N° National de Thèse : XXX

Aix*Marseille université
Initiative d'excellence

irphé
Institut de Recherche sur les Phénomènes Hors Equilibre

Ph.D. Thesis

Submitted for the degree of
Doctor of Fluid Mechanics,
Aix Marseille University

Fields : **Fluid Mechanics**

Institut de Recherche sur les Phénomènes Hors Equilibre (IRPHE)

École Doctorale 353

Numerical simulation of wave-body interaction: development of a fully nonlinear potential flow solver and assessment of two local coupling strategies with a CFD solver

---

Presented and defended in public on December 15, 2020
by **Fabien Robaux**

Ph.D. supervisor : M. Michel Benoit, Ecole Centrale Marseille & Irphé, Marseille

Doctoral committee:

Pr.	Marilena Greco	NTNU, Norway	Rapportrice/Referee
Pr.	Pierre Ferrant	Ecole Centrale Nantes & LHEEA, Nantes	Rapporteur/Referee
Dr.	Yanlin Shao	DTU, Denmark	Examinateur/Examiner
Pr.	Sylvain Guillou	Univ. Caen Normandie & LUSAC, Cherbourg	Examinateur/Examiner
Dr.	Eric Serre	M2P2 & CNRS, Marseille	Examinateur/Examiner
Dr.	Bernard Molin	Ecole Centrale Marseille & Irphé, Marseille	Examinateur/Examiner
M.	Christophe Peyrard	EDF R&D, Chatou	Invité/invited

Institut de Recherche sur les
Phénomènes Hors Équilibre (IRPHE)
49 Rue Frédéric Joliot Curie
13013 Marseille

École doctorale 353
5 rue Enrico Fermi
13013 Marseille

Summary: This thesis is devoted to the mathematical modeling and numerical simulation of interactions between nonlinear ocean waves and fixed marine structures of arbitrary shape, which can be either submerged or surface-piercing. Particular attention is paid to the prediction of nonlinear loads on the structure, with emphasis on the analysis of viscous and turbulent effects.

First, a Numerical Wave Tank (NWT) is developed within the fully nonlinear potential flow theory (FNPT) in two spatial dimensions. It uses a combination of the Harmonic Polynomial Cell (HPC) method for solving the Laplace Boundary Value Problem (BVP) on the wave potential and the Immersed Boundary Method (IBM) for capturing the free surface motion. To compute the flow around the body and associated pressure field, an original multi overlapping grid method is implemented. Each grid having its own free surface, a two-way communication is ensured between the problem in the body vicinity and the larger scale wave propagation problem. Nonlinear loads on the structure are computed from an accurate pressure field obtained as a solution of a BVP formulated on the time derivative of the potential, at the cost of a second matrix inversion at each time step. This potential NWT was developed from scratch during this thesis. The mathematical formalism and the numerical methods are first presented. Afterwards, the stability and convergence properties of the NWT on a highly nonlinear standing wave case are assessed. Then, the NWT is tested against both numerical and experimental data on bodies subjected to various regular wave conditions. The wave field and associated loads are shown to be predicted accurately, for instance up to the third harmonic component for the horizontal and vertical forces. However, limitations are encountered in some cases, mostly due to the intrinsic assumptions of the potential flow theory.

For this reason, in a second part, a Reynolds Averaged Navier-Stokes (RANS) solver is selected and extensively tested with a focus on one case that involves important effects laying outside of the validity range of the potential theory (occurrence of vortex shedding, detachment of the flow, re-circulations, *etc.*). This solver is based on the finite volume method available in the OpenFOAM® toolkit, using a Volume of Fluid (VoF) technique to capture the free surface. The model is extensively applied and validated against experiments in a wave flume with a submerged horizontal cylinder of rectangular cross-section. However, the modeling of turbulent variables is shown to be challenging: it is difficult to model simultaneously and accurately the dynamics of the fields of turbulent variables both on a larger scale (wave propagation from the wavemaker to the body) and on a local scale close to the body.

To overcome this issue and optimize the computational resources, two one-way coupling strategies with the above-mentioned potential model are set up within the OpenFOAM® toolkit and tested on the same case. A domain coupling strategy is first considered. A mesh is defined in the body vicinity on which the RANS equations are solved. Boundary conditions and interpolation operators from the potential results are developed in order to enforce the HPC values at the outer boundary in a stable and accurate manner. Extensive comparisons with the OpenFOAM® simulations show very good agreement for a range of Keulegan-Carpenter numbers.

The second coupling strategy considers a decomposition of variables on a local grid in the body vicinity. Indeed, the HPC simulation provides fields that are solution of the Euler equations: thus, solving the complete RANS equations in the body vicinity can be considered as largely redundant. It is possible to define complementary velocity

and pressure components that need to be added to the potential velocity and pressure in order to obtain a solution of the complete RANS equations. Those complementary variables are solutions of a modified version of the RANS equations, that are derived and implemented, again within the OpenFOAM® software. This newly developed solver shows good performance when compared to either the complete OpenFOAM® simulations or the domain coupling solver.

Finally, the four simulation methods (fully potential, fully viscous, potential-viscous coupling with domain decomposition, and potential-viscous coupling with variable decomposition) are compared on the same set of experiments, and differences in body forces and hydrodynamic coefficients are discussed.

Contents

Contents v

1 Introduction 1
- 1.1 General context of the *Ph.D* research . . . 2
- 1.2 Review of existing models and literature . . . 5
  - 1.2.1 Potential models . . . 5
  - 1.2.2 Fields solvers and the Harmonic Polynomial Cell method (HPC) . . . 6
  - 1.2.3 Introduction of viscous and turbulent effects . . . 7
  - 1.2.4 Free surface treatment . . . 8
- 1.3 Potential-viscous coupling methods . . . 10
- 1.4 Introduction of some dimensionless parameters . . . 11
- 1.5 Objectives and outlines of this thesis . . . 13
- 1.6 Publications and release of the programs . . . 14

2 Solving of the Laplace problem with HPC method 15
- 2.1 Introduction . . . 16
- 2.2 Fully nonlinear potential flow modeling approach . . . 17
- 2.3 The Harmonic Polynomial Cell method (HPC) with immersed free surface 19
  - 2.3.1 General principle of the HPC method . . . 19
  - 2.3.2 Treatment of nodes inside the fluid domain . . . 21
  - 2.3.3 Nodes where a Dirichlet or Neumann boundary condition is imposed . . . 21
  - 2.3.4 Treatment of the free surface . . . 22
  - 2.3.5 Linear solver and advance in time . . . 24
  - 2.3.6 Computation of the time derivative of the potential . . . 25
- 2.4 Introduction of a fixed body in the NWT . . . 27
  - 2.4.1 A double mesh strategy to adapt the resolution in the vicinity of a body . . . 27
  - 2.4.2 The two-way communication inside the fluid domain . . . 28
  - 2.4.3 Free surface piercing body . . . 29
- 2.5 Treatment of a sharp corners . . . 31

3 Results and discussion about the HPC model 35
- 3.1 Introduction . . . 36
- 3.2 Validation and convergence study on a nonlinear standing wave . . . 37
  - 3.2.1 Presentation of the test-case . . . 37
  - 3.2.2 Evolution of $L_2(\eta)$ error with space and time discretizations . . . 38

3.2.3 Convergence with time discretization . . . 39
3.2.4 Convergence with spatial discretization . . . 41
3.2.5 Summary of numerical convergence study . . . 42
3.3 Validation and limits of wave-body interaction against flume experiments 43
3.3.1 Fixed horizontal submerged cylinder . . . 43
3.3.2 Free surface piercing body, experiments and numerical comparison . 45
3.3.3 Analysis of the limits of the potential approach: rectangular horizontal cylinder . . . 51
3.4 Fitted mesh method: results and limitations . . . 59
3.4.1 Standing wave case . . . 59
3.4.2 Circular horizontal cylinder case . . . 61
3.4.3 Conclusion on the fitted mesh method . . . 64
3.5 Summary of conclusions . . . 65

4 Investigations on the volume of fluid method with OpenFOAM® 67
4.1 Introduction . . . 68
4.1.1 Limitations of the potential approach . . . 68
4.1.2 OpenFOAM®, a widely used toolbox . . . 69
4.1.3 Turbulence modeling . . . 69
4.1.4 Outline of this chapter . . . 70
4.2 Models and equations . . . 71
4.2.1 Navier-Stokes equations . . . 71
4.2.2 Reynolds Averaged Navier-Stokes (RANS) method . . . 71
4.2.3 Volume of Fluid (VoF) method . . . 73
4.2.4 Turbulence modeling . . . 74
4.3 Numerical methods . . . 79
4.3.1 Finite Volume Method (FVM) . . . 79
4.3.2 Coupled velocity-pressure system and its resolution . . . 82
4.4 Description of the test case and base parameters of the simulations . . . 84
4.4.1 Generated and absorbed waves . . . 84
4.4.2 Mesh generation and description . . . 85
4.4.3 Numerical parameters . . . 87
4.5 Comparison of the different turbulence models . . . 89
4.5.1 Laminar computation . . . 89
4.5.2 Native OpenFOAM® $k-\omega$ SST model . . . 90
4.5.3 Modified $k-\omega$ SST models . . . 90
4.5.4 Suggested $k-\omega$ SST models . . . 94
4.5.5 Instability at long time of the turbulence models. . . . 96
4.5.6 Conclusion and selection of a turbulence model . . . 97
4.6 Sensitivity and independence studies . . . 99
4.6.1 Local periodicity . . . 99
4.6.2 A significant sensitivity to mesh size and time step . . . 99
4.6.3 Boundary layer and wall functions. . . . 106
4.6.4 Conclusions on the investigations studies. . . . 108
4.7 Application and comparison with experimental results . . . 110
4.7.1 Errors during the wave propagation . . . 110

4.7.2 Results and comments . . . 112
4.8 Conclusions on OpenFOAM® as a NWT. . . . 114

5 Development of a one-way modular domain coupling method. 117
5.1 Introduction . . . 118
5.1.1 Method presentation . . . 118
5.1.2 Literature review . . . 119
5.1.3 Presentation and objectives of the current approach . . . 119
5.1.4 Organisation of this chapter . . . 120
5.2 Method, implementation and usage . . . 121
5.2.1 Workflow, usage and setup of a domain coupled case . . . 121
5.2.2 The main algorithm . . . 122
5.2.3 The CFD problem . . . 123
5.2.4 Spatial interpolation . . . 124
5.2.5 Time interpolation . . . 125
5.2.6 Boundary and initial conditions . . . 126
5.3 Validation and comparisons of the different approaches . . . 128
5.3.1 Base parameters of the simulation . . . 128
5.3.2 Results . . . 130
5.4 Sensitivity studies and parameters exploration . . . 139
5.4.1 Hotstart performance . . . 139
5.4.2 Sensitivity to CFD mesh . . . 140
5.4.3 Boundary conditions . . . 142
5.4.4 Numerical parameters, algorithm & schemes . . . 144
5.4.5 Turbulence modeling . . . 146
5.5 Comparison of the hydrodynamic coefficients with results from the literature 149
5.6 Conclusion . . . 152

6 Restriction of the resolution to the complementary variables: a velocity decomposition approach 155
6.1 Introduction . . . 156
6.1.1 The velocity decomposition method . . . 156
6.1.2 Previous works . . . 156
6.2 Theoretical formulation . . . 159
6.2.1 Complementary Navier-Stokes equations - Momentum and mass conservation . . . 159
6.2.2 Complementary RANS momentum equation . . . 161
6.2.3 Boundary conditions . . . 162
6.3 Numerical implementation . . . 164
6.3.1 `PIMPLE` loop . . . 164
6.3.2 `PISO` loop . . . 165
6.3.3 Momentum equation . . . 166
6.3.4 Turbulence object and equations . . . 167
6.3.5 Boundary conditions . . . 168
6.3.6 Various other contributions and implementations . . . 169
6.4 Intrinsic differences with OpenFOAM . . . 170
6.4.1 Upwind schemes . . . 170

6.4.2 Residual threshold . . . 171
6.5 Stability, convergence and various investigations . . . 173
6.5.1 Test case . . . 173
6.5.2 Numerical scheme and parameters . . . 173
6.5.3 Tolerances and residual controls . . . 175
6.5.4 Boundary conditions . . . 177
6.5.5 Time step influence investigation . . . 179
6.6 Validation of the velocity coupling method . . . 180
6.6.1 Local comparisons with other models . . . 180
6.6.2 Obtained loads with the velocity decomposition method . . . 183
6.6.3 Comparisons with experiments . . . 184
6.7 Conclusions on the velocity decomposition scheme . . . 186

7 General conclusions of the presented research 189
7.1 Summary of the main results . . . 190
7.1.1 Fully potential approach (HPC) . . . 190
7.1.2 Fully viscous CFD approach . . . 190
7.1.3 Potential-viscous coupling using a domain decomposition . . . 191
7.1.4 Potential-viscous coupling using a velocity decomposition . . . 191
7.2 Analysis of CPU costs . . . 193
7.3 Future works and further developments . . . 195
7.3.1 HPC numerical wave tank . . . 195
7.3.2 Domain and velocity decompositions . . . 195

References 199

List of Figures 212

List of Tables 218

A Convergence of a perturbed cosine in both period and amplitude at long time 219
A.1 Taylor expansion . . . 221
A.2 Representation and interpretation . . . 222

B Finite Volume Method (FVM) applied in OpenFOAM for solving the RANS equations 225

Chapter 1

Introduction

Contents

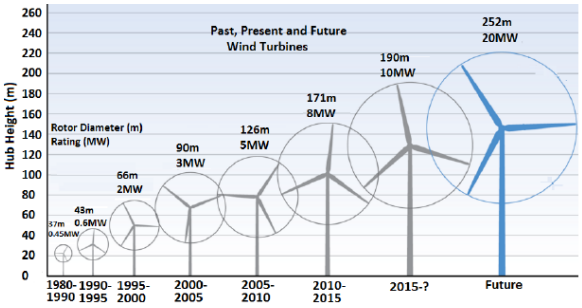
1.1 General context of the *Ph.D* research **2**
1.2 Review of existing models and literature **5**
1.3 Potential-viscous coupling methods **10**
1.4 Introduction of some dimensionless parameters **11**
1.5 Objectives and outlines of this thesis **13**
1.6 Publications and release of the programs **14**

1.1 General context of the *Ph.D* research

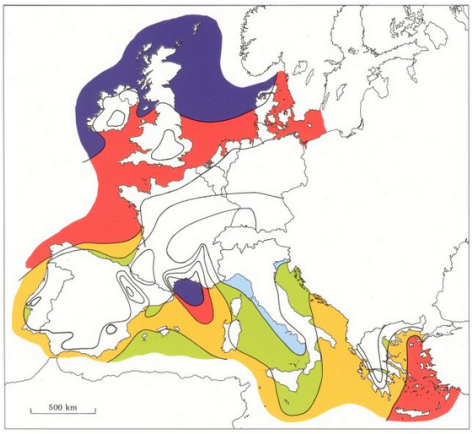
In a general context of global warming and climate change, countries worldwide have at heart to reduce their non renewable footprints. In Europe, commitments to reduce greenhouse gas emissions have been made, such as for example to increase the share of renewable energy in energy production to 32% by the 2030 horizon (Monti and Martinez Romera, 2020). In the majority of the scenarios considered nowadays, the wind turbines occupy a considerable part. While the technology itself is mature and the costs are predictable and contained, it becomes harder to start new project on land mainly due to population acceptability. Moreover, the constant increase in size and nominal power of the most efficient turbines (fig. 1.1a) accentuate this difficulty.

