solver in the time interval $1T - 10T$, a resolution level higher than $\Delta x = \lambda/100 \,\&\, \Delta t = T/100$ is required.

For the SWENSE solver, the most efficient resolution to target 5% error is $\Delta x = \lambda/50 \,\&\, \Delta t = T/50$. The results for the averaged time interval $90T - 100T$ show that the averaged wave amplitude is less affected by the time-step size and the time scheme except for the resolution $\Delta x = \lambda/100 \,\&\, \Delta t = T/50$.

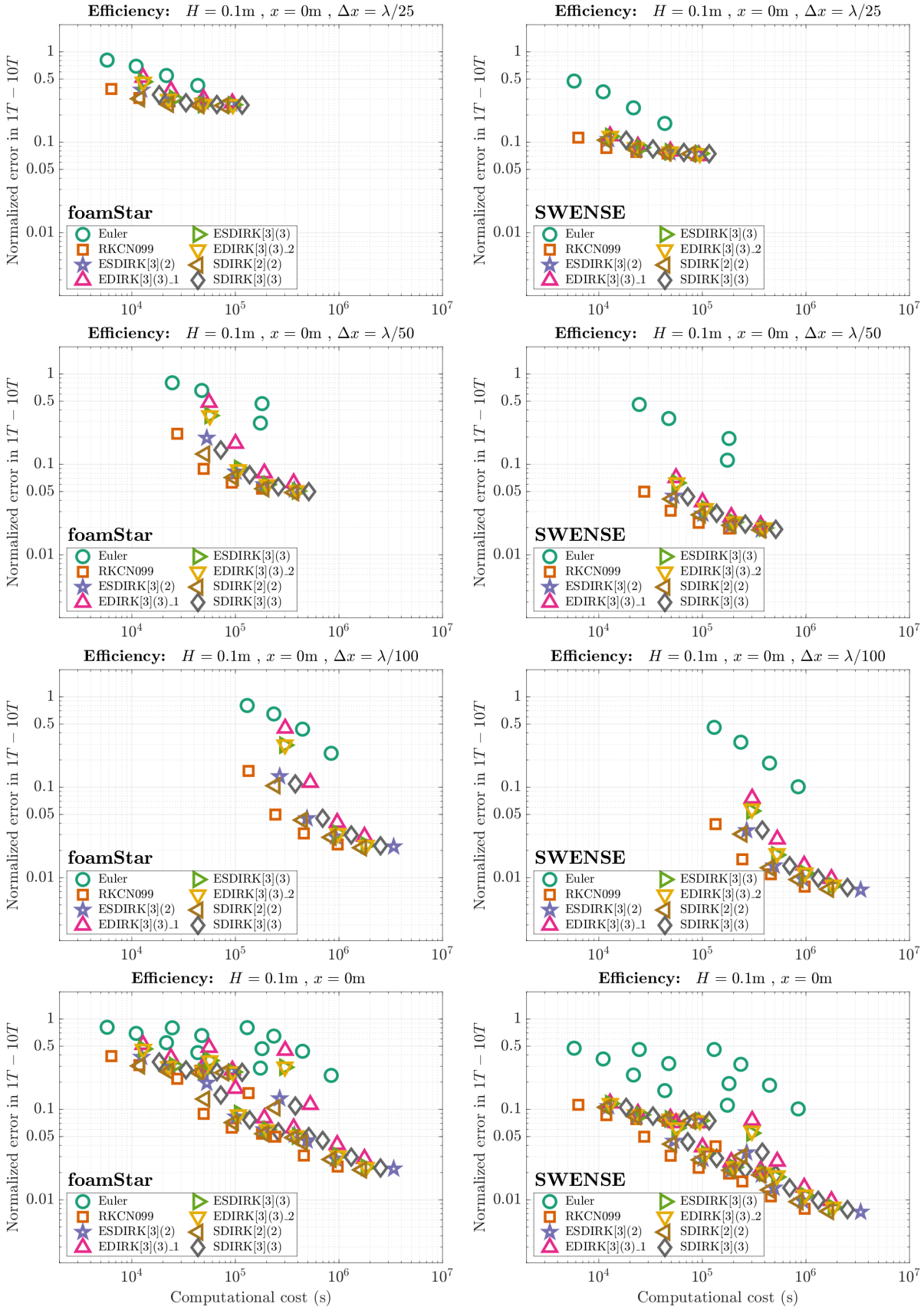


Figure 7.7. Comparison of the efficiency of the different time integration schemes using the normalized error of wave amplitude averaged from $1T - 10T$ at $x = 0$ in the NWT. Left: two-phase flow solver (*foamStar*). Right: SWENSE solver (*foamStarSWENSE*).

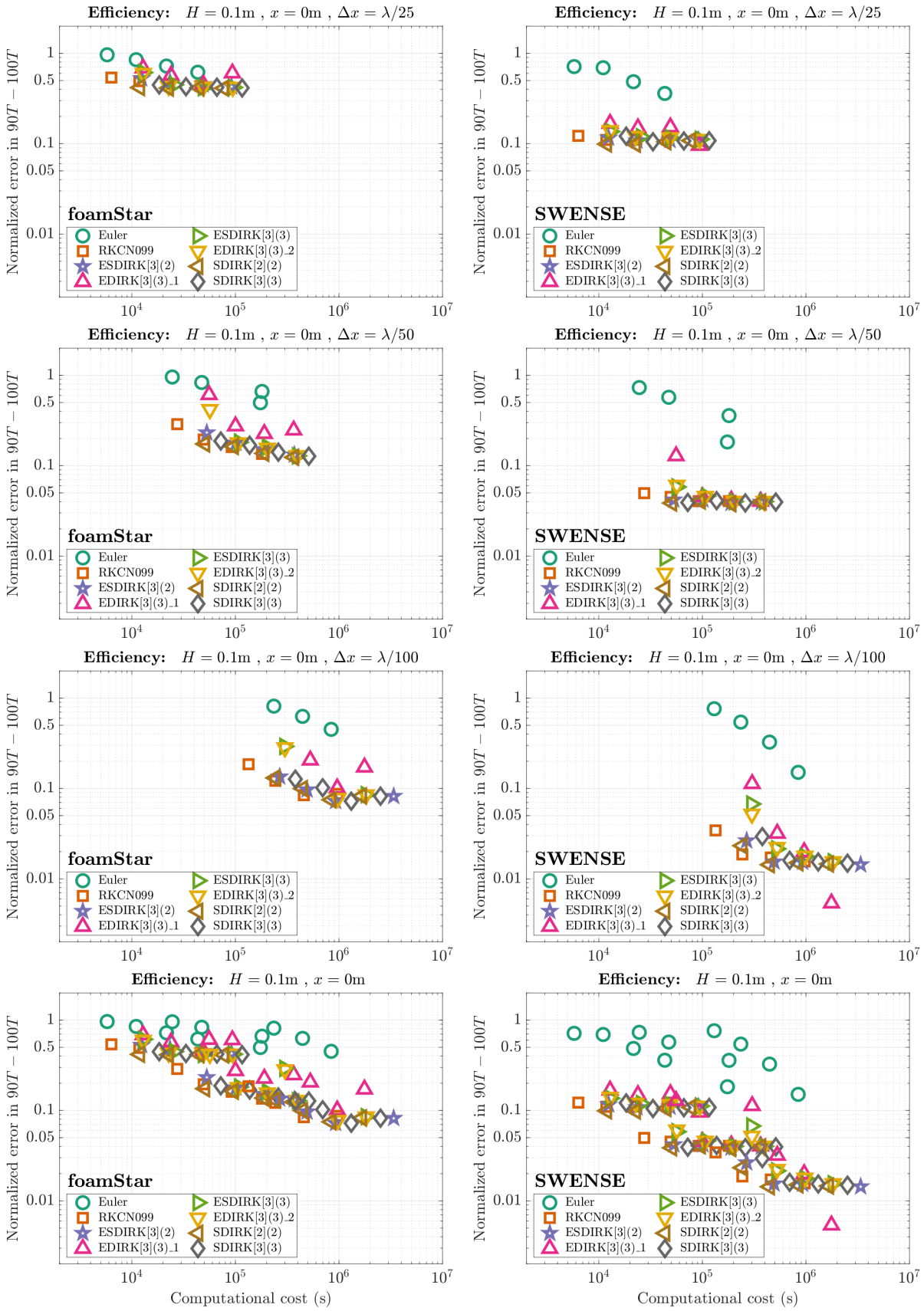


Figure 7.8. Comparison of the efficiency of the different time integration schemes using the normalized error of wave amplitude averaged from $90T - 100T$ at $x = 0$ in the NWT. Left: two-phase flow solver (*foamStar*). Right: SWENSE solver (*foamStarSWENSE*).

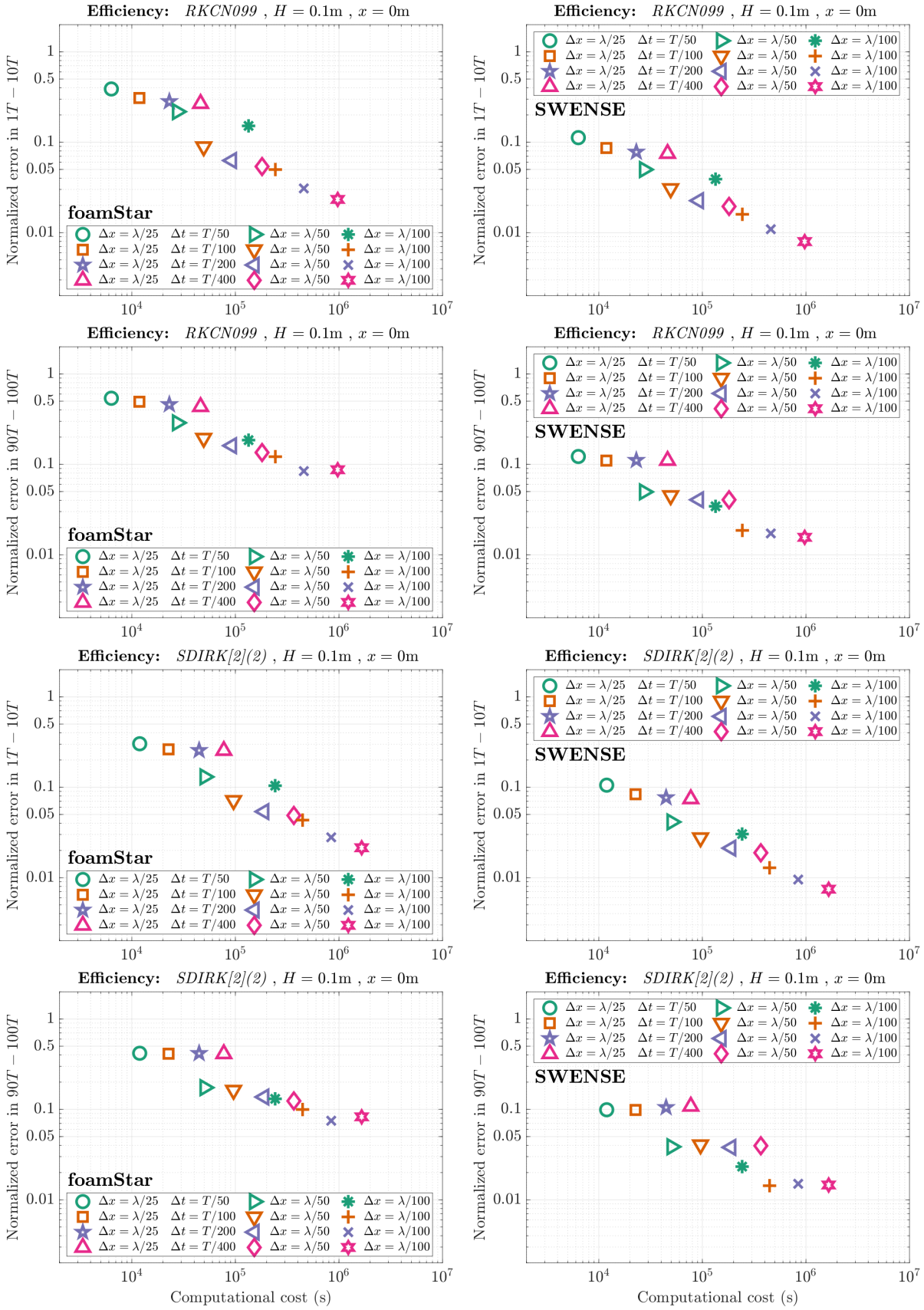


Figure 7.9. Influence of time and space resolutions on the efficiency, with *RKCN099* and *SDIRK[2](2)* schemes using the normalized error of wave amplitude at $x = 0$ in the NWT. Left: Two-phase flow solver (*foamStar*). Right: SWENSE solver (*foamStarSWENSE*).

7.2 Irregular wave propagation in a NWT

The floating bodies that operate in open-ocean encounter various sea-states during their lifetime. The common industry practice for accounting for ocean waves in an experimental or numerical study is to consider a typical 3 hours duration of long-crested irregular waves for each design sea-state. Given a wave spectrum each three-hour realization is generated using different sets of random phases. A sufficient number of three-hour cases with different sets of random phases are required to gather the statistical properties of the sea-state and to obtain the extreme response expected during the operation lifetime. Therefore, being capable of simulating accurately three-hour long-crested irregular wave evolution in a numerical wave tank is essential for realistic naval application.

This section presents the generation and propagation of irregular waves using a Numerical Wave Tank equipped with a wavemaker and an absorption beach, based on the High-Order Spectral method (HOS-NWT) and the CFD solver, OpenFOAM. A complete description of the HOS-NWT solver can be found in Ducrozet et al. [84], and the summarized version is described in Section 3.2. Therefore, three types of irregular waves are compared: waves measured from the experiment [87], waves from the HOS-NWT simulation, and waves from the OpenFOAM simulation. The experiment has been performed in the ECN Hydrodynamics and Ocean Engineering Tank. The experiment and the HOS-NWT solver used the same flap-type wavemaker. It should be acknowledged that the experiment is funded by a joint industry project *Reproducible Offshore CFD JIP* [120]. In theory, the motion of the wavemaker of HOW-NWT and experiment should be identical. However, the wavemaker motion of the experiment is modified within the spectrum correction iteration. Deterministic analysis and stochastic analysis are carried out for the numerical simulations (both for potential flow and CFD solutions) and the experimental results to measure the quality of the irregular waves.

7.2.1 NWT with irregular waves: Description

The JONSWAP parameters of two sea-states, Case 1 and Case 3, are presented in Table 7.3 at real scale. H_s is the significant wave height; T_p is the peak period; λ_p is the peak wavelength; ϵ is the steepness parameter; γ is the spectral peakedness parameter for the JONSWAP spectrum (Eqn. (3.10)). The sea-state Case 1 is a relatively mild sea-state, and almost no wave breaking events are expected. Case 3 corresponds to the Gulf of Mexico (GOM) 1000-year return period wave condition and frequent and severe wave breaking events are expected. The Benjamin-Feir Index (BFI) parameter represents the sea-state stability, which following the definition of Serio et al. [121] is given as:

$$\begin{aligned}
&\text{BFI} = \sqrt{m_0} k_m Q_p \sqrt{2\pi} v \sqrt{\frac{|\beta|}{\alpha}}\,, \quad \alpha = -\nu^2 + 2 + 8(k_m D)^2 \frac{\cosh(2k_m D)}{\sinh^2(2k_m D)}\,, \\
&v = 1 + 2\frac{k_m D}{\sinh(2k_m D)}\,, \quad Q_p = \frac{2}{m_0^2}\int f S^2(f) df\,, \\
&\beta = \frac{\cosh(4k_m D) + 8 - 2\tanh^2(k_m D)}{8\sinh^4(k_m D)} - \frac{(2\cosh^2(k_m D) + 0.5v)^2}{\sinh^2(2k_m D)\left[\frac{k_m D}{\tanh(k_m D) - (\frac{v}{2})^2}\right]}
\end{aligned} \tag{7.1}$$

where k_m is the mean wavenumber; m_n is the moment of order n of the spectrum; v is the correction of group velocity for finite depth D; α and β are coefficients; Q_p is the peak quality factor.

Figure 7.10 illustrates the target JONSWAP spectrum of Case 1 and Case 3. The peakedness parameter (γ) of Case 3 is higher than for Case 1, and the Case 3 spectrum has a higher peak than Case 1. Since many wave breaking events are expected for Case 3, a Tian breaking model is applied to the HOS-NWT solver to represent at best energy dissipation due to breaking and, at least impede the solution to blow up. No breaking model is applied for Case 1. The wave breaking model used was introduced by Tian et al. [64], its applicability was investigated by Seiffert et al. [86], and it is also shortly summarized in Section 3.2.2.

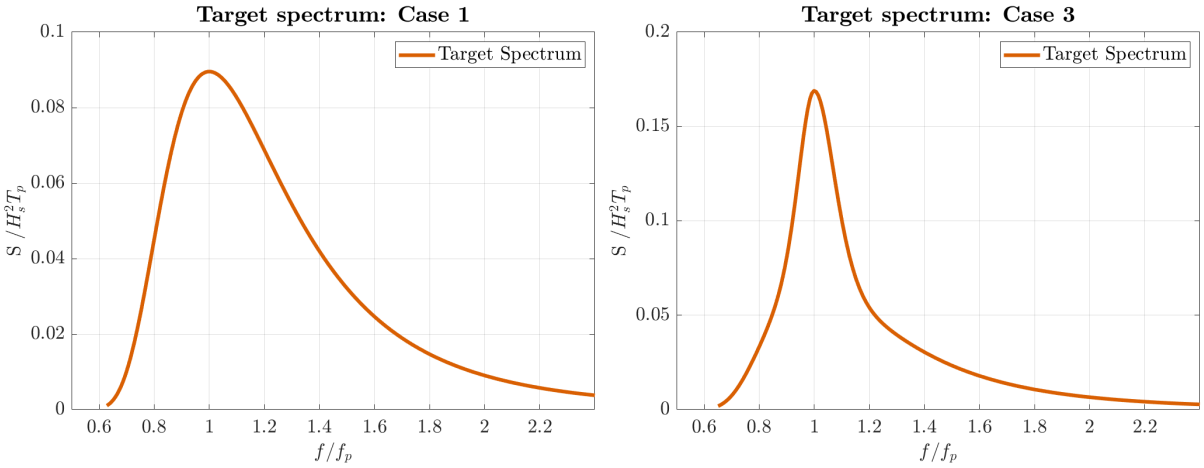


Figure 7.10. Target wave spectra for Case 1 and Case 3. Corresponding spectrum parameters are listed in Table 7.3.

Table 7.3. Irregular wave parameters based on the JONSWAP spectrum.

Case	H_s (m)	T_p (s)	D (m)	$\lambda_p(m)$	γ	ϵ (%)	BFI
Case 1	6	12.25	500	234	1	2.5	0.318
Case 3	17	15.5	500	375	2.6	4.5	0.677

Ten realizations with different random phases are generated for each sea-state to measure

the stochastic properties accurately enough to obtain a reliable estimation of the wave height corresponding to a probability of occurrence as low as 5×10^{-3}. Following the wave qualification criteria presented in Fouques et al. [120], the HOS-NWT solver uses iterative correction of the wavemaker motion to achieve a qualified spectrum obtained at a target location. Practically, this iteration ends when the target location spectrum matches the target spectrum with a tolerance of $\pm 5\%$, in the range of frequency considered. Further details on the iterative procedure and the wave qualification of HOS-NWT waves for the same two sea-states can be found in Canard et al. [87].

Figure 7.11 illustrates a sketch of 2D rectangular NWT for a real scale HOS-NWT and OpenFOAM simulation. Two coordinate systems, one for the HOS domain and one for the CFD domain, are defined for an easier definition of each solver numerical setup. The full scale length of the HOS domain (L_{HOS}) is 5000m, and it is around 21 wavelengths and 13 wavelengths for Case 1 and Case 3. The numerical wavemaker of the HOS-NWT solver is located at $x_{HOS} = 0$. The numerical absorption beach is implemented on the right side of the domain, and no reflection is expected from the beach. The seabed is placed 500m under the free surface ($D = 500$m). The HOS simulation has been run at a $1/100$ model scale, which makes the domain identical to the ECN ocean wave tank experimental facility: length $L_x = 50$m and $D = 5$m.

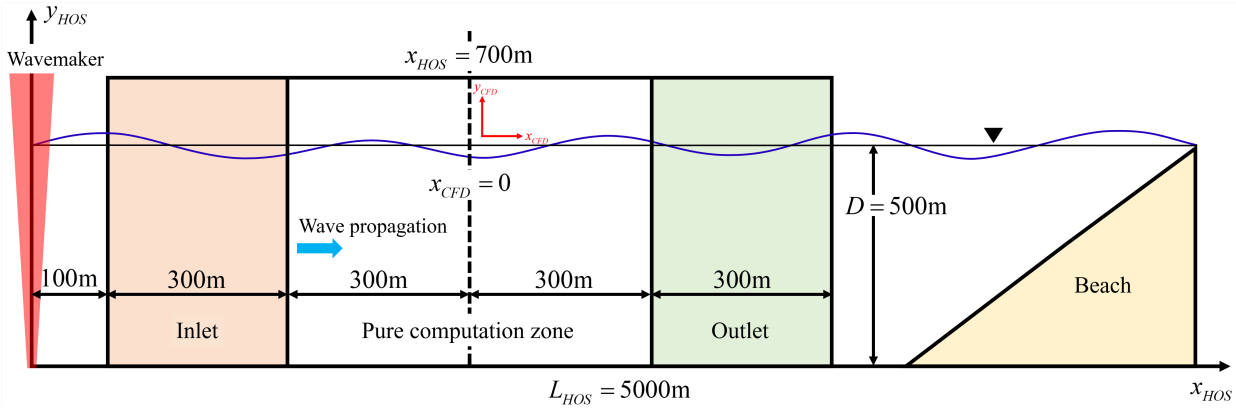


Figure 7.11. Sketch of the HOS and CFD domains for irregular wave simulation. All dimensions are in real scale and the position $x_{HOS} = 700$m is identical to $x_{CFD} = 0$m.

Another coordinate system for the OpenFOAM computational domain is introduced inside the HOS domain. The full-scale length of the CFD domain is 1200m, and it is positioned from $x_{HOS} = 100$m to $x_{HOS} = 1300$m. The relaxation zones length is set to 300m for both inlet and outlet, respectively. The CFD simulation uses the same scale ratio (1/100) as the HOS simulation. Therefore, the length of the model-scale CFD domain is 12m, and the length of the relaxation zones is 3m on both sides of the domain. The relaxation zone explicitly blends the wave properties from HOS-NWT with CFD results using the `Grid2Grid` wrapper, which uses inverse FFTs and interpolation to reconstruct the wave amplitude and the velocity to an arbitrary time and space [97]. The HOS domain must thus fully include the CFD computational domain.

There is a significant limitation linked with computational costs to run irregular wave simulations in a 3D domain. To be prepared for a 3D irregular wave simulation, the grid refinement must be carefully considered. Similar to the grid refinement for regular waves, the free-surface refinement region is defined between $z = -H_s$ and $z = H_s$, and the upper and lower refinement zones are linearly stretched from cell sizes comparable to the free-surface refinement zone ones to large cells at the top and bottom domain boundaries.

Table 7.4 summarizes the (full scale) computation parameters adopted for each case, based on the conclusions obtained for regular waves. Δx and Δz are the cell size in the free-surface refinement region, and the cell aspect ratio ($\Delta z/\Delta x$) is 0.5. Considering that the peak wavelength is either 234m or 375m, the refinement is similar to the reference setup used for the regular wave propagation ($\lambda/\Delta x = 100 \sim 150$).

Table 7.4. Mesh refinement parameters in real scale for the free surface refinement region ($-H_s < z < H_s$)

Wave case	Δx (m)	Δz (m)	$H_s/\Delta z$	h (s)	Total N cells
Case 1	2	1	6	0.04	60000
Case 3	2	1	17	0.04	73200

For each sea-state, ten random sets of phases (seeds) are applied for the realization of the irregular waves in time domain from the frequency spectrum. Table 7.5 gives the numbers of the seeds used. These numbers are not in regular sequence since they are selected from the successful HOS-NWT runs (100 seeds).

Table 7.5. Numbers of the seeds used for Case 1 and Case 3.

Wave case	**Seed numbers**
Case 1	seeds: 1, 2, 4, 5, 9, 10, 12, 16, 17, 18
Case 3	seeds: 2, 3, 5, 6, 8, 9, 10, 12, 14, 15

Table 7.6. Time intervals of the irregular wave simulations for Case 1 and Case 3 in real scale.

	Case 1 $H_s = 6\text{m}$ $T_p = 12.25\text{s}$		**Case 3** $H_s = 17\text{m}$ $T_p = 15.5\text{s}$	
Run	t_{start} (s)	t_{end} (s)	t_{start} (s)	t_{end} (s)
Run 1	1077.5	2425	1050	2700
Run 2	2302.5	3650	2550	4200
Run 3	3527.5	4875	4050	5700
Run 4	4752.5	6100	5550	7200
Run 5	5977.5	7325	7050	8700
Run 6	7202.5	8550	8550	10200
Run 7	8427.5	9775	10050	11700
Run 8	9652.5	11000	-	-

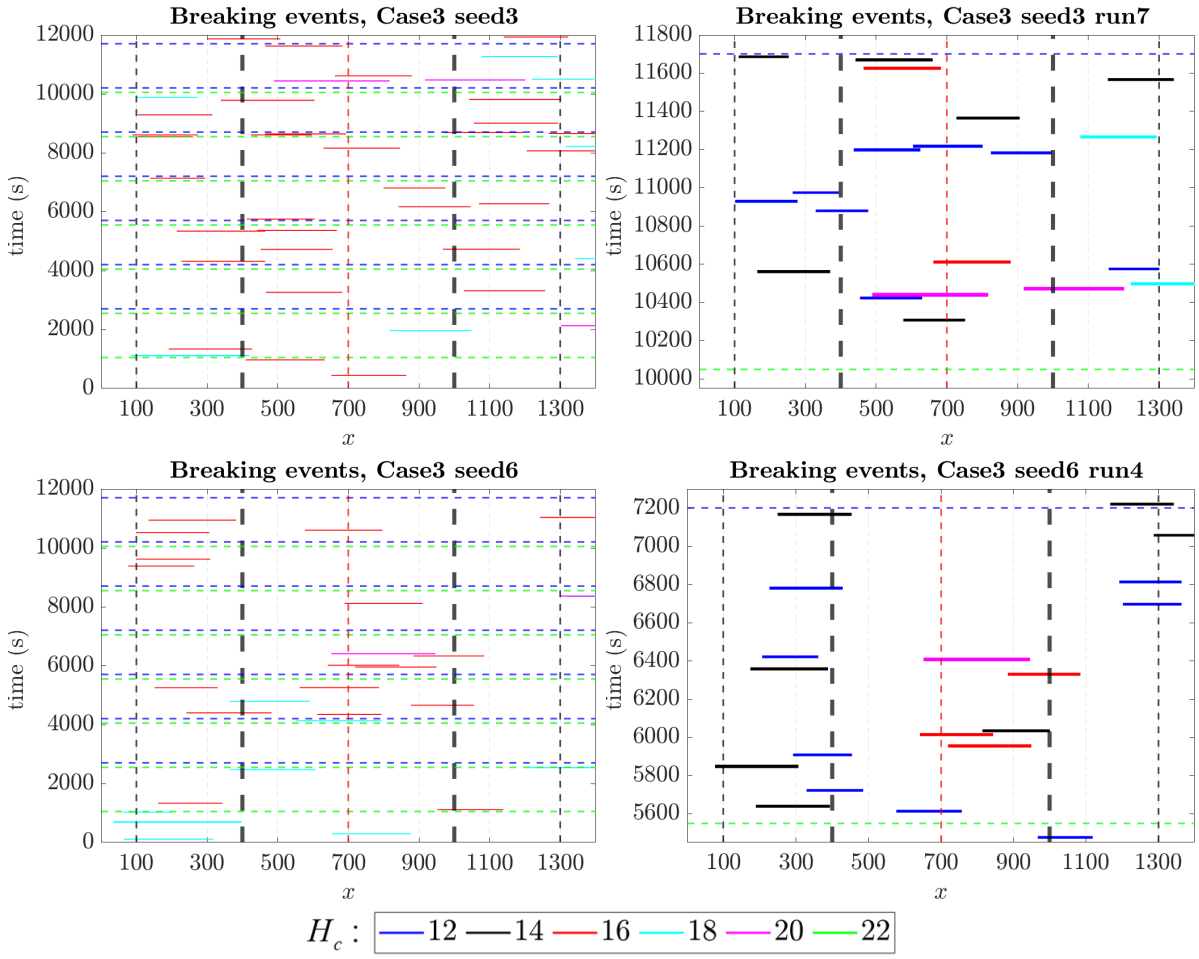


Figure 7.12. Examples of wave breaking events prediction from HOS-NWT and Tian breaking model. The color of the box corresponds to its wave crest height (H_c)

Each seed full scale simulation time excluding the initial transient period is around 3 hours (992.25s for Case 1 and 1065s for Case 3 in model scale), which corresponds to model scale 1080s.

So as to get shorter simulations, the full 3 hours sequence is split into 7 or 8 intervals of smaller duration. For each interval, ten peak periods (T_p) are overlapped with the previous and later run. Table 7.6 shows the start-time (t_{start}) and the end-time (t_{end}) of each interval. Each run corresponds to 110 peak periods, respectively 1225s and 1550s for Case 1 and Case 3. Also, a full simulation of 3 hours without splitting is tested in the following subsection to check the differences between both approaches.

Figure 7.12 shows an example of the wave breaking events identified by the Tian model during HOS-NWT simulations. The x-axis represents the computational domain, and the y-axis represents the simulation time. The black vertical dotted lines represent the CFD computational domain and the relaxation zones, and the horizontal dotted lines represent the time intervals of the simulation cut into parts. The plots on the left illustrate the entire time series, and the plots on the right illustrate a zoom on a run with maximum breaking events in the domain. Each colored box in the figure represents a breaking event where the color given to the box depends on its wave crest height (H_c). The colored boxes indicated the space and time where and when the breaking model acts in the potential flow simulation. The corresponding real breaking event is expected to lie in that colored box range. In the left column of the figure, the breaking events under 14m are not pictured for better visualization.

A campaign of irregular wave simulations with *foamStar*, with and without the free-surface k-ω SST turbulence model, is performed. The results from *foamStarSWENSE* solver is not reported in this thesis due to the diverging error in the computational domain. The simulations are performed using the DIRK-based Crank-Nicolson scheme with 0.99 off-centering (*RKCN*), which was the most efficient time integration scheme for the regular wave simulations. Other higher-order DIRK schemes are not tested due to their higher computational cost and lower cost-error efficiency. The spatial discretization schemes and the boundary conditions, described respectively in Table 4.1 and in Section 4.9, are identical to those used in regular wave simulations. 8 PIMPLE iterations and 2 PISO iterations are used to minimize the error from non-linear iterations. Since meshes are fully orthogonal, no non-orthogonal corrector is used.

The results of the numerical runs performed are presented first in a deterministic way and then in a stochastic approach.

7.2.2 NWT with irregular waves: Deterministic analysis

This section aims to deterministically compare the time series of the waves obtained with different computational setups. The deterministic analysis is divided into three parts. The first part compares the difference between the split run and the full-duration simulation and checks the validity of the split run approach. The second part compares the wave elevation results from the irregular wave experiment done at ECN, the HOS-NWT solver, and the *foamStar* solver with and without turbulence model. The third part focuses on the breaking wave events. The wave profiles from the HOS-NWT solver and the simulations with three different resolutions are compared in terms of wave breaking events.

Two difference functions are introduced to measure the wave elevation difference between HOS-NWT and CFD results. These difference functions are named $D1$ and $D2$. The first one captures the maximum difference in space and normalizes it with the significant wave height. The second difference function calculates the integration of the absolute difference over the CFD domain and normalizes it with the integration of absolute wave elevation over the domain. For both difference functions, the HOS-NWT data is used as reference data for computing the difference.

$$D1 = \max_x \left(\frac{|\eta_{HOS} - \eta_{CFD}|}{H_s} \right) \qquad D2 = \frac{\int_{x_{CFD}} (|\eta_{HOS} - \eta_{CFD}|)\, dx}{\int_{x_{CFD}} (|\eta_{HOS}|)\, dx} \tag{7.2}$$

Comparison between the split runs and the full run

Figures 7.13 and 7.14 compare the result of the last split run to the last part of the full 3-hour simulation for Case 1 seed 1 and Case 3 seed 5. The computational setup is identical for both approaches, but for the starting time. Both full-duration simulation and split simulation are performed without turbulence model. The upper two pictures compare the wave time series at $x_{CFD} = 0$m, and the lower two pictures compare the corresponding $D1$ and $D2$ difference function. Generally, the differences between the target HOS-NWT waves and the simulation results are not increasing in time for both simulations. For both Case 1 and Case 3, the difference of the time series of wave elevation between the split run and the full-duration simulation is almost non-visible, while the difference functions $(D1, D2)$ clearly show the differences. For Case 3, the difference between the two simulations almost disappears after 20 wave periods. However, for Case 1, the difference functions do not show the same values after a transient, whereas the D2 difference is smaller than Case 3. This results imply that the memory effect from the non-linearity of irregular waves and the wave breaking events is minimal.

In conclusion, both the split run and the full run approaches work well for these 3-hour long irregular wave simulations. Further study is needed to explain the small differences observed between the two strategies, especially on Case 1.

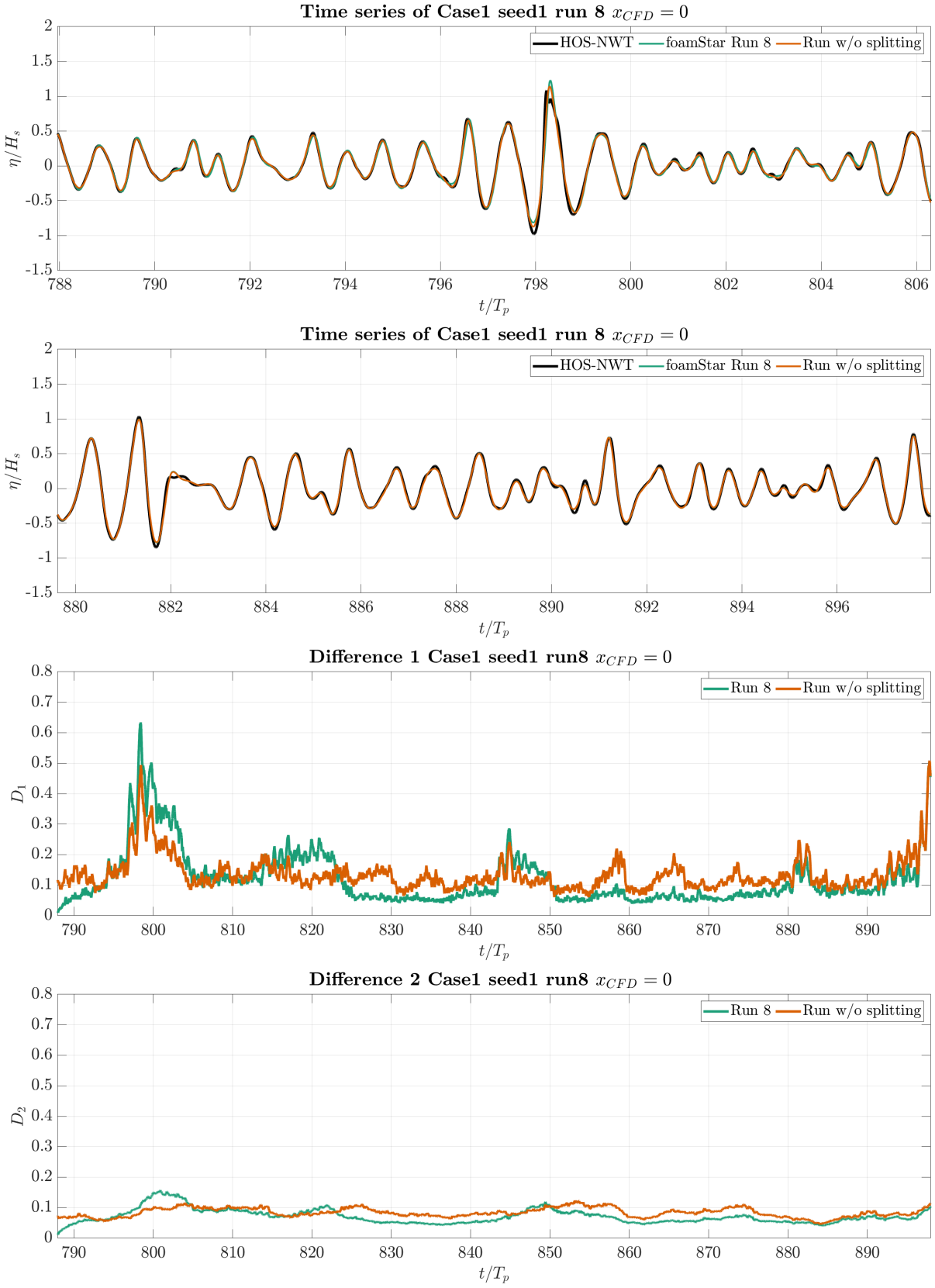


Figure 7.13. Comparison of the last split run to the last part of the simulation without splitting for Case 1 with the two-phase flow solver *foamStar* at the target location $x_{CFD} = 0$m.

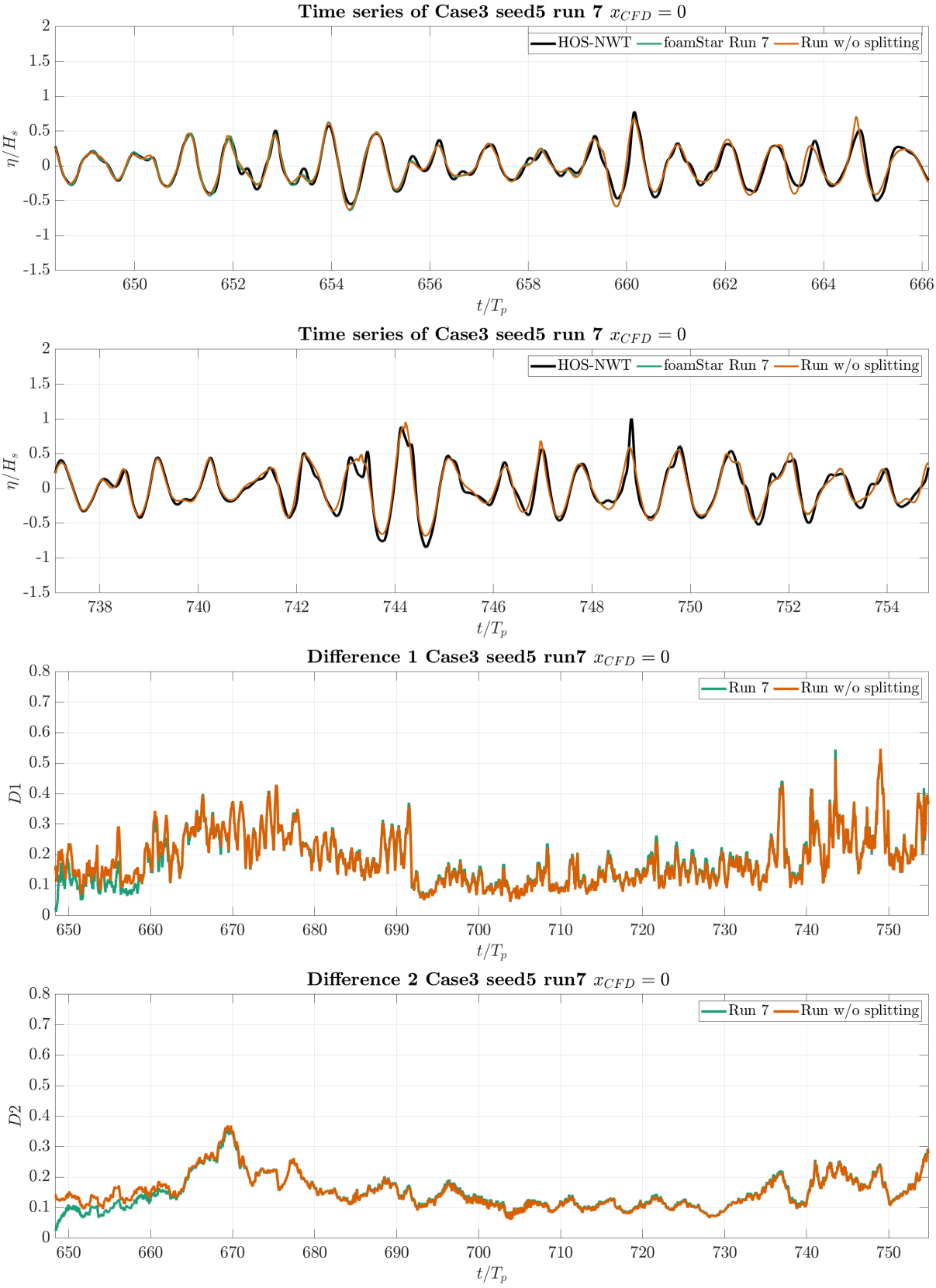


Figure 7.14. Comparison of the last split run to the last part of the simulation without splitting for Case 3 with the two-phase flow solver *foamStar* at the target location $x = 0$m.

Irregular waves with and without turbulence model

Figure 7.15 plots a comparison of the wave elevation at $x_{CFD} = 0$m obtained from the ECN experiment, the HOS-NWT solver, and the simulations with and without the free-surface k-ω SST turbulence model. Note that only parts of full time series are presented. The difference between the *foamStar* simulations with and without the free-surface k-ω SST turbulence model is non-distinguishable in the time series, even for Case 3 where there are breaking events. There are differences of phase and amplitude between the simulations and the experimental signal, especially for wave crests.

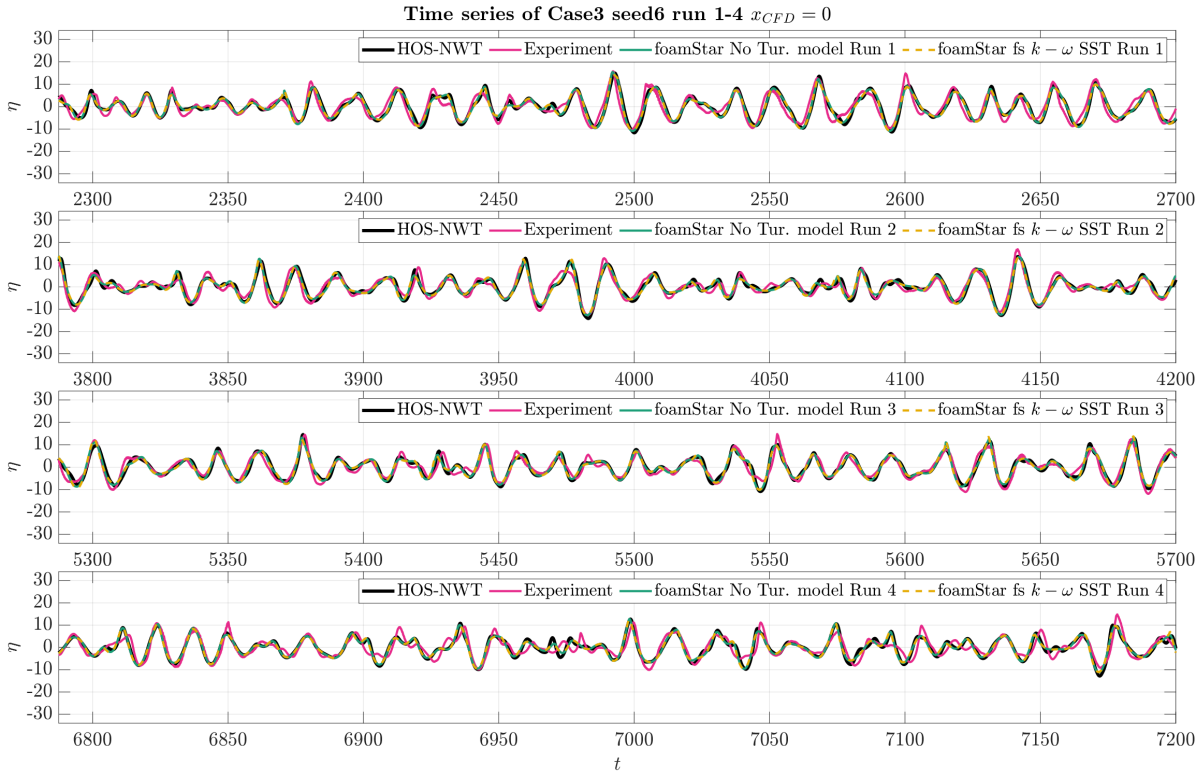


Figure 7.15. Comparison of the time series of wave elevation at $x_{CFD} = 0$m for Case 3 seed 6. Experimental signal is compared to simulations with and without free-surface k-ω SST turbulence model.

Wave breaking events and simulation resolution

The last part of this deterministic analysis mainly compares the difference in wave profile during the wave breaking events. Three levels of refinement parameters are applied to the *foamStar* solver, and the resulting wave profiles are compared with the HOS-NWT wave profiles. Table 7.7 presents the three resolutions used only for this breaking wave event analysis. The suggested resolutions used is presented in Table 7.4, and it is identical to 'Case 3 Dx 2m'.

Table 7.7. Real scale parameters of the three resolutions used in the free-surface refinement region. 'Case 3 Dx 2m' is the suggested parameter for the irregular wave simulations.

Refinements	Δx (m)	Δz (m)	$H_s/\Delta z$	h (s)	Total N cells
Case 3 Dx 4m	4	2	8.5	0.08	18300
Case 3 Dx 2m	2	1	17	0.04	73200
Case 3 Dx 1m	1	0.5	34	0.02	292800

Figure 7.16 presents the wave breaking events used for the wave profile comparison. Figure 7.17 presents the wave profiles corresponding to Case 3 seed 3 run 7, and Figure 7.18 presents the wave profiles corresponding to Case 3 seed 6 run 4. For both cases, one can see that the simulation is converged since the differences between the results from the three resolutions are small except for details of the wave crest shape during the wave breaking events.

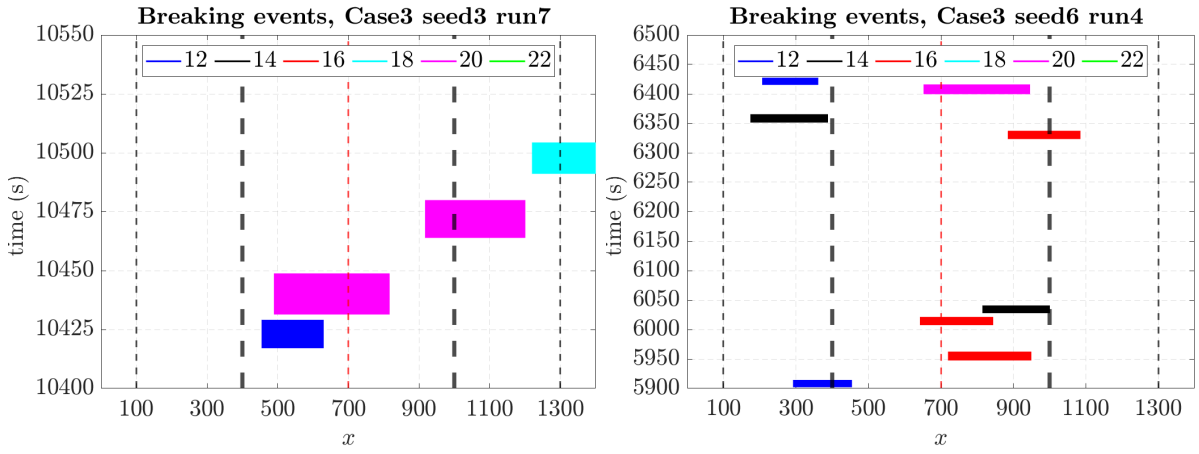


Figure 7.16. The wave breaking events used for the wave profile comparison.

In Figure 7.17, the upper three plots show the breaking event in the time interval [10425s, 10450s], and the lower four plots show the breaking event around 10475s. For both breaking events, the HOS-NWT wave profile and the *foamStar* wave profile look similar until they reach the highest wave crest amplitude. Then obvious increase of the phase velocity is captured with large phase differences. These phase differences are quickly removed as the waves enter the relaxation zone. When the wave breaks in the pure computational domain, it makes local wiggling of the free-surface, and it makes another phase difference in the domain. Similar conclusion can be also found for Case 3 seed 6 run 4 in Figure 7.18. Comparing two breaking events for seed 6, it seems that higher increase of the phase velocity is observed when the local wave steepness is high.

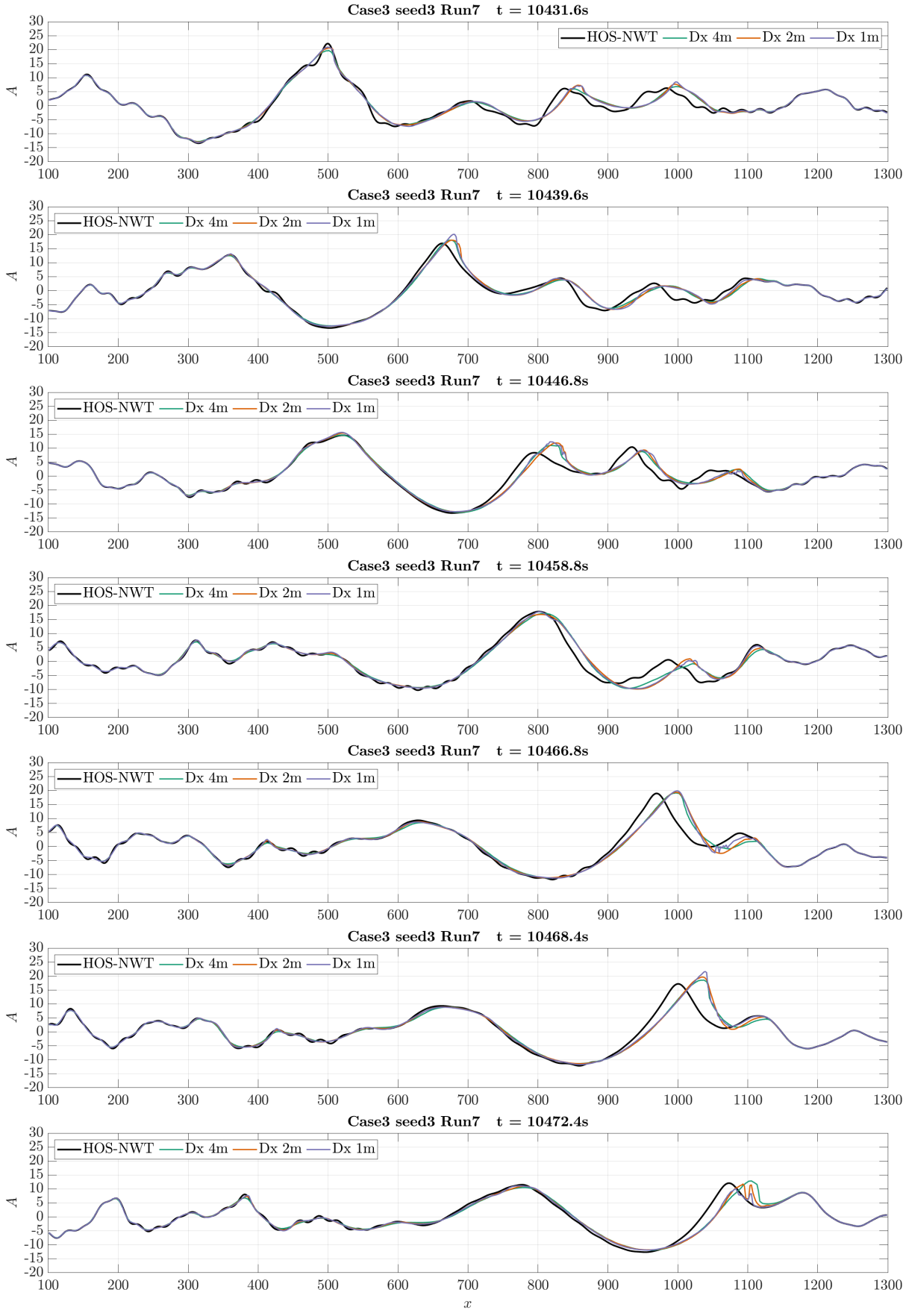


Figure 7.17. Comparison of the wave profiles from HOS-NWT solver and *foamStar* solver with three resolutions during the selected wave breaking events of Case 3 seed 3 run 7.

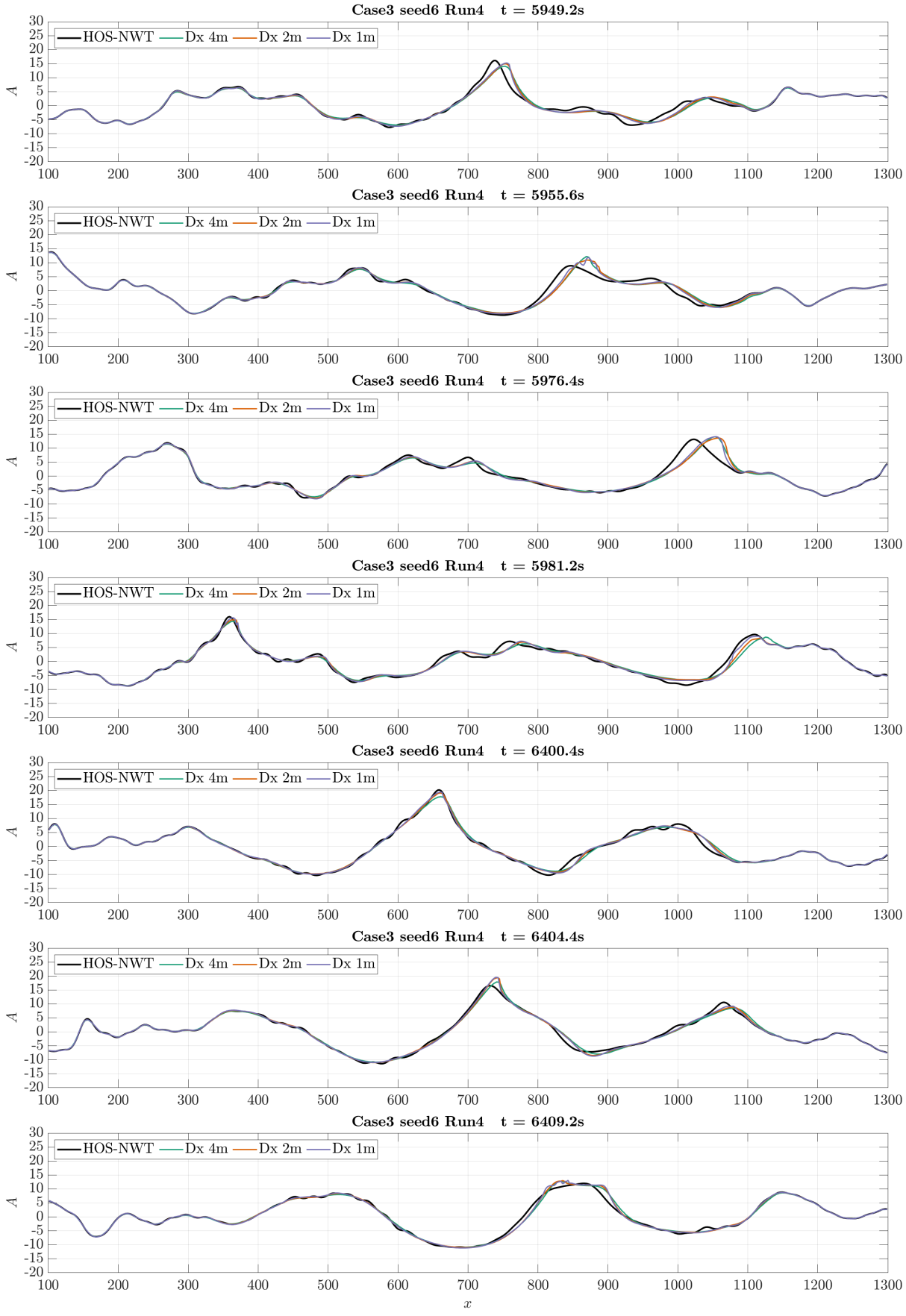


Figure 7.18. Comparison of the wave profile from HOS-NWT solver and *foamStar* solver with three resolutions during the selected wave breaking events of Case 3 seed 6 run 4.

7.2.3 NWT with irregular waves: Stochastic analysis

This section presents the stochastic analysis of the irregular waves obtained from the experiment and from the numerical solver to estimate the quality of irregular waves statistical description obtained in CFD. As mentioned earlier, the wavemaker motion is the result of an iterative process to fulfill the criterion of ±5% difference to the target spectrum for the frequencies in the range $f \in [0.75f_p\ 1.5f_p]$, where f_p is the peak frequency of the target spectrum (see procedure in [120]). The wave is measured at the center of the CFD domain ($x_{CFD} = 0$m), as illustrated in Figure 7.11. The stochastic analysis is performed for ten seeds, and each seed corresponds to three hours of measurement in the ocean.

The averaged wave spectra of Case 1 and Case 3 are compared with the target spectrum in Figure 7.19. The definition of the averaged wave spectrum is the average of all wave spectra obtained from each seed. In the figure, five different wave spectra are compared: the target wave spectrum and the averaged wave spectra obtained from the HOS-NWT, the experiment, and two simulation results with and without free-surface k-ω SST turbulence model.

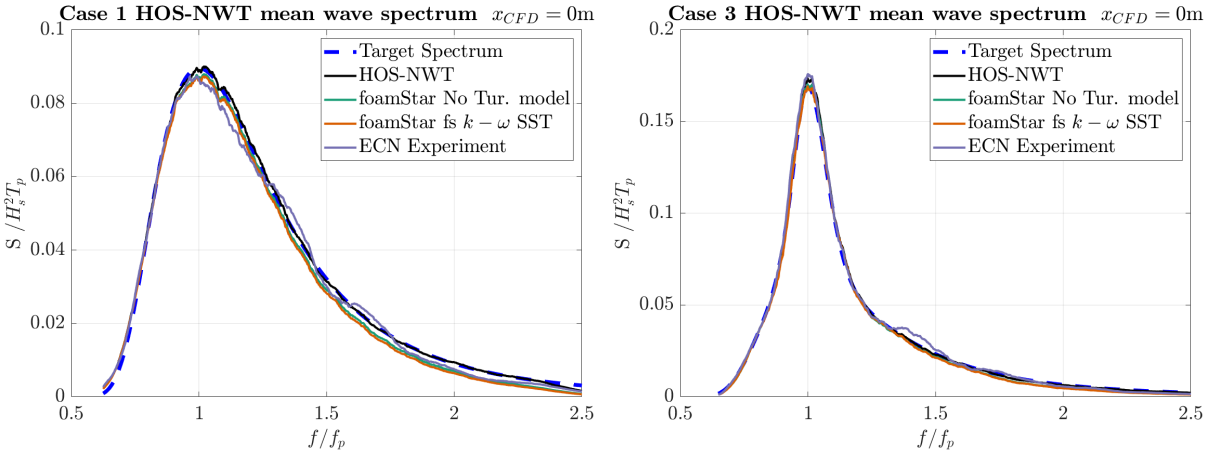


Figure 7.19. Comparison of the irregular wave spectrum from HOS-NWT, experiment and *foamStar* for Case 1 and Case 3 at the target location $x_{CFD} = 0$m.