Trying to harvest the offshore wind, more stable and powerful, is a natural step. Acceptance is easier and the size of the turbine - closely related to power - is almost only limited by the available technology. One of the drawbacks is the environmental impact, especially when located on migration lines (Bailey *et al.*, 2014). On the other hand, the reef provided by the artificial immersed structure is rapidly used as a habitat by the marine ecosystem (see *e.g.* Gercken and Schmidt, 2014).

In the nearshore area, fixed solution (*e.g.* monopiles, gravity based structures, jackets, *etc.*) are a natural solution. However, the bathymetry can drastically limit the usage of the commonly employed fixed solutions. Indeed, for water depth larger than a few dozen of meters, the cost of the structure is prohibitive and floating support are considered instead (see fig. 1.2a). For example, while in the south of France, the Mediterranean exhibits a significant available wind resource (fig. 1.1b), fixed structures are hardly conceivable due to large water depth even in close coastal vicinities. For these reasons, three of the four floating wind turbines "pilot projects" attributed in 2015 are located in the Mediterranean Sea.



(a)



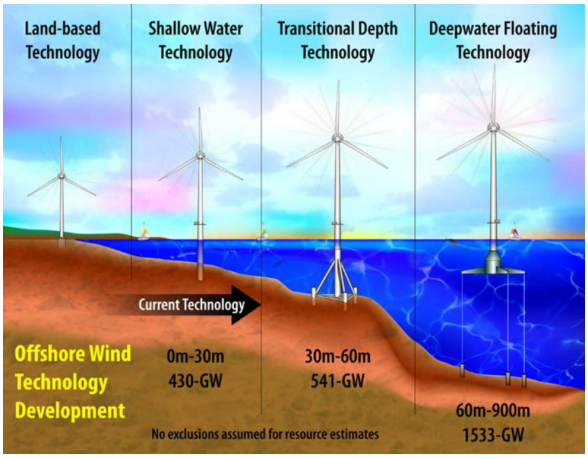
Wind resources over open sea (more than 10 km offshore) for five standard heights

	10 m		25 m		50 m		100 m		200 m	
	ms^{-1}	Wm^{-2}	ms^{-1}	Wm^{-2}	ms^{-1}	Wm^{-2}	ms^{-1}	Wm^{-2}	ms^{-1}	Wm^{-2}
	> 8.0	> 600	> 8.5	> 700	> 9.0	> 800	> 10.0	> 1100	> 11.0	> 1500
	7.0-8.0	350-600	7.5-8.5	450-700	8.0-9.0	600-800	8.5-10.0	650-1100	9.5-11.0	900-1500
	6.0-7.0	250-300	6.5-7.5	300-450	7.0-8.0	400-600	7.5- 8.5	450- 650	8.0- 9.5	600- 900
	4.5-6.0	100-250	5.0-6.5	150-300	5.5-7.0	200-400	6.0- 7.5	250- 450	6.5- 8.0	300- 600
	< 4.5	< 100	< 5.0	< 150	< 5.5	< 200	< 6.0	< 250	< 6.5	< 300

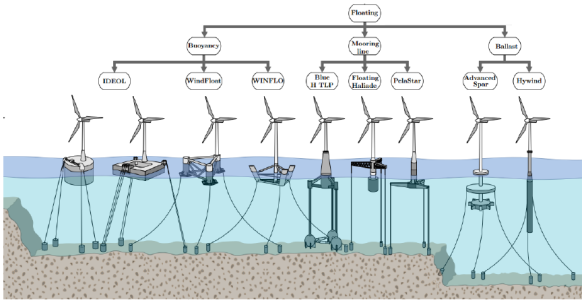
(b)

Figure 1.1: (a) Growth in size and power of wind turbine (Igwemezie *et al.*, 2019). (b): European offshore wind resource (Troen and Petersen, 1989).

Anchoring solutions and energy transfer methods to onshore are some examples of new engineering and research challenges arising with these technologies. Moreover waves impact and interact with the structure. In order to maintain and control the state and position of the structure, this interaction must be predicted accurately. The envi-



(a)



(b)

Figure 1.2: (a): bathymetry dependent solutions and associated ressource estimate by depth in the US (Musial, 2007). (b): existing solutions and proposed projects for floating offshore wind turbines (FOWT) (Igwemezie *et al.*, 2019).

ronmental loads applied on the structure can be extreme and offshore floating structures must be designed for the worst case scenario. For example, DNV GL (2018) states that "sea state with return periods of 100 years shall be considered". Earlier this year, during the storm Gloria, significant wave heights up to 8 m were measured in the west Mediterranean (Amores *et al.*, 2020). In the North Sea, the suggested requirement specification yields a design scenario of significant wave height of 15 m.

Depending on the depth and environmental conditions, different solutions for the floating structure are possible. Several projects, currently at different stage, exist. Some of them, and the associated selected technology are shown on fig. 1.2b. Note that a detailed review, from which this the figure was extracted, of the state of the art in the offshore wind industry was done by Igwemezie *et al.* (2019). Current propositions can be separated into three general categories that are distinguished by the main physical effect used for the stabilization, namely the buoyancy force (barge, semi-submersible, *etc.*), the mooring tension (tension leg platform) and the ballast gravity force (center of gravity lower that the center of buoyancy, SPAR).

Even though offshore structures are subjected to many different sollicitations, especially when considering floating wind turbine, this thesis focuses solely on the wave loads aspect. Thus, in order to design systems that withstand the mentioned environmental conditions, an accurate prediction of the loads applied on the structure, as well as its movements, must be available.

On the one hand, a first possible approach is to conduct experimental studies. A physical scaled model of the structure is placed into a wave flume or a wave tank, on which several external conditions are tested. This approach remains mandatory for a safe design, and experiments are often conducted multiple times during the

Figure 1.3: Examples of experiment of floating wind turbine (Lacaze, 2015).

design process. However, the associated costs are relatively significant making it difficult to test a wide range of models or conditions.

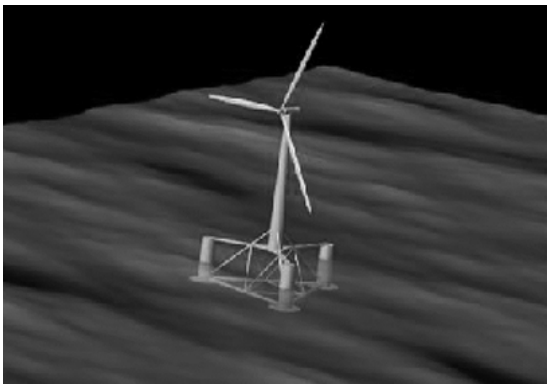


Figure 1.4: Examples of numerical simulation of floating wind turbines.

Numerical simulations, on the other hand can, at a lower cost, accurately model a range of phenomena. However, errors are always made, which are not always straight forward to quantify or interpret: part of the underlying physics might be lost or ignored without even realizing it. Those models are often validated by comparison with experiments, and trust in the model is extrapolated afterwards to other cases. However, many different models exist and the associated computation costs vary greatly, as well as the associated accuracy of results.

1.2 Review of existing models and literature

Remember that all models are wrong, but some are useful.

George Box, 1978

Many different numerical models and approaches are available, and commonly used, to predict the wave-structure interaction. In this section, a description of some of them is given. A more extensive and very detailed literature review and model description can be found in the work of Davidson and Costello (2020) where several existing approaches are discussed from, but not limited to, the wave energy converter modeling point of view.

1.2.1 Potential models

Amongst the different models, the potential theory is often put to use. The potential assumptions consist in neglecting the vortical part of the flow (which in turns means that turbulent effects cannot be taken into account) and in assuming that the fluid is inviscid. On top of this, the linear version is a further reduction that is based on a small wave amplitude hypothesis.

This simplified linear version is often used in the engineering field due to its low computational cost, allowing to capture the main effects on a wide range of parameters at a very contained CPU cost, for example in a design optimization process. WAMIT (Lee, 1995), ANSYS-AQWA (Ansys, 2013) or Nemoh (Fàbregas Flavià *et al.*, 2016) are examples of such widely used linear models. However, the linear assumption is often used outside its prescribed validity domain, for example in extreme cases where the allegedly small parameters, usually the wave steepness, become large. In those conditions many aspects of the dynamics of both the incident waves and the wave-body interaction are not properly modeled.

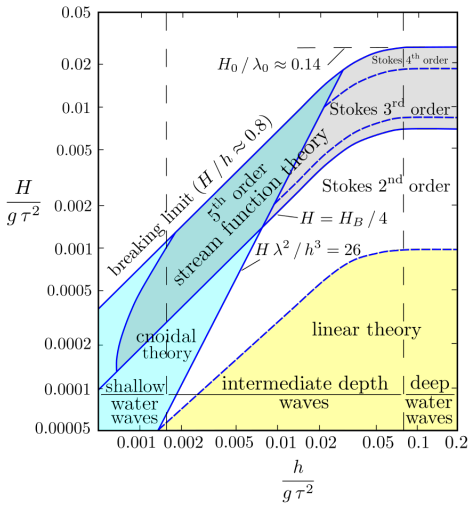


Figure 1.5: Validity domains of various wave theories. h is mean water depth, τ the wave period, H the wave height and g the gravitational acceleration. From wikimedia, after Le Méhauté (1976).

Efforts have been done to extend these potential linear models to weakly nonlinear conditions, *e.g.* at second order, mostly by adding terms like the Froude-Krylov forces or the complete Quadratic Transfer Functions, see *e.g.* Pinkster (1980) or more recently Philippe *et al.* (2015).

However in some cases, mostly extreme wave conditions, even third order effects can have a significant effect on the wave loads (see *e.g.*, Fedele *et al.*, 2017). Both the nonlinear wave dynamics and the nonlinear wave-structure interaction need to be taken into account. In those cases, alternative approaches capturing higher-order or fully nonlinear effects are needed. Several developments have been made to compute such nonlinear effects exactly,

or at high orders, in the case of periodic regular waves in uniform water depth. For example, models based on analytical theories such as the Stokes wave theory (see *e.g.* Sobey, 1989) or the so-called stream function method (Dean, 1965; Rienecker and Fenton, 1981; Fenton, 1988) can only be applied with constant or simple geometry of the sea floor. A review of several methods to describe wave propagation in a potential flow framework is given by Fenton (1999).

High Order Spectral (HOS) methods are also fast and accurate to compute the flow behavior even up to large wave steepness. Such methods, though very efficient in computing the wave elevation even for large domains, are difficult to apply for complex geometry as well as for wide ranges of parameters. They are mostly applied for wave maker modeling (Ducrozet *et al.*, 2012), or to compute the incident wave field within a more complex method to resolve the wave-structures interaction. The SWENSE method (see *e.g.* Luquet *et al.*, 2007b) on which the diffracted field is computed separately with a Navier-Stokes based solver, provides an example of the usage of such approaches.

In order to develop a versatile numerical wave tank (NWT), with the possibility to include one (or several) body or sea bed with complex geometries, a time domain resolution involving a mesh in the spatial coordinates appears to be practical at the cost of an increase in the computational time. The commonly used Boundary Element Method (BEM), in which the Laplace equation is projected onto the spatially discretized boundaries with the Green's identities, has been proven to be effective in both 2D and 3D cases. For example, Grilli and Horrillo (1997) used a high-order BEM method to generate and absorb waves in 2D. For an overview of work on the BEM methods up to the end of the 20$^{\text{th}}$ century, the reader is referred to Kim *et al.* (1999). More recently Guerber *et al.* (2012) presented in great detail and implemented a complete NWT. Note that under the BEM potential scheme, a special attention must be given to the treatment of the sharp corners at body or fluid domain boundaries (Hague and Swan, 2009).

1.2.2 Fields solvers and the Harmonic Polynomial Cell method (HPC)

By opposition to the BEM methods, in which only boundaries of the fluid domain are discretized, the name "field" or "volume" solver is attributed when the entire fluid domain has to be meshed. Numerically, this drastically increases the number of unknowns, however, the resulting matrix is mostly sparse allowing the use of efficient solvers. For example, the Finite Element Method (FEM), which falls in this category, was successfully applied to solve the nonlinear potential model by Yan and Ma (2007).

A different potential model that solves for the volume field was recently proposed by Shao and Faltinsen (2012b, 2014b,a) and tested against several methods including the BEM, FDM and FEM. This innovative technique, called the "Harmonic Polynomial Cell" (HPC) method, was proven to be promising both in 2D and 3D (Shao and Faltinsen, 2014b; Hanssen *et al.*, 2015, 2017a). Although relatively new, this method was used to study a relatively large range of flows and phenomena: from a closed flexible fish cage (Strand and Faltinsen, 2019) to hydrodynamic lifting problems (Liang *et al.*, 2015), but also, very recently, a sloshing problem in a circular tank (Liang *et al.*, 2020). It was also extended to solve the Poisson equation by Bardazzi *et al.* (2015). The numerical aspects of the method were studied in details by Ma *et al.* (2017) and applied as a 2D NWT by

Zhu *et al.* (2017). For a complete presentation of the method, the reader is also referred to the *Ph.D.* of Hanssen (2019), where a significant part is dedicated to this approach.

However, today, only a few implementations of this method exist. In a first step, the current study will focus on the development and implementation of a solver based on this HPC approach.