The significant wave height (H_s) and peak wave period (T_p) can be computed from these wave spectra. Table 7.8 summarizes the wave parameters calculated from the wave elevation at $x_{CFD} = 0$m. The peak periods of *foamStar* simulations are not exactly the same as the target wave spectrum but follow the peak period of HOS-NWT. For both cases, the significant wave height obtained with HOS-NWT simulations is larger than the one obtained with CFD simulations, and the significant wave height with the free-surface turbulence model is slightly smaller than the one without.

To complete this analysis, averaged wave spectra are also quantitatively compared using the difference ratio between the target spectrum and the calculated wave spectra. Figure 7.20 plots the difference ratio of each wave spectrum. The black dotted lines are the limits of the ±5% criterion region. The red dotted lines are the $[0.75f_p\ 1.5f_p]$ target frequency range for

the ±5% criterion. Both the *foamStar* simulation and the experimental measurements do not satisfy the ±5% criterion for the target frequency range. Comparing to HOS-NWT results, the simulations by *foamStar* have lower high-frequency components for both Case 1 and Case 3, and the experimental measurement has higher high-frequency components. For the *foamStar* simulations, the underprediction of the spectrum frequencies which is more and more apparent when going to higher frequencies is very likely to be linked with the yet too coarse resolution. Further tests with finer mesh should yield less damped high-frequency components.

Table 7.8. Significant wave height (H_s) and peak period (T_p) of irregular waves measured from the wave elevations.

Case 1	Target	HOS-NWT	*foamStar* No Tur.	*foamStar* f-s $k-\omega$ SST	Experiment
H_s (m)	6	5.95	5.76	5.71	5.85
T_p (s)	12.25	12.01	12.01	12.01	12.36
Case 3	Target	HOS-NWT	*foamStar* No Tur.	*foamStar* f-s $k-\omega$ SST	Experiment
H_s (m)	17	16.91	16.57	16.51	17.05
T_p (s)	15.5	15.52	15.52	15.52	15.49

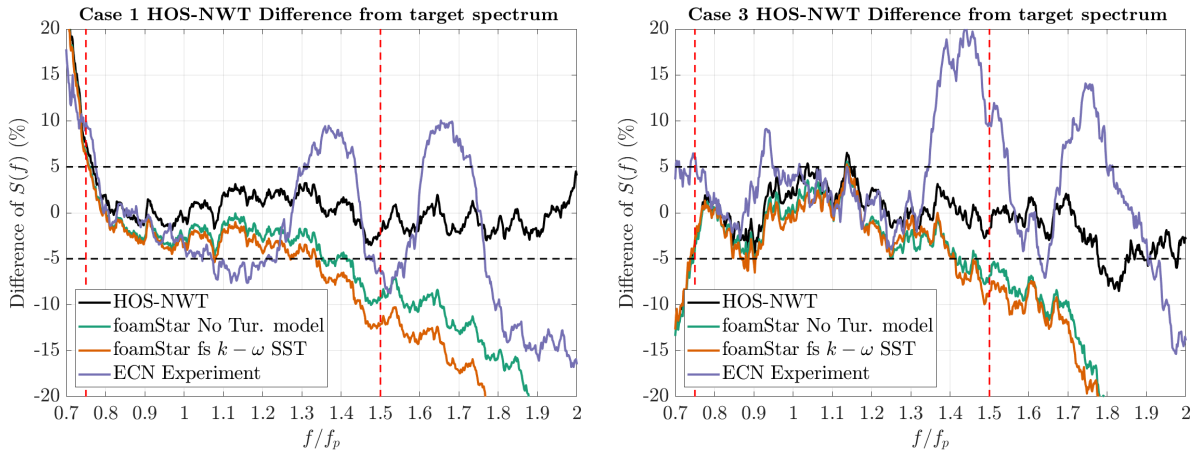


Figure 7.20. Difference ratio between target wave spectrum and other averaged wave spectra in Figure 7.19. Left plot: Case 1. Right plot: Case 3.

Figure 7.21 compares the Probability of Exceedance (POE) of the wave crests measured at the target location x_{CFD} = 0m. The upper plots show the crest distributions for each single seed while the lower plots show distributions for the ten seed ensemble. The qualification criteria for the wave crests height follow the previous study done by Canard et al. [87]. The single seed crest distributions from the experiment and numerical simulations are compared with a

Forristall distribution [122] and ±99% bounds built by Huang and Zhang [123]. The ensemble crest distributions are compared with ±5% of Huang's distribution and the Forristall distribution. The POEs of single realizations (PDSR) of the experiment, HOS-NWT, and *foamStar* mostly lie between the 99% lower and upper bounds of Huang's distribution. The distribution of the wave crests of *foamStar* is generally smaller than the others. The ensemble of realizations (PDER) also fits well within the ±5% bounds of Huang's distribution.

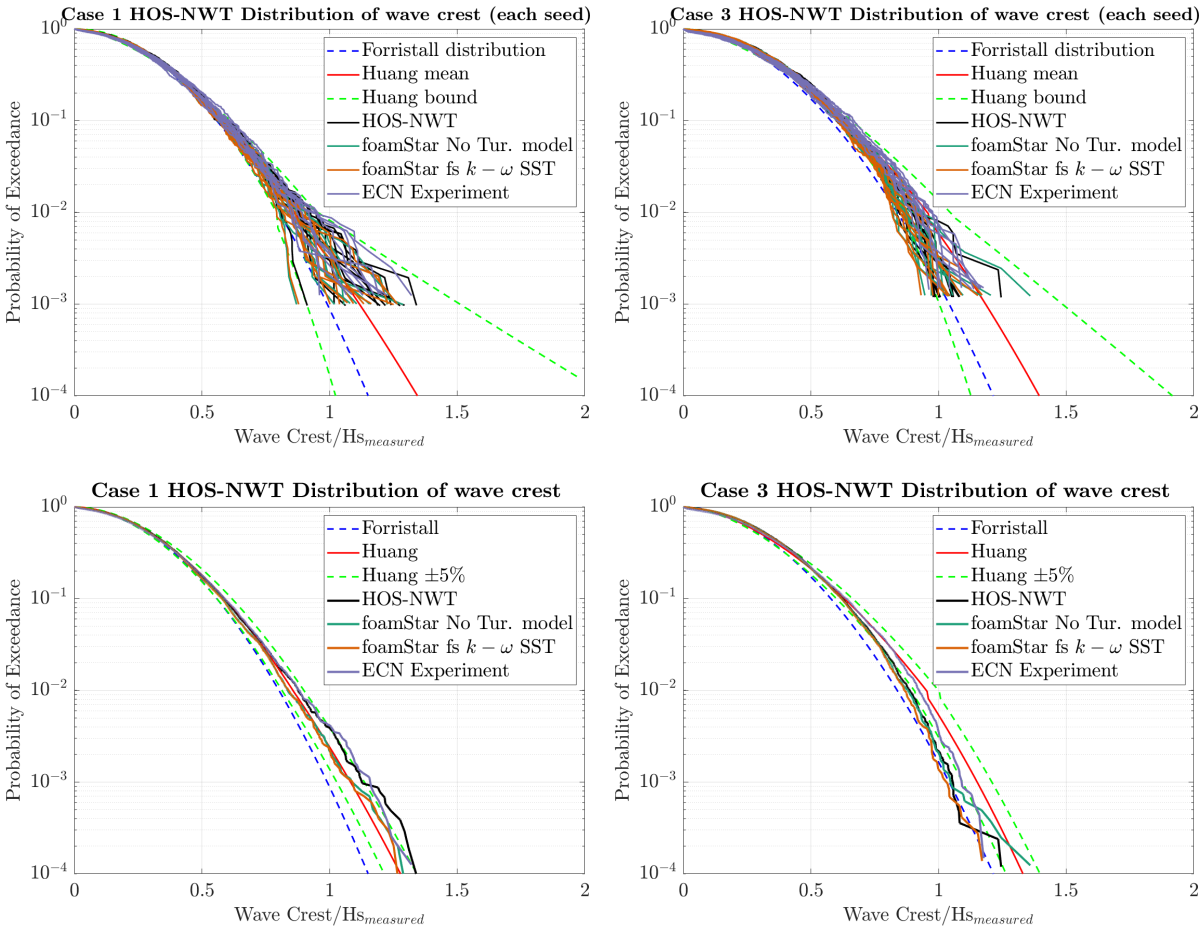


Figure 7.21. Distribution of wave crests at the target location $x_{CFD} = 0$m. Upper plots show the POEs of each seed, and lower plots show the POEs of the ten seeds altogether.

7.3 Conclusions and summary of the NWT performances

The main objective of this chapter was to find qualification criteria for regular and irregular wave propagation in a Numerical Wave Tank (NWT) with various numerical schemes. The relaxation method was applied to generate and absorb waves in the NWT. Several non-linear wave theories were applied to the relaxation zone to generate waves in the NWT. Mainly two solvers were used. One is the *foamStar* solver, which uses the averaged two-phase flow model. Another is the *foamStarSWENSE* solver, which uses the SWENSE formulation. Both formulations are described in Chapter 4. With these two solvers, the accuracy and efficiency of various DIRK-

based time integration schemes were tested.

The following summarizes the conclusions we can draw on regular and irregular wave propagation in NWT for both *foamStar* and *foamStarSWENSE* solvers.

Regular waves

- A 2D ten-wavelength long numerical wave tank and three deepwater wave conditions were considered.
- The quality of wave propagation for four different cell aspect ratios ($\Delta z/\Delta x$) with the same horizontal cell size and different vertical cell sizes were compared. For an horizontal discretization $\lambda/\Delta x = 100$, it was found that the appropriate range of cell aspect ratio for a qualified regular wave propagation in NWT was $\Delta z/\Delta x \in [0.5, 1.0]$.
- The free-surface $k-\omega$ SST turbulence model was introduced to prevent non-physical turbulent damping of the waves. The free-surface $k-\omega$ SST model was compared with the standard $k-\omega$ SST model. The conclusion made is that the free-surface $k-\omega$ SST model did not generate excessive turbulence viscosity near the free-surface, and the difference between using this model or no model at all was small.
- The efficiency of DIRK time integration schemes with both the two-phase flow solver and the SWENSE solver was investigated. Within eight time integration schemes, *SDIRK[2](2)* and *RKCN099* methods were found to be the most efficient schemes.

Irregular waves

- A 2D NWT and two long-crested irregular wave conditions were considered for irregular wave analysis. One wave condition was mild, and the other was a wave breaking condition. These two sea-states were generated using the HOS-NWT solver with the Tian breaking model before the OpenFOAM simulations, and were then coupled to OpenFOAM using a wrapper program named `Grid2Gird`. Ten seeds were run for each wave condition to obtain the stochastic properties of the irregular waves. Computationally speaking, it was also tested whether splitting a single seed (3 hours) into several shorter intervals had an influence on the results compared to making a simulation of the full 3-hour duration.
- A deterministic analysis, which compared the time series of wave elevation with different computational setups, was performed. Two difference functions were introduced to measure the difference between HOS-NWT results and others quantitatively. Using these difference functions, the time series of wave elevation of the split run and the full-duration run were compared. It was found that the difference between the two strategies was small, and the memory effect from the non-linearity of irregular waves was tiny.
- The deterministic analysis also compared the wave elevation acquired from the ECN experiment with the results from the HOS-NWT solver and the *foamStar* solver. Specific wave breaking events were selected, and the wave profiles from the HOS-NWT solver and *foamStar* solver with three refinement levels during the wave breaking events were compared. The *foamStar* results were apparently converged but for details of the breaking wave crest evolution, and the small differences with HOS-NWT also concerned mainly the amplitude and phase of the breaking wave crest.

- A stochastic analysis was then performed for the irregular waves from the experiment and the numerical solvers to measure the quality of the irregular waves. This quality was measured using the averaged wave spectrum and the wave crest distribution. Both *foamStar* and the experiment failed to satisfy the ±5% criterion for the target frequency range. Compared with the HOS-NWT results for both Case 1 and Case 3, the simulations done with *foamStar* showed underestimation of high-frequency components, which is likely to be linked to still insufficiently fine resolution to properly propagate these high-frequency components.

8 Naval applications: Seakeeping analysis

Seakeeping analysis is the study of ship motions and ship performance subjected to incoming waves and other environmental forces. The seakeeping performance is one of the primary concerns for designing ships because the prediction of ship motions and wave-induced additional ship resistance are essential to ship performance and safety in operation. Besides, there are also important factors in the upcoming regulation for reducing greenhouse gas emissions from the International Maritime Organisation (IMO). The mean contribution of wave-induced additional resistance is usually named 'added resistance'. Mathematically, the added resistance is often defined in non-dimensional form as:

$$R_{AR} = \frac{R_T - R_S}{\rho g (B^2/L)(H/2)^2}. \tag{8.1}$$

where R_{AR} is the added resistance, R_S is the calm water resistance measured without waves, and R_T is the total resistance measured in waves.

Many articles present successful RANS-based CFD seakeeping simulations of a ship in head waves. Orihara and Miyata performed a simulation for the SR108 container ship and M58 series medium-speed tankers using a RANS-based solver, WISDAM-X [124]. Carrica et al. applied the unsteady single-phase level set method and the rigid overset grids method to DTMB model 5512. The authors evaluated the coupled heave-pitch motions in regular head waves [125]. Simonsen et al. compared the ship motion and the resistance from experiment with CFD results for the KRISO container ship (KCS) in regular head waves. The numerical codes used were AEGIR based on potential theory and RANS-based codes CFDSHIP-Iowa and Star-CCM+ [126]. Tezdogan et al. used the RANS-based solver Star-CCM+ to predict the motion and added resistance of a full-scale KCS with two design speeds [127]. Recently, the capability of modern CFD codes to perform more advanced simulations such as ship manoeuvring simulation with propeller and rudders in waves was demonstrated. For instance, Broglia et al. investigated the turning ability of a twin-screw single-rudder model performing a self-propelled free manoeuvring [128].

In an effort to benchmark the numerical codes, the 'CFD workshop' is a famous naval workshop where numerous participants solve the same problem using their respective solvers. The recent workshop, Tokyo 2015 workshop [129], demonstrated the capability of CFD codes for seakeeping analysis in a head sea. Some benchmarks were also defined with oblique seas but only a few institutions were successful on these cases at that time.

Other numerical formulations have been found useful to tackle seakeeping and added resistance problem. In particular the efficiency of the SWENSE method has been shown with a single phase solver based on the finite difference method with and without forward speed [22, 130, 131]. However, the two-phase SWENSE method efficiency has not been validated with the averaged two-phase flow model for simulations with forward speed. Therefore, the main objective of this chapter is to compare the ability of the *foamStar* and *foamStarSWENSE* solvers with the DIRK time integration scheme ability for naval seakeeping applications, and to test the efficiency of these solvers.

Two ship models are considered, the Wigley III hull, and the KCS hull. Both ships are studied at design speeds. The numerical results obtained with the *foamStar* and *foamStarSWENSE* solvers are compared to experimental data in regular wave conditions with 3DoF motions and forward speed.

8.1 Wigley III

8.1.1 Wigley III: Geometry and test matrix

The first seakeeping analysis is conducted on the Wigley hull, which is a mathematically defined hull shape already used as a validation case in numerous articles. The Wigley hull used in the present study is commonly named 'Wigley III', and its hull surface ($\mathbf{x}(x, y, z)$) is defined in Eqn. (8.2):

$$\frac{2y}{B} = \left[1 - \left(\frac{z}{d}\right)^2\right]\left[1 - \left(\frac{2x}{L}\right)^2\right]\left[1 + 0.2\left(\frac{2x}{L}\right)^2\right] \tag{8.2}$$

where L, B and d are the length, beam and draft of the Wigley hull, respectively. The main dimensions and some other important parameters of the Wigley hull are listed in Table 8.1. The initial trim of the Wigley hull is zero.

Table 8.1. Main dimensions of the Wigley III model for both experiments and CFD simulations

Wigley III	**Unit**	**Value**
Length (L)	$[m]$	3.0
Beam (B)	$[m]$	0.3
Draft (d)	$[m]$	0.1875
Volume of displacement (∇)	$[m^3]$	0.0780
Vertical distance from keel to COG	$[m]$	0.1875
Radius of inertia (K_{yy})	$[m]$	0.750

Table 8.2 presents the motion constraints of the Wigley III hull. The heave and pitch motions are free, and the surge motion is predefined from the ship design velocity and the ramp function.

Table 8.2. The six degrees of freedom of the Wigley hull used for experiments and numerical simulations

Surge	**Sway**	**Heave**	**Roll**	**Pitch**	**Yaw**
Towing	*Fixed*	*Free*	*Fixed*	*Free*	*Fixed*

Table 8.3 presents the full numerical test matrix based on the experiments of Journée [132]

with the Wigley III hull. Three Froude numbers (Fr) are available in the campaign. For each of them the calm water resistance is also obtained through dedicated simulations to later estimate the added resistance. Six different wavelengths (λ_0) are selected from cases available in the experimental campaign. The wavelength (λ_0) represents the linear wave theory wavelength with deep water assumption. Most of the cases are run with wave height $H = 2A = 0.04m$, which is similar to the one of the experiments. Two additional wave heights are tested only for wavelength $\lambda_0/L = 1.25$ to evaluate the effect of the wave steepness on the motions and the added resistance. For each condition, the regular waves are reconstructed using the stream function wave theory with the given wave height and wave period. The study is performed in two steps. First, some parametric studies and convergence tests are performed on the case defined with the parameters indicated in bold in Table 8.3. From this preliminary step the mesh refinement and the time step are fixed and further tests are performed on the full numerical test matrix varying the wavelength.

Table 8.3. Test matrix for Wigley III simulations

Wigley III	**Unit**	**Values**		
Froude Number (Fr)	-	**0.2**	0.3	0.4
Normalized wavelength (λ_0/L)	-	0.75, 1.0, **1.25**, 1.5, 1.75, 2.0		
Wave height ($H = 2A$)	$[m]$	**0.04**, 0.06, 0.08		

8.1.2 Wigley III: Numerical setup

The numerical setup of the computations is presented in Figures 8.1 and 8.2.

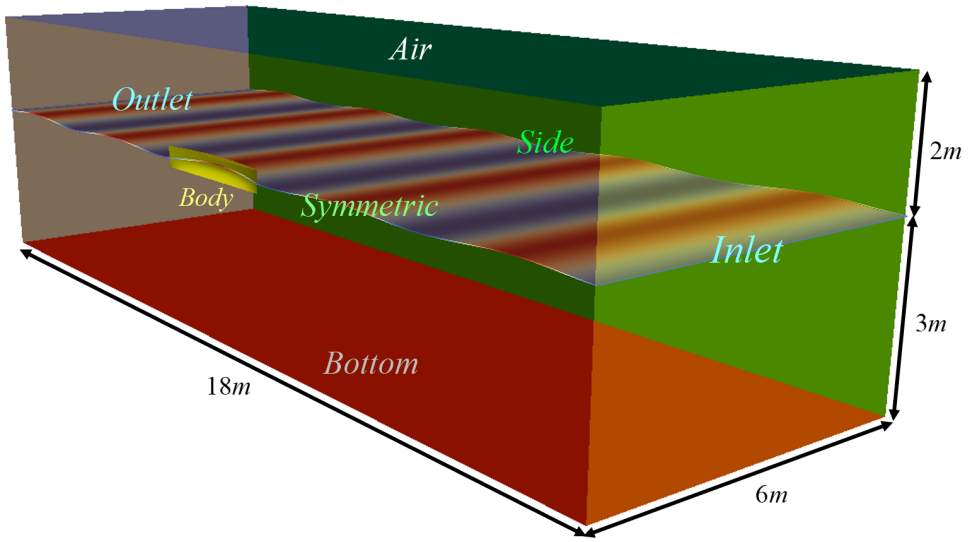


Figure 8.1. The boundary conditions for Wigley III simulation.

Figure 8.1 illustrates the Wigley III hull positioned in the domain together with the initialized

waves at t=0. The initial trim of the Wigley hull is zero. The shape of the hull over the draft line is identical to the waterplane, and its length is $1.5d$. The names of the OpenFOAM patches used to apply the boundary conditions are also indicated. The incident wavefield is initialized at the beginning of the simulation to facilitate the convergence of the simulation. As presented in Section 4.5, the waves are generated and absorbed with explicit relaxation zones positioned at the inlet, outlet and side boundaries. Figure 8.2 illustrates the configuration of the relaxation zones (length and exponential weight function through color contours). The lengths of the relaxation zones are $1.5L$ for 'inlet' and 'outlet' boundaries and $1L$ for the 'side' boundary.

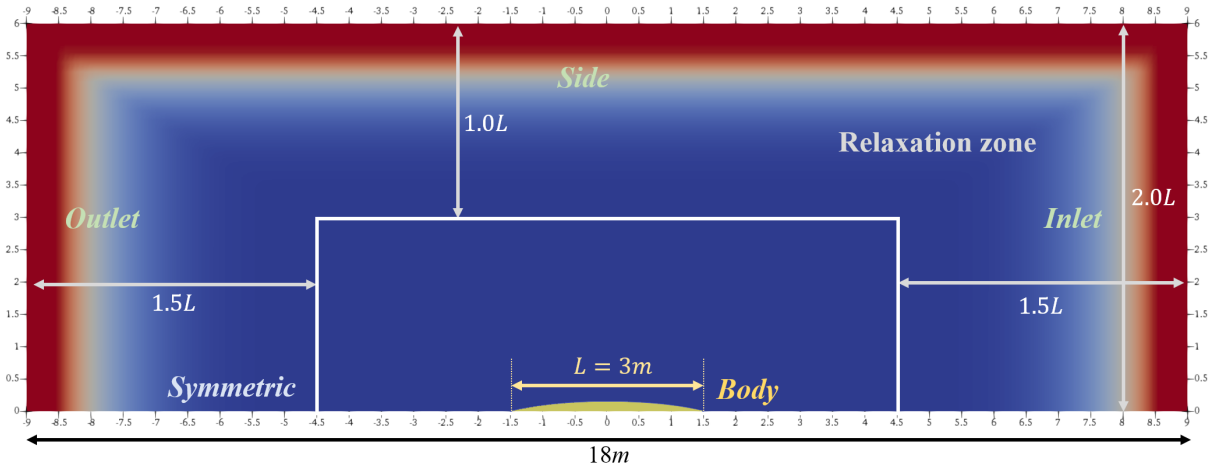


Figure 8.2. The top view of the Wigley III hull and the relaxation zones. The color contours represent the weight function intensity: red represents $w = 1$ and the blue represents $w = 0$.

Table 8.4 presents the boundary conditions for the Wigley III simulations. The variables with subscript $(...)_C$ are used only in SWENSE simulations. The details on each type of boundary condition can be found in Section 4.9. Note that the body boundary conditions for the turbulence variables are selected using trial and error between various selectable wall functions in OpenFOAM minimizing the difference with the calm water resistance obtained experimentally.

Table 8.4. Boundary conditions for Wigley III simulations, the blue rows represent the variables only used in the SWENSE formulation

	Body	**Inlet/Outlet/Side**	**Air**	**Bottom**
u	$\mathbf{u}_{\text{Body}}$	$\mathbf{u}_{\text{I}}$	Pressure outlet	$\mathbf{u}_{\text{Wall}}$
$\mathbf{u}_{\text{C}}$	$\mathbf{u}_{\text{Body}} - \mathbf{u}_{\text{Inc}}$	zero vector	Pressure outlet	slip
p_{d}	zeroGradient	zeroGradient	zeroGradient	zeroGradient
p_{C}	fixedFluxPressure	zeroGradient	zeroGradient	zeroGradient
α	zeroGradient	α_{water}	inlet-outlet	zeroGradient
k	kqRWallFunction	zeroGradient	zeroGradient	zeroGradient
ω	omegaWallFunction	zeroGradient	zeroGradient	zeroGradient
ν_t	nutUSpaldingWallFunction	calculated	calculated	calculated

The forward speed velocity is imposed by blocking the surge and using the COG as the towing point (the COG being free to heave). The forward velocity is gradually increased using the ramp function in Eqn. (4.22). The total length of the simulation is 20s, and the ramp time (t_R) is 4s. Table 8.5 presents the parameters of two mesh refinements of the computational domain which are referred to as *Coarse* (around 250000 cells) and *Medium* (around 500000 cells). The reference resolution is the *Medium* mesh.

Table 8.5. Wigley III meshes with two resolutions

Refinements	$L/\Delta x$	$L/\Delta y$	Δz (m)	$\Delta z/\Delta x$	δ (m)	N Cell
Coarse	20	20	0.0375	0.25	0.0089	240,557
Medium (Ref)	40	20	0.0187	0.25	0.0049	498,590

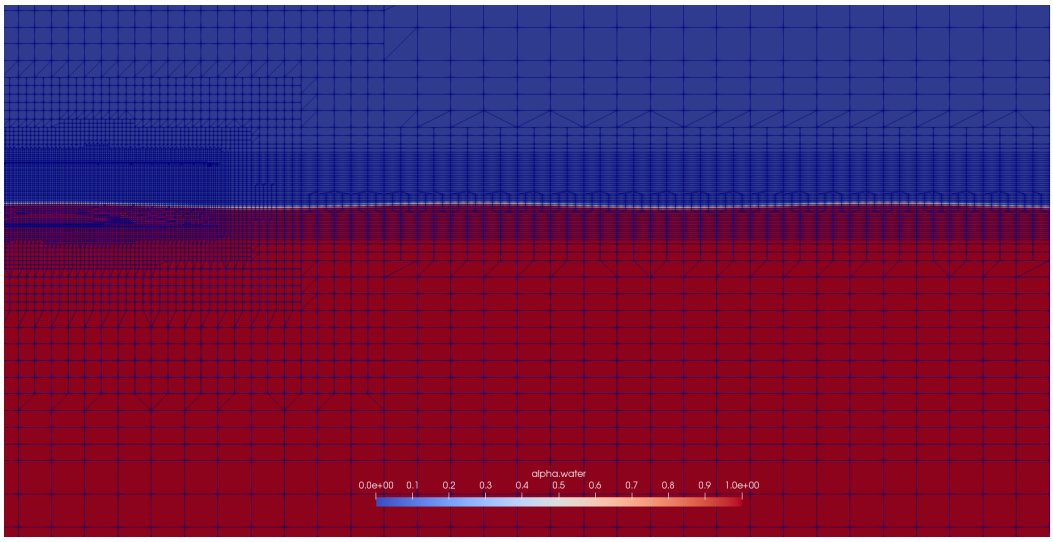


Figure 8.3. Wigley *Normal* mesh with initialized VOF field

Figure 8.3 presents the initialized VOF field on the computational mesh. It is made using the Star-CCM+ mesh generator to generate the unstructured polyhedral mesh. While it is a polyhedral mesh generator, the free-surface mesh is close to be a Cartesian mesh. Table 8.5 presents the size of the free-surface cells which are far from the body. The smallest cell in the domain is attracted to the body surface, and their vertical cell size is same to the free-surface vertical cell size (Δz) to avoid the separation of VOF field during the forwarding motion. A boundary layer mesh is applied along the Wigley hull to measure correctly the viscous force on the body. As the mesh refinement level increases, the boundary layer cell size (δ) is reduced to half. The maximum cell size in the domain is 0.3m $\times$ 0.3m $\times$ 0.3m cubic at the bottom of the relaxation zone.

8.1.3 Wigley III: Efficiency of DIRK methods for the *foamStar* and *foamStar-SWENSE* solvers: effect of time-step size and mesh refinement

This section presents the efficiency of the DIRK schemes for both *foamStar* and *foamStar-SWENSE* for the advancing Wigley hull. The following summarizes the computational setups used for the efficiency study:

- Computational condition: Fr= 0.2, $\lambda_0/L = 1.25$, $H = 0.04$m
- Two solvers: *foamStar* & *foamStarSWENSE*
- Six time integration schemes listed in Table 8.6
- Three time-step sizes: $h = 0.004$s, 0.008s, 0.016s
- Two mesh resolutions: *Coarse* & *Medium* mesh conditions listed in Table 8.5
- 8 PIMPLE iterations, 4 PISO iterations, and one non-orthogonal correction

Table 8.6 presents the DIRK schemes compared in this section. These schemes were already detailed in Chapter 6. Note that it was found that varying τ when applying the DIRK method causes instability in the dynamic pressure in the presence of a body, therefore *SDIRK[2](2)* and *SDIRK[3](3)* were not used in this section. The numerical setups include the implicit Euler time integration scheme, even if it makes significant numerical damping, since this scheme is still commonly used for various CFD naval applications due to its robustness. With the six time-integration schemes, three time-step sizes and two mesh resolutions, 72 cases are used to study convergence and efficiency.

Table 8.6. Names and properties of the DIRK schemes used for Wigley III simulations.