1.2.3 Introduction of viscous and turbulent effects

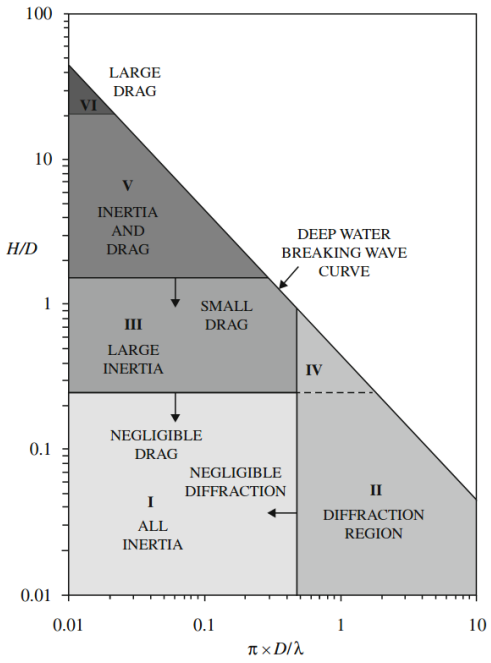


Figure 1.6: Dominant effect(s) of the wave-structure interaction depending on the wave height H, the wavelength λ and the characteristic dimension of the object D. From Bergdahl (2017).

Remember that all models presented above are potential-based. Thus, they intrinsically cannot capture rotational and viscous effects. For a range of applications those assumptions are mostly verified, and those models are well-adapted to capture the dominant effects. This is for instance the case when considering a ship-shaped structure submitted to a common sea state. Figure 1.6 shows that, in a more general manner, diffraction and inertial effects are dominant when the structure is of large dimension compared to the wave typical dimensions, namely wavelength and wave height. Note that these effects can be captured by a potential approach. However, on other applications a complete description of all the significant phenomena taking place cannot be correctly captured by this model. For example, loads applied on a fish farm, fish net or on an anchor cable are mostly governed by the viscous drag, neglected in a pure potential approach. More generally, when the object is of lower dimension relative to the wave spatial characteristics, the drag must be taken into account. At a certain point, when $D \ll \lambda$ and $D \ll H$ the drag largely dominates (upper left corner on fig. 1.6), in that case, the Morison formula can be applied independently (potential effects can be neglected), with the underlying assumption that the wave field is not perturbed by the object presence. Moreover for any application where wave breaking or vortex emissions occur and are of prime importance, the use of a potential model is not recommended: many of the underlying physics will not be captured.

It is possible, however, for intermediate cases, to add an *ad-hoc* correction to a potential model. For instance, one can apply, in addition to the potential solution yielding the potential part of the loads, the Morison formula to estimate the viscous loads based on the surrounding velocity. This model is applicable when the considered object does not perturb the wave field in a significant manner (for example an anchor cable) but is often used outside of its prescribed range, *e.g.* to evaluate the viscous drag of a heave plate of a semi-submersible. Another example of such correction is the use of numerical lid to force local dissipation of energy: this technique is often used to model wave breaking or wave

dissipation over very long distance.

The last type of model that we will present here completely changes of focus. Computational Fluid Dynamics (CFD) codes focus on solving directly the complete Navier-Stokes (NS) equations. Within this framework, one is attached to solve the NS equations with the minimal number of hypotheses. Thus, viscous and vortical effects are taken into account. However, solving all scales of turbulence is rarely possible in this context, because of the very large required numerical cost, due to the wide simulation domain. Thus, once again, small scales of the flow can be modeled in a macro manner. The first and most used of these models is the Reynolds-Averaging Navier-Stokes (RANS) method. Another commonly used and studied is called the Large Eddy Simulation (LES) method. The latter is more expensive while often considered as more accurate with interesting properties in terms of convergence. Both however are really expensive when compared to a potential model.

In the last decade, the use of CFD code (mainly in conjunction with the RANS method) has become increasingly popular in the scientific community, see for instance the applications of industrial or research codes like the Open Field Operation And Manipulation (OpenFOAM®) package (Jacobsen *et al.*, 2011; Higuera *et al.*, 2013; Hu *et al.*, 2016; Windt *et al.*, 2019), STAR-CCM+ (Oggiano *et al.*, 2017; Jiao and Huang, 2020), ANSYS-FLUENT (Kim *et al.*, 2016; Feng and Wu, 2019), and REEF3D (Bihs *et al.*, 2016), just to mention a few of them.

A notable work on a NWT based on the finite difference method (FDM) is given by Tavassoli and Kim (2001). The FEM was also successfully applied to both the potential problem (*e.g.* Ma and Yan, 2006; Yan and Ma, 2007) and the NS equations (*e.g.* Wu *et al.*, 2013).

Although these CFD models may be perfectly adapted to capture complex wave-structure interaction down to small scales, in particular when complex physical processes such as wave breaking, formation of jets, air entrapment, *etc.* occur, their use remains quite limited due to their computational cost. This is particularly true when targeting applications on ocean domains whose extent is larger than, say, about 10 typical wavelengths. Limitations associated with the employed numerical methods (*e.g.* numerical diffusion and difficulty to resolve accurately the dynamics of the free surface) are other reasons which still hinder the applicability of such models to large scale wave propagation problems. Thus, models based on a potential flow approach (*i.e.* neglecting viscous effects and assuming irrotational flow) remain widely used to describe the dynamics of wave-structure interaction flows (see *e.g.* the review by Tanizawa, 2000)

1.2.4 Free surface treatment

The treatment of the free surface conditions and body boundary condition can be done in several ways. A classical and straightforward approach is to use a grid that conforms the boundary shape. With this method, the boundary nodes values are explicitly enforced in the linear system. The drawback of this method is that the grid needs to be deformed at each time step so as to match the free surface or body position. A lot of solutions have been successfully applied to tackle this issue. For instance, Ma and Yan (2006) used a Quasi Arbitrary Lagrangian-Eulerian (ALE) method combined with the FEM spatial discretization to prevent the mesh to have to be regenerated at each time step. Yan and

Ma (2007) extended the mesh conformation technique to include a freely floating body. With this same FEM discretization, Wu *et al.* (2013) used, in addition, a hybrid Cartesian Immersed Boundary Method (IBM) for the body boundary condition.

Using an IBM for the free surface, *i.e.* describing the interface with a discretized surface evolving independently from the background mesh, is also a possibility. This method was for example applied in the context of a HPC method by Hanssen *et al.* (2017a).

Another type of approach consists in the introduction of a global function that indicates in which fluid a given location lays on. Level Set methods, introduced by Osher and Sethian (1988) and extended in the context of two phase flows by Sussman *et al.* (1994) belong in this category: the indicator function is the signed distance from the interface. The Volume of Fluid (VoF) method (Hirt and Nichols, 1981) is another example. This time, the function takes the form of a discretized field indicating the fraction of one of the fluid in a given cell. Both methods have been extensively used in the context of wave-body interaction.

In the current study, a description of the implementation of an IBM for the free surface will be given in the context of the HPC method, and afterwards, in the context of OpenFOAM®, the VoF method will be presented and employed.

1.3 Potential-viscous coupling methods

CFD approaches are able to capture several effects that lie outside of the range of the potential assumptions. Note that, in the context of wave-structure interaction, these effects are mostly contained in the close vicinity of the object. On the other hand, potential models are well suited to simulate the propagation phase of waves, because of their intrinsic low energy dissipation rate. An intuitive idea is thus to apply each model in the zone where it best performs. This type of method is referred to as a "coupled" or "hybrid" method.

A brief generic presentation of those approaches is given here, even though a more detailed analysis and literature review is presented in the corresponding chapters, namely sections 5.1.2 and 6.1.2.

The idea of using different models to capture different effects is not new: the studies that tried to include the boundary layer viscous effects, within an otherwise laminar simulation, can be seen as a hybrid model. For example, the very boundary layer theory (Prandtl, 1904) falls into that category, as well as the further work by Lighthill (1958).

The introduction in the context of wave flows was done by Dommermuth *et al.* (1997), who applied a decomposition of the flow into an irrotational and vortical part to solve the contact line problem in bow waves.

For quite a while now authors have suggested and successfully developed coupling schemes that use each model in the areas where they are the most adequate. Those coupling schemes can be separated into two main categories: domain decomposition methods and velocity decomposition methods. The first one simply uses different resolution methods for different domains, that, most of the time, do not overlap each other. Information is passed between the two models only at common boundary conditions.

The second type of decomposition leads to a modification of the equations themselves. A flow that is solution of a simplified set of equations already verifies a significant portion of the more generic equations (respectively Euler equations and NS equations in our case). Thus, if the simplified equations are solved on the whole domain of interest, solving the generic set of equations from scratch can be seen as largely redundant. Modifying the second set of equations to take into account the already computed part is possible, and is hoped to lead to significant computational cost savings. In this framework we state that $\mathbf{u} = \mathbf{u}_1 + \mathbf{u}_2$ where $\mathbf{u}$ is the real total velocity and $\mathbf{u}_1$, $\mathbf{u}_2$ the velocities computed by the model 1 and 2 respectively (similar decomposition can be applied to other variables, *e.g.* pressure).

A further distinction can be made between one- and two-way coupling methods (also referred to as weak and strong couplings). Within a one-way framework, the first model is used to solve a given wave field. The second model imports the (boundary or volume) values to solve in a more contained zone the more complex set of equations. No feedback from the second model to the first one happens, which implies that the first one is independent and can be employed *a priori*. In a two-way coupling method, both models receive informations from the other (at their common boundary conditions on a domain decomposition approach), each one having an effect on the other. Thus, they have to evolve simultaneously and "wait" for the other.

In this study, both domain and velocity decomposition approaches will be tackled. However, as a first step, only one-way couplings are tackled.

1.4 Introduction of some dimensionless parameters

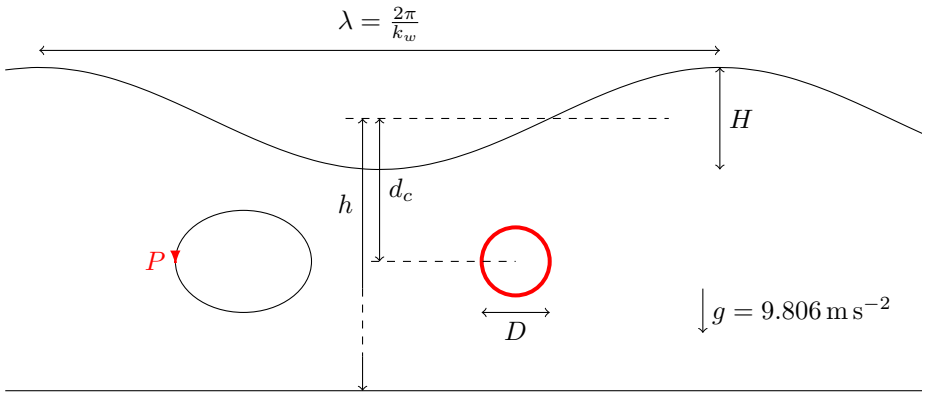


Figure 1.7: Schematics of a body submitted to regular water waves.

Some of the notation used throughout this work are introduced in fig. 1.7. From these physical parameters, others can be defined and are dependent of the latter. For example the wave number $k_w = 2\pi/\lambda$, but also the wave period, denoted T. It is for example, assuming linear potential waves given by:

$$T = \frac{2\pi}{\sqrt{k_w g \tanh(k_w h)}} \tag{1.1}$$

where g is the gravitational acceleration ($g = 9.806\,\mathrm{m\,s^{-2}}$).

In order to have a relative overview of the present physical effects, some dimensionless numbers can be defined and will be used extensively in the manuscript. A brief overview is given here, as well as their main influence on the flow physics.

Wave steepness $\frac{H}{\lambda}$, $\frac{H}{gT^2}$ or $\frac{k_w H}{2}$ This number compares the wave height with the wavelength and is used to quantify the nonlinearity of a given wave. For example, at significant wave steepness ($H/\lambda > 1\%$, at large water depth) linear models are expected to yield some errors in the wave description and a nonlinear model is needed (see fig. 1.5). In the deep water limit, wave breaking occurs at $H/\lambda = 14\%$.

Relative body dimension to the wavelength $\pi\frac{D}{\lambda}$ In fig. 1.6 are shown the different expected dominant effects of the fluid-structure interaction for different regimes. When $\pi\frac{D}{\lambda} < 0.4$ the interaction is mostly driven by inertial and/or drags effects. These effects can be modeled by the Morison eq. (3.3), on which further discussions are conducted in section 3.3.3.2. These equations neglects however the diffraction effects, making them unfit to model the wave-body interaction when the body is of greater relative dimension ($\pi\frac{D}{\lambda} > 0.4$).

Wave height relative to the body dimension H/D This parameter draw one of the limit of the potential assumptions: at important H/D (see regimes VI,V, and III on fig. 1.6) viscous drag effects are of prime importance, and diffraction+inertial effect, captured by a potential model, are not sufficient to describe the entire flow physics.

Reynolds number $Re = \dfrac{U_m D}{\nu}$ The classically used Reynolds number is extended to oscillatory flows of velocity amplitude in the direction of the wave U_m (U_m can also denote the maximum horizontal velocity of the flow). This number parametrises the relative contribution of the inertial effects to viscous effects. At high Reynolds number, inertial effects are dominant over viscous effects and the flow is expected to transition to turbulent. For example, at $Re \geq 10^3 - 10^4$ for a uniform flow around a sphere.

Keulegan-Carpenter number $KC = \dfrac{U_m T}{D}$ It is used to compare the horizontal motion amplitude of a fluid particule at the body depth to the body characteristic dimension in the horizontal direction. Using the linear assumption this parameter can also be expressed as

$$KC = \pi \frac{H}{D} \exp(-k d_c) \tag{1.2}$$

At $KC \geq 1$, the particule motion amplitude is wider than the body horizontal breath. At this stage, strong circulation are expected (Arai, 1995) and potential models are expected to be outside of their range of validity. For further discussions about the influences of this parameter, the reader is referred to Fredsoe and Sumer (1997), Molin and Ferziger (2003), and Guerber (2011)

Frequency parameter $\beta = \dfrac{Re}{KC} = \dfrac{D^2}{\nu T}$ Dimensionless number representing the wave frequency.

1.5 Objectives and outlines of this thesis

During this study, only cases in two spatial dimensions are considered, and structures will remain fixed. The overall objective is to evaluate different approaches: the accuracy of their prediction of wave loads exerted on a body, and their associated computational costs.