Code name	Identification name	Order	Implicit stage
Euler	Implicit Euler	1	1
RKCN	ESDIRK[2](2)_SA	2	1
ESDIRK32	ESDIRK[3](2)_SA	2	2
EDIRK331	EDIRK[3](3)_SA_1	3	2
ESDIRK33	ESDIRK[3](3)_SA	3	2
EDIRK332	EDIRK[3](3)_SA_2	3	2

Comparison of DIRK schemes with different time-step sizes

Figures 8.4 and 8.5 illustrate the heave and pitch motions, and the total resistance obtained from the *foamStar* solver with the *Medium* mesh. The dashed line in Figure 8.5 is the mean of last ten periods of the total resistance. The experimental data plotted in the total resistance graphs refer to the measured calm water resistance of the Wigley hull without waves.

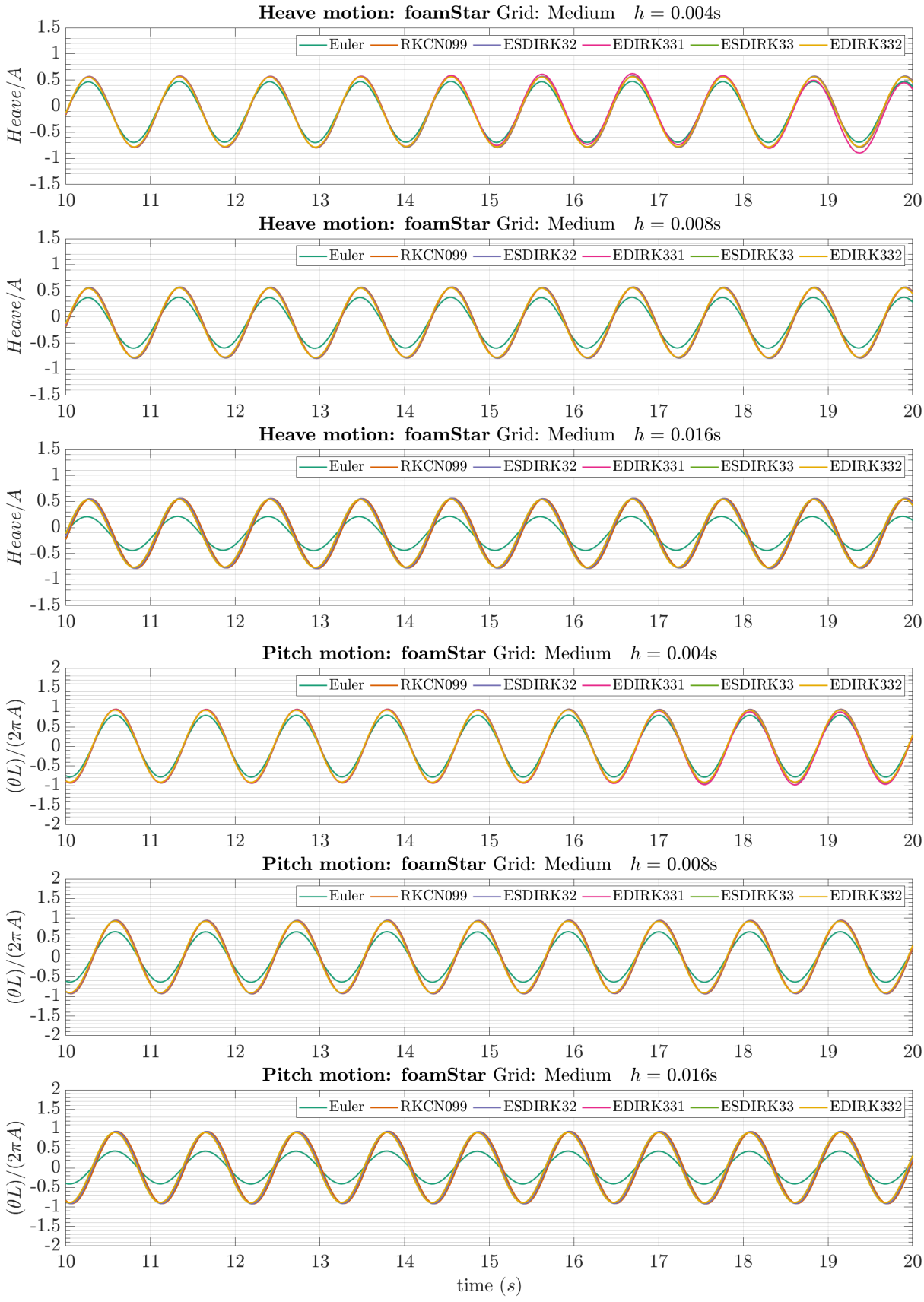


Figure 8.4. Heave and pitch motions predicted by the *foamStar* solver for the *Medium* mesh and time-step sizes $h = 0.004$s, 0.008s, 0.016s

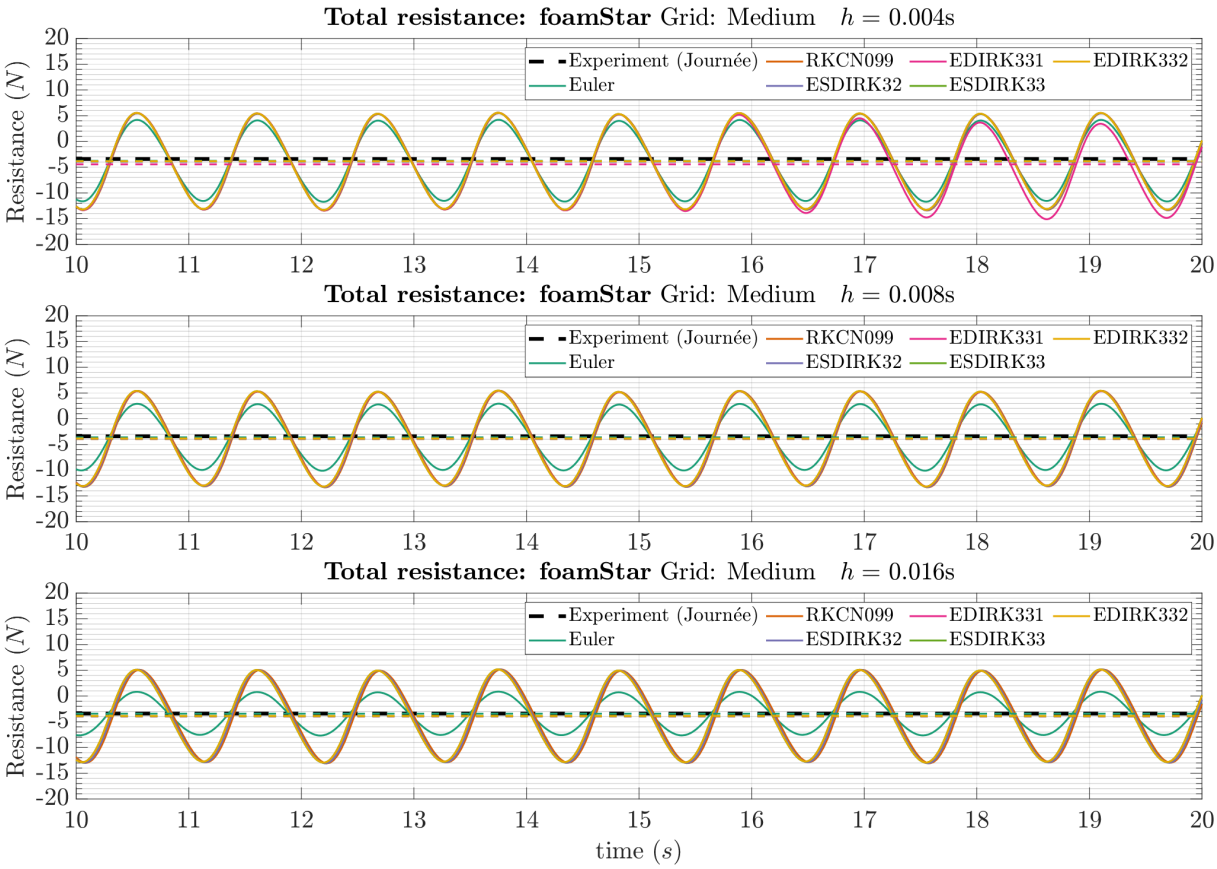


Figure 8.5. Total resistance predicted by the *foamStar* solver for the *Medium* mesh and for the three time-step sizes $h = 0.004$s, 0.008s, 0.016s

For all cases, the *implicit Euler* scheme showed a significant decrease in motions and total resistance, which was less important as the used time-step size decreased. The *EDIRK331* scheme showed a significant change of the mean heave motion and the mean total resistance after some evolution time, with the small time-step size ($h = 0.004$s) showing the unstable behaviour of this specific scheme. For the other DIRK schemes, the difference of motions and total resistance between the different used time-step sizes are minor, showing that the simulation is already converged in time with the largest used time-step size, and stable with those schemes.

Comparison of DIRK schemes with the same time-step size and different mesh resolutions with *foamStar* and *foamStarSWENSE* solvers

Figure 8.6 compares the heave and pitch motions, and total resistance between the *Coarse* and *Medium* meshes, with the same time-step size $h = 0.004$s. As shown in the previous results, with the *EDIRK331* scheme spurious mean heave motion and mean total resistance are captured. As the mesh resolution increased from *Coarse* to *Medium*, the increase of mean heave and mean total resistance is observed.

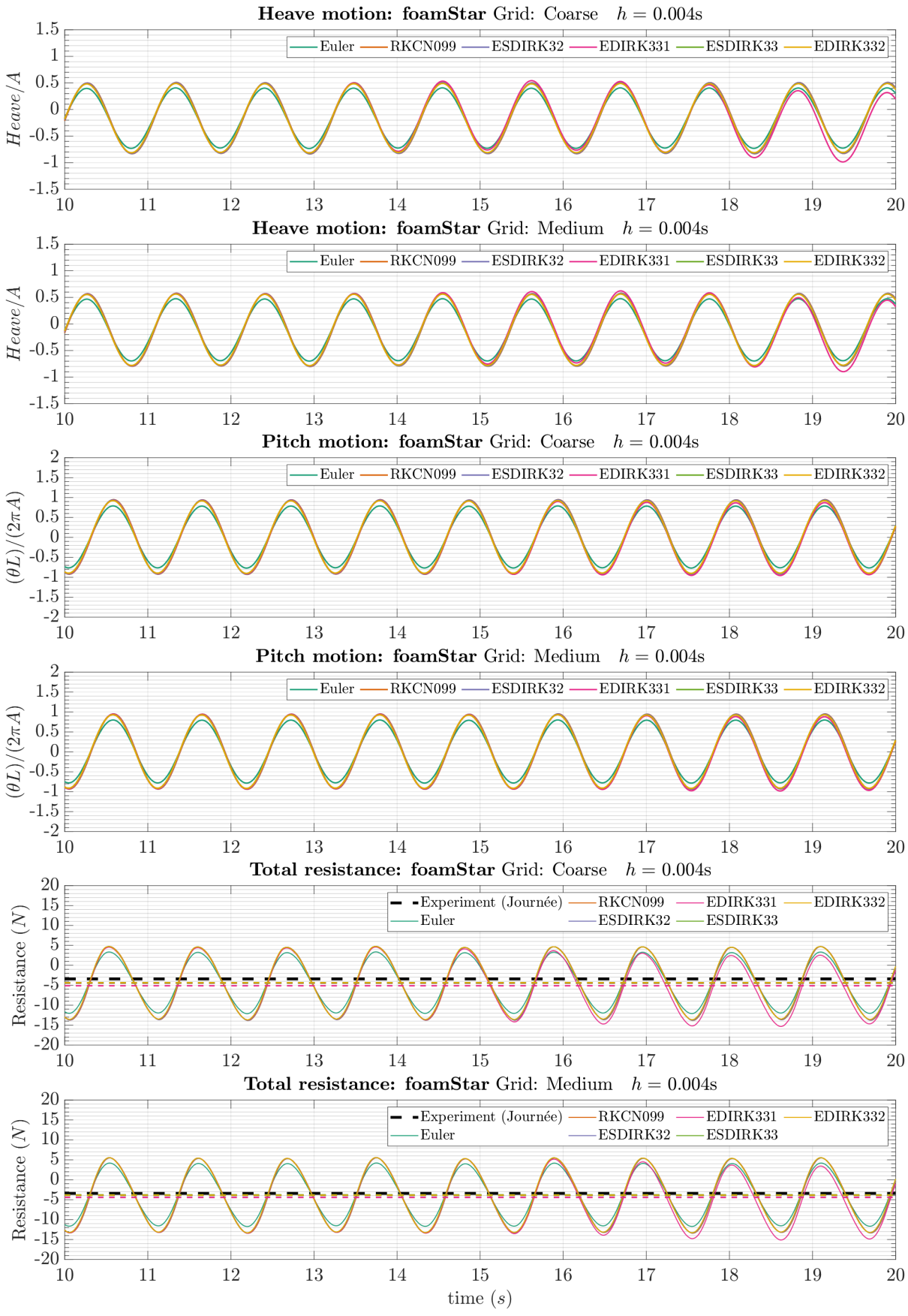


Figure 8.6. Heave and pitch motions, and total resistance for the *Coarse* and *Medium* meshes, with time-step size $h = 0.004$s, for the *foamStar* solver

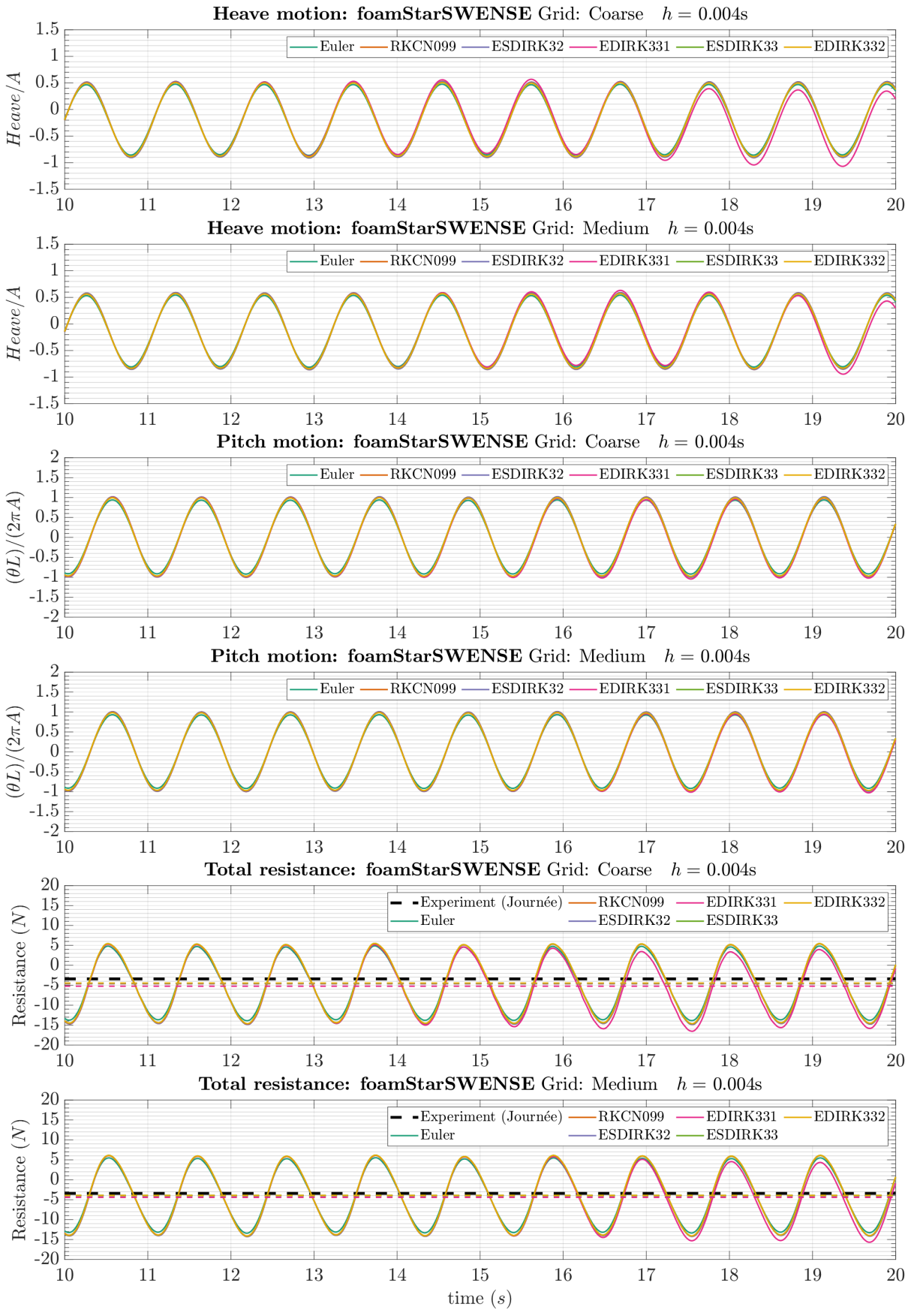


Figure 8.7. Heave and pitch motions, and total resistance for the *Coarse* and *Medium* meshes, with time-step size $h = 0.004$s, for the *foamStarSWENSE* solver

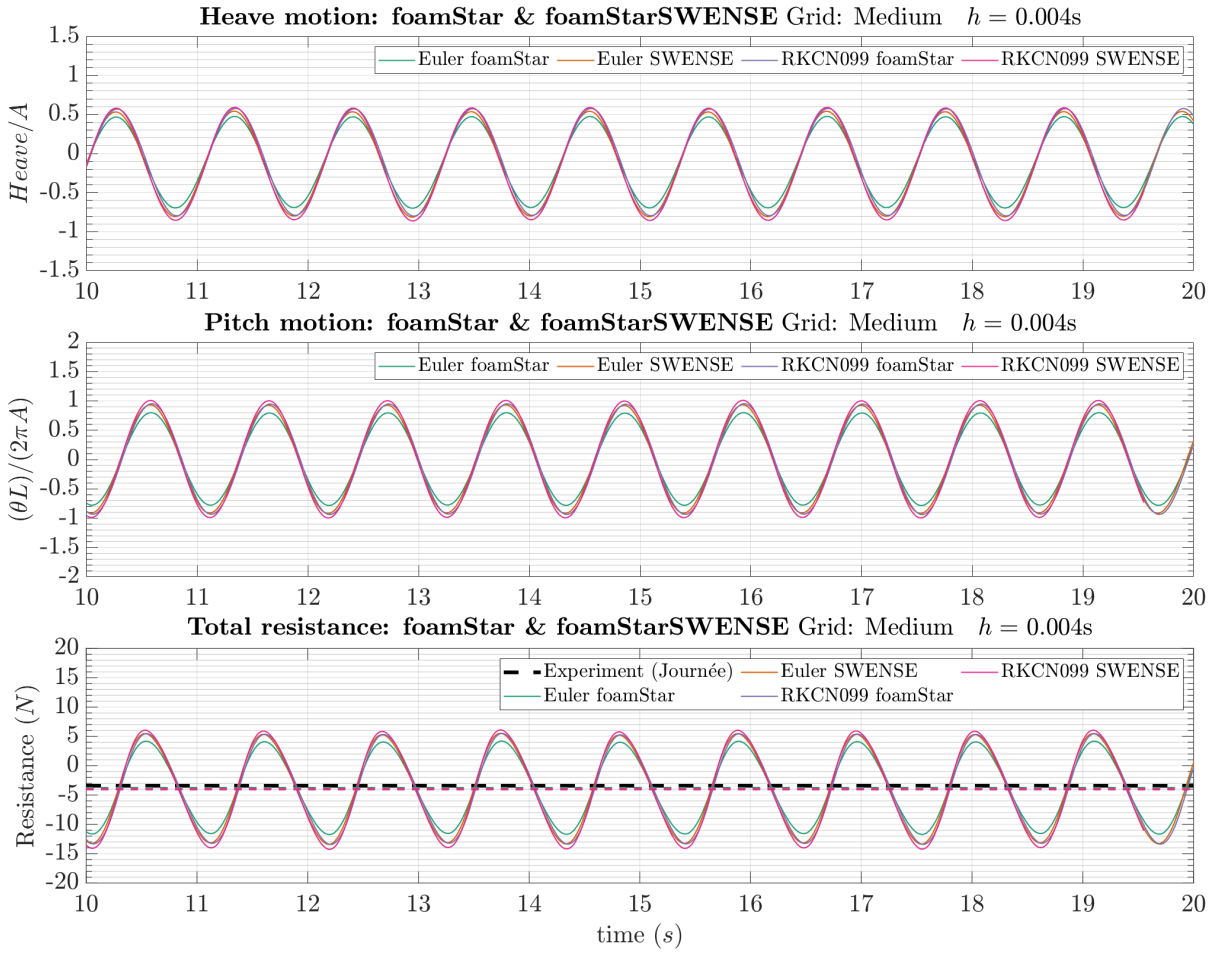


Figure 8.8. Heave and pitch motions, and total resistance for the *Medium* mesh, with time-step size $h = 0.004$s, for the *foamStar* and *foamStarSWENSE* solvers with *implicit Euler* and *RKCN099* schemes.

Similarly, Figure 8.7 compares the motions and resistance obtained from the *foamStarSWENSE* solver for the two mesh refinements. Compare to the *foamStar* solver, *foamStarSWENSE* solver gives better results with the *implicit Euler* scheme, and shows similar strange mean heave motion with the *EDIRK331* scheme. For a better comparison between *foamStar* and *foamStarSWENSE*, Figure 8.8 compares the motions and resistance for two time schemes, *implicit Euler* and *RKCN099*, with the *Medium* mesh. The amplitude of motions and total resistance for *foamStarSWENSE* is larger than *foamStar* for both *implicit Euler* and *RKCN099* schemes.

Accuracy and efficiency

Since the current study had to be less systematic than the previous NWT one for computational load reasons, the ship motions and resistance are here compared more qualitatively in terms of average of the last ten periods and CPU times. Tables 8.8 and 8.9 list the results of averaged total resistance, added resistance, and body motions from the *foamStar* and *foamStarSWENSE* solvers. In both tables, the two mesh resolutions, three time-step sizes, and six

time integration schemes are compared. Not all simulations successfully run up to the end of the 20s of physical time. If the computed time was less than 15s of physical time, the result was considered non-valid and the corresponding cell of the table is empty. In general, except for the low-order *implicit Euler* scheme and the unstable *EDIRK331* scheme, the differences between the different time schemes were found to be small, especially for the finer mesh. For the rest, the *foamStarSWENSE* solver was less robust to the increase of the time-step size since no simulation except with the *implicit Euler* scheme run up to the end of the 20s with the *Medium* mesh resolution and the finest time-step size $h = 0.016$s.

Table 8.7 presents the computational cost of each simulation. When the simulation is stopped due to any reason, the computational time is not reported in the table. The DIRK methods with more than two implicit sub-stages require more computational cost than the *implicit Euler* and *RKCN099* methods. However, the benefit of multiple sub-stages is not clear from the different figures shown in this section. Another conclusion is that the SWENSE method requires higher computational costs compared to the *foamStar* solver, which is due to the slower convergence of complementary pressure. However, the extra CPU cost is limited, on average of around 30%, and it should be verified with a more complete convergence study and comparison to reference results whether the use of SWENSE keeps a better global efficiency by more than compensating this extra cost with extra accuracy for a given resolution.

Table 8.7. Comparison of the computational time of Wigley III hull simulations.

Mesh resolutions	*Coarse* mesh			*Medium* mesh		
time-step size	0.004s	0.008s	0.016s	0.004s	0.008s	0.016s
	computational time (hour) *foamStar*					
implicit Euler	60.21	43.39	22.06	260.12	135.07	69.93
RKCN099	95.43	48.61	24.73	290.09	145.97	71.55
ESDIRK32	171.71	89.15	43.49	587.16	299.52	152.84
EDIRK331	196.45	97.40	49.62	588.90	315.15	-
ESDIRK33	201.81	100.44	52.69	612.71	323.04	168.29
EDIRK332	162.15	81.41	44.06	615.76	319.72	168.80
	computational time (hour) *foamStarSWENSE*					
implicit Euler	103.79	56.57	29.49	343.77	181.67	99.40
RKCN099	107.99	58.22	30.22	355.56	190.48	-
ESDIRK32	213.86	119.36	61.08	859.02	378.89	-
EDIRK331	224.68	120.22	64.87	819.10	385.99	-
ESDIRK33	225.41	164.67	66.43	876.75	391.32	-
EDIRK332	224.04	123.54	55.78	837.78	397.15	-

Table 8.8. Wigley III computations results from *foamStar* solver.

Mesh resolutions	*Coarse* mesh			*Medium* mesh		
time-step size	0.004s	0.008s	0.016s	0.004s	0.008s	0.016s
	Added resistance $R_{AR} = \frac{R_T - R_S}{\rho g (B^2/L)(H/2)^2}$					
Experiment	Averaged experiment result: $R_{AR} = 6.58$					
implicit Euler	7.94	6.61	4.84	2.96	1.57	-0.02
RKCN099	9.12	9.00	9.00	4.49	4.41	4.54
ESDIRK32	9.05	8.95	8.98	4.42	4.36	4.52
EDIRK331	14.57	8.47	8.26	8.75	3.75	-
ESDIRK33	8.10	8.07	8.17	3.60	3.60	3.89
EDIRK332	8.08	8.05	8.14	3.59	3.57	3.82
	Total resistance 1st harmonic amplitude $[N]$					
implicit Euler	7.56	6.19	4.04	7.78	6.38	4.18
RKCN099	9.03	8.91	8.64	9.22	9.13	8.87
ESDIRK32	9.02	8.91	8.67	9.21	9.12	8.90
EDIRK331	8.82	8.66	8.27	9.20	9.05	-
ESDIRK33	8.96	8.83	8.56	9.19	9.08	8.83
EDIRK332	8.96	8.82	8.55	9.19	9.08	8.82
	Non-dimensional heave amplitude					
Experiment	Averaged experiment result: $\frac{heave}{(H/2)} = 0.716$					
implicit Euler	0.566	0.468	0.313	0.583	0.486	0.326
RKCN099	0.667	0.660	0.646	0.684	0.680	0.668
ESDIRK32	0.666	0.660	0.647	0.683	0.679	0.669
EDIRK331	0.650	0.636	0.611	0.674	0.664	-
ESDIRK33	0.650	0.643	0.629	0.670	0.666	0.653
EDIRK332	0.649	0.642	0.627	0.670	0.665	0.652
	Non-dimensional pitch amplitude					
Experiment	Averaged experiment result: $\frac{L\theta}{(2/\pi)(H/2)} = 1.106$					
implicit Euler	0.775	0.630	0.406	0.788	0.643	0.419
RKCN099	0.933	0.924	0.905	0.941	0.937	0.922
ESDIRK32	0.930	0.922	0.907	0.938	0.934	0.923
EDIRK331	0.914	0.893	0.856	0.933	0.916	-
ESDIRK33	0.909	0.898	0.879	0.921	0.914	0.899
EDIRK332	0.908	0.897	0.877	0.920	0.913	0.897

Table 8.9. Wigley III computations results from *foamStarSWENSE* solver.

Mesh resolutions	*Coarse* mesh			*Medium* mesh		
time-step size	0.004s	0.008s	0.016s	0.004s	0.008s	0.016s
	Added resistance $R_{AR} = \frac{R_T - R_S}{\rho g (B^2/L)(H/2)^2}$					
Experiment	Averaged experiment result: $R_{AR} = 6.58$					
implicit Euler	8.98	8.20	7.28	4.13	3.20	2.04
RKCN099	10.11	9.93	9.88	5.36	5.05	5.52
ESDIRK32	10.04	9.89	9.88	5.13	5.04	5.27
EDIRK331	15.52	9.24	9.07	8.55	4.28	-
ESDIRK33	8.88	8.85	9.00	4.25	4.28	-
EDIRK332	8.84	8.81	8.94	4.22	4.22	-
	Total resistance 1st harmonic amplitude $[N]$					
implicit Euler	9.07	8.30	7.20	9.19	8.43	7.27
RKCN099	9.80	9.65	9.38	9.87	9.74	9.48
ESDIRK32	9.79	9.66	9.40	9.87	9.74	9.50
EDIRK331	9.65	9.50	9.17	9.77	9.63	-
ESDIRK33	9.74	9.59	9.32	9.84	9.70	-
EDIRK332	9.73	9.59	9.31	9.84	9.70	-
	Non-dimensional heave amplitude					
Experiment	Averaged experiment result: $\frac{heave}{(H/2)} = 0.716$					
implicit Euler	0.663	0.609	0.529	0.672	0.621	0.542
RKCN099	0.713	0.705	0.689	0.720	0.713	0.701
ESDIRK32	0.712	0.704	0.689	0.718	0.712	0.699
EDIRK331	0.697	0.683	0.663	0.702	0.692	-
ESDIRK33	0.694	0.686	0.671	0.702	0.696	-
EDIRK332	0.693	0.685	0.670	0.701	0.694	-
	Non-dimensional pitch amplitude					
Experiment	Averaged experiment result: $\frac{L\theta}{(2/\pi)(H/2)} = 1.106$					
implicit Euler	0.925	0.845	0.729	0.920	0.843	0.726
RKCN099	1.006	0.994	0.975	0.997	0.989	0.977
ESDIRK32	1.004	0.993	0.975	0.996	0.988	0.976
EDIRK331	0.991	0.971	0.942	0.977	0.964	-
ESDIRK33	0.981	0.970	0.950	0.975	0.966	-
EDIRK332	0.980	0.968	0.949	0.974	0.964	-

8.1.4 Wigley III: RAOs comparison with experiments

This section presents a comparative analysis of the heave, pitch and added resistance of the numerical results with the experimental data reported by Journée [132] on a set of wavelengths. Table 8.10 gives the test matrix and computational parameters for the Wigley III simulations considered in this subsection. For each Froude number, six wave conditions with wave height $H = 0.04$m are computed. Two additional wave heights ($H = 0.06$m and $H = 0.08$m) are tested only for $\lambda_0/L = 1.25$ to check the effect of wave steepness on the added resistance. As in the efficiency tests of the previous subsection, the initial trim is zero, and the ship is free to heave and pitch. The total simulation time is 20s, and the ramp time is 4s regardless of the Froude number. The time-step size of the simulation is specific to the Froude number, and listed in Table 8.10. For Froude number Fr= 0.4, a smaller time-step size ($h = 0.0025$) is used for the SWENSE solver due to an unstable behaviour.

The following describes the motion and added resistance results from the *foamStar* and *foamStarSWENSE* solvers. The FFT analyses are performed when the computation has run for more than 15 s. For Froude number 0.2, the analysis was performed on the whole data set, but some of the simulations failed to complete for Froude numbers 0.3 and 0.4 so that the analysis was made only on a partial data set for those Froude numbers. The simulation RAOs are computed from the first harmonic amplitudes at their encounter frequencies dividing by target wave amplitude.

Table 8.10. Detailed test matrix for Wigley III simulations

Wigley III	**Unit**	**Values**		
Froude Number (Fr)	-	0.2	0.3	0.4
Normalized wavelength (λ_0/L)	-	0.75, 1.0, 1.25, 1.5, 1.75, 2.0		
Wave height ($H = 2A$)	$[m]$	0.04, 0.06, 0.08		
Time-step size (h)	$[s]$	0.008	0.004	0.004 / 0.0025
Mesh resolution	-	Medium mesh		
Free-surface cell size (Δx, Δy)	$[m]$	$\Delta x = L/40$, $\Delta y = L/20$		

Figure 8.9 compares the first harmonic motion Response Amplitude Operators (RAOs) obtained with various numerical schemes to experimental data for three Froude numbers. The black round markers denote the experimental data. For each computed frequency, the simulation results with the SWENSE method are shifted a little to the left, and the results from *foamStar* are shifted a little to the right of the experimental results for better visualization.

In general, both *foamStar* and *foamStarSWENSE* solvers estimate motion RAOs satisfactorily when comparing to the experiment. Generally, the motion RAOs results with *foamStarSWENSE* solver are closer to the experiment than with *foamStar* solver. This is attributed to the better wave field obtained with the SWENSE solver: if the measured amplitude at the ship position is lower than the target one, the RAOs will also be lower. This partly answers the

previous question raised on the efficiency, since the accuracy gained with the SWENSE solver probably overcompensate the 20% $\sim$ 30% extra CPU cost found in the previous subsection.

The two implicit stage DIRK schemes, *ESDIRK33* and *EDIRK332* schemes, do not give significant benefit in the motion RAOs' estimation with respect to the *RKCN099* method even if their convergence order is higher. Considering that simulations maximum Courant numbers were over ten during the simulation, the choice of time integration scheme seems not to affect much the body motions.

Figure 8.10 emphasizes the effect of wave steepness on the motion RAOs for the normalized wavelength 1.25 and three different Froude numbers. The simulation data are again a little shifted for better visualization. As expected, the wave amplitude has a moderate influence on the motion RAOs. The largest variations are observed for Froude number 0.3 compared to Froude numbers 0.2 and 0.4. This result seems to be related to the heave and pitch resonance near the non dimensional wavelength 1.25. At this Froude number, the non-dimensional heave motion increases with the wave steepness while the non-dimensional pitch motion decreases.

Figure 8.11 illustrates the comparison of the added resistance obtained from the experiments and the numerical simulations. For Froude number 0.2, the general trend of added resistance from the two solvers shows good agreement with the experiment, even if for wavelength $\lambda = L$, the simulation results are a bit lower than the experiment. For Froude numbers 0.3 and 0.4, the SWENSE solver over-estimates the added resistance compared to the experiment and the *foamStar* results.

The bottom right plot of Figure 8.11 emphasizes the effect of wave steepness on the added resistance. For Froude numbers 0.3 and 0.4, the non dimensional added resistance decreases as the wave steepness increases.

Overall, the *foamStarSWENSE* solver showed slightly better motion RAO than *foamStar* for same grid resolution. For the added resistance, the *foamStarSWENSE* solver showed over-estimation of added resistance for high Froude number, while the *foamStar* solver showed better agreement of added resistance with the experiment. However, since motions are underpredicted with the latter solver, this better agreement for added resistance should be taken with care and further investigated, since it could be coincidental.

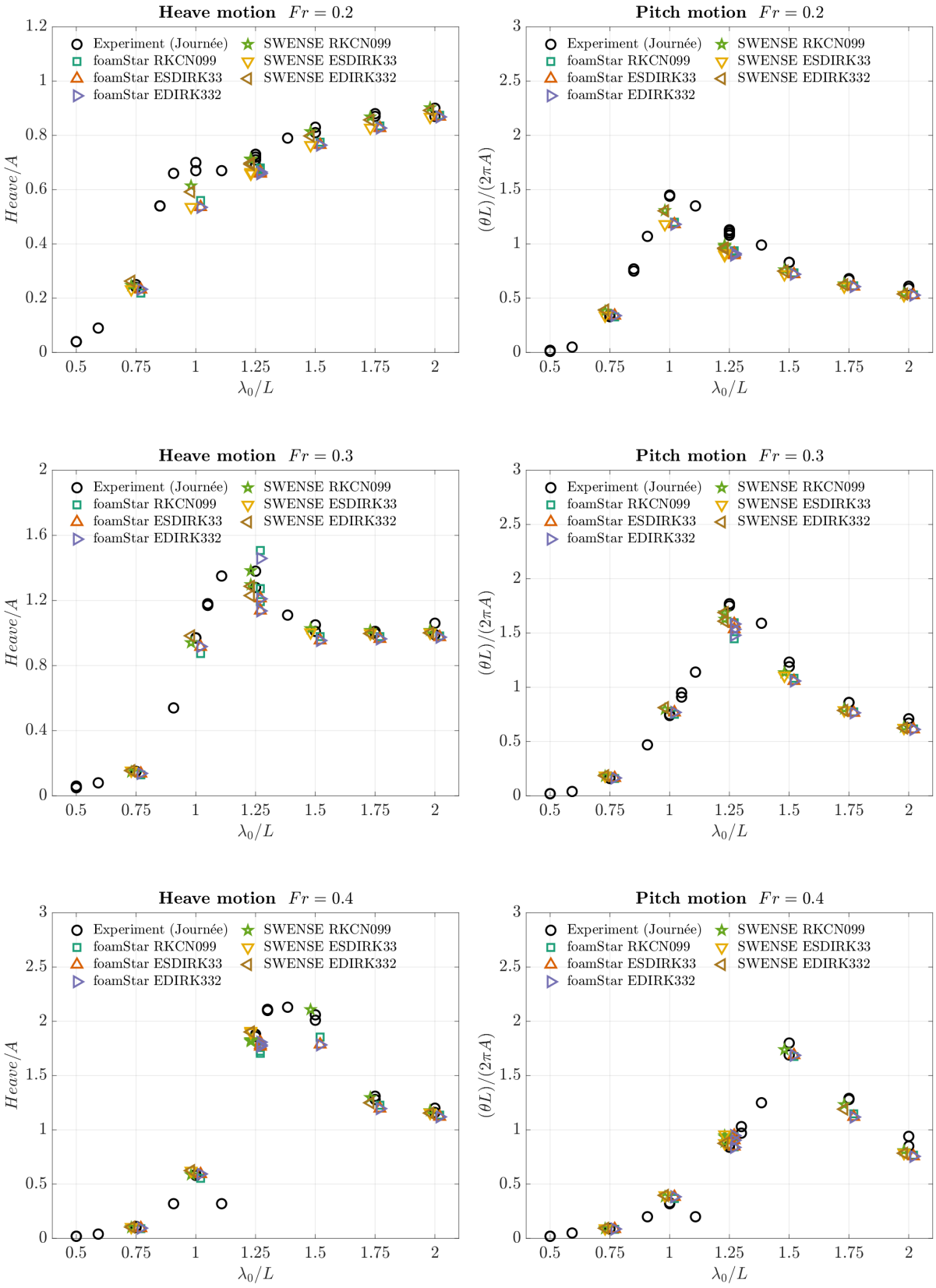


Figure 8.9. Wigley III motion RAOs in head waves for Froude numbers Fr = 0.2, Fr = 0.3 and Fr = 0.4. At each computed frequency, the *foamStar* and *foamStarSWENSE* results are shifted a little to the left and to the right for better visualization.

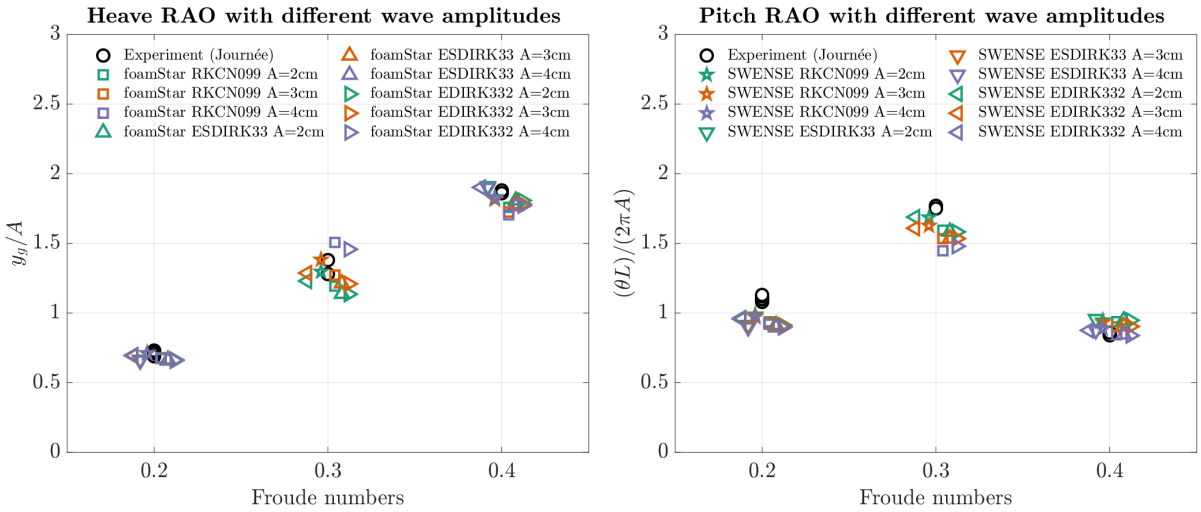


Figure 8.10. Comparison of Wigley III motion RAOs in head waves for Fr = 0.2, Fr = 0.3 and Fr = 0.4 with three wave heights. The numerical results are shifted a little for better visualization.

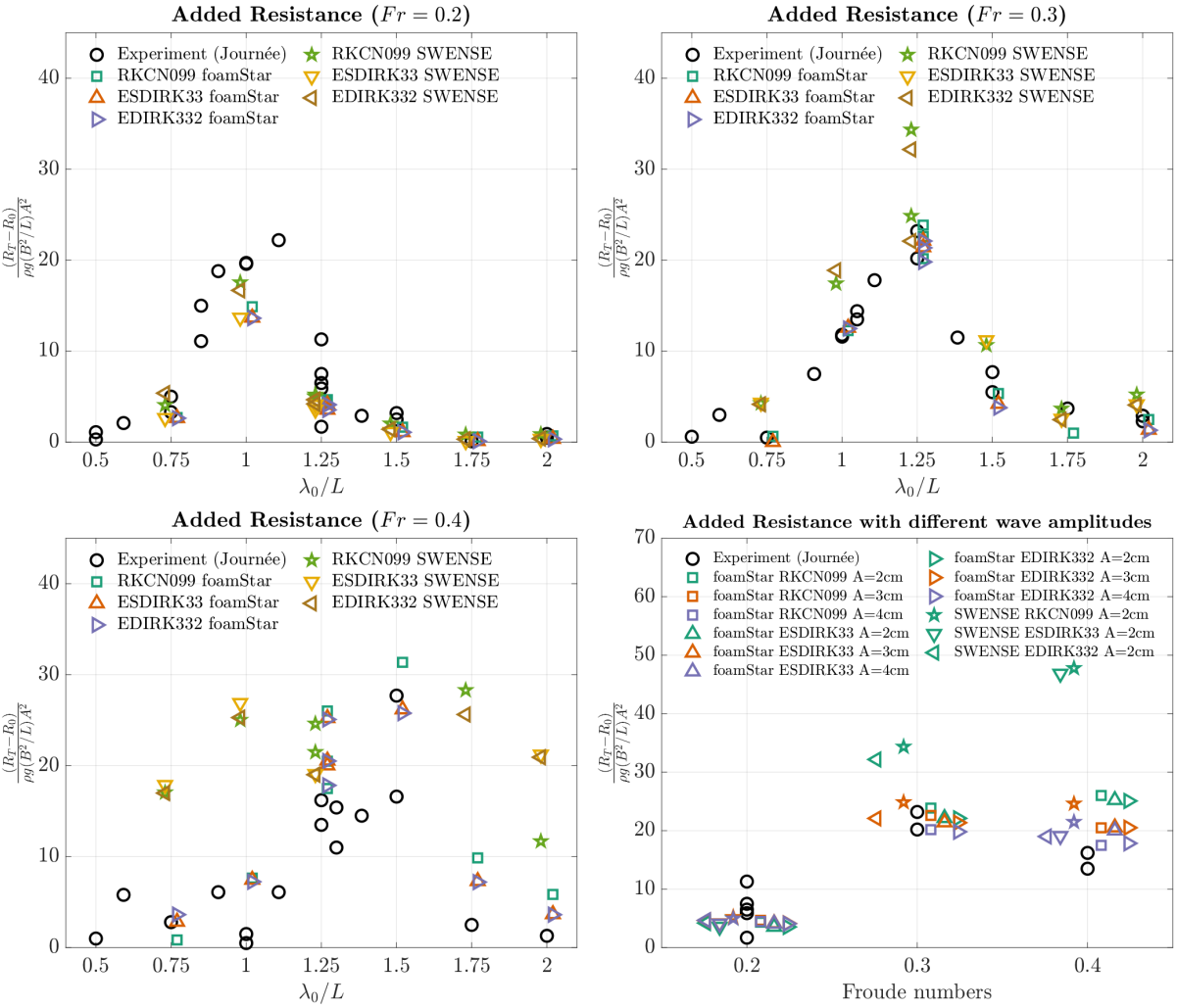


Figure 8.11. Added resistance of the Wigley III hull in head waves for Froude numbers Fr = 0.2, Fr = 0.3 and Fr = 0.4.

8.2 KCS

This section presents another numerical application with the (Korea Research Institute for Ships and Ocean Engineering) KRISO Container ship (KCS), using the same numerical schemes used for Wigley III hull applications. The KCS is a 3,600 TEU container carrier that has been designed for the validation of CFD prediction [133]. The design speed of the KCS is 24 kn ($Fr = 0.26$), and the KCS hull has a low block coefficient [134]. The KCS hull form was also used in the Gothenburg 2010 CFD Workshop [135], and Tokyo 2015 CFD Workshop [129]. In the latter workshop model experiment data of a 6.07m KCS course keeping in head and oblique waves were provided. [136].

8.2.1 KCS: Geometry and test matrix

Table 8.11 lists KCS full and model scale dimensions, mass properties, and natural frequency of motions. The model ship's scale ratio is 37.89. Case 2.10 (head waves) and Case 2.11 (oblique waves) in the Tokyo 2015 workshop reports give more details on the KCS experiment and numerical setups. The current simulations focused on the free-running KCS in head wave condition (Case 2.10), where a fixed rudder without rotation is used. The surge motion is predefined, and the ship is free to heave and pitch.

Table 8.11. Main particulars of KCS geometry

Main particulars	**Full scale** [133]	**Model scale** [136]
Length between perpendiculars ($L = L_{PP}$) [m]	230	6.0702
Length of waterline (L_{WL}) [m]	232.5	6.1357
Maximum beam of waterline (B_{WL}) [m]	32.2	0.8498
Depth (D) [m]	19.0	-
Draft (T) [m]	10.8	0.285
Displacement volume (Δ) [m^3]	52030	0.9571
Wetted surface area w/o rudder (S_W) [m^2]	9424	6.6177
Wetted surface area of rudder (S_R) [m^2]	115	0.0801
LCB (%LPP), fwd+ (L_{CB})	-1.48	-1.48
Vertical center of gravity from the keel (K_G/L) [m]	0.0622	0.0622
Metacentric height (GM) [m]	0.623	0.016
Block coefficient (CB)	0.6505	0.6505
Midship section coefficient (CM)	0.9849	-
Moment of inertia (K_{xx}/B)	0.40	0.40
Moment of inertia (K_{yy}/L_{PP})	0.25	0.252
Heave natural frequency f_{nz} [Hz]	0.11	0.73
Pitch natural frequency $f_{n\theta}$ [Hz]	0.11	0.73

Table 8.12. Degrees of freedom during KCS experiments and numerical simulations

Surge	Sway	Heave	Roll	Pitch	Yaw
Towing	*Fixed*	*Free*	*Fixed*	*Free*	*Fixed*

The following equations define the total resistance coefficient (Eqn. (8.3)), the added resistance coefficient (Eqn. (8.4)), and the motion RAOs (Eqn. (8.5)), where ρ is the water density, $S = S_W$ is the wetted surface excluding the wetted area of the rudder, $L = L_{PP}$ is the reference length of the ship, $B = B_{PP}$ is the reference ship beam, $A = H/2$ is the wave amplitude, n is the n^{th}-order harmonic amplitude. U_0 is the ship design speed given in Table 8.13.

$$C_{Tn} = \frac{R_n}{0.5\rho S_w \mathrm{U}_0} \tag{8.3}$$

$$R_{AR} = \frac{R_T - R_{calm}}{\rho g (B^2/L) A^2} = \frac{S\mathrm{U}_0^2}{2gA^2B^2/L}(C_{T0} - C_{T,calm}) \tag{8.4}$$

$$\mathrm{RAO}_{heave} = heave/A \quad , \quad \mathrm{RAO}_{pitch} = (\theta\lambda)/(2\pi A) \tag{8.5}$$

Table 8.13. Physical parameters for KCS tests in model scale, used in the experiments and numerical simulations

Ship design speed (U_0) [m.s^{-1}]	Froude number (Fr)	Reynolds number (Re)
2.017	0.26	$1.074 \cdot 10^7$

Table 8.14. Regular waves conditions for KCS tests in model scale, used in the experiments and numerical simulations

Case	λ [m]	λ/L_{PP}	f_e/f_{nz}	H [m]	H/λ	T [s]
C1	3.949	0.651	1.56	0.062	0.016	1.5884
C2	5.164	0.851	1.29	0.078	0.015	1.8166
C3	6.979	1.150	1.05	0.123	0.018	2.1110
C4	8.321	1.371	0.93	0.149	0.018	2.3049
C5	11.840	1.951	0.73	0.196	0.017	2.7501

Table 8.14 tabulates the wave conditions. The five wave conditions have a wavelength of $\lambda/L = 0.65$, 0.85, 1.15, 1.37 and 1.95, respectively, with the wave steepness around $H/\lambda = 1/60$. The wave conditions are thus with low steepness, and the second harmonic wave amplitude is

around 1% of the first harmonic wave amplitude. The table lists also the ratios between encounter frequencies (f_e) and heave natural frequency (f_{nz}). The case *C3* has the wave encounter frequency closest to the heave natural frequency, which induced a large dispersion of published results for that case (in particular at the Tokyo 2015 workshop).

8.2.2 KCS: Numerical setup

The viscous flow solver based on SWENSE (*foamStarSWENSE*) and the averaged two-phase viscous flow solver (*foamStar*) are again compared with different computational setups. Figures 8.12 and 8.13 illustrate the dimensions of the computational domain and of the relaxation zones with the name of each boundary patch. The symmetric boundary condition is applied at the 'symmetric' patch, and only half of the domain is modelled. Table 8.15 summarizes the other boundary conditions for the KCS simulations. The blue rows are the boundary conditions which apply only to *foamStarSWENSE* computations.

The origin of the x-axis is the Aft Perpendicular (AP) of the ship, and the one of the z-axis is the mean free-surface level. The dimension of the computational domain is 32m × 12m × 5.5m. The water depth of the experiment is not described in the Tokyo 2015 Workshop report; therefore, the water depth of the FORCE technology's towing tank (240m × 12m × 5.5m) is used. The 'Air' patch is set 5.5m above the free-surface to avoid spurious effect from the open-air boundary condition. Similar to what has been done with the Wigley III setup, explicit relaxation zones are positioned at the inlet, outlet and side boundaries. Figure 8.13 illustrates the configuration of the relaxation zones. The relaxation zone lengths are 6m each, and the exponential weight function is given as color contours. Depending on the wave conditions in Table 8.14, the relaxation zone lengths (6m) are between 0.5 and 1.5 wavelength.

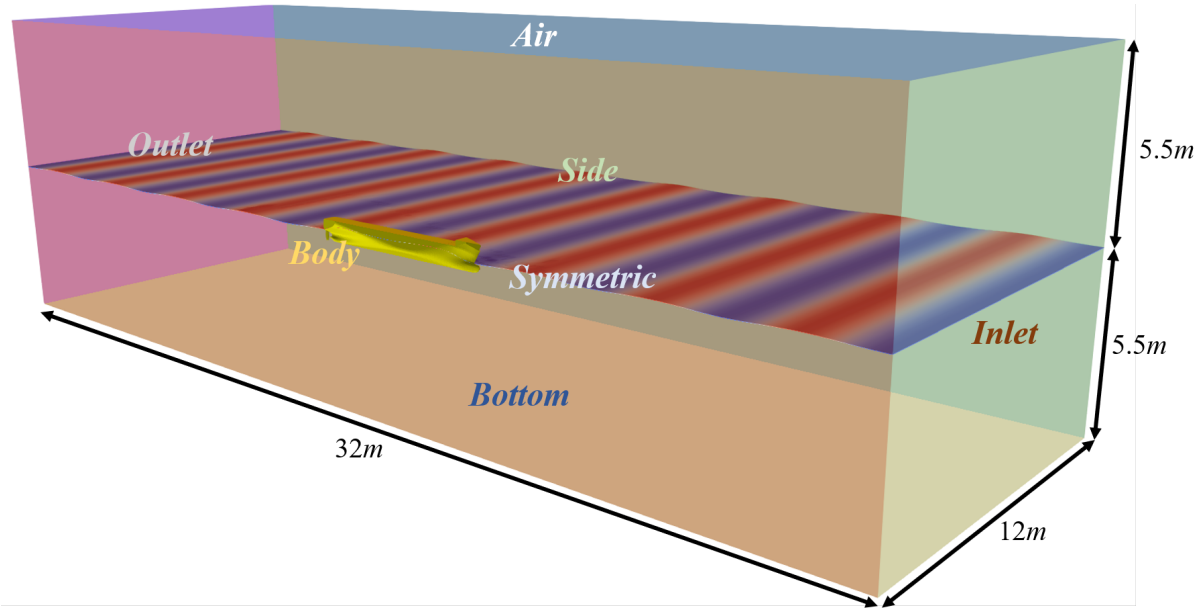


Figure 8.12. Boundary patches and computational domain for the KCS simulations

Mesh convergence study is performed using three mesh refinement levels and only one time-

step size in each case. The computational mesh does not change with respect to the wave conditions. Table 8.16 summarizes the computational conditions for each resolution. The cell sizes in the table are given for cells inside the free-surface refinement zone. The $x-z$ aspect ratio is $\Delta z/\Delta x = 1/8$, and the $x-y$ aspect ratio is $\Delta y/\Delta x = 2$. The smallest cell size can be found at the boundary layer of the body, and the boundary layer size (δ) is listed in Table 8.16. The refinement ratios between the mesh refinement levels are $r = 1.333$ and $r = 1.5$. The time-step size of the *Coarse* refinement is 0.002s, and the time-step size of the *Medium* and *Fine* refinements is 0.001s. The *Medium* and *Fine* refinements used same time-step size due to the high computational cost.

Table 8.15. Boundary conditions for the KCS simulations. The blue rows represent variables used only in *foamStarSWENSE* computations.

	Body	**Inlet/Outlet/Side**	**Air**	**Bottom**
$\mathbf{u}$	$\mathbf{u}_{\text{Body}}$	$\mathbf{u}_{\text{I}}$	Pressure outlet	$\mathbf{u}_{\text{Wall}}$
$\mathbf{u}_{\text{C}}$	$\mathbf{u}_{\text{Body}} - \mathbf{u}_{\text{I}}$	zero vector	Pressure outlet	slip
p_{d}	zeroGradient	zeroGradient	zeroGradient	zeroGradient
p_{C}	fixedFluxPressure	zeroGradient	zeroGradient	zeroGradient
α	zeroGradient	α_{I}	inlet-outlet	zeroGradient
k	kqRWallFunction	zeroGradient	zeroGradient	zeroGradient
ω	omegaWallFunction	zeroGradient	zeroGradient	zeroGradient
ν_t	nutUSpaldingWallFunction	calculated	calculated	calculated

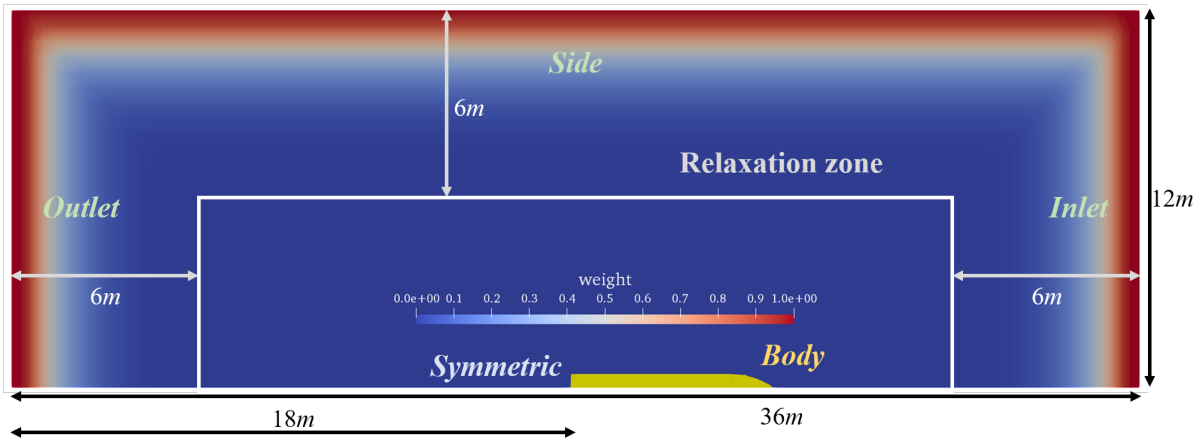


Figure 8.13. Configuration of the relaxation zones and visualization of the exponential weight function field. The color contours represent the weight function: red represents $w = 1$ and the blue represents $w = 0$.