First of all, this work aims to develop and validate a new implementation of the HPC method to solve the Boundary Value Problem (BVP) on the potential. The necessary features to obtain an accurate and versatile NWT, such as to allow for a body inclusion, are also developed. Afterwards, the main objective is to develop a one-way modular velocity decomposition model within the OpenFOAM® framework. Natural steps towards this objective include the validation of OpenFOAM® to capture the turbulent and viscous effects and the development of a domain decomposition approach. The coupling solvers developed are modular in sense that any other OpenFOAM® formatted case, can serve as the outer model. The structure of this work reflects those overall goals and steps.

Firstly, a fully nonlinear potential model is developed in FORTRAN. Chapter 2 presents the mathematical formalism and the various numerical methods implementations, both developed during this work and inspired from existing literature. It lays on the method introduced by Shao and Faltinsen (2012a), presented as an efficient solution of solve the BVP. Afterwards, in chapter 3, the implemented NWT is evaluated. As a first step (section 3.2), a complete convergence study in both space and time is conducted on a highly nonlinear standing wave. Then, in section 3.3, an extensive validation is conducted against several experimental and numerical results. Note that the deforming mesh approach, used to fit the free surface, is evaluated in section 3.4, where reasons that led us to reconsider it are detailed.

However, some limits of this NWT are encountered, the main one being related to its intrinsic potential assumptions. On many engineering cases of practical interest, neglecting the viscosity and turbulence leads to missing out significant aspects of the wave-body interaction. Thus, in chapter 4, an open source toolkit, OpenFOAM®, and particularly the RANS multiphase FVM VoF solver is presented, applied, and validated against experiments. Many variations of parameters are assessed and a conclusion is drawn about the capabilities of the solver "waveFoam" (part of the wave dedicated toolbox developed by Jacobsen *et al.*, 2011) to be used as an accurate and reliable NWT.

The last two chapters follow a common objective: use the best of both worlds. Capturing, thanks to the capabilities of OpenFOAM®, the complex effects of the body vicinity flow, while employing the developed HPC method to propagate and reflect the potential part of the wave field. These decomposition approaches are both one-way coupling methods. Chapter 5 focuses on developing a modular domain decomposition and all the associated tools. On the contrary, the study conducted in chapter 6 concerns a velocity decomposition method further taking advantage of the fact that the main part of the NS equations has already been solved, and that we thus have direct access to a solution of the Euler equations. Both methods are validated against other models presented in this manuscript, as well as against experimental results.

1.6 Publications and release of the programs

This work, and more particularly the first part about the HPC methods, has been presented in a conference, with the following proceeding:

- Fabien Robaux and Michel Benoit (2018). "Modeling Nonlinear Wave-Body Interaction with the Harmonic Polynomial Cell Method Combined with the Immersed Boundary Method on a Fixed Grid". In: *Proc. 32th International Workshop on Water Waves and Floating Bodies, Guidel-Plages, France, April 4 to 7*.

A journal article was also submitted:

- Fabien Robaux and Michel Benoit (2020). "Development and Validation of a Numerical Wave Tank Based on the Harmonic Polynomial Cell and Immersed Boundary Methods to Model Nonlinear Wave-Structure Interaction". In: *arXiv:2009.08937 [physics]*. arXiv: 2009.08937 [physics]. URL: http://arxiv.org/abs/2009.08937 (visited on 09/21/2020). A large part is common, between this article and chapters 2 and 3. However, methods (section 2.5) and several validations results are further detailed here.

All the developed codes aim to be released in open source under different licenses. While they are not published at the time these lines were written, the plan is to make them available by the end of 2020. They will probably be hosted in a corresponding github repository at https://github.com/fabienRobaux/. This includes

- The HPC program. However, for licensing reasons, the stream-function model for the enforcement of the wave field at inlet will not be present. The plan is to include a Stokes order 5 model instead.
- The conversion routines HPC→OpenFOAM® format.
- The variations of the turbulence model presented in section 4.2.4, as well as the Larsen and Fuhrman (2018) version slightly modified, only to allow the outputs of many different fields, investigated here in section 4.5.
- the domain decomposition solver requiring the OpenFOAM® libraries.
- the velocity decomposition solver requiring the OpenFOAM® libraries.

In order to make this work as reproductible as possible, we will try to include the mesh generation routines and methods as well as tutorial cases when adequate.

Declaration of competing interests The author declare that he has no known competing financial interest or personal relationships that could have appeared to influence the work reported in the thesis. This thesis was funded by Ecole Normale Supérieure de Paris-Saclay (formerly ENS Cachan), trough a Specific Doctoral Contract for students of Normale school (CDSN).

Chapter 2

Solving of the Laplace problem with HPC method

Contents

2.1 Introduction 16
2.2 Fully nonlinear potential flow modeling approach 17
2.3 The Harmonic Polynomial Cell method (HPC) with immersed free surface 19
2.4 Introduction of a fixed body in the NWT 27
2.5 Treatment of a sharp corners 31

2.1 Introduction

In order to model waves and their interaction with offshore structures, up to high degree of nonlinearity, the potential approach is often selected. A good candidate to resolve such model was introduced by Shao and Faltinsen (2012b) and denoted HPC. Based on overlapping cells, the velocity potential is approximated as a weighted sum of harmonic polynomials inside a given cell. In this framework, the governing Laplace equation is automatically verified in each cell independently, thus, the linear system resolution is there to ensure that cells that overlaps yield consistent values in their shared area. In this document, a HPC cell will be denoted a macro-cell.

The first part of this *Ph.D* (chapters 2 and 3) focuses on a NWT, developed within the HPC framework and implemented from scratch. We first describe the various mathematical formalisms and numerical methods associated with this NWT. Afterwards, in chapter 3, its convergence will be assessed, and validations against both numerical and experimental data will be conducted.

Relaxations zones are used to generate and absorb the waves in a similar manner as in the OpenFOAM® toolbox waves2foam (Jacobsen *et al.*, 2011): the values and positions of the free surface nodes are imposed from a stream-function theory over a given distance. The free surface is tracked in a semi-Lagrangian way following Hanssen *et al.* (2017a) whereas for the solid bodies, an additional grid fitted to the body boundaries is defined, following the recommendations given by Ma *et al.* (2017).

In order for NWT to accept free surface piercing bodies several numerical developments are implemented, for example the addition of a new free surface marker list defined in the body fitted grid, denoted the "body fitted" free surface. "Background" and "boddy fitted" free surfaces thus overlap each other and communicate through relaxation zones. In order to tackle the singular nodes, a velocity flux method (studied in further details by Zhu *et al.*, 2017), is imposed at the sharp corners that lay on a Neumann boundary condition.

The presentation of the different methods and their developments implemented here, within the HPC framework, is organized in this chapter as follows. First, section 2.2 presents the associated mathematical formulation of the different equations governing the problem. Then, the different numerical methods are presented in section 2.3, with particular attention devoted to the treatment of the free surface dynamics (section 2.3.4). Afterwards, a fitted mesh overlapping grid method is described and implemented in section 2.4, along with the treatment of the free surface(s) employed when the body pierces the free surface. Finally, the approach selected to enforce a Neumann condition on a sharp body corner is presented in section 2.5.

In the following chapter 3, the main results obtained during this work will be presented, and discussions will be conducted on some of the emphasized advantages and limits of this NWT.

2.2 Fully nonlinear potential flow modeling approach

Three main assumptions are used in the potential model: i. we consider a fluid of constant and homogeneous density ρ (incompressible flow); ii. the flow is assumed to be irrotational, implying that $\nabla \times \mathbf{v} = 0$, where $\mathbf{v}(x, y, z, t)$ denotes the velocity field; and iii. viscous effects are neglected (ideal fluid). Here, (x, y) denote horizontal coordinates, z the vertical coordinate on a vertical axis pointing upwards, and t the time. The gradient operator is defined as $\nabla f \equiv (f_x, f_y, f_z)^T$, where subscripts denote partial derivatives (*e.g.* $f_x = \frac{\partial f}{\partial x}$).

The complete description of the velocity field $\mathbf{v}$ can thus be reduced to the knowledge of the potential scalar field $\phi(x, y, z, t)$, such that: $\mathbf{v} = \nabla\phi$. Due to the incompressibility of the flow, the potential ϕ satisfies the Laplace equation inside the fluid domain:

$$\nabla^2\phi = 0, \qquad -h(x, y) \leq z \leq \eta(x, y, t) \tag{2.1}$$

where $\eta(x, y, t)$ is the free surface elevation and $h(x, y)$ the water depth relative to the still water level (SWL). In order to solve this equation, boundary conditions need to be considered. On the time varying free surface $z = \eta(x, y, t)$, the Kinematic Free Surface Boundary Condition (KFSBC) and the Dynamic Free Surface Boundary Condition (DFSBC) apply:

$$\eta_t + \nabla_H\eta \cdot \nabla_H\phi - \phi_z = 0 \quad \text{on } z = \eta(x, y, t), \tag{2.2}$$

$$\phi_t + \frac{1}{2}\left(\nabla_H\phi\right)^2 + g\eta = 0 \quad \text{on } z = \eta(x, y, t), \tag{2.3}$$

where $\nabla_H f \equiv (f_x, f_y)^T$ denotes the horizontal gradient operator and g the acceleration due to gravity. At the bottom (impermeable and fixed in time), the Bottom Boundary Condition (BBC) reads:

$$\nabla_H h \cdot \nabla_H\phi + \phi_z = 0 \quad \text{on } z = -h(x, y). \tag{2.4}$$

On the body surface, the slip boundary condition expresses that the velocity component of the flow normal to the body face equals the normal component of the body velocity. Here, we restrict our attention to fixed bodies, thus, denoting $\mathbf{n}$ unit vector normal to the body boundary, this condition reduces to:

$$\frac{\partial\phi}{\partial\mathbf{n}} = \nabla\phi \cdot \mathbf{n} = 0 \quad \text{on the body.} \tag{2.5}$$

Note that, using the free surface velocity potential and the vertical component of the velocity at the free surface, defined respectively as:

$$\tilde{\phi}(x, y, t) = \phi(x, y, \eta(x, y, t), t) \tag{2.6}$$

$$\tilde{w}(x, y, t) = \frac{\partial\phi}{\partial z}(x, y, \eta(x, y, t), t) \tag{2.7}$$

the KFSBC and the DFSBC can be reformulated following Zakharov (1968) as:

$$\eta_t = -\nabla_H\eta \cdot \nabla_H\tilde{\phi} + \tilde{w}(1 + (\nabla_H\eta)^2) \tag{2.8}$$

$$\tilde{\phi}_t = -g\eta - \frac{1}{2}(\nabla_H\tilde{\phi})^2 + \frac{1}{2}\tilde{w}^2(1 + (\nabla_H\eta)^2) \tag{2.9}$$

It can be noted that the Laplace equation (2.1), the BBC (2.4) and the body BC (2.5) are all linear equations. Thus, the nonlinearity of the problem originates uniquely from the free surface boundary conditions (2.2-2.3) or (2.8-2.9). At that point, the commonly used linear wave theory assumes small amplitude surface waves, so that these boundary conditions can be linearized and applied at the SWL (*i.e.* at $z = 0$). Here, we intend to retain full nonlinearity of wave motion by considering the complete conditions (2.8 and 2.9). These two equations are used to compute the time evolution of the free surface elevation η and the free surface potential $\tilde{\phi}$. This requires obtaining $\tilde{w}$ from $(\eta, \tilde{\phi})$, a problem usually referred to as Dirichlet-to-Neumann (DtN) problem.

For given values of $(\eta, \tilde{\phi})$, the DtN problem is here solved by solving a BVP problem in the fluid domain on the wave potential $\phi(x, y, z, t)$, composed of the Laplace equation (2.1), the BBC (2.4), the body BC (2.5), the imposed value $\phi(z = \eta) = \tilde{\phi}$ (Dirichlet condition) on the free surface $z = \eta$, supplemented with boundary conditions on lateral boundaries of the domain (of *e.g.* Dirichlet, Neumann, *etc* type). The numerical methods to solve the BVP are presented in the next section.

2.3 The Harmonic Polynomial Cell method (HPC) with immersed free surface

2.3.1 General principle of the HPC method

In order to solve the above mentioned BVP at a given time, the HPC method introduced by Shao and Faltinsen (2012b) is used. It is briefly described here, and more details can be found in Shao and Faltinsen (2014a), Hanssen *et al.* (2015, 2017a), Hanssen (2019) and Ma *et al.* (2017). In this work, the HPC approach is implemented and tested in 2 spatial dimensions, *i.e.* in the vertical plane (x, z), for a wide range of parameters.

The fluid domain is discretized with overlapping macro-cells which are composed of 9 nodes in 2 dimensions. Those macro-cells are obtained by assembling four adjacent quadrilateral cells on an underlying quadrangular mesh. The four cells of a macro-cell share a same vertex node, called the "central node" or "center" of the macro-cell. A typical macro-cell is schematically shown in figure (2.1), with the corresponding local index numbers of the 9 nodes. With this convention, any node with global index n has the local index "9" in the considered macro-cell and is considered as an interior fluid point, whereas for example node with local index "4" can either be a fluid point or a point lying an a boundary. If node "4" is also inside the fluid domain, it also defines another macro-cell whose the center right point (with local index "5") will then correspond to the node with global index n.

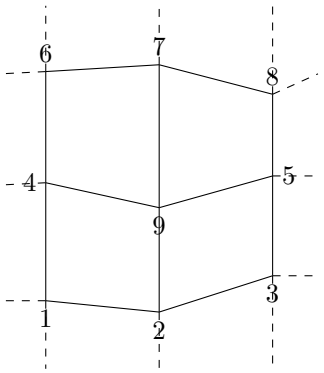


Figure 2.1: Definition sketch of 9-node macro-cell used for the HPC method, with local numbering of the nodes

In each macro-cell, the velocity potential is approximated as a weighted sum of the 8 first harmonic polynomials (HP), the later being fundamental polynomial solutions of the Laplace equation (2.1). A discussion about which of the HP are to be chosen is given in Ma *et al.* (2017). Here, we follow Shao and Faltinsen (2012b), and select all polynomials of order 0 to 3 plus one fourth-order polynomial, namely: $f_1(\mathbf{x}) = 1$, $f_2(\mathbf{x}) = x$, $f_3(\mathbf{x}) = z$, $f_4(\mathbf{x}) = x^2 - z^2$, $f_5(\mathbf{x}) = xz$,$f_6(\mathbf{x}) = x^3 - 3xz^2$, $f_7(\mathbf{x}) = -z^3 + 3x^2z$ and $f_8(\mathbf{x}) = x^4 - 6x^2z^2 + z^4$. Here, $\mathbf{x} = (x, z)$ represents the spatial coordinates. Thereafter, we define $\bar{\mathbf{x}} = \mathbf{x} - \mathbf{x}_9$ the same spatial coordinate in the local reference frame of the macro-cell, with $\mathbf{x}_9$ being the center node of the macro cell. From a given macro-cell, the

potential can be approximated at a location $\mathbf{x}$ as:

$$\phi(\mathbf{x}) = \sum_{j=1}^{8} b_j f_j(\bar{\mathbf{x}}) \tag{2.10}$$

As every HP is a solution of the Laplace equation (2.1) which is linear, any linear combination of them is also solution of this equation. Thus, the goal now becomes to match the local expressions (LE) given by equation (2.10) such that every local expression is verified at each participating node. For that reason, macro-cells overlap each other. Note that this study deals with 2D problems, but the method can be extended to 3D cases as shown by Shao and Faltinsen (2014b) considering cubic-like macro-cells with 27 nodes.