Table 8.16. KCS computational conditions and mesh size information inside the free-surface refinement zone

Mesh	Δx	$L/\Delta x$	$\Delta x/\Delta y$	Δz (m)	$\Delta z/\Delta x$	h(s)	δ (m)	N Cell
Coarse	0.25	24.28	2	0.03125	1/8	0.002	0.004	237,743
Medium	0.1875	32.37	2	0.02344	1/8	0.001	0.003	470,797
Fine	0.125	48.56	2	0.01563	1/8	0.001	0.002	1,181,558

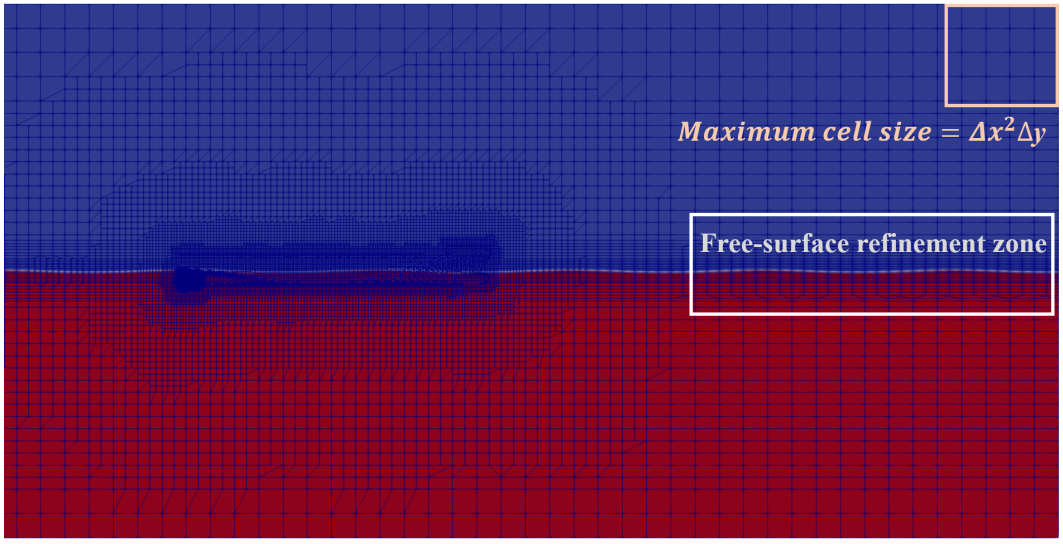


Figure 8.14. An example of computational mesh (Coarse mesh)

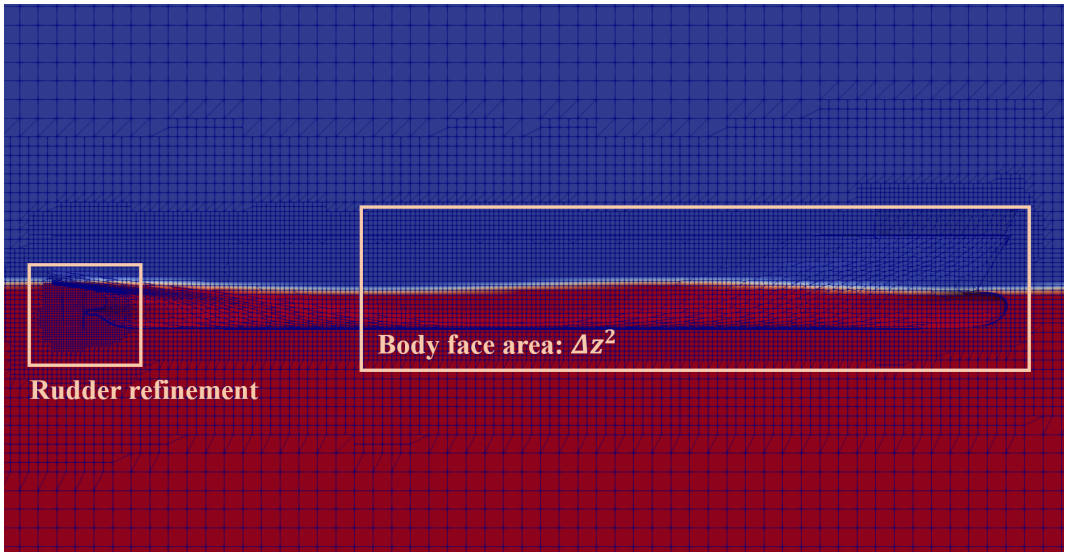


Figure 8.15. An example of computational mesh near the body (Coarse mesh)

The KCS simulation meshes are generated with the STAR-CCM+ mesher. Figures 8.14 and 8.15 show the coarse mesh used. The free-surface refinement zone follows the cell sizes presented

in Table 8.16. The maximum cell size in the domain is $\Delta x^2 \times \Delta x$. The vertical refinement does not change along the free-surface to prevent VOF separation due to the ship forward velocity. The body surface area is determined based on the vertical cell size (Δz) as Δz^2, due to the Star-CCM+ mesher algorithm. The rudder zone is refined once more for proper modelling of the sharp rudder shape. The boundary layer cell size (δ) is reduced as the refinement level increased. The boundary layer only exists on the KCS hull but is ignored on the transom, the deck, and the rudder.

8.2.3 KCS: Results

The following presents the results of KCS simulations together with the previous experimental and numerical results given at the Tokyo 2015 CFD Workshop. Table 8.17 tabulates the computational costs of all simulations. The computational cost is again defined as 'cost = (computational time) × (number of CPUs)'. Not all simulations attempts were successful. Especially, many of the simulations with the *foamStarSWENSE* solver and the *ESDIRK33* and *EDIRK332* schemes with the *Medium* resolution, failed to reach the final physical time of 20s. The table remains empty if the simulation fails to reach the final time of 20s. Also, when the simulation final time was less than 15s, no FFT analysis has been performed. This is marked by 'x' in Table 8.17. Therefore, if a simulation stopped between 15s and 20s, the FFT analysis was performed but the computational cost was not measured. This is marked by 'o' in the table.

As for the computational cost of Wigley III simulations, the SWENSE solver requires higher cost for the same computational condition, and this is mainly due to the slower convergence of variables in the non-linear iterations. However the extra cost is smaller than for the Wigley III hull simulations, around $20\% \sim 30\%$ on average. And again, the simulation with the *foamStarSWENSE* solver turned to be more unstable, especially for the *Medium* resolution, and in terms of cases, for case *C4*.

With the *foamStar* solver, which proved to be more robust, the *ESDIRK33* and *EDIRK332* schemes were nevertheless unstable with the *Medium* resolution, so that those schemes were discarded for the *Fine* resolution simulations.

Mean motions and first harmonic motions of KCS

Figure 8.16 and Figure 8.17 illustrates the mean motions and first harmonic motions with different computational setups. The black round markers denote the experimental data, and the coloured small round markers denote the numerical results from the other institutions which participated in the Tokyo 2015 CFD Workshop [129]. Again for better visualization, the *foamStarSWENSE* results are shifted little left from the experimental ones, and the *foamStar* results are shifted little right. The x-axis is the normalized wavelength, and the y-axis represents either the mean motions or the first harmonic motion RAOs. The mean motions are defined as half the zeroth harmonic amplitudes.

Table 8.17. Comparison of the computational cost (Hr) of KCS simulations.

Computational cost (Hr) ***foamStar (Coarse)***					
Wave condition	***C1***	***C2***	***C3***	***C4***	***C5***
implicit Euler	175.3	182.1	206.1	198.6	187.8
RKCN099	194.1	199.9	376.0	268.2	264.5
ESDIRK33	314.5	319.4	360.1	343.9	345.2
EDIRK332	329.8	340.0	337.5	352.6	336.4
Computational cost (Hr) ***foamStarSWENSE (Coarse)***					
Wave condition	***C1***	***C2***	***C3***	***C4***	***C5***
implicit Euler	244.4	253.3	307.0	304.1	264.8
RKCN099	255.5	264.1	271.0	269.9	288.6
ESDIRK33	396.4	390.0	420.4	412.5	421.0
EDIRK332	394.0	390.5	416.1	434.8	424.1
Computational cost (Hr) ***foamStar (Medium)***					
Wave condition	***C1***	***C2***	***C3***	***C4***	***C5***
implicit Euler	1160.8	1166.3	1194.3	1231.3	1159.4
RKCN099	1312.9	1303.1	1321.1	1322.1	1273.1
ESDIRK33	2602.7	2657.4	x	2720.1	2667.4
EDIRK332	2605.4	2652.3	2690.7	o	2696.6
Computational cost (Hr) ***foamStarSWENSE (Medium)***					
Wave condition	***C1***	***C2***	***C3***	***C4***	***C5***
implicit Euler	1305.8	1339.0	1376.0	1361.9	1384.2
RKCN099	1390.0	1413.3	1471.9	x	o
ESDIRK33	x	x	x	x	x
EDIRK332	x	x	x	x	x
Computational cost (Hr) ***foamStar (Fine)***					
Wave condition	***C1***	***C2***	***C3***	***C4***	***C5***
implicit Euler	3360.8	3292.5	3488.8	3486.9	3252.0
RKCN099	3692.3	3625.6	3785.5	3829.8	3561.2
Computational cost (Hr) ***foamStarSWENSE (Fine)***					
Wave condition	***C1***	***C2***	***C3***	***C4***	***C5***
implicit Euler	3817.4	3892.8	4161.0	4096.1	3995.5
RKCN099	4042.8	4158.5	o	o	4180.6

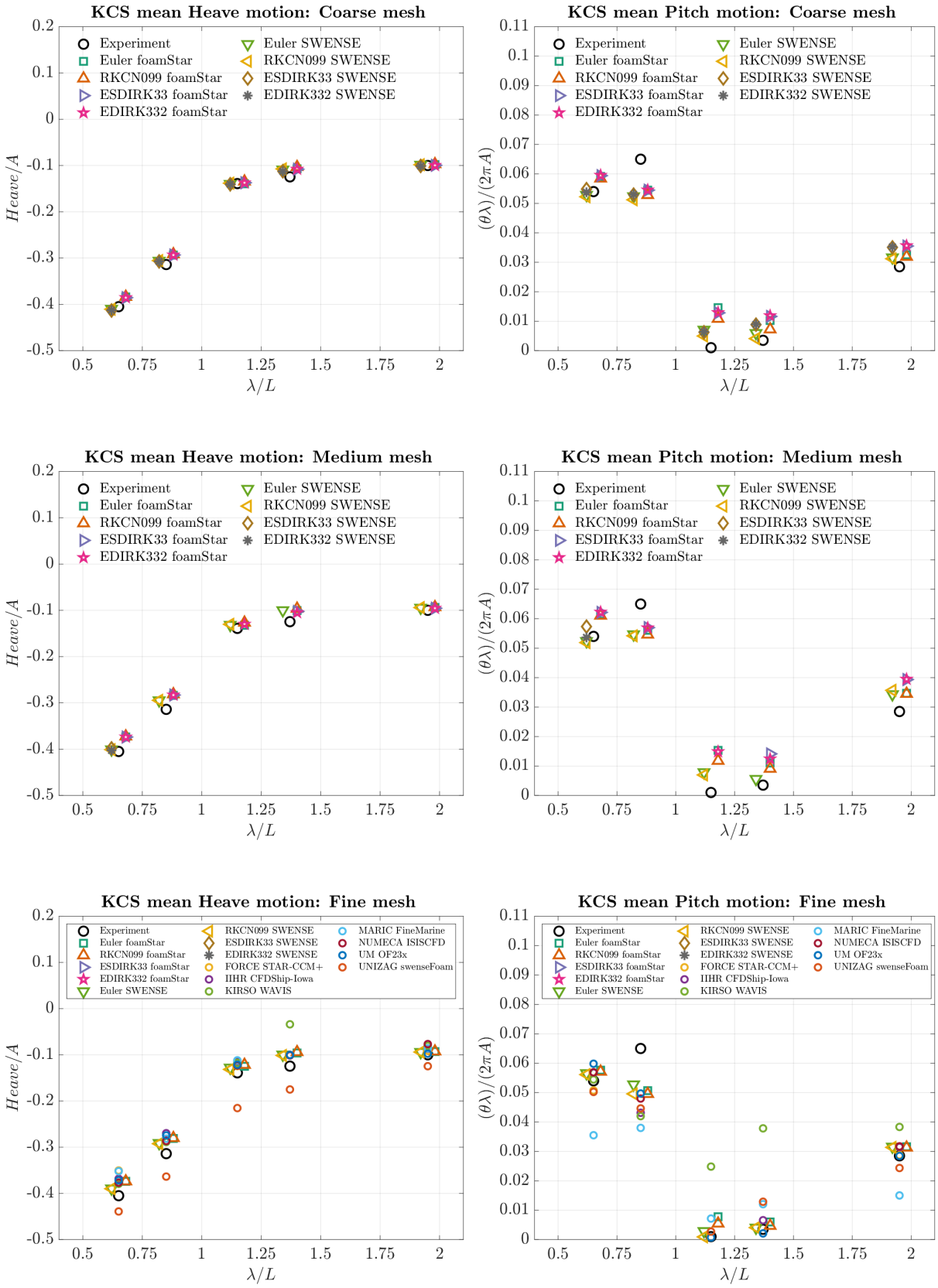


Figure 8.16. Mean heave and pitch motions of KCS in head waves

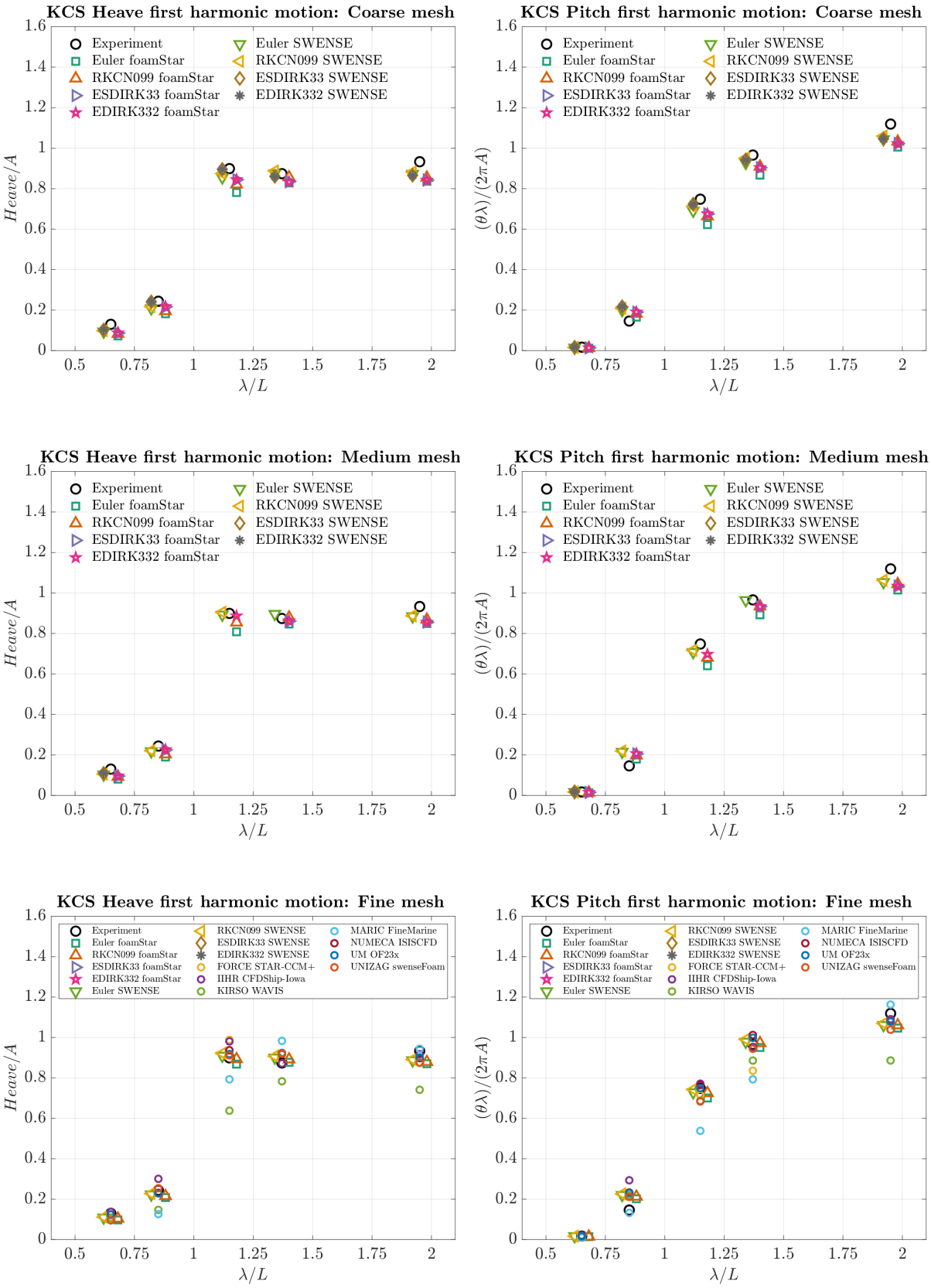


Figure 8.17. First harmonic motions of KCS in head waves

In general, the mean motions obtained with numerical simulations agree well with the ex-

periments except for the mean pitch motion of case *C2* ($\lambda = 0.85$). However, the mean pitch motion obtained for that case compares well with the numerical results of the Tokyo 2015 CFD Workshop. The mean heave motions from the simulations are slightly larger than those of the experiment. With the *Fine* resolution, the *foamStarSWENSE* solver results have smaller difference from the experiment than the *foamStar* results.

Like the mean motions, the first harmonic motions globally converge towards the experimental values when increasing the resolution. The *foamStarSWENSE* solver results in better first harmonic amplitudes than *foamStar* with lower resolution. With the *foamStar* solver, the accuracy of first harmonic motions is less influenced by the time integration scheme (except for the first order Euler scheme).

Finally, it can be underlined that compared to the Tokyo 2015 CFD Workshop results, the ones obtained with our two solvers belong to the closest to the experiments, especially when using the *foamStar* solver.

Total resistance and added resistance of KCS

Figure 8.18 compares the mean total resistance and the first harmonic total resistance. The mean and first harmonic total resistance generally shows good agreement with the experiment. Similar to the motion RAOs result, the higher-order time schemes did not give better results. Besides, the numerical results obtained with the two solvers do not change much with the resolution showing that they are already close to converged from the coarsest resolution, or that convergence is very slow. These 'converged' values are anyway from very close to rather close to the experimental data, depending on the wavelength.

Figure 8.19 compares the added resistance with different time integration schemes. The added resistance is calculated as a subtraction of the mean total resistance from the experiment steady-state resistance. Therefore, the added resistance is very sensitive to a little change of mean resistance. Actually, for cases *C1* and *C2* the mean resistance is close to the experimental value, but there is a larger difference looking at the non-dimensional added resistance. As well, convergence with resolution on this more sensitive quantity is clearly visible throughout these three plots, backing up the fact that mean resistance was already converged with the coarsest resolution in Figure 8.18, but not yet the additional resistance. The added resistance from the *foamStar* and the *foamStarSWENSE* solvers generally show good agreement with the experiment and other numerical simulations from the Tokyo 2015 CFD Workshop. The added resistance computed with the SWENSE method are larger than the results obtained with *foamStar*, and generally closer to the experimental ones for the finest resolution.

Considering that the *Fine* resolution in this study is not as fine as in other KCS studies and that only one time-step size per mesh size was considered, it is difficult to conclude about the accuracy and convergence order of DIRK schemes on this application. However, taking into account that the maximum Courant number is already over ten with a given time-step size, the high-order DIRK schemes were not expected to be efficient in that range, which was found in practice. Finally, it should be highlighted again that despite this not so fine resolution, the results obtained compare very well to the other numerical solutions from the Tokyo 2015 CFD

Workshop since they belong again to the closest results to the experiment, both in terms of total and added resistances.

Summary

In summary, good agreement was found between the two solver results and experimental or other numerical results, especially with the *foamStarSWENSE* solver, both in terms of ship motions and resistance, including added resistance.

In terms of stability the *foamStar* solver on the one hand, and the *Euler* and *RKCN099* time integration scheme, on the other hand, were found to be the more stable. The *foamStarSWENSE* solver met some stability issues, especially with the finest resolutions used, while other DIRK time integration schemes showed unstable in different configurations.

Besides, the higher-order DIRK time integration methods were not effective for these simulations with a free-running ship probably due to the high Courant number occurring in the boundary cells near the moving body. Also, some of the SDIRK methods which were efficient for NWT simulations, were not stable in presence of a body due to unstable pressure found when there were large changes of the time-step size during the sub-steps.

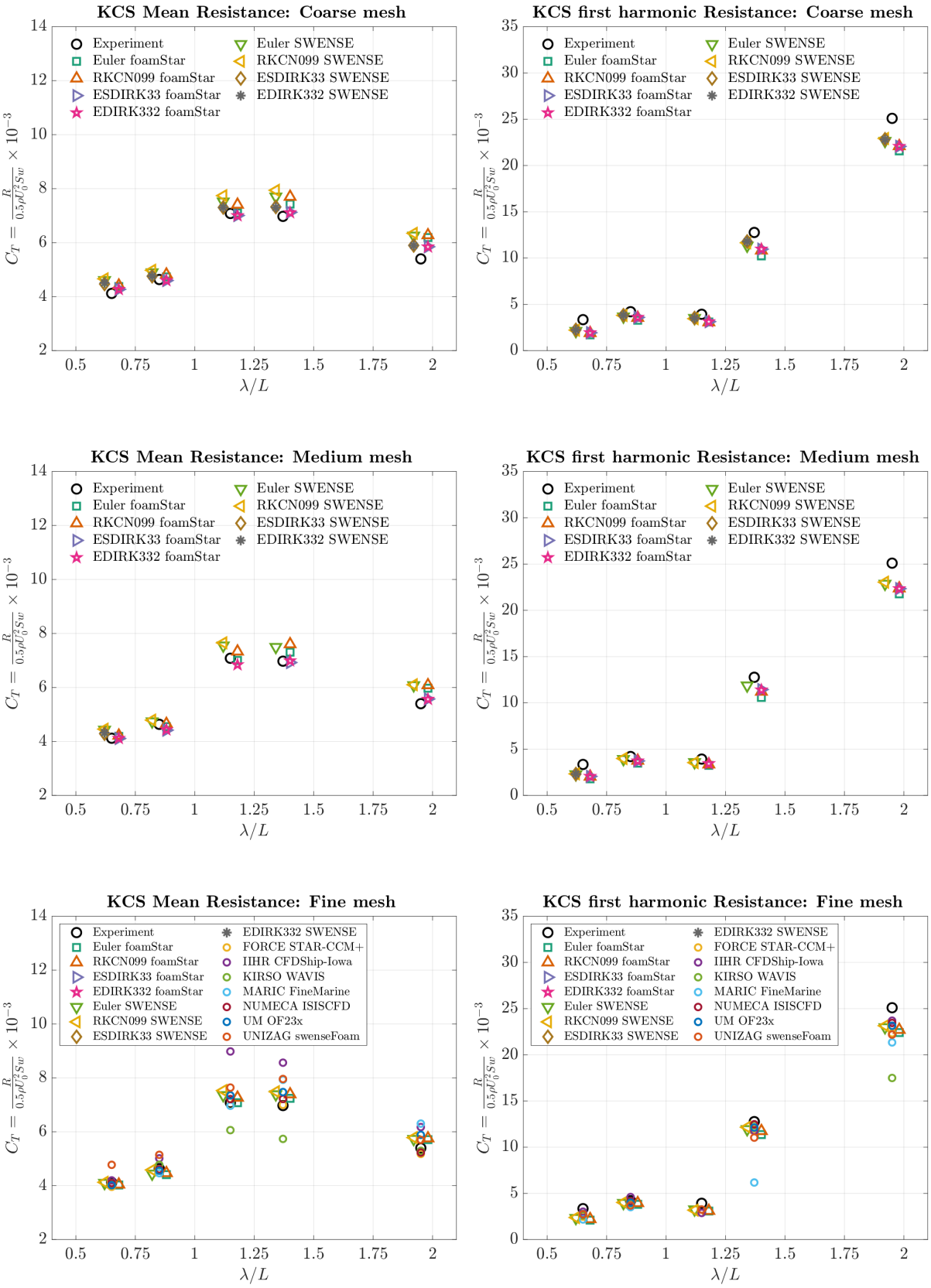
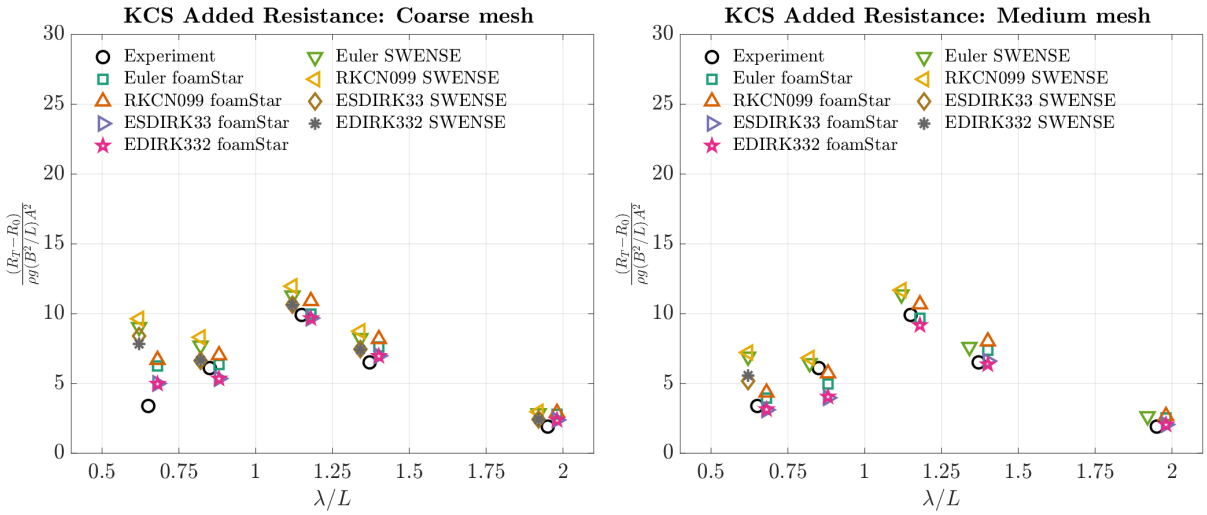


Figure 8.18. Mean and first harmonic resistance of KCS in head waves



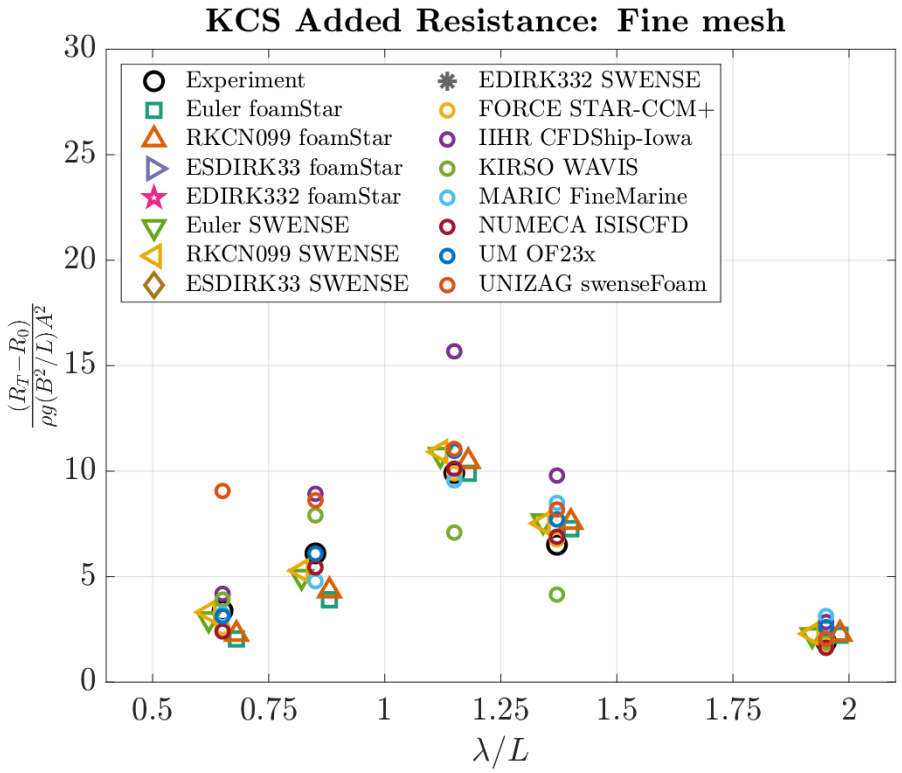


Figure 8.19. Added resistance of KCS in head waves

9 Conclusions

In the present thesis, several numerical models were tested to improve the efficiency and the accuracy of a two-phase flow CFD solver dedicated to wave-structure interaction. The solver is an incompressible and semi-implicit solver discretized with the Finite Volume Method, developed in OpenFOAM from existing solvers (interFoam and Waves2Foam in particular). The applications targeted are typical naval and ocean engineering design problems. After numerical developments and studies, the thesis treats with more details two hydrodynamic problems: the long-crested wave generation and the seakeeping of a ship with forward speed in head waves.

The first part of the thesis investigates separately a series of key aspects of a finite volume semi-implicit two phase solver. First, selections of interface treatment schemes are tested and compared. Those schemes mathematically or numerically control the discontinuous free-surface properties and play a major role in the accuracy of wave propagation and during impact. They also affect the robustness of the solver. Then, some developments have been made with the intent to increase the order of the solver and check the gain in efficiency (the efficiency being a ratio cost/error). Preliminary tests were first performed during the PhD with high-order spatial schemes, but those were not further developed since the results of these preliminary tests did not give a path to easy efficiency gain due to the complicated algorithms and high additional computational burden of these high-order schemes. Consequently the thesis did not present results on this, but a significant part was rather dedicated to increasing the order of the time integration scheme of the two-phase flow solver. This development has been done using Diagonally Implicit Runge Kutta (DIRK) schemes. Several of those were tested to evaluate whether their use could improve the solver accuracy and efficiency. This study revisited the conservative version of the SWENSE solver suggested by Li [1] and evaluated the loss or gain in efficiency and accuracy for the wave interaction applications discussed.

During the PhD study mentioned above, other than OpenFOAM and the foamStar layer developed in collaboration between ECN and Bureau Veritas, the following software was developed, imported or updated.

- **interFlow** & **interFlowGFM** & **interFoamGFM**: Three additional two-phase flow solvers with different interface treatment methods were introduced. The VOF convection equation of the *interFlow* and *interFoamGFM* was imported from the GitHub *isoAdvector* [32]. The ghost fluid method was applied to the *interFoamGFM* and *interFlowGFM*.
- **DIRK methods**: The user-selectable diagonally implicit Runge-Kutta (DIRK) time integration schemes were added to the *foamStar* solver. Butcher matrices are used to express each DIRK method. The Butcher matrix of each DIRK method used in this thesis is presented in Appendix B. Within these DIRK methods, an important option `NEulerStep`, which gives control to start the simulation with the implicit *Euler* time scheme for a user-selectable number of time steps, was included. This option helps to stabilize the pressure shock occurring due to the presence of waves on the body surface at the beginning of the simulation.
- **foamStarSWENSE**: A conservative version of the SWENSE solver (*foamStarSWENSE*)

developed by Li [1] was updated to couple with the developed DIRK time schemes.

9.1 Summary and conclusions and perspectives

9.1.1 Interface treatment schemes

Several interface treatment schemes, which mathematically or numerically control the discontinuous free-surface properties evolution, are studied, and the accuracy of each method is compared. Four combinations of the solver are proposed using two VOF convection methods and two interface boundary conditions. One of the two VOF convection methods is algebraic and uses an artificial interface compression term to minimize the VOF smearing. Another VOF convection method is geometric, it is the `isoAdvector` scheme proposed by Roenby et al. [137]. The two interface boundary conditions are the averaged two-phase flow model and the ghost fluid method (GFM).