The first objective is to determine the vector of coefficients $b_j, j = 1, ..., 8$ for the selected macro-cell. Recalling that equation (2.10) should be verified at the location of each point of the macro-cell (with local index running from 1 to 9), this equation applied at the 8 neighboring nodes $(1-8)$ of the center yields:

$$\phi_i = \phi(\mathbf{x}_i) = \sum_{j=1}^{8} b_j f_j(\bar{\mathbf{x}}_i) \quad \text{for } i = 1, ..., 8 \tag{2.11}$$

which represents, in vector notation, a relation between the vector of size 8 of the values of the potential ϕ_i at the outer nodes with the vector of size 8 of the b_j coefficients. The 8x8 local matrix linking this two vectors is denoted $\mathbb{C}$, and defined by $C_{ij} = f_j(\mathbf{x}_i)$. Note that $\mathbb{C}$ is defined geometrically, thus it only depends on the position of the outer nodes i relatively to the position of the central node. $\mathbb{C}$ can be inverted and its inverse is denoted $\mathbb{C}^{-1}$. The b_j coefficients are then obtained for the given macro-cell as a function of the potentials at the 8 neighboring nodes of the central node of that macro-cell:

$$b_j = \sum_{i=1}^{8} C_{ji}^{-1} \phi_i \quad \text{for } j = 1, ..., 8. \tag{2.12}$$

Injecting this result into the interpolation equation (2.10), a relation is found providing an approximation for the potential at any point located inside the macro-cell using the values of the potential of the eight surrounding nodes of the central node:

$$\phi(\mathbf{x}) = \sum_{i=1}^{8} \left(\sum_{j=1}^{8} C_{ji}^{-1} f_j(\bar{\mathbf{x}}) \right) \phi_i \tag{2.13}$$

This equation will be referred to as local expression (LE) of the potential. It will be used to derive the boundary conditions equations and the fluid node equations that need to be solved in the BVP. Also note that this LE provides a really good interpolation function that can be used for every additional computation once the nodal values of the potential are known (*i.e.* potential derivatives at the free surface or close to the body to compute the pressure field).

Note that the accuracy of LE depends only on the geometry: coordinates at which this equation is applied, shape of the macro-cell, *etc.* Those dependencies are investigated in details by Ma *et al.* (2017).

2.3.2 Treatment of nodes inside the fluid domain

We first consider the general case of macro-cells whose central node is an interior node of the fluid domain. Applying the LE (2.13) at the central node yields a linear relation between the values of the potential at the nine nodes of this macro-cell:

$$\phi_9 = \phi(\mathbf{x}_9) = \sum_{i=1}^{8} \left(\sum_{j=1}^{8} C_{ji}^{-1} f_j(\bar{\mathbf{x}}_9) \right) \phi_i \tag{2.14}$$

We may further simplify this equation by noting that, as $\bar{\mathbf{x}}_9 = (0,0)$ in local coordinates, all $f_j(\bar{\mathbf{x}}_9)$ vanish, except $f_1(\bar{\mathbf{x}}_9)$ which is constant and equal to 1. Equation (2.14) then simplifies to:

$$\phi_9 = \sum_{i=1}^{8} C_{1i}^{-1} \phi_i \tag{2.15}$$

meaning that only the first row of the matrix $\mathbb{C}^{-1}$ is needed here.

In order to solve the global potential problem, *i.e.* to find the value of the nodal values of the potential at all grid points (whose total number is denoted N), a global linear system of equations is formed, with general form $\mathbb{A}.\phi = \mathbf{B}$, or:

$$\sum_{l=1}^{N} A_{kl} \phi_l = B_k \quad \text{for } k = 1, ..., N. \tag{2.16}$$

where k and l are global indexes of the nodes. For each interior node in the fluid domain, with global index k and associated macro-cell, an equation of the form (2.15) allows to fill a row of the global matrix $\mathbb{A}$. This row k of the matrix involves only the considered node and its 8 neighboring nodes, making the matrix $\mathbb{A}$ very sparse (at most 9 non-zero elements out of N terms). Moreover, the corresponding right-hand-side (RHS) term B_k is null. Note that all the 8 neighboring nodes of the macro-cell associated with center k should also have a dedicated equation in the global matrix in order to close the system.

2.3.3 Nodes where a Dirichlet or Neumann boundary condition is imposed

If a Dirichlet boundary condition with value ϕ_D of the potential has to be imposed at the node of global index k, the corresponding equation is simply $\phi_k = \phi_D$, so that only the diagonal element of the global matrix is non-null and equal to 1 for the corresponding row k: $A_{kl} = \delta_{kl} \ \ \forall l \in [1, N]$. The corresponding term on the RHS is set to $B_k = \phi_D$.

If a Neumann condition has to imposed at a given node of global index k, the relation set in the global matrix is found trough the spatial derivation along the imposed normal $\mathbf{n}$ of the LE (2.13) of any macro-cell on which k appears. In practice, the macro-cell whose center is the closest from the node k is chosen, and we then use:

$$\nabla\phi(\mathbf{x}_k) \cdot \mathbf{n} = \sum_{i=1}^{8} \left(\sum_{j=1}^{8} C_{ji}^{-1} \nabla f_j(\bar{\mathbf{x}}_k) \cdot \mathbf{n} \right) \phi_i \tag{2.17}$$

Thus, a relation is set in the row k of the global matrix to enforce the value of $B_k = \nabla\phi(\mathbf{x}_k) \cdot \mathbf{n}$ at position $\mathbf{x}_k$. In that case, a maximum of 8 non-zero values appear in this row on the global matrix as the potential of the central node of the macro-cell does not intervene here.

2.3.4 Treatment of the free surface

As already mentioned, in order to solve the BVP at a given time-step, the system of equations needs to be closed, meaning that each neighbor of a node in the fluid domain should have a dedicated equation. We consider now the case of nodes lying on or in the vicinity of the (time varying) free surface. The free surface potential should be involved here, either directly at a node fitted to the free surface through a Dirichlet condition described in the previous sub-section, or through alternative techniques.

For instance, an Immersed Boundary Method (IBM) was first suggested in the HPC framework by Hanssen *et al.* (2015) to tackle body boundary conditions. More recently, Ma *et al.* (2017) compared a modified version of the IBM with two different multi-grid (MG) approaches (fitted or combined with an IBM) for both body and free surface boundary conditions. Hanssen *et al.* (2017a) and Hanssen *et al.* (2017a) also made in-depth comparisons of the MG and IB approaches, focusing on the free surface tracking. Both methods showed promising results. Zhu *et al.* (2017) introduced a similar yet slightly different IB approach with one or two ghost node layers, then realized a comparison between this IB approach and the original fitted mesh approach. In the present work, the IBM was chosen for the treatment of the free surface, though the fitting mesh method is shortly described thereafter.

2.3.4.1 Fitted mesh approach for the free surface

The first possibility is to fit the mesh to the actual free surface position at any time when the BVP has to solved. The mesh is deformed so that the upper node at any abscissa always lies on the free surface. That way, the computational domain is completely closed and the free surface potential is simply enforced as a Dirichlet boundary condition at the correct position $z = \eta$ as explained in section 2.3.3. With this approach, the algorithm, given the boundary values at the considered time, can be summarized as:

- Deform the mesh to fit the current free surface elevation,
- Build and then invert the local geometric matrices $\mathbb{C}$,
- Fill the global matrix $\mathbb{A}$ and RHS $\mathbf{B}$, using the corresponding Dirichlet conditions at nodes lying on the free surface,
- Invert the global problem to obtain the potential everywhere

Recently, Ma *et al.* (2017) pointed out that the HPC method efficiency (in terms of accuracy and convergence rate) is greatly improved when a fixed mesh of perfectly-squared cells is used. In this work, the negative effects of a deforming mesh outlined in the previous subsection were also encountered. Especially, for some particular cell shapes, a high increase of the local condition number was observed, leading to difficulty of matrix inversion and important errors on the approximated potential. As a consequence, results were highly dependent on the mesh deformation method employed, especially in the vicinity of a fixed fully-immersed body.

2.3.4.2 Immersed free surface approach

In order to work with regular fixed grids, an IBM technique was developed and implemented to describe the free surface dynamics. Hanssen *et al.* (2015) introduced a first version of this method applied on the boundaries of a moving body. This method was

recently extended to the free surface and compared to a fitted MG method by Ma *et al.* (2017) and Hanssen *et al.* (2017a). In the current work, a semi-Lagrangian IB method introduced by Hanssen *et al.* (2017a) is chosen.

In this method, the free surface is discretized with markers, evenly spaced and positioned at each vertical intersection with the background fixed grid, as shown in figure (2.2). Those markers are semi-Lagrangian in such a way that they are only allowed to move vertically, following equations (2.8-2.9).

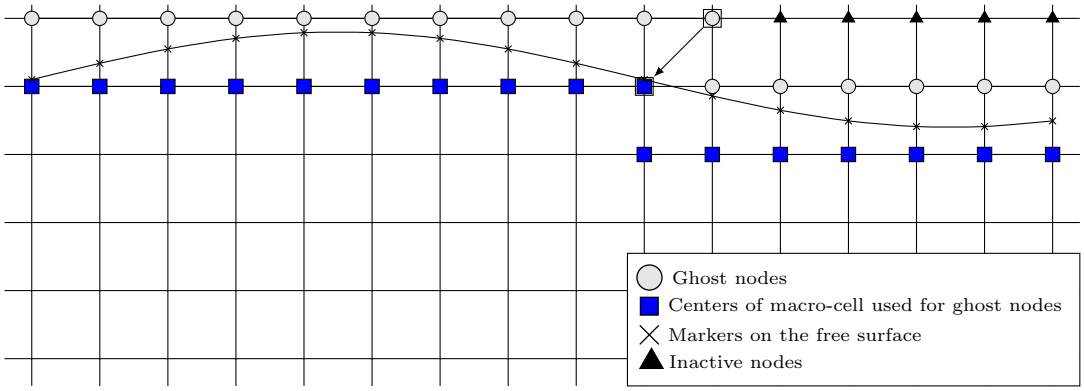


Figure 2.2: Schematic representation of the immersed free surface in a fixed grid

At a given time, every node located below the free surface (*i.e.* below a marker) is considered as a node in the fluid domain ("fluid" node), and defines a macro-cell with its 8 neighbors. The global matrix is classically filled with the local expression (2.13) at those nodes. As a consequence, in order to close the system, each neighbor of a node just above the free surface must also have a dedicated equation in the global matrix. These neighbors, represented with grey circles on figure (2.2), are denoted as "ghost" nodes. The chosen equation to close the system at a node of this type is the local expression (2.13) applied at the marker position in a given macro-cell:

$$\phi_m = \sum_{i=1}^{8} \left(\sum_{j=1}^{8} C_{ji}^{-1} f_j(\bar{\mathbf{x}}_m) \right) \phi_i \tag{2.18}$$

where $\bar{\mathbf{x}}_m = (x_m, \eta(x_m)) - \mathbf{x}_c$ is the position of the marker in the macro-cell's reference frame ($\mathbf{x}_c$ is the global position of the center node of the chosen macro-cell) and ϕ_m its potential (known at this stage). This ensures that the potential at free surface point is equal to the potential at the position of the marker from the interpolation equation. In other words, if one wants to interpolate the computed field ϕ at the particular location of the marker $\mathbf{x}_m$, the results should be consistent and yield the potential ϕ_m.

Note that this equation (2.18) is cell dependent (through C_{ji}^{-1}, the involved ϕ_i and the position of the center node $\mathbf{x}_c$) , but also depends on the chosen marker (trough $\mathbf{x}_m$ and ϕ_m). The only mathematical restriction on the choice the macro-cell to consider is that the ghost point potential should intervene as one of the ϕ_i in order to impose the needed constraint at this point.

An important note is that the later equation (2.18) is not dependent on the ghost point in any fashion. This implies that if the same couple (marker, macro-cell) is chosen

to close the system at two different ghost points, the global matrix will have to strictly identical rows. Its inversion would thus not be possible Particularly, two vertically aligned ghost points cannot use the same macro-cell equation at the same marker position. Here stands the differences between the IB method of Hanssen *et al.* (2017b), Ma *et al.* (2017) and the one chosen by Zhu *et al.* (2017). Zhu *et al.* (2017) decided to only impose the marker potential once in the first layer (or two first layers) and to constraint the upper potentials to an arbitrary value (in practice if the point is not used directly, the potential is set to the first point below which potential is used). The method used during this work is closer to the one by Hanssen *et al.* (2017b) and Ma *et al.* (2017): if a node needs a constraint but does not have a maker directly underneath (case of two ghost points vertically aligned), the ghost point on the top should invoke the local expression of the macro-cell centered on the closest fluid point instead of the cell centered on the vertically aligned fluid point (case indicated by an arrow in figure 2.2). With that method, in such a situation, the potential of vertically aligned marker is imposed twice in two different adjacent macro-cells. A comparison between those two methods had not been conducted and would be of great interest.

Whatever IB the method, the main goal is achieved: it is not needed to deform the mesh in time. As a consequence, the computation and inversion of the local (geometric) matrices is only done once, at the beginning of the computation. However, a step of identification of the type of each node, which was proven to be time consuming, is needed instead. Note that this identification algorithm could be greatly improved and is relatively slow in its current implementation. The general algorithm at one time step becomes:

- Identify nodes inside the fluid domain,
- Identify ghost nodes needed to close the system, associated markers and macro-cells,
- Fill global matrix $\mathbb{A}$ and RHS (B),
- Invert global problem to obtain the potential everywhere,

2.3.5 Linear solver and advance in time

To solve the global linear sparse system of equations, an iterative GMRES solver, based on Arnoldi inversion, was used for all computations. The base solver was developed by Saad (2003) for sparse matrix (SPARSEKIT library), and includes an incomplete LU factorization preconditionner. During this work, a modified version was implemented with the improvement proposed by Baker *et al.* (2009). Except for staling during the study of a standing wave at very long time, this solver was proven to be robust. Improvements of the construction step of the global matrix could further made in order to increase the efficiency of its inversion. Also, the initial guess in the GMRES solver could also be improved taking advantages of the already computed potential values. The number of inner iterations of the GMRES algorithm was chosen as $m \in [30, 60]$ and the iterative solution is considered converged when the residual is lower that 5.10^{-9}.

Marching in time thanks to equations (2.8-2.9) yields the free surface elevation and the free surface potential at the next time step. Note that the steps of computing the RHS terms of these equations are straightforward for most terms directly from the local expression (2.13) of the closest macro-cell. In addition, the spatial derivative of η is computed with a finite difference method. A centered scheme of order 4 is chosen for this work with the objective to maintain the theoretical order 4 of spatial convergence

provided by the HPC method.

In order to integrate equations (2.8-2.9), the classical four-step explicit Runge-Kutta method of order 4 (RK4) was selected as time-marching algorithm. During a given simulation, the time step (δt) was chosen to remain constant. Its value is made nondimensional by considering the Courant-Friedrichs-Lewy (CFL) number C_o based on the phase velocity $C = \lambda/T$, where λ is the wavelength and T the wave period:

$$C_o = \frac{C\delta t}{\delta x} = \frac{\lambda/\delta x}{T/\delta t} \tag{2.19}$$

The CFL number thus corresponds to the ratio of the number of spatial grid-steps per wavelength ($N_x = \lambda/\delta x$) divided by the number of time-steps per wave period ($N_t = T/\delta t$), *i.e.* $C_o = N_x/N_t$.

2.3.6 Computation of the time derivative of the potential

The pressure inside the fluid domain is obtained from the Bernoulli equation:

$$p(x,z,t) = -\rho\left(\frac{\partial\phi}{\partial t} + \frac{1}{2}(\nabla\phi)^2 + gz\right) \tag{2.20}$$

Any potential based NWT needs to solve this equation as the loads applied on the tested bodies are most of the time of prime importance. In a first attempt, the time derivative of the potential was estimated using a backward finite difference scheme. However, this method is not well suited when important variations of the potential are at play. Moreover, in the case of the IB method, it is not possible to obtain the value of the pressure at a point that was previously above the free surface, and thus for which a time derivative of the potential cannot be computed by the finite difference scheme.

A fairly accurate method is to introduce the (Eulerian) time derivative of the potential as a new variable $\phi_t = \dfrac{\partial\phi}{\partial t}$ and to solve a similar BVP as described previously on this newly defined variable, noting that ϕ_t has to satisfy the same Laplace equation as ϕ in the fluid domain. This method has been used by *e.g.* Guerber (2011) in the BEM framework or by Ma *et al.* (2017) in the HPC method.

Note that the local macro-cell matrices and coefficients, which are only geometrically dependent, do not change. In the different expressions presented above that are used to fill the global matrix, the coefficients linking the different potentials are not time dependent. Thus, the matrix to invert is exactly the same for the ϕ_t field and the ϕ field. However, the constant boundary conditions (and thus the RHS) may change. This is the case only when the RHS is different from zero, as for example, for free surface related closure points.

Remember that the equations at a (non-moving) Neumann condition and at a point inside the fluid domain yield a zero value in the RHS, and thus the equations at those points are exactly the same for the potential variable and for its derivative. At a (non-moving) Dirichlet boundary condition, one would simply impose $\phi_t = 0$ instead of $\phi = \phi_D$. At the IB ghost points, the ϕ_t is imposed to match the derivative of the potential with respect to time, known at the marker positions from equations (2.8-2.9).

Even tough the global matrices are exactly the same, the RHS being different and the chosen resolution method being iterative (GMRES solver), the easiest way is just to

solve twice the almost same problem. A more clever way could maybe be investigated by taking advantage of the previous inversion, but this is left for future work.

2.4 Introduction of a fixed body in the NWT

2.4.1 A double mesh strategy to adapt the resolution in the vicinity of a body

The HPC method applied to nonlinear waves, and particularly the immersed free-surface strategy will be validated and a convergence study will be conducted in the next chapter section 3.2. Here, we present the next objective which is to include a body in the fluid domain, either fully submerged or floating. Obviously, a desirable solution would retain cells of square shape and constant geometry as much as possible, even in the case of a moving body (not treated here however).

Once again, different strategies are possible. Hanssen *et al.* (2015) first introduced an IB method for bodies in waves in the HPC framework. Ma *et al.* (2017) compared this method with an immersed overlapping grid fitted to the boundaries (corresponding to the body in our case). This newly introduced grid will often be referred to as the "fitted mesh" for simplicity. In the current study, the later strategy is chosen. Two main reasons led to this choice: first, an oscillatory behavior was exhibited by Ma *et al.* (2017, Figs. 24, 25) when studying the spatial convergence with an IBM. This oscillatory behavior is also present when the body is moving, with a large magnitude. This is mainly due to an incremental change in the chosen ghost nodes which can turn to be favorable at some time steps (*i.e.* for certain grid configurations) and unfavorable at some other ones. Moreover, this oscillatory behavior does not come with a reduction of the error, both for the fixed and oscillating body.

The second reason is that adding a new fitted mesh allows to decouple the discretization of the wave propagation part (usually defined with respect to the wavelength, *e.g.* approximately $N_x \in [40 - 90]$ as shown in the previous section) from the discretization appropriate for the resolution of the potential close to the body (usually defined with respect to the body characteristic dimension, denoted D). Thus, by using two different grid, a suitable discretization for both the wavelength and the computation of the loads would be possible. For instance, including a small body relative to the incoming wavelength would be challenging with the IBM: too many nodes would be required so as to correctly solve the BVP in the vicinity of the body whereas less nodes would be needed further away. From a quantitative point of view, the case inspired from Chaplin (1984) and treated in section 3.3.1 hereafter involves an important ratio $\lambda/D \approx 15$. Thus, if the far-field discretization is set as $N_x = 90$ to correctly capture the propagation of the waves, then D is discretized with only 6 nodes. That high ratio λ/D, often encountered in practical engineering applications, is more easily taken into account with a second grid fitting the body than with a choice of higher order cell or with a local refinement of the grid. Note that the solution combining both a secondary grid and a solid immersed boundary has not yet been tested and would be of great interest. On this subject Ma *et al.* (2017, § 4.2.2) applied this combination to model the free surface and obtained an important reduction of the resulting error.

2.4.2 The two-way communication inside the fluid domain

Thus, a boundary fitted grid (BFG) is added locally around the body, overlapping the background grid (BGG). These grids and the points associated to the method are represented on figure (2.3). The Laplace problem is solved on both these grids simultaneously, *i.e.* both domains are solved in the same global matrix problem. The global matrix size is increased by the number of nodes of the BFG and decreased by the number of nodes inactivated in the BGG. One can note that the global matrix is thus almost defined by block, each corresponding to a grid.

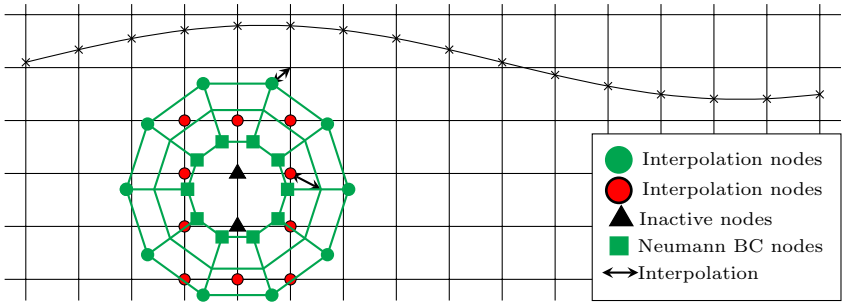


Figure 2.3: Schematic representation of the immersed free surface and body below the free surface. A circle means an "interpolation" node (described in the text) while the colors are used to identify particular nodes on the two grids, as indicated in the legend.

Thus, the boundary nodes of the new BFG also need a dedicated equation in order for the system to be closed. For a node laying on the body boundary, a simple Neumann boundary condition is set and enforced in the global matrix, as described in section 2.3.3. For an "interpolation node" P_f (green circle markers on figure 2.3) located on the outer contour of the BFG, the imposed equation in the global matrix is the interpolation equation from the closest macro-cell in the BGG (equation 2.13). On figure (2.3), a double arrow gives a representative example of the link between P_f and the center of the closest macro-cell in the BGG. This ensures - in an implicit manner - that the potential at the location $\mathbf{x}_{P_f}$ is the same in both meshes:

$$\phi_{P_f}^{(f)} = \phi^{(bg)}(\mathbf{x}_{P_f}) \tag{2.21}$$

where $\phi_{P_f}^{(f)}$ is the potential of the particular node P_f (directly an unknown of our system of equations), and $\phi^{(bg)}(\mathbf{x}_{P_f})$ represents the value of the interpolation equation from the background potential field at the given coordinate $\mathbf{x}_{P_f}$. Further developing equation (2.21) and using the closest macro-cell local expression (2.13) yield an implicit interpolation equation:

$$\phi_{P_f}^{(f)} = \sum_{i=1}^{8} \left(\sum_{j=1}^{8} C_{ji}^{-1} f_j(\bar{\mathbf{x}}_{P_f}) \right) \phi_{P_i} \Big|_{(bgc)} \tag{2.22}$$

where the notation $|_{(bgc)}$ emphasis that the local expression is applied on the closest background macro-cell. ϕ_{P_i} are the potential of the bounding nodes of this cell. Only for those interpolation nodes, the part of the matrix corresponding to the BFG is not

defined by block: the potential of a node P_f in the fitted grid is implicitly linked with the potentials of the neighboring background nodes.

Points belonging to the BGG situated inside the body are inactivated (black triangles on figure (2.3). Thus, equations are needed for the points of the BGG surrounding those inactive points (red circle markers on figure 2.3). The same method is used here: the interpolation equation 2.13 is enforced such that the interpolation of the fitted grid potential matches the node potential. In other words, the interpolation is effective from the BFG to the BGG, using the LE of the closest cell of the fitted mesh. Thus, denoting this point P_{bg} and its coordinates $\mathbf{x}_{bg}$:

$$\phi^{(bg)}_{P_{bg}} = \phi^{(f)}(\mathbf{x}_{P_{bg}}) \tag{2.23}$$

Here again, this relation is represented for one particular node on figure (2.3) by a double arrow.

So, by considering the various type of nodes discussed above, the proposed method ensures a consistent implicit two-way communication between the two meshes of interest, as the BVP problems (on ϕ and ϕ_t) are solved on both grids simultaneously.

2.4.3 Free surface piercing body

At this stage, a two-way communication is ensured between the fitted grid (BFG) and the background grid (BGG). The problem is closed in the sense that every node involved in the global matrix has its own dedicated equation. Still, a difficulty arises when the fitted mesh pierces the free surface. Indeed, it is not possible to interpolate outer points of the fitted mesh where no solution is computed (above the free surface): This issue is solved by introducing a new free surface, evolving in the BFG. This method is a variation of the presented technique in Tong *et al.* (2019) applied on a piston-type wave maker. This allows to solve and advance the free surface locally at the scale of the body. In this work, having a dedicated discretization in the vicinity of the body was proven to be necessary, when for example the reflection on the body resulted in waves of short wavelength and large steepness. The free surface evolving in the background grid is truncated such that no marker is defined inside the body (*i.e.* markers are only present in the fluid domain) as can be seen in figure (2.4)

For simplification purposes, the new free surface evolving in the BFG will be called "fitted free surface" even tough this free surface also uses the IBM described in section 2.3.4.2. Thus, we obtain two free surfaces, with different resolutions in space, following their respective grid discretizations, that overlap each other in the vicinity of the body. To ensure the communication between both free surface curves, the outer nodes positions and values of variables ϕ and ϕ_t of one free surface are interpolated and enforced trough a 1D B-spline interpolation from the other free surface. Referring to the schematic representation (2.4), the position and values of the outer right node of the background free surface is enforced so as to match the fitted free surface. Reciprocally, the position and values of the outer left node of the fitted free surface is enforced so as to match the background free surface.

However, if this enforcement affects only one marker at the extremity, instabilities may occur. For example, a stencil of two points on each side is the minimal length to maintain a 4^{th} order of spatial convergence with a 1D centered finite difference scheme. To prevent this from having an important impact, relaxation zones are set to incrementally match

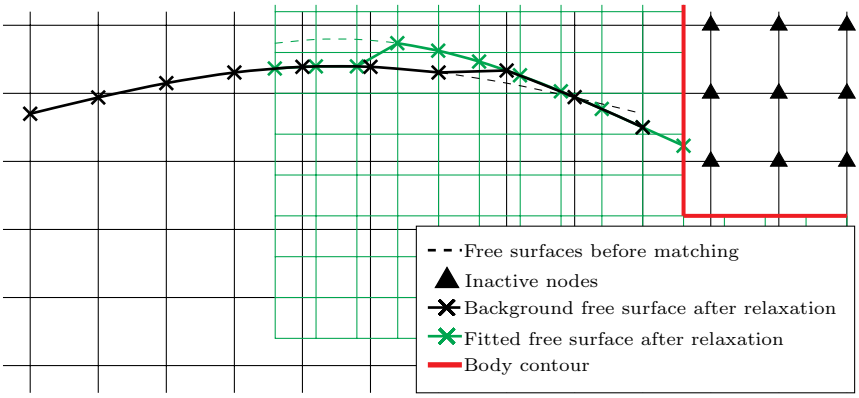


Figure 2.4: Schematic representation of the background free surface and the fitted free surface. A matching is enforced between the two free surfaces. The difference between the two free surfaces is here exaggerated for clarity.

the free surfaces at their extremities. Relaxation formulas and weights are thus needed for every marker:

$$\gamma_e = (1 - \alpha)\gamma_i + \alpha\gamma_t \tag{2.24}$$

where γ_e represents the value to enforce, γ_i the initial marker value, γ_t the target value (interpolated value from the other free surface) and α an arbitrary weight function of the marker position that evolves between 0 and 1. Note that for this application, γ stands for either ϕ, ϕ_t or η. Many different functions for α were tested and implemented without significant impact. In practice free surfaces are completely matched over a given length (*i.e.* $\alpha = 1$ if the marker distance to the free surface extremity is lower than a certain threshold, $\alpha = 0$ otherwise). The figure (2.4) emphasizes the effect of such relaxation functions: it shows the free surfaces before (dotted lines) and after the matching (solid lines) using this method. Note that if the body is not at the extremity of the computational domain, a second fitted free surface is needed on the other side of the body. This will be used in section 3.3.2. From a numerical point of view, the fitted mesh is considered as an unstructured grid. At a price of additional coding efforts and an increase of CPU time when identifying points (as well as an increase in the memory usage), a gain is made on the simplicity of inclusion of complex bodies of arbitrary shape. However, as already stated, the HPC method implemented requires square cells to be most effective (Ma *et al.*, 2017). Taking advantages of the fact that the BGG will remain a structured mono block, it would thus be possible to modify the methods on this grid to reduce both the RAM requirement and the necessary CPU time associated with identification of node types and interpolations between the resolutions of the BVP themselves.