The benchmark test cases used to study these four combinations of interface schemes are one periodic wave propagation and two slamming impact cases. The first benchmark case is used to test wave propagation performances for the four solvers in a periodic one wavelength domain, to remove relaxation zones effects. The solver using the VOF convection method with the compression term (*interFoam*) gave the most stable and predictable wave elevation among the four solvers. The solver using `isoAdvector` method and averaged two-phase flow model (*interFlow*) showed the best preservation of wave amplitude with the fine mesh, but it was less stable with the coarser meshes and, more importantly, yielded much larger phase shifts than *interFoam*. The solvers using GFM (*interFoamGFM* and *interFlowGFM*) gave good result with the coarse mesh, but they were less accurate than the averaged two-phase flow solvers with finer meshes.

Two 2D slamming impact benchmark cases were then used to test the four solvers. One is an asymmetric wedge entry, and the other is a water entry-exit problem. Both case results were compared with Smoothed Particle Hydrodynamics (SPH) results. The *interFlow* solver showed the closest free-surface deformation to the SPH one, but it generated high pressure oscillations during the impact. Similar to the wave propagation, the best solver was found to be *interFoam* since it gave similar water jet profiles as those of SPH and obtained the most stable pressure and force signals. Alteration of the free-surface during the free-fall was observed with the two solvers using GFM due to the mesh motions. Moreover, the GFM solvers gave more unstable water jets, especially for the water entry-exit test case, which did not compare well with the SPH solution.

In conclusion, the *interFoam* solver was not always the most accurate solver, but was always accurate and gave stable and predictable results. The *interFlow* solver simulated accurately wave profiles, but its pressure and the wave phase were less stable than *interFoam*. For the test cases considered, the GFM method showed large defects with the current computation setup and would require further investigation. The *interFoam* and the *interFlow* solvers show good jet flow and free-surface profile for two slamming test cases. With those test cases, two subjects are remain as future works. First, the pressure oscillation near the body during the impact obtained from *interFlow* solver needs to be improved. Second, more detailed study on the fluid

detachment on the body is required, which is critical for the water entry exit problem.

9.1.2 High-order time integration schemes

The DIRK method was implemented in the two-phase flow solver and in the SWENSE solver to improve the solvers accuracy and efficiency in view of naval applications. The correct implementation of these higher-order DIRK methods was verified using a single-phase flow test case, the Taylor Green Vortices. The DIRK method worked as expected on this test case, and the convergence order was verified. The method was then tested on two-phase flow benchmark cases, such as regular wave propagation in a periodic domain, the convergence order of the DIRK methods was lower than first-order with a Courant number range of $[0, 1]$. Besides, differences in accuracy were noticeable between the results of the different DIRK methods, especially with high Courant number.

The difference between the 'native' OpenFOAM Crank-Nicolson (*OFCN*) and the ESDIRK22 (*RKCN*) method was also an important finding in this study. The main difference between the two methods lies in the VOF convection algorithm. This difference affects the stability and accuracy of the simulations significantly. For the two-phase wave propagation, the *OFCN* scheme could not complete the simulation without off-centering. This is not the case with the *RKCN* scheme which did not require the off-centering and thus gave less dissipative results. Note that further applications in this thesis used *RKCN* scheme with 0.99 off-centering for better stability of the solver, and it is named *RKCN099*.

The efficiency of the different time schemes was compared using two variables, the computational cost and the accuracy, computed in an absolute way when an analytical solution was available, i.e. especially for the periodic wave propagation test case. Comparing all time integration schemes, *RKCN* and *SDIRK[2](2)* schemes showed the best efficiency for all wave conditions.

Summarizing the different test cases made, the most efficient and stable scheme turned out to be clearly the *RKCN* scheme, which was confirmed with the more complex cases of the next chapters. This is one of the main contribution of the present work.

9.1.3 Applications of the developed and updated solvers

In this thesis, mainly two types of applications were studied. One is the regular and irregular wave qualification assessment in a NWT and the other is the seakeeping analysis of a ship with forward speed in waves. With detailed analysis of these two applications, the validation and efficiency study of the updated solvers (*foamStar* and *foamStarSWENSE*) was thoroughly performed.

The qualifications analysis of the regular waves in NWT was performed using three wave conditions, three computational resolutions, and eight time integration schemes. The computational time for each combination of setup was measured, and the efficiency of the DIRK method for the two solvers *foamStar* and *foamStarSWENSE* was compared. As a result, within eight time integration schemes, *SDIRK[2](2)* and *RKCN099* methods were the most efficient schemes.

The *RKCN099* method was then used for the irregular wave simulations since this scheme was the most efficient and stable time scheme for the regular wave simulation in a NWT. Two long-crested irregular wave conditions were used for irregular wave analysis. For regular wave test cases, the most accurate and efficient solver was found to be the `foamStarSWENSE` solver. However, the `foamStarSWENSE` solver was not able to simulate the irregular wave simulations due to the cumulative error in the computational domain. The `foamStar` was found to be the most stable and predictable solver for regular wave and irregular wave simulations.

The quality of the long crested wave sea-states was assessed with both deterministic and stochastic analysis. This was performed for the irregular waves measured experimentally and for those computed with the numerical solvers. Both experimental and numerical results were generated with an input based on a solution obtained beforehand with a Higher Order Spectral (HOS) method. The deterministic analysis compared the time series of the wave elevation obtained with different numerical setups and experiments to the HOS solution. The differences were quantified for the wave conditions tested. For high-steepness cases which include breaking events the wave elevation predicted by HOS is not exactly the same as in the experiment because HOS uses an approximate breaking model. Conversely, in CFD the interface model fully resolves the two-phase Navier-Stokes evolution of waves, including the breaking events as in the real experiment, but it is rather the accuracy of the schemes (interface and turbulence schemes in particular) and the limited resolution of the simulations which do not permit to reach accurate wave breaking details. Then, stochastic analysis was performed on 30 hours of 2D CFD simulations in order to catch variations in the wave crest distribution. The quality of the irregular waves simulated was assessed using averaged wave spectrum and the wave crest probability of exceedance. This study showed that for a situation where the computational cost is not prohibitive, CFD can be used in reliable manner for design with the proposed setup.

In a second step, applications with the presence of ship were studied. At first, a comparative analysis with the Wigley III hull was performed with three Froude numbers and six wavelengths. Appropriate time-step size, mesh resolution, and time integration schemes were selected from an efficiency study. Then, the numerical results from `foamStar` and `foamStarSWENSE` solvers were compared with the experiment reported by Journée [132]. The motions and added resistance predicted by the `foamStar` solver generally showed good agreement with the experiment. The *foamStarSWENSE* solver showed slightly better motion RAO than *foamStar* for same grid resolution. For the added resistance, the *foamStarSWENSE* solver showed over-estimation of added resistance for high Froude number, while the *foamStar* solver showed better agreement of added resistance with the experiment. The effect of the wave steepness was also studied for a given wavelength. The effect was found to be minimal for the motion RAOs except for Froude number 0.3. In terms of added resistance, it was found to slightly decrease as the wave steepness increased.

The efficiency and accuracy of the DIRK schemes was studied again on this more complex problem using one test case ($Fr = 0.2,\ \lambda_0/L = 1.25,\ H = 0.04\text{m}$) of the Wigley III benchmark campaign. A massive set of computations were performed using the two solvers (`foamStar` and `foamStarSWENSE`), six DIRK schemes, two mesh resolutions, and three temporal resolutions. The

SWENSE solver is generally 20% $\sim$ 30% slower than `foamStar` for the same computational setup and sometime more unstable than `foamStar` for some cases on the simulations with forwarding body. However with low spatial and temporal resolutions, the `foamStarSWENSE` solver gives better result.

Finally, the more realistic KCS hull was used as last application. A similar but shorter analysis to the one conducted with the Wigley III hull was conducted. Besides experiments, numerical results from the 2015 Tokyo CFD workshop were also available for comparison. Three mesh resolutions and a time-step size for each mesh resolution were applied. With these three resolutions, mean motions, first harmonic motions, total resistance and added resistance were measured for five wave conditions. The computational cost of each computational setup was also qualitatively compared.

Generally, good agreement was found between the two solver results and experimental or other numerical results, especially with the *foamStarSWENSE* solver, both in terms of ship motions and resistance, including added resistance. Also with the Coarse resolution, the *foamStarSWENSE* solver predicts better ship motions than the *foamStar* solver. However, the *foamStarSWENSE* solver overestimates the added resistance with the Coarse resolution and converged to the experiment result as the refinement level increased.

Among the DIRK schemes, *Euler* and *RKCN099* were found to be the more stable with two solvers than other schemes. Considering that the most refined mesh in this study has number of cells only around one million, and it is less refined compared to other KCS studies. Also, only one time-step size is applied. Therefore, it is challenging to conclude the efficiency and the accuracy of DIRK schemes. However, taking into account that the maximum Courant number is already over ten with a given small time-step size, the high-order DIRK schemes (*ESDIRK33* and *EDIRK332*) are difficult to be efficient for naval applications with a free-running ship.

In conclusion, for most of the naval applications, the applicable time schemes for two solvers were *Euler* and *RKCN099*. In general, the *RKCN099* scheme was more accurate with the *foamStar* solver, and the *Euler* scheme with the *foamStarSWENSE* was more efficient than others. For naval applications, the higher-order DIRK methods were often less stable than the *RKCN099* method. Moreover, their accuracy was not outstanding, considering that their computational cost is double of the *RKCN099* method. Also, further investigations are required for the SDIRK scheme, which were unstable in presence of a body due to unstable pressure, since the SDIRK schemes were as efficient as *RKCN* scheme for NWT simulations.

A Second order upwind biased interpolation scheme with V-option

The numerical details concerning `linearUpwind` and `linearUpwindV` are outlined in the following. The finite volume discretization of the convection term reads:

$$\iiint_{V_P} \nabla \cdot (\mathbf{uf}) dV \approx \sum_f^N (\mathbf{S}_f \cdot \mathbf{u}_f)(\mathbf{f})_f = \sum_f^N \phi_f (\mathbf{f})_f, \tag{A.1}$$

where $\mathbf{f}$ is a vector field, $\mathbf{u}$ is a transport velocity, $(\mathbf{f})_f$ is an interpolated vector at the face center $\mathbf{x}_f$. The normal face vector $\mathbf{S}_f$ has the magnitude of the face area, and the face flux is defined as $\phi_f = \mathbf{S}_f \cdot \mathbf{u}_f$. The index $(...)f$ represents the face index between cells P and N. The computation of $(\mathbf{f})_f$ using the second-order upwinding scheme is as follows:

$$(\mathbf{f})_f = \mathbf{f}^{\text{upwind}} + \Delta\mathbf{f}^{\text{corr}} \tag{A.2}$$

where $\mathbf{f}^{\text{upwind}}$ is determined from the cell value in the upwinding direction using the sign of the flux:

$$\mathbf{f}^{\text{upwind}} = \begin{cases} \mathbf{f}_P, & \phi_f > 0 \\ \mathbf{f}_N, & \phi_f \leq 0 \end{cases} \tag{A.3}$$

The correction term $\Delta\mathbf{f}^{\text{corr}}$ differs when selecting `linearUpwind` or `linearUpwindV`. The `linearUpwind` scheme proceeds as follows:

$$\Delta\mathbf{f}^{\text{corr}} = \Delta\mathbf{f}_o^{\text{corr}} \quad , \quad \Delta\mathbf{f}_o^{\text{corr}} = \begin{cases} (\mathbf{x}_f - \mathbf{x}_P) \cdot (\nabla\mathbf{f})_P, & \phi_f > 0, \\ (\mathbf{x}_f - \mathbf{x}_N) \cdot (\nabla\mathbf{f})_N, & \phi_f \leq 0. \end{cases} \tag{A.4}$$

The gradients $(\nabla\mathbf{f})_P$ and $(\nabla\mathbf{f})_N$ are the cell averaged gradients computed separately by other means. The `V`-version of this scheme (i.e. `linearUpwindV`) contains an additional limiter:

$$\Delta\mathbf{f}^{\text{corr}} = \lambda_f^{\text{limiter}} \Delta\mathbf{f}_o^{\text{corr}} \quad , \quad \lambda_f^{\text{limiter}} = \min\left(1, \quad \frac{\Delta\mathbf{f}_o^{\text{corr}} \cdot \Delta\hat{\mathbf{f}}^{corr}}{|\Delta\mathbf{f}_o^{\text{corr}}|^2}\right) \tag{A.5}$$

with

$$\Delta\hat{\mathbf{f}}^{corr} = \begin{cases} (1-w)(\mathbf{f}_N - \mathbf{f}_P), & \phi_f > 0, \\ w(\mathbf{f}_P - \mathbf{f}_N), & \phi_f \leq 0 \end{cases} \tag{A.6}$$

where w is the distance weight corresponding to a central difference linear interpolation from cell-centered P and N to the shared face f. This linear interpolation is written as

$$(\mathbf{f})_f = w\mathbf{f}_P + (1-w)\mathbf{f}_N \tag{A.7}$$

The effect of $\lambda_f^{\text{limiter}}$ is to limit $(\mathbf{f})_f$ of Eqn. (A.2) to be at most the same as in Eqn. (A.7).

B Stiffly accurate DIRK methods

This section provides the coefficients and some numerical properties of the second- and third-order stiffly-accurate (E)(S)DIRK methods used in this study. The stability of a numerical method is measured using a test equation $\dot{y} = ky$ ($\text{Re}(k) < 0$) with initial condition $y(0) = 1$. The stability of Runge-Kutta methods can be determined using the stability function in Eqn. (B.1).

$$R(z) = \frac{\det(\mathbf{I} - z\mathbf{A} + z\mathbf{e}\mathbf{b}^T)}{\det(\mathbf{I} - z\mathbf{A})} \quad , \qquad z = kh \tag{B.1}$$

where $\mathbf{e}$ is a unity vector and $\mathbf{A}$ and $\mathbf{b}^T$ are the components of the Butcher matrix (see Eqn. (6.4)). If $|R(z)| \leq 1$ for $\text{Re}(k) < 0$, the stability function is A-acceptable and the numerical method is A-stable. In addition to A-stability, if $R(z)_{z\to-\infty} = 0$, this stability function is L-stable and the method is L-stable. The L-stability provides damping of high-frequency modes which may be non-physical, and perfect damping of modes at infinity. A simple way to achieve L-stability is to use the stiffly-accurate assumption. Nonetheless, the stability properties are introduced on the test equation, not on the Navier-Stokes equations. Therefore, these stability properties do not fully represent the stability of the Navier-Stokes equations with DIRK method.

In the following subsection, the DIRK coefficients and their numerical properties are shortly described. In this study, only stiffly-accurate schemes are considered. The schemes are labeled using the convention given in Eqn. (B.2), where each letter's significance is described in Table B.1. The backgrounds and the derivation of each DIRK method are not treated in this thesis. For more details on each DIRK method, the interested reader is referred to a review paper by Christopher A. Kennedy and Mark H. Carpenter [7].

$$\text{(E)(S)}\mathbf{DIRK[p](s)_SA_}\text{i} \tag{B.2}$$

Table B.1. Identification of DIRK method. Ex: ESDIRK[p](s)_SA_i.

E	Explicit first stage	S	Singly
DI	Diagonally-implicit	RK	Runge-Kutta
p	Global order	s	Number of stages
SA	Stiffly accurate	i	identifier

B.1 Second-order scheme

B.1.1 Implicit Euler

The simplest implicit time integration scheme is the Implicit Euler scheme. For a generalized formulation of governing equation with DIRK method, the Implicit Euler scheme can be given as Eqn. (B.3).

$$\begin{array}{c|c} 1 & 1 \\ \hline & 1 \end{array} \tag{B.3}$$

B.1.2 ESDIRK[2](2)_SA

Excluding the first-order L-stable Implicit Euler method, Eqn. (B.4) is one of the simplest DIRK method. ESDIRK[2](2)_SA scheme is second-order accurate, two stages, one implicit stage ($s_I = 1$), explicit first, stiffly-accurate and A-stable. It is also well known as Crank-Nicolson (or Lobatto IIIA or trapezoidal rule) method, and it is already implemented in OpenFOAM.

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 0.5 & 0.5 \\ \hline & 0.5 & 0.5 \end{array} \tag{B.4}$$

For the stability reason, an off-centering can be applied to the ESDIRK[2](2)_SA method. Applying the Crank-Nicolson coefficient defined as $C_{CN} = 1/(C_{oc} + 1)$ in to Eqn. (B.5), the off-centering coefficient C_{oc} controls the accuracy and stability of the scheme. The off-centering gives more weight to the implicit term which makes scheme more stable but less accurate. If $C_{oc} = 1.0$, the scheme is ESDIRK[2](2)_SA method. If $C_{oc} = 0$, the scheme is implicit Euler.

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 - C_{CN} & C_{CN} \\ \hline & 1 - C_{CN} & C_{CN} \end{array} \tag{B.5}$$

B.1.3 SDIRK[2](2)_SA

The implicit two stages, stiffly-accurate and L-stable SDIRK method is presented here [138]. Two possible solutions exist with $\gamma = (2 \pm \sqrt{2})/2$. Both solutions are L-stable, but $\gamma = (2 - \sqrt{2})/2$ is more stable because γ is below the unity. Therefore, only $\gamma = (2 - \sqrt{2})/2$ is used.

$$\begin{array}{c|cc} \gamma & \gamma & 0 \\ 1 & 1 - \gamma & \gamma \\ \hline & 1 - \gamma & \gamma \end{array} \tag{B.6}$$

B.1.4 ESDIRK[2](3)_SA

ESDIRK[2](3)_SA scheme is a second order, three stage, stiffly-accurate, and explicit first SDIRK methods. L-stability is achieved when $b_2 = (1-2\gamma)/(4\gamma)$ and $\gamma = (2-\sqrt{2})/2$.

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ 2\gamma & \gamma & \gamma & 0 \\ 1 & (1-b_2-\gamma) & b_2 & \gamma \\ \hline & (1-b_2-\gamma) & b_2 & \gamma \end{array} \tag{B.7}$$

B.2 Third-order scheme

B.2.1 SDIRK[3](3)_SA

SDIRK[3](3)_SA is a three implicit stages, third-order, stiffly-accurate, L-stable SDIRK method [138]. The L-stability is achieved only with $\gamma = 0.435866521508458999416019 45$.

$$\begin{array}{c|ccc} \gamma & \gamma & 0 & 0 \\ \frac{(1+\gamma)}{2} & \frac{(1+\gamma)}{2}-\gamma & \gamma & 0 \\ 1 & -\frac{(6\gamma^2-16\gamma+1)}{4} & \frac{(6\gamma^2-20\gamma+5)}{4} & \gamma \\ \hline & -\frac{(6\gamma^2-16\gamma+1)}{4} & \frac{(6\gamma^2-20\gamma+5)}{4} & \gamma \end{array} \tag{B.8}$$

B.2.2 EDIRK[3](3)_SA_1

EDIRK[3](3)_SA_1 scheme is a A-stable, third-order, three stages, two implicit stages, non-singly diagonal, and stiffly-accurate, EDIRK methods [139].

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ \frac{7}{6} & \frac{7}{12} & \frac{7}{12} & 0 \\ 1 & \frac{5}{14} & -\frac{6}{7} & \frac{3}{2} \\ \hline & \frac{5}{14} & -\frac{6}{7} & \frac{3}{2} \end{array} \tag{B.9}$$

B.2.3 ESDIRK[3](3)_SA

The third-order, three stage, stiffly-accurate and explicit first SDIRK method is presented here. The L-stability is achieved when $b_2 = (1-2\gamma)/(4\gamma)$ and $\gamma = (3+\sqrt{3})/6$.

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ 2\gamma & \gamma & \gamma & 0 \\ 1 & (1-b_2-\gamma) & b_2 & \gamma \\ \hline & (1-b_2-\gamma) & b_2 & \gamma \end{array} \tag{B.10}$$

B.2.4 EDIRK[3](3)_SA_2

The A-stable, third-order, three stages, two implicit stages, stiffly-accurate, and explicit first SDIRK method is presented here. In the review article [7], is presented as one of the early DIRK method developed by Alt [140].

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ \frac{3}{2} & \frac{3}{4} & \frac{3}{4} & 0 \\ 1 & \frac{7}{18} & \frac{-4}{18} & \frac{15}{18} \\ \hline & \frac{7}{18} & \frac{-4}{18} & \frac{15}{18} \end{array} \tag{B.11}$$

B.2.5 ESDIRK[3](4)_SA

ESDIRK[3](4)_SA scheme is a third-order, four stages, three implicit stages, stiffly-accurate, and explicit first SDIRK method [7]. The L-stability requires that γ is a root of the polynomial $6\gamma^3 - 18\gamma^2 + 9\gamma - 1 = 0$ and $0 \leq c_3 \leq 1.743$. Various variations of schemes are studied with the choice of γ and c_3 by many authors. Current study used $\gamma = 0.435866521508458999416019 45$ and Eqn. (B.13).

$$\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ 2\gamma & \gamma & \gamma & 0 & 0 \\ c_3 & (c_3 - a_{32} - \gamma) & a_{32} & \gamma & 0 \\ 1 & (1 - b_2 - b_3 - \gamma) & b_2 & b_3 & \gamma \\ \hline & (1 - b_2 - b_3 - \gamma) & b_2 & b_3 & \gamma \end{array} \tag{B.12}$$

$$c_3 = \frac{3 - 20\gamma + 24\gamma^2}{4 - 24\gamma + 24\gamma^2} \tag{B.13}$$

$$a_{32} = \frac{c_3(c_3 - 2\gamma)}{4\gamma}, \qquad b_2 = \frac{-2 + 3c_3 + 6\gamma(1 - c_3)}{12\gamma(c_3 - 2\gamma)}, \qquad b_3 = \frac{1 - 6\gamma + 6\gamma^2}{3c_3(c_3 - 2\gamma)} \tag{B.14}$$

References

[1] Z. Li, Two-phase spectral wave explicit Navier-Stokes equations method for wave-structure interactions, Theses, École centrale de Nantes (2018).
URL https://tel.archives-ouvertes.fr/tel-02096992

[2] Y. Choi, Two-way coupling between potential and viscous flows for a marine application, Ph.D. thesis, École centrale de Nantes (2019).

[3] Kraaiennest, Le Méhauté's diagram (2009).

[4] Ž. Tuković, H. Jasak, A moving mesh finite volume interface tracking method for surface tension dominated interfacial fluid flow, Computers and Fluids 55 (2012) 70–84. doi:10.1016/j.compfluid.2011.11.003.

[5] S. Seng, Slamming and Whipping Analysis of Ships, Ph.D. thesis, Technical University of Denmark (2012).

[6] G. Oger, M. Doring, B. Alessandrini, P. Ferrant, Two-dimensional SPH simulations of wedge water entries, Journal of Computational Physics 213 (2) (2006) 803–822. doi:10.1016/j.jcp.2005.09.004.

[7] C. A. Kennedy, M. H. Carpenter, Diagonally Implicit Runge-Kutta Methods for Ordinary Differential Equations. A Review, Tech. Rep. March, NASA (2016).

[8] J. V. Wehausen, E. V. Laitone, Surface waves. In: Handbuch der Physik, Springer, New York, USA, 1960.

[9] D. E. Nakos, Ship wave patterns and motions by a three dimensional Rankine panel method (1985).
URL http://hdl.handle.net/1721.1/13651

[10] Bo Woo Nam, A Numerical Study on Berthing Problem between Two Floating Bodies in Waves, Ph.D. thesis, Seoul National University (2015).

[11] X. Chen, Offshore hydrodynamics and applications, IES Journal Part A: Civil and Structural Engineering 4 (3) (2011) 124–142. doi:10.1080/19373260.2011.595903.

[12] WAMIT Inc., WAMIT User Manual - Version 7.1 (2015).
URL www.wamit.com

[13] DNV-GL, SESAM USER MANUAL WADAM, DNV-GL Ver. 9.3 (March).

[14] I. Ansys, AQWA User Manual, Ansys Inc. 15317 (October) (2012) 724–746.

[15] A. Babarit, G. Delhommeau, Theoretical and numerical aspects of the open source BEM solver NEMOH, Proceedings of the 11th European Wave and Tidal Energy Conference. 1 (September 2015) (2015) 1–12.

[16] Simcenter, Simcenter STAR-CCM+ User Guide, 2019 Siemens PLM Software.

[17] NUMECA, Theoretical Manual FINEMarine v3.1, NUMECA.

[18] ANSYS FLUENT, ANSYS Fluent Theory Guide, ANSYS Inc., USA.

[19] OpenFOAM, OpenFOAM - The Open Source CFD Toolbox - User Guide, OpenFOAM Foundationdoi:10.1023/A.

[20] S. V. Patankar, D. B. Spalding, A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows, International Journal of Heat and Mass Transfer 15 (10) (1972) 1787–1806. doi:10.1016/0017-9310(72)90054-3.

[21] R. I. Issa, Solution of the implicitly discretised fluid flow equations by operator-splitting, Journal of Computational Physics 62 (1) (1986) 40–65. doi:10.1016/0021-9991(86)90099-9.

[22] P. Ferrant, L. Gentaz, B. Alessandrini, D. Le Touzé, A Potential/RANSE Approach for Regular Water Wave Diffraction about 2-d Structures, Ship Technology Research 50 (4) (2003) 165–171. doi:10.1179/str.2003.50.4.004.

[23] C. Monroy, G. Ducrozet, F. Bonnefoy, A. Babarit, L. Gentaz, P. Ferrant, RANS simulations of CALM buoy in regular and irregular seas using SWENSE method, International Journal of Offshore and Polar Engineering 21 (4) (2011) 264–271.

[24] G. Reliquet, A. Drouet, P. E. Guillerm, E. Jacquin, L. Gentaz, P. Ferrant, Simulation of wave-body interaction using a single-phase level set function in the swense method, in: Proceedings of the International Conference on Offshore Mechanics and Arctic Engineering - OMAE, Vol. 7, 2013, pp. OMAE2013–11097. doi:10.1115/OMAE2013-11097.

[25] V. Vukcevic, Numerical Modelling of Coupled Potential and Viscous Flow for Marine Applications, Ph.D. thesis, University of Zagreb (2016).

[26] C. W. Hirt, B. D. Nichols, Volume of fluid (VOF) method for the dynamics of free boundaries, Journal of Computational Physics 39 (1) (1981) 201–225. doi:10.1016/0021-9991(81)90145-5.

[27] S. Osher, J. A. Sethian, Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations, Journal of Computational Physics 79 (1) (1988) 12–49. doi:10.1016/0021-9991(88)90002-2.

[28] W. J. Boettinger, J. A. Warren, C. Beckermann, A. Karma, Phase-field simulation of solidification, Annual Review of Materials Science 32 (2002) 163–194. doi:10.1146/annurev.matsci.32.101901.155803.

[29] A. Di Mascio, R. Broglia, R. Muscari, On the application of the single-phase level set method to naval hydrodynamic flows, Computers and Fluids 36 (5) (2007) 868–886. doi:10.1016/j.compfluid.2006.08.001.
URL http://www.sciencedirect.com/science/article/pii/S0045793006000910

[30] D. L. Youngs, Time-dependent multi-material flow with large fluid distortion., Methods Fluid Dynynamic 1 (1982) 41–51.

[31] J. Roenby, H. Bredmose, H. Jasak, A computational method for sharp interface advection, Royal Society Open Science 3 (11) (2016) 1–27. arXiv:1601.05392, doi:10.1098/rsos.160405.

[32] J. Roenby, IsoAdvector, https://github.com/isoadvector, 2021.

[33] S. Muzaferija, M. Perić, Computation of free surface flows using interface-tracking and interface- capturing methods, Nonlinear Water Wave Interaction (1999) 59–100.

[34] O. Ubbink, R. I. Issa, A Method for Capturing Sharp Fluid Interfaces on Arbitrary Meshes, Journal of Computational Physics 153 (1) (1999) 26–50. doi:10.1006/jcph.1999.6276.

[35] H. Jasak, H. Weller, Interface tracking capabilities of the inter-gamma differencing scheme, Tech. Rep. 2, Department of Mechanical Engineering, Imperial College of Science, Technology and Medicine (1995).
URL http://powerlab.fsb.hr/ped/kturbo/OpenFOAM/docs/InterTrack.pdf

[36] P. W. Hogg, X.-J. Gu, D. R. Emerson, an Implicit Algorithm for Capturing Sharp Fluid Interfaces in the Volume of Fluid Advection Method, European Conference on Computational Fluid Dynamics ECCOMAS CFD (2006) 1–20.

[37] M. Darwish, F. Moukalled, Convective schemes for capturing interfaces of free-surface flows on unstructured grids, Numerical Heat Transfer, Part B: Fundamentals 49 (2006) 19–42. doi:10.1080/10407790500272137.

[38] J. A. Heyns, A. G. Malan, T. M. Harms, O. F. Oxtoby, Development of a compressive surface capturing formulation for modelling free-surface flow by using the volume-of-fluid approach, International Journal for Numerical Methods in Fluids 71 (6) (2013) 788–804. doi:10.1002/fld.3694.

[39] D. Zhang, C. Jiang, D. Liang, Z. Chen, Y. Yang, Y. Shi, A refined volume-of-fluid algorithm for capturing sharp fluid interfaces on arbitrary meshes, Journal of Computational Physics 274 (2014) 709–736. doi:10.1016/j.jcp.2014.06.043.