2.5 Treatment of a sharp corners

In a potential framework, on a wall boundary condition is classically imposed with a non-penetration: the velocity normal to the wall is enforced as null. Numerically, this means that a null Neumann condition which vector is the surface normal is enforced on the velocity potential ϕ. When dealing with sharp corners, the problem of choosing the wall normal arises, as classically, only one equation can be set at a particular point.

Within the HPC framework, a first approach was suggested by Liang *et al.* (2015). They apply a domain decomposition method is the extreme vicinity of the apex of the corner: a local corner-flow solution is coupled with the HPC problem.

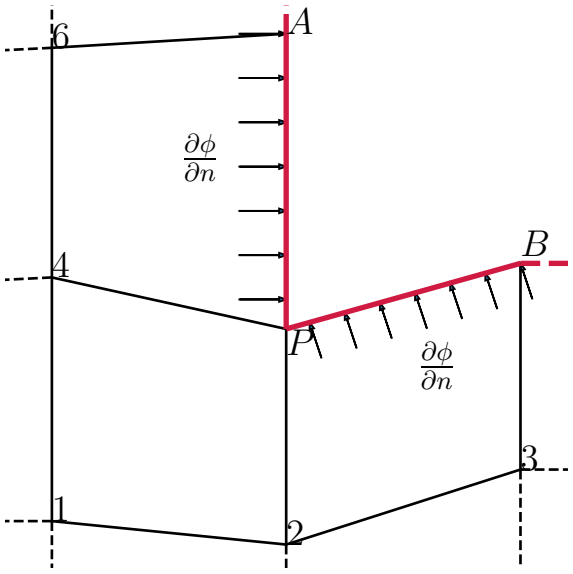


Figure 2.5: Schematic representation of a sharp coner (body in red)

Another method, suggested by Zhu *et al.* (2017), was to nullify the total flux accross the two closest boundary segments. They compared multiple versions of the application of this flux method, and their respective convergence properties. The different compared approaches are:

- The original HPC method.
- Overlapped: flux method applied everywhere, where the sum of the flux over the two neighboring faces are nullified.
- Non-overlapped: flux method applied everywhere, where the sum of the flux over the *half* of two neighboring faces are nullified. Two versions of this one where tested which differ in the treatment of the intersection with a Dirichlet condition.
- Corner-only: the flux method is only applied at sharp corners (in their case, the lower corners of the studied domain).

They found the same convergence rate whatever the applied approach. However, the corner-only method exhibited a lower overall error.

Thus, this method was selected and implemented in the current version. Currently, the user is expected to manually enter the corners that needs to be treated with this method. Note that in the next chapter, every body corners will be described using this method.

However, tank corners will not make use of this. It was not thought to be necessary in cases involving wave propagation, as relaxation zones will impose the selected wave theory over a given length and thus reduce the error associated with the domain external corners. It would, however, be interesting to apply on the standing wave case.

The objective is to set an implicit relation between our unknowns (potential at fluid points) such that the flux F_{APB} over the considered length ($A-P-B$, see fig. 2.5) is null. Lets denote s the curvi-linear scaled abscissa in a given segment **CD**, such that a position on this segment is given by $\mathbf{x}(s) = \mathbf{x}_C + s(\mathbf{x}_D - \mathbf{x}_C)$, where $\mathbf{x}_{C,D}$ represents the 2D coordinates of C and D respectively. The total flux can be derived as:

$$\begin{aligned} F_{APB} &= 0 \\ &= F_{AP} + F_{BP} \\ &= l_{AP} \int_{AP} \frac{\partial \phi}{\partial n}(\mathbf{x}(s))ds + l_{PB} \int_{PB} \frac{\partial \phi}{\partial n}(\mathbf{x}(s))ds \end{aligned} \tag{2.25}$$

where l_{AP} and l_{PB} are the length of the segments **AP** and **PB** respectively.

Let's recall that our local expression was derived - eq. (2.17) - to yield, at any location, the derivative of ϕ along a given normal $\mathbf{n}$. We here restrict the analysis to straight boundary segments, it is thus assumed that $\mathbf{n}$ does not vary during the integration over a given segment. The flux across the line **PA** - selecting the nearest macro-cell of A for the local expression (*i.e.* macro-cell centered by the node "6" in fig. 2.5) - can be written as:

$$\frac{F_{PA}}{l_{PA}} = \int_{PA} (\nabla\phi(\mathbf{x}(s)) \cdot \mathbf{n})\, ds = \sum_{i=1}^{8} \left(\sum_{j=1}^{8} C_{ji}^{-1} \underbrace{\left[\int_{PA} \nabla f_j(\bar{\mathbf{x}}(s))ds \right]}_{\mathbf{I}_j} \cdot \mathbf{n} \right) \phi_i \tag{2.26}$$

The inner integral ($\mathbf{I}_j$) of the above expression (eq. (2.26)) is separated into its spatial components.

$$\mathbf{I}_j = \int_{PA} \nabla f_j(\bar{\mathbf{x}}_k)ds = (\underbrace{\int_{PA} \nabla f_j(\bar{\mathbf{x}}(s)) \cdot \mathbf{e}_x ds}_{I_{xj}})\mathbf{e}_x + (\underbrace{\int_{PA} \nabla f_j(\bar{\mathbf{x}}(s)) \cdot \mathbf{e}_z ds}_{I_{zj}})\mathbf{e}_z \tag{2.27}$$

where $(\mathbf{e}_x, \mathbf{e}_z)$ are the two spatial unit vectors.

Given the expressions of the polynomials f_j (section 2.3.1 but also recalled in table 2.1), it is possible to obtain the analytical form of the different terms in eq. (2.27). The integration is done over s, by using $\bar{\mathbf{x}}(s) = (x_P + s\Delta x, z_P + s\Delta z)$, where $\bar{\mathbf{x}}_P = (x_P, z_P)$ are the coordinates of P relative to the considered macro-cell center, and $\mathbf{PA} = (\Delta x, \Delta z)$. The exemple of analytical expressions of I_{xj} are given in table 2.1, and the same procedure can be repeated over the vertical axis. Note that however, the expression were not implemented directly in the form presented in the table. Rather, a function $i_{a,b}$ was defined:

$$i_{a,b}(x_P, z_P, \Delta x, \Delta z) = \int_{s=0}^{s=1} (x_P + s\Delta x)^a (z_P + s\Delta z)^b ds \tag{2.28}$$

which was analytically integrated for every whole number (a, b) such that $a + b \leq 3$.

Remark. *The same procedure can be applied in the computation of the integration of the pressure on a wall segment: using the Bernoulli equation and integrating the derivative of*

j	f_j	$\frac{\partial f_j}{\partial x}$	I_{xj}
1	1	0	0
2	x	1	1
3	z	0	0
4	$x^2 - z^2$	$2x$	$2(\Delta x/2 + x_A)$
5	xz	z	$\Delta z/2 + z_A$
6	$x^3 - 3xz^2$	$3x^2 - 3z^2$	$\Delta x^2 + 3\Delta x x_P + 3x_P^2 + \Delta z^2 + 3\Delta z z_P + 3z_P^2$
7	$3x^2z - z^3$	$6xz$	$6(1./3\Delta x\Delta z + 1./2.(\Delta x z_P + \Delta z x_P) + z_P x_P)$
8	$x^4 - 6x^2z^2 + z^4$	$4x^3 - 12xz^2$	$4(\Delta x^3/4 + 3\Delta x^2 x_P/3 + 3\Delta x x_P^2/2 + x_P^3$ $+12(\Delta x\Delta z^2/4. + (\Delta z^2 x_P + 2\Delta x\Delta z z_P)/3.$ $+(2\Delta z x_P z_P + \Delta x z_P^2)/2. + x_P z_P^2)$

Table 2.1: Analytical expressions of the integral over a straight line

*the potential with respect to time. More analytical expressions of $i_{a,b}$ should be calculated however (*i.e. *for more a, b), due to the presence of the velocity square in the integrand.*

Once the different integrals are computed ($\mathbf{I}_j = (I_{jx}, I_{jz})$) for the segment **PA** in the selected macro-cell, it is possible to derive an implicit expression for the flux:

$$F_{PA} = \sum_{i=1}^{8} \left(\sum_{j=1}^{8} C_{ji}^{-1} l_{AP} \mathbf{I}_j \cdot \mathbf{n} \right) \phi_i \tag{2.29}$$

The exact same procedure can be repeated to obtain the flux across **PB**, for which a different macro-cell can be used. A null total flux eq. (2.25) is enforced in the global matrix $\mathbb{A}$. Note that it means that two macro-cells potentials might be linked by this equation, leading to an associated matrix row having up to 16 non null terms. In practice, for instance in the situation depicted in fig. 2.5, the selected macro-cells (centered by "6" and "3") will share some nodes (only "P" here), leading to a maximum of 15 non zero coefficients.

Note that no specific study evaluating the influence of such approach will be conducted in this work. However, a validation of the obtained flux was conducted by computing *a posteriori*, by finite difference, the obtained fluxes on the two segments, to verify their canceling. In practice, both fluxes were consistently of small amplitude, and the results proved to be more reliable and stable with this approach than using the classical singular Neumann node approach.

Chapter 3

Results and discussion about the HPC model

Contents

3.1 Introduction 36
3.2 Validation and convergence study on a nonlinear standing wave . . . 37
3.3 Validation and limits of wave-body interaction against flume experiments 43
3.4 Fitted mesh method: results and limitations 59
3.5 Summary of conclusions 65

3.1 Introduction

In chapter 2, the HPC method was presented, along with various associated approaches. Namely, we described the employed IBM to capture and simulate the evolution of the free surface, the body fitted grid overlapping the background mesh, but also the treatment of a sharp body corner. Following this presentation, this chapter aims to validate the obtained numerical wave tank on various test cases.

In section 3.2, a convergence study is performed on a highly nonlinear freely evolving standing wave case compared to an accurate numerical solution from the stream function theory. Therefore, the main objective is to validate the correct resolution of the BVP, as well as the IBM employed for the free surface, and the associated DtN problem.

Afterwards, the double mesh strategy is evaluated on three selected cases in section 3.3. The first one (section 3.3.1) is a horizontal fixed circular cylinder, completely immersed although close from the free surface. Available numerical and experimental results on the literature will be used as references, in particular the experiments carried by Chaplin (1984). The second one (section 3.3.2) involves a free surface piercing body with sharp body corners. The obtained loads and surface elevation are compared against dedicated experimental results obtained during this work. During the experiment, effects laying outside of the potential theory (breaking, air-entrapment, *etc.*) were observed. Hence, a perfect comparison is not expected, and the test case purpose is more about finding the limits of our approach, but also about evaluating the use of the NWT outside its prescribed range. In order to analyze some of the present phenomena in a finer way, without some of the free surface strongly non-potential effects, a fully immersed horizontal cylinder of rectangular base is tested in section 3.3.3.

Then, in section 3.4, a side study will be conducted on the more basic way of treating the free surface movement: deforming the mesh so as to fit the latter and enforce its values as a simple Dirichlet condition. Main advantages and drawbacks will be discussed in there, as well as the reasons for its abandon.

Finally, in section 3.5, the main findings from this study will be summarized. Note that the possible future works and direct follow-ups of this method are mainly discussed in the general conclusion, in section 7.3.1.

3.2 Validation and convergence study on a nonlinear standing wave

3.2.1 Presentation of the test-case

The first case consists in simulating a nonlinear standing wave in a domain of uniform water depth h whose extent is equal to one wavelength λ. This case is actually challenging as the wave height H (difference between the maximum and minimum values of free surface elevation at anti-node locations) is fixed by choosing a large value of wave steepness $H/\lambda = 10\%$ (or $kH/2 = \pi/10 \approx 0.314$). We also choose to work in deep water conditions by selecting $h = \lambda = 64$ m (or $kh = 2\pi \approx 6.28$). The water domain at rest has thus a square shape in the (x, z) plane, as illustrated in figure (3.1). Initial elevations of free surface $\eta(x, t = 0)$ are computed from the numerical method proposed by Tsai and Jeng (1994). The initial phase is chosen such that the imposed potential field is null at $t = 0$ at any point in the water domain. This initial state corresponds to a maximum wave elevation at the beginning and the end of the domain ($x/\lambda = 0$ and 1), and a minimum wave elevation at the center point of the domain ($x/\lambda = 0.5$), these three locations being anti-nodes of the standing wave.

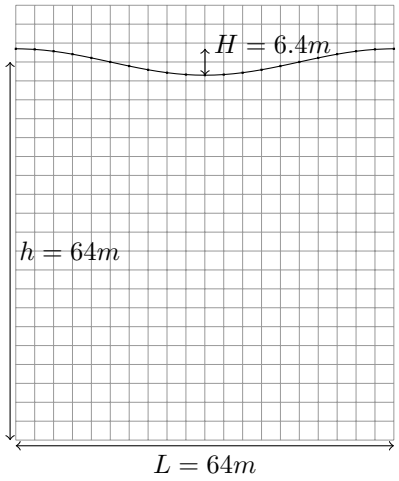


Figure 3.1: Schematic representation of the nonlinear standing wave with steepness $H/\lambda = 10\%$ at $t = 0$.

The wave is freely evolving under the effect of gravity: in theory one should observe a fully periodic motion without any damping as the viscosity is neglected. At each time step, a spatial $L_2(\eta)$ error on η is computed relative to the theoretical solution of Tsai and Jeng (1994) (denoted η^{th} hereafter), and normalized with the wave height:

$$L_2(\eta, t) = \frac{1}{H}\sqrt{\frac{1}{n_p}\sum_{i=1}^{n_p}\left(\eta(x_i, t) - \eta^{th}(x_i, t)\right)^2} \tag{3.1}$$

where i represents the index of a point on the free surface and n_p the total number of points on the free surface.