[40] B. Xie, S. Ii, F. Xiao, An efficient and accurate algebraic interface capturing method for unstructured grids in 2 and 3 dimensions: The THINC method with quadratic surface representation, International Journal for Numerical Methods in Fluids 76 (12) (2014) 1025–1042. doi:10.1002/fld.3968.

[41] B. Xie, F. Xiao, Toward efficient and accurate interface capturing on arbitrary hybrid unstructured grids: The THINC method with quadratic surface representation and Gaussian quadrature, Journal of Computational Physics 349 (2017) 415–440. doi:10.1016/j.jcp.2017.08.028.

[42] N. G. Jacobsen, D. R. Fuhrman, J. Fredsøe, A wave generation toolbox for the open-source CFD library: OpenFoam®, International Journal for Numerical Methods in Fluids 70 (9) (2012) 1073–1088. doi:10.1002/fld.2726.

[43] Z. Shen, D. Wan, P. M. Carrica, Dynamic overset grids in OpenFOAM with application to KCS self-propulsion and maneuvering, Ocean Engineering 108 (2015) 287–306. doi:10.1016/j.oceaneng.2015.07.035.
URL http://dx.doi.org/10.1016/j.oceaneng.2015.07.035

[44] H. Islam, C. Guedes Soares, Uncertainty analysis in ship resistance prediction using OpenFOAM, Ocean Engineering 191 (May) (2019) 105805. doi:10.1016/j.oceaneng.2019.02.033.
URL https://doi.org/10.1016/j.oceaneng.2019.02.033

[45] J. hua Wang, W. wen Zhao, D. cheng Wan, Development of naoe-FOAM-SJTU solver based on OpenFOAM for marine hydrodynamics, Journal of Hydrodynamics 31 (1). doi:10.1007/s42241-019-0020-6.

[46] Y. J. Kim, Y.-M. Choi, B. Bouscasse, Sopheak Seng, D. Le Touzé, Influence of interface treatment in wave propagation in CFD, in: 16èmes Journées de l'Hydrodynamique, 2018, pp. 1–10.

[47] M. Dumbser, Arbitrary high order PNPM schemes on unstructured meshes for the compressible Navier-Stokes equations, Computers and Fluids 39 (1) (2010) 60–76. doi:10.1016/j.compfluid.2009.07.003.
URL http://dx.doi.org/10.1016/j.compfluid.2009.07.003

[48] S. Seng, C. Monroy, Š. Malenica, On the use of Euler and Crank-Nicolson time-stepping schemes for seakeeping simulations in Openfoam, in: 7th International Conference on Computational Methods in Marine Engineering, MARINE 2017, Vol. 2017-May, 2017, pp. 905–920.

[49] C. W. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, Journal of Computational Physics 77 (2) (1988) 439–471. doi:10.1016/0021-9991(88)90177-5.

[50] H. Bihs, A. Kamath, M. Alagan Chella, A. Aggarwal, Ø. A. Arntsen, A new level set numerical wave tank with improved density interpolation for complex wave hydrodynamics, Computers and Fluids 140 (2016) 191–208. doi:10.1016/j.compfluid.2016.09.012.

[51] Q. Wang, Y.-X. Ren, J. Pan, W. Li, Compact high order finite volume method on unstructured grids III: Variational reconstruction, Journal of Computational Physics 337 (2017) 1–26. doi:https://doi.org/10.1016/j.jcp.2017.02.031.

[52] J. C. Butcher, Diagonally-implicit multi-stage integration methods, Applied Numerical Mathematics 11 (5) (1993) 347–363. doi:10.1016/0168-9274(93)90059-Z.

[53] A. Kværnø, S. P. Nørsett, B. Owren, Runge-Kutta research in Trondheim, Applied Numerical Mathematics 22 (1-3 SPEC. ISS.) (1996) 263–277. doi:10.1016/s0168-9274(96)00037-2.

[54] F. Nazari, A. Mohammadian, M. Charron, High-order low-dissipation low-dispersion diagonally implicit Runge-Kutta schemes, Journal of Computational Physics 286 (2015) 38–48. doi:10.1016/j.jcp.2015.01.020.

[55] Y. P. Marx, Time integration schemes for the unsteady incompressible Navier-Stokes equations, Journal of Computational Physics 112 (1) (1994) 182–209. doi:10.1006/jcph.1994.1091.

[56] V. Vuorinen, J. P. Keskinen, C. Duwig, B. J. Boersma, On the implementation of low-dissipative Runge-Kutta projection methods for time dependent flows using OpenFOAM®, Computers and Fluids 93 (2014) 153–163. doi:10.1016/j.compfluid.2014.01.026.

[57] H. Bijl, M. H. Carpenter, V. N. Vatsa, C. A. Kennedy, Implicit time integration schemes for the unsteady compressible Navier-Stokes equations: Laminar flow, Journal of Computational Physics 179 (1) (2002) 313–329. doi:10.1006/jcph.2002.7059.

[58] V. D'Alessandro, L. Binci, S. Montelpare, R. Ricci, On the development of OpenFOAM solvers based on explicit and implicit high-order Runge–Kutta schemes for incompressible flows with heat transfer, Computer Physics Communications 222 (2018) 14–30. doi:https://doi.org/10.1016/j.cpc.2017.09.009.

[59] T. J. Barth, C. F. D. Branch, M. Field, P. Frederickson, M. Field, T. J. Barth, C. F. D. Branch, M. Field, Higher order solution of the Euler equations on unstructured grids using quadratic reconstruction, NTRS.

[60] M. Delanaye, Y. Liu, Quadratic reconstruction finite volume schemes on 3D arbitrary unstrtictured polyhedral grids, 14th Computational Fluid Dynamics Conference di (c) (1999) 80–90. doi:10.2514/6.1999-3259.

[61] O. Friedrich, Weighted Essentially Non-Oscillatory Schemes for the Interpolation of Mean Values on Unstructured Grids, Journal of Computational Physics 144 (1) (1998) 194–212. doi:10.1006/jcph.1998.5988.

[62] B. Cockburn, S. Y. Lin, C. W. Shu, TVB runge-kutta local projection discontinuous galerkin finite element method for conservation laws III: One-dimensional systems, Journal of Computational Physics 84 (1) (1989) 90–113. doi:10.1016/0021-9991(89)90183-6.

[63] T. Martin, I. Shevchuk, Implementation and validation of semi-implicit WENO schemes using OpenFOAM®, Computation 6 (1) (2018) 6. doi:10.3390/computation6010006.

[64] Z. Tian, M. Perlin, W. Choi, An eddy viscosity model for two-dimensional breaking waves and its validation with laboratory experiments, Physics of Fluids 24 (3). doi:10.1063/1.3687508.

[65] H. Jasak, Error Analysis and Estimation for the finite volume method with applications to fluid flows., Ph.D. thesis, Imperial College of Science, Technology and Medicine (1996).

[66] E. Berberović, N. P. Van Hinsberg, S. Jakirlić, I. V. Roisman, C. Tropea, Drop impact onto a liquid layer of finite thickness: Dynamics of the cavity evolution, Physical Review E - Statistical, Nonlinear, and Soft Matter Physics 79 (3). doi:10.1103/PhysRevE.79.036306.

[67] H. Rusche, Computational Fluid Dynamics of Dispersed Two-Phase Flows at High Phase Fractions, Ph.D. thesis, Imperial College of Science, Technology and Medicine (2002). doi:10.1145/1806799.1806850.
URL http://portal.acm.org/citation.cfm?doid=1806799.1806850

[68] Z. Li, G. Deng, P. Queutey, B. Bouscasse, G. Ducrozet, L. Gentaz, D. Le Touzé, P. Ferrant, Comparison of wave modeling methods in CFD solvers for ocean engineering applications, Ocean Engineering 188 (2019) 106237. doi:10.1016/j.oceaneng.2019.106237.

[69] V. Eswaran, S. B. Pope, Direct numerical simulations of the turbulent mixing of a passive scalar, Physics of Fluids 31 (3) (1988) 506. doi:10.1063/1.866832.

[70] J. W. Deardorff, A numerical study of three-dimensional turbulent channel flow at large Reynolds numbers, Journal of Fluid Mechanics 41 (2) (1970) 453–480. doi:10.1017/S0022112070000691.

[71] B. E. Larsen, D. R. Fuhrman, On the over-production of turbulence beneath surface waves in Reynolds-averaged Navier-Stokes models, Journal of Fluid Mechanics 853 (2018) 419–460. doi:10.1017/jfm.2018.577.

[72] F. R. Menter, M. Kuntz, R. Langtry, Ten Years of Industrial Experience with the SST Turbulence Model (2003).

[73] N. L. R. Center, The Menter Shear Stress Transport Turbulence Model (2015).

[74] G. G. Stokes, On the Theory of Oscillatory Waves, Mathematical and Physical Papers vol.1 8 (2010) 197–229. doi:10.1017/cbo9780511702242.013.

[75] J. D. Fenton, A fifth-order stokes theory for steady waves, Journal of Waterway, Port, Coastal and Ocean Engineering 111 (2) (1985) 216–234. doi:10.1061/(ASCE)0733-950X(1985)111:2(216).

[76] M. M. Rienecker, J. D. Fenton, A fourier approximation method for steady water waves, Journal of Fluid Mechanics 104 (1981) 119–137. doi:10.1017/S0022112081002851.

[77] G. Ducrozet, B. Bouscasse, M. Gouin, P. Ferrant, F. Bonnefoy, CN-Stream: Open-source library for nonlinear regular waves using stream function theory, arXiv:1901.10577 [physics.flu-dyn] (2019) 1–31arXiv:1901.10577.
URL http://arxiv.org/abs/1901.10577

[78] W. J. Pierson, L. Moskowitz, A proposed spectral form for fully developed wind seas based on the similarity theory of S. A. Kitaigorodskii, Journal of Geophysical Research 69 (24) (1964) 5181–5190. doi:10.1029/jz069i024p05181.

[79] K. Hasselmann, T. P. Barnett, E. Bouws, H. Carlson, D. E. Cartwright, K. Eake, J. A. Euring, A. Gicnapp, D. E. Hasselmann, P. Kruseman, A. Meerburg, P. Mullen, D. J. Olbers, K. Richren, W. Sell, H. Walden, Measurements of wind-wave growth and swell decay during the joint North Sea wave project (JONSWAP)., Ergnzungsheft zur Deutschen Hydrographischen Zeitschrift Reihe 8 (12).

[80] Bureau Veritas, Guidance for Long-term Hydro-structure Calculations, Bureau Veritas, 2019.

[81] DNV-GL, Fatigue assessment of ship structures Class guideline DNVGL-CG-0129, DNV-GL, 2015.

[82] G. Ducrozet, F. Bonnefoy, D. Le Touzé, P. Ferrant, 3-D HOS simulations of extreme waves in open seas, Natural Hazards and Earth System Science 7 (1) (2007) 109–122. doi:10.5194/nhess-7-109-2007.

[83] G. Ducrozet, F. Bonnefoy, D. Le Touzé, P. Ferrant, HOS-ocean: Open-source solver for nonlinear waves in open ocean based on High-Order Spectral method, Computer Physics Communications 203 (2016) 245–254. doi:10.1016/j.cpc.2016.02.017.

[84] G. Ducrozet, F. Bonnefoy, D. Le Touzé, P. Ferrant, A modified High-Order Spectral method for wavemaker modeling in a numerical wave tank, European Journal of Mechanics, B/Fluids 34 (2012) 19–34. doi:10.1016/j.euromechflu.2012.01.017.

[85] X. Barthelemy, M. L. Banner, W. L. Peirson, F. Fedele, M. Allis, F. Dias, On a unified breaking onset threshold for gravity waves in deep and intermediate depth water, Journal of Fluid Mechanics 841 (2018) 463–488. arXiv:1508.06002, doi:10.1017/jfm.2018.93.

[86] B. R. Seiffert, G. Ducrozet, Simulation of breaking waves using the high-order spectral method with laboratory experiments: wave-breaking energy dissipation, Ocean Dynamics 68 (1) (2018) 65–89. doi:10.1007/s10236-017-1119-3.

[87] M. Canard, G. Ducrozet, B. Bouscasse, Generation of 3hr long-crested waves of extreme sea states with HOS-NWT solver, in: Proceedings of the International Conference on Offshore Mechanics and Arctic Engineering - OMAE, 2020, pp. OAME2020–18930.

[88] H. G. Weller, G. Tabor, H. Jasak, C. Fureby, A tensorial approach to computational continuum mechanics using object-oriented techniques, Computers in Physics 12 (6) (1998) 620. doi:10.1063/1.168744.

[89] Y. Choi, B. Bouscasse, S. Seng, G. Ducrozet, L. Gentaz, P. Ferrant, Generation of regular and irregular waves in navier-stokes CFD solvers by matching with the nonlinear potential wave solution at the boundaries, in: Proceedings of the International Conference on Offshore Mechanics and Arctic Engineering - OMAE, Vol. 2, 2018, pp. OMAE2018–78077. doi:10.1115/OMAE2018-78077.

[90] S. Seng, A. Benhamou, C. Monroy, S. Malenica, Hydroealstic Simulations in OpenFOAM: An Efficient Numerical Implementation of the Modal Equations, in: 8th International Conference on Hydroelasticity in Marine Technology, 2018, pp. 341–350.

[91] S. Mayer, A. Garapon, L. S. Sørensen, A fractional step method for unsteady free-surface flow with applications to non-linear wave dynamics, International Journal for Numerical Methods in Fluids 28 (2) (1998) 293–315. doi:10.1002/(SICI)1097-0363(19980815)28:2<293::AID-FLD719>3.0.CO;2-1.

[92] M. Engineering, D. R. Fuhrman, Numerical Solutions of Boussinesq Equations for Fully Nonlinear and Extremely Dispersive Water Waves MEK Department of Mechanical Engineering, Ph.D. thesis, Technical University of Denmark, Kgs. Lyngby (2004).

[93] A. P. Engsig-Karup, P. A. Madsen, H. B. Bingham, P. G. Thomsen, Unstructured nodal DG-FEM solution of high-order Boussinesq-type equations - DTU Findit, Ph.D. thesis, Technical University of Denmark, Kgs. Lyngby (2007).
URL http://findit.dtu.dk/en/catalog/2185768794

[94] J. Kim, J. O'Sullivan, A. Read, Ringing analysis of a vertical cylinder by Euler overlay method, Proceedings of the International Conference on Offshore Mechanics and Arctic Engineering - OMAE 4 (2012) 855–866. doi:10.1115/OMAE2012-84091.

[95] R. Perić, M. Abdel-Maksoud, Analytical prediction of reflection coefficients for wave absorbing layers in flow simulations of regular free-surface waves, Ocean Engineering 147 (2018) 132–147. arXiv:1705.06940, doi:10.1016/j.oceaneng.2017.10.009.

[96] Y. M. Choi, Y. J. Kim, B. Bouscasse, S. Seng, L. Gentaz, P. Ferrant, Performance of different techniques of generation and absorption of free-surface waves in Computational Fluid Dynamics, Ocean Engineering 214. doi:10.1016/j.oceaneng.2020.107575.

[97] Y. Choi, M. Gouin, G. Ducrozet, B. Bouscasse, P. Ferrant, Grid2Grid : HOS Wrapper Program for CFD solvers, ArXiv e-printsarXiv:1801.00026.
URL http://arxiv.org/abs/1801.00026

[98] M. Frigo, S. Johnson, The Design and Implementation of {FFTW3}, Proceedings of the IEEE 93 (2) (2005) 216–231.

[99] J. Williams, bspline-fortran (2018). doi:10.5281/zenodo.1405787.

[100] H. Jasak, Ž. Tuković, Automatic mesh motion for the unstructured Finite Volume Method, Transactions of Famena 30 (2) (2006) 1–20.

[101] J. H. Ferziger, M. Perić, Computational Methods for Fluid Dynamics, 3rd Edition, Vol. 3, Springer-Verlag Berlin Heidelberg, Berlin, 2002. doi:10.1007/978-3-642-56026-2.

[102] B. Van Leer, Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection, Journal of Computational Physics 23 (3) (1977) 276–299. doi:10.1016/0021-9991(77)90095-X.

[103] J. P. Boris, D. L. Book, Flux-corrected transport. I. SHASTA, a fluid transport algorithm that works, Journal of Computational Physics 11 (1) (1973) 38–69. doi:10.1016/0021-9991(73)90147-2.

[104] S. T. Zalesak, Fully multidimensional flux-corrected transport algorithms for fluids, Journal of Computational Physics 31 (3) (1979) 335–362. doi:10.1016/0021-9991(79)90051-2.

[105] S. M. Damián, N. M. Nigro, An extended mixture model for the simultaneous treatment of small-scale and large-scale interfaces, Ph.D. thesis, Universidad Nacional del Litoral (2014). doi:10.1002/fld.3906.

[106] H. K. Versteeg, Malalasekera Weeratunge, An introduction to computational fluid dynamics : the finite volume method / H.K. Versteeg and W. Malalasekera, second edi Edition, Pearson Education Ltd., Harlow (GB) New York Boston [etc, 2007.

[107] F. R. Menter, T. Esch, Elements of Industrial Heat Transfer Predictions, 16th Brazilian Congress of Mechanical Engineering (2001) 117–127.

[108] Y. J. Kim, S. Seng, B. Bouscasse, D. Le Touze, Numerical study on the temporal discretization schemes in two-phase wave simulation, in: Proceedings of the International Conference on Offshore Mechanics and Arctic Engineering - OMAE, Vol. 2, 2019, pp. OMAE2019–96278. doi:10.1115/OMAE2019-96278.

[109] J. Roenby, H. Bredmose, H. Jasak, IsoAdvector: Geometric VOF on General Meshes, in: OpenFOAM®, Springer, 2019, pp. 281–296. doi:10.1007/978-3-319-60846-4_21.

[110] V. Vukčević, H. Jasak, I. Gatin, Implementation of the Ghost Fluid Method for free surface flows in polyhedral Finite Volume framework, Computers and Fluids 153 (2017) 1–19. doi:10.1016/j.compfluid.2017.05.003.

[111] L. Eça, M. Hoekstra, A procedure for the estimation of the numerical uncertainty of CFD calculations based on grid refinement studies, Journal of Computational Physics 262 (2014) 104–130. doi:10.1016/j.jcp.2014.01.006.

[112] P. Ghadimi, M. A. Feizi Chekab, A. Dashtimanesh, Numerical simulation of water entry of different arbitrary bow sections, Journal of Naval Architecture and Marine Engineering 11 (2) (2014) 117–129. doi:10.3329/jname.v11i2.18724.

[113] Julien MICHEL, Développements numériques de la méthode SPH couplée aux Eléments Finis appliqués au phénomène de l'hydroplanage, phdthesis, École centrale de Nantes (2020).

[114] L. Xu, A. W. Troesch, R. Peterson, Asymmetric hydrodynamic impact and dynamic response of vessels, Journal of Offshore Mechanics and Arctic Engineering 121 (2) (1999) 83–89. doi:10.1115/1.2830082.

[115] D. J. Piro, K. J. Maki, Hydroelastic analysis of bodies that enter and exit water, Journal of Fluids and Structures 37 (2013) 134–150. doi:10.1016/j.jfluidstructs.2012.09.006. URL http://dx.doi.org/10.1016/j.jfluidstructs.2012.09.006

[116] E. Hairer, G. Wanner, Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems, 2nd Edition, Vol. 14, Springer Berlin Heidelberg, 1996. doi:10.1007/978-3-642-05221-7.

[117] P. D. Boom, D. W. Zingg, Optimization of high-order diagonally-implicit Runge–Kutta methods, Journal of Computational Physics 371 (2018) 168–191. doi:10.1016/j.jcp.2018.05.020.

[118] A. Prothero, A. Robinson, On the Stability and Accuracy of One-Step Methods for Solving Stiff Systems of Ordinary Differential Equations, Mathematics of Computation 28 (125) (1974) 145. doi:10.2307/2005822.

[119] B. E. Larsen, D. R. Fuhrman, J. Roenby, Performance of interFoam on the simulation of progressive waves, Coastal Engineering Journal 61 (3) (2019) 380–400. arXiv:1804.01158, doi:10.1080/21664250.2019.1609713.

[120] S. Fouques, E. Croonenborghs, A. Koop, H.-J. Lim, B. Zhao, J. Kim, M. Canard, G. Ducrozet, B. Bouscasse, W. Wang, H. Bihs, QUALIFICATION CRITERIA FOR THE VERIFICATION OF NUMERICAL WAVES – PART 1: POTENTIAL-BASED NUMERICAL WAVE TANK (PNWT), Proceedings of the ASME 2021 40th International Conference on Ocean, Offshore and Arctic Engineering (2021) OMAE2021–63884.

[121] M. Serio, M. Onorato, A. Osborne, P. Janssen, On the computation of the Benjamin-Feir Index, Nuovo Cimento della Societa Italiana di Fisica C 28 (2005) 893–903. doi:10.1393/ncc/i2005-10134-1.

[122] G. Z. Forristall, Wave crest distributions: Observations and second-order theory, Journal of Physical Oceanography 30 (8) (2000) 1931–1943. doi:10.1175/1520-0485(2000)030<1931:WCDOAS>2.0.CO;2.

[123] Z. Huang, Y. Zhang, Semi-empirical single realization and ensemble crest distributions of long-crest nonlinear waves, Proceedings of the International Conference on Offshore Mechanics and Arctic Engineering - OMAE 1 (June 2018). doi:10.1115/OMAE2018-78192.

[124] H. Orihara, H. Miyata, Evaluation of added resistance in regular incident waves by computational fluid dynamics motion simulation using an overlapping grid system, Journal of Marine Science and Technology 8 (2) (2003) 47–60. doi:10.1007/s00773-003-0163-5.

[125] P. M. Carrica, R. V. Wilson, R. W. Noack, F. Stern, Ship motions using single-phase level set with dynamic overset grids, Computers and Fluids 36 (9) (2007) 1415–1433. doi:10.1016/j.compfluid.2007.01.007.

[126] C. D. Simonsen, J. F. Otzen, S. Joncquez, F. Stern, EFD and CFD for KCS heaving and pitching in regular head waves, Journal of Marine Science and Technology (Japan) 18 (4) (2013) 435–459. doi:10.1007/s00773-013-0219-0.

[127] T. Tezdogan, Y. K. Demirel, P. Kellett, M. Khorasanchi, A. Incecik, O. Turan, Full-scale unsteady RANS CFD simulations of ship behaviour and performance in head seas due to slow

steaming, Ocean Engineering 97 (2015) 186–206. doi:10.1016/j.oceaneng.2015.01.011.
URL http://dx.doi.org/10.1016/j.oceaneng.2015.01.011

[128] R. Broglia, G. Dubbioso, D. Durante, A. Di Mascio, Turning ability analysis of a fully appended twin screw vessel by CFD. Part I: Single rudder configuration, Ocean engineering 105 (2015) 275–286.

[129] T. Hino, F. Stern, L. Larsson, M. Visonneau, N. Hirakata, J. Kim, Numerical Ship Hydrodynamics: An Assessment of the Tokyo 2015 Workshop, 1st Edition, Vol. 94, Springer International Publishing, 2020. doi:10.1007/978-3-030-47572-7.

[130] G. Reliquet, A. Drouet, P. Guillerm, L. Gentaz, P. Ferrant, L. De, Simulation of Wave-Ship Interaction in Regular and Irregular Seas under Viscous Flow Theory using the SWENSE Method, Proc. 30th Symp. Naval Hydrod.(Tasmania, 11/2014) (2014) 2–7.

[131] Y. Zhang, G. Reliquet, B. Bouscasse, L. Gentaz, D. Le Touzé, Numerical Investigation on the Added Resistance and Seakeeping Performance of KVLCC2 with the SWENSE Method, Journal of Ship Research (2020) 1–18doi:10.5957/JOSR.03200023.

[132] J. Journée, Experiments and Calculations on 4 Wigley Hull Forms in Head Waves, Tech. Rep. May 1992, Delft University of Technology (1992).
URL http://www.shipmotions.nl/DUT/PapersReports/0909-DUT-92.pdf

[133] S. H. Van, W. J. Kim, G. T. Yim, D. H. Kim, C. J. Lee, Experimental Investigation of the Flow Characteristics around Practical Hull Forms, 3rd Osaka Colloquium on Advanced CFD Applications to Ship Flow and Hull Form Design (1998) 215–227.

[134] W. J. Kim, S. H. Van, D. H. Kim, Measurement of flows around modern commercial ship models, Experiments in Fluids 31 (5) (2001) 567–578. doi:10.1007/s003480100332.

[135] L. Larsson, F. Stern, M. Visonneau, CFD in ship hydrodynamics - Results of the Gothenburg 2010 workshop, Vol. 29, Springer, 2013. doi:10.1007/978-94-007-6143-8_14.

[136] J. F. Otzen, Uncertainty Assessment for KCS Added Resistance in Waves, Tech. rep., FORCE Technology, Lyngby, Denmark (2015).

[137] J. Roenby, H. Bredmose, H. Jasak, IsoAdvector: Geometric VOF on general meshes, Selected papers of the 11th Workshop, 2017.

[138] R. Alexander, Diagonally Implicit Runge–Kutta Methods for Stiff O.D.E.'s, SIAM Journal on Numerical Analysis 14 (6) (1977) 1006–1021. doi:10.1137/0714068.

[139] F. Cameron, Class of low order DIRK methods for a class of DAEs, Applied Numerical Mathematics 31 (1) (1999) 1–16. doi:10.1016/S0168-9274(98)00123-8.

[140] R. Alt, A-stable one-step methods with step-size control for stiff systems of ordinary differential equations, Journal of Computational and Applied Mathematics 4 (1) (1978) 29–35. doi:10.1016/0771-050X(78)90016-5.
URL http://www.sciencedirect.com/science/article/pii/0771050X78900165

Titre: Amélioration de la rapidité et de la precision puis validation d'un solveur hydrodynamique naval CFD pour des interactions houle navire complexes

Mots clés: Schéma à haut ordre; OpenFOAM; Intéraction houle-structure; Génération de vagues; SWENSE; Résistance ajoutée

Résumé : La thèse a pour sujet l'efficacité et la précision des schémas numériques utilisés dans un solveur CFD dédié aux applications navales, en particulier pour les problèmes d'interaction vague-structure. Des études spécifiques sont réalisées sur des sous-parties du solveur numérique, le modèle d'interface air/eau, l'intégration en temps, et l'utilisation du modèle SWENSE. Pour chaque aspect, l'influence de la discretisation spatiale et temporelle est également étudiée.

Une première étude est effectuée sur les schémas de traitement d'interface. Dans le cadre du schéma de capture d'interface Volume of Fluid, deux méthodes de convection d'interface et deux conditions d'interface sont testées pour la propagation d'ondes 2D et des cas typiques d'impact.

Les schémas d'intégration temporelle DIRK (Diagonally Implicit Runge-Kutta) sont ensuite implémentés dans le solveur biphase moyenné incompressible (foamStar) et sur le solveur modifié sur les équations de Navier-Stoke Explicit Wave Spectral (FoamStarSWENSE).

La validité et l'ordre de convergence des méthodes DIRK d'ordre supérieur sont confirmés à l'aide de cas tests ou la solution analytique est connue comme un flux de Taylor Green Vortex 2D puis sur la propagation d'onde régulière dans le domaine périodique.

Deux types d'applications sont effectuées avec les solveurs développés. D'abord des vagues régulières et irrégulières dans le bassin numérique sont qualifiée à l'aidre d'une procédure déterministe et stochastique. L'efficacité de chaque schéma DIRK est estimée à différentes résolutions.

La deuxième application détaillée est la tenue en mer de la coque Wigley III et du navire KCS avec vitesse d'avance. L'efficacité et la précision du schéma DIRK et du modèle SWENSE sont étudiés.

Title: Numerical improvement and validation of a naval hydrodynamics CFD solver in view of performing fast and accurate simulation of complex ship-wave interaction

Keywords: High-order scheme; Diagonally Implicit Runge-Kutta (DIRK) scheme; OpenFOAM; Wave-structure interaction (WSI); Wave generation; SWENSE; Added resistance analysis

Abstract: The present thesis aims to study the efficiency and the accuracy of numerical schemes for naval applications, especially for unsteady wave-structure interaction problems. Specific studies are done on subparts of the solver: the interface model, the time integration schemes and the use of SWENSE model.

The study on the interface treatment schemes is performed with the Volume of Fluid interface capturing scheme. Two interface convection methods and two interface conditions are tested for 2D wave propagation and impact cases.

The Diagonally Implicit Runge-Kutta (DIRK) time integration schemes are applied to two-phase flow solvers. The DIRK method is applied to an incompressible averaged two-phase flow solver (*foamStar*) and a two-phase flow solver based on the Spectral Wave Explicit Navier-Stoke Equations (*foamStarSWENSE*).

The validity and the order of convergence of the higher-order DIRK methods are confirmed using a 2D Taylor Green Vortex flow. The efficiency of the DIRK method is then studied on the two-phase regular wave propagation in the periodic domain.

With two solvers and various DIRK schemes, two types of applications are performed. First, the qualification analysis on the regular and the irregular wave in the numerical wave tank is performed. The efficiency of each DIRK scheme is compared with various resolutions, and the parameters for qualified wave propagation are proposed.

Second, the seakeeping analysis of the Wigley III hull and the KCS with a forward speed is then performed, and the efficiency and the accuracy of the DIRK scheme with two solvers are compared.