3.2.2 Evolution of $L_2(\eta)$ error with space and time discretizations

The result of this $L_2(\eta)$-error is represented as a color map at four different times $t/T =$ 1, 10, 50 and 100 in figure (3.2) as a function of the number of nodes per wavelength ($N_x = \lambda/\delta x$, where δx is the spatial step-size) and the CFL number C_o (defined in section 2.3.5). Wide ranges of the two discretization parameters are explored, namely $N_x \in [10, 90]$ and $C_o \in [0.05, 4.0]$. Simulations that ran till the end of the requested duration of $100T$ are represented with coloured squares. A circle is chosen as a marker when the computation breaks down before the end of that duration. Nonetheless the markers are colored if the computation did not yet diverged at the time instant shown on the corresponding panel.

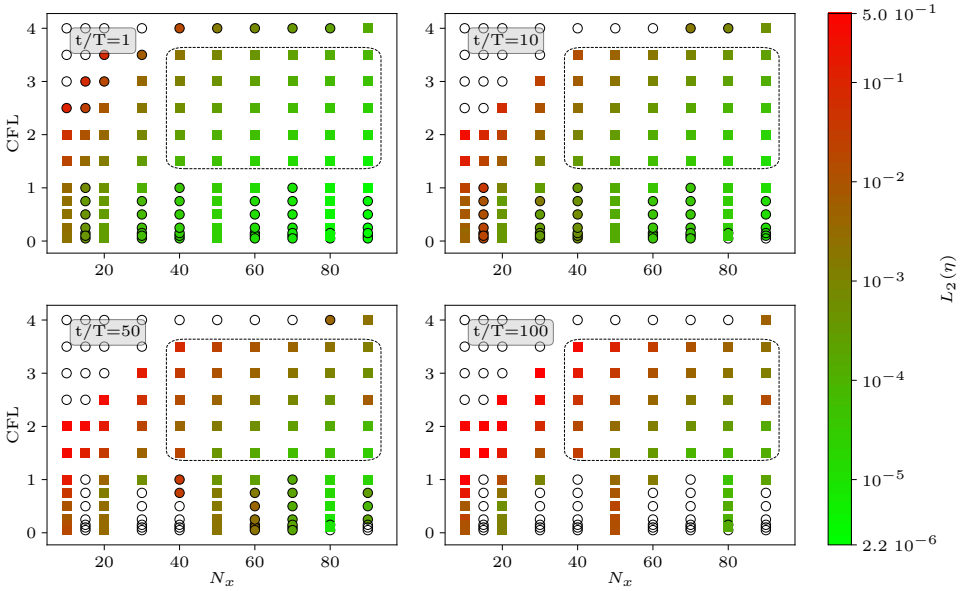


Figure 3.2: L_2 error on η on the nonlinear standing wave case at four time instants ($t/T = 1$, 10, 50 and 100) as a function of the spatial and temporal discretizations. The color scale indicates the $L_2(\eta)$ error respective to the theoretical solution by Tsai and Jeng (1994). See text for explanations on the significance of the markers shapes.

As seen on figure (3.2), a large number of simulations were completed over this rather long physical time of $100T$. Note that no filter were used along this work, so some of the numerical simulations tend to be unstable for extreme values of the discretization parameters. For instance, when $C_o \leq 1$, the computation is mostly unstable and breaks down: before $50T$ when N_x is small (*i.e.* below 40) and between $50T$ and $100T$ when N_x is larger. Note that $C_o = 1$ corresponds to a time step ranging from $N_t = T/\delta t = 10$ to 90, for $N_x = 10$ and 90 respectively. This value of $C_o = 1$, and associated time step, is the lower stable limit exhibited by these simulations.

On the other hand, when the C_o is too high (*i.e.* larger than 3.5) instabilities also occur almost at the beginning of the simulation ($t/T < 10$), particularly when N_x is small. For very small N_x (in the range 10-15) and whatever the C_o, the computation tends to be unstable. This is probably due to the discretization of the immersed free surface being the same as the discretization of the background mesh. A coarse discretization of the free

surface leads to an inaccurate computation of the spatial derivative of η: instabilities may then occur.

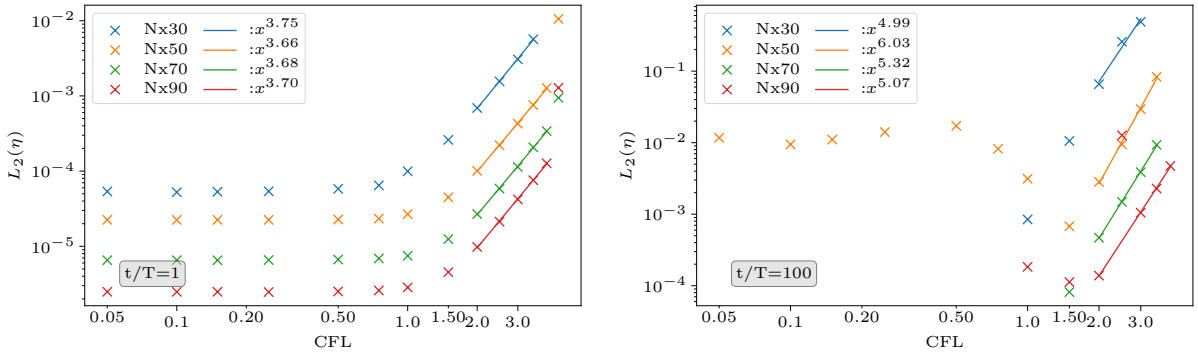
A suitable range of parameters is thus determined to avoid instabilities: $1.5 \leq C_o \leq 3.5$ and $40 \leq N_x \leq 90$. This zone is represented in figure (3.2) as a rectangular box with a dashed contour. In that zone, all the computations ran with the requested time step over a duration of $100T$. Note that the CPU cost scales with $N_x^2 N_t \sim N_x^3/C_o$ and thus the most expensive computations in this stable zone are approximately 25 times slower than the least expensive ones in the same zone. Also note that the lowest error is almost systematically reached in this zone. The $L_2(\eta)$ error is as small as 2.10^{-6} after $1T$. After $100T$, the lowest error is approximately 10^{-4}. Moreover, the evolution of the value of the error is qualitatively consistent with the mesh refinement and time refinement.

The stability was not assessed for finer mesh than $N_x = 90$ points per wavelength, due to increasing computation cost on one hand, and the fact that finer resolutions would lie out of the range of discretizations targeted for real-case applications. In addition, at long time, a discretization of $N_x = 90$ already exhibits behavior that does not match the expected convergence rate, as will be discussed hereafter in greater detail.

3.2.3 Convergence with time discretization

In order to study the convergence of the method in a more quantitative manner, the $L_2(\eta)$ error is shown as a function of the C_o number for different spatial discretizations N_x at $t/T = 1$ in figure (3.3a) and at $t/T = 100$ in figure (3.3b). A C_o^α regression line is computed and fitted on the linear convergence range of the log-log of the error. That will be called "linear range" for simplicity, though it correspond to an algebraic rate of convergence of the error. Note that this linear range corresponds exactly to the zone in which the computations remain stable (with the exception of one particular point at $t/T = 100$ and $N_x = 90$ excluded from the determination of the convergence rate). At $t/T = 1$, the minimum error is, as expected, obtained for small C_o numbers and large N_x: $L_2(\eta) \sim 10^{-5}$ in the linear range and the minimal error reached is 2.10^{-6} for the finer discretization $N_x = 90$. The algebraic order of convergence is close to 4. This was expected as the temporal scheme is the RK4 method at order 4. Moreover, at this early stage of the simulation the error decreases with a power 4 law only when $C_o \gtrsim 1.0$. The lowest errors are achieved at $C_o \approx 0.75$. Below that C_o number, a threshold is met: the error remains constant when the time step (and C_o number) is further decreased; it is then controlled by the spatial discretization. It is also possible to note that the CFL number C_o seems to be a relevant metric when testing the convergence of the method: the range of C_o in which the results converge is the same across the 4 considered spatial discretizations.

Remark. *Significant differences in terms of C_o with the work of Hanssen* et al. *(2017a) have to be stressed. In their simulations the chosen numbers of points per wavelength were similar to the ones used here ($N_x \in [15, 90]$), but the time step was constant and fixed at a small value of $\delta t/T = 1/N_t = 1/250$. This value yields a C_o between 0.06 and 0.36. This range of C_o was shown to be out of the domain of convergence in time in our case. For the same C_o (and apparently the same RK4 time scheme), the computation is indeed converged with respect to the time discretization and yields low error during the first periods, but instabilities then occur when the wave are freely evolving on a longer time scale.*



(a) $t/T = 1$ (b) $t/T = 100$

Figure 3.3: Convergence of the L_2 error on η (crosses) with respect to the temporal discretization at two different physical times: $t/T = 1$ (left panel) and $t/T = 100$ (right panel). The spatial discretization is fixed for a given line. Solid lines represent power regression of the error in the "linear range", the computed power is reported in the legend of the fitted straight lines.

Hanssen et al. *(2017a) also encountered instabilities with this IB method. To counteract these instabilities, they used a* 12^{th} *order Savitzky-Golay filter in order to suppress, or at least, attenuate them. No filtering nor smoothing was used in our simulations. This may explain the differences of behavior with Hanssen* et al. *(2017a) in terms of* C_o *number. Similarly, in Zhu* et al. *(2017), also with the RK4 time scheme, the time-step is chosen as* $\delta t/T = 1/200$ *for a spatial discretization of* $\delta x = h/10$*. Converted to our numerical case, this would correspond to* $N_x = 100$*, and so a* C_o *fixed at* 0.5*. On the contrary, the time-step chosen by Ma* et al. *(2017) to compute the potential flow around a rigid body in infinite fluid domain,* $\delta t/T = 1/40$*, is closer to the current range of time-steps. During their investigations on periodic wave propagation, their spatial discretizations ranged from* $N_x = 16$ *to* $N_x = 128$*. The equivalent* C_o *number is thus comprised between* 0.4 *and* 3.2*.*

At long time $t/T = 100$ (see figure 3.3b), the error behaves differently. First, the error is approximately one order of magnitude higher compared to the time $t/T = 1$, but the convergence rate is also slightly different, actually higher. As a matter of fact, at $t/T = 100$, an order 4 of convergence is still found on the wave period and on the amplitude of the computed wave: fig. 3.4 shows the evolution of the error on wave period and amplitude at the center of the domain $x/\lambda = 0.5$ at $t/T = 100$ for a fixed $N_x = 30$ as a function of the CFL number C_o. The reference case used here is the one with $N_x = 100$ at $C_o = 0.05$ computed on one period. On a given case, the period is computed trough the mean time separating two successive maximums, then a sliding Fast Fourier Transformation (FFT) is performed to obtained an accurate estimation of the amplitude of the free surface elevation.

However, the behavior of the $L_2(\eta)$ error results from a combined effect of both the error on the wave period and the error on the amplitude. The relative effect of those errors on the total error is analyzed in detail in A, and a brief summary is given here. Let e be the relative error between two cosine functions. The first is the target function and the second one tends to the first one in amplitude as $\epsilon_a = f_A d^4$ and in period as

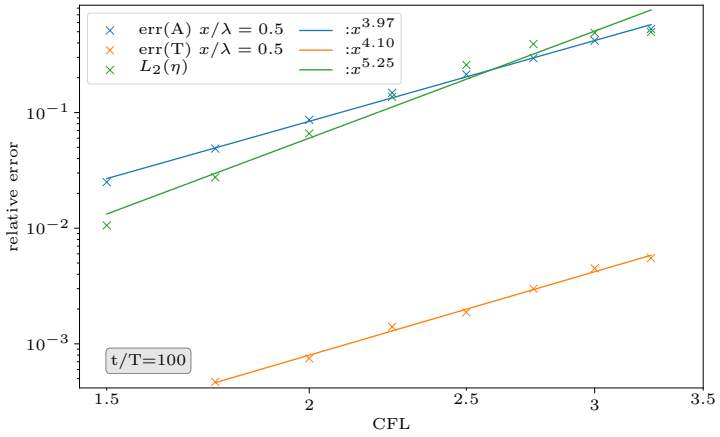


Figure 3.4: Convergence of the error on wave period and amplitude of the wave elevation at $x/\lambda = 0.5$ for $N_x = 30$. The $L_2(\eta)$ error -combination of both- is also added.

$\epsilon_t = f_T d^4$. Here d is a discretization variable -either C_o or $1/N_x$ in our case-, which drives the convergence. f_A and f_T are constants with respect to d. A Taylor expansion of e in the vicinity of a time t/T corresponding to a whole number of periods gives:

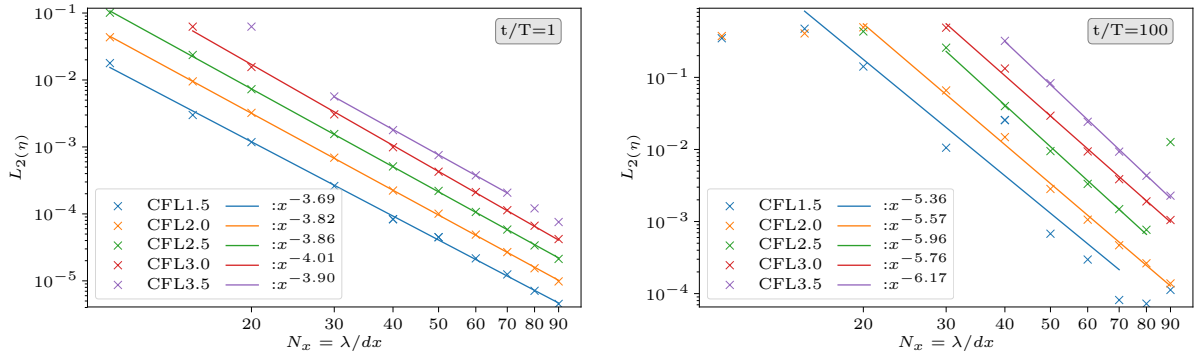
$$e = f_A d^4 + 2\pi^2 \frac{t^2}{T^2} f_T^2 d^8 + (2\pi \frac{t}{T} f_T^3 - 2 f_A \pi^2 \frac{t^2}{T^2} f_T^2) d^{12} + O(d^{16}) \tag{3.2}$$

Note that f_A depends on t/T because the error on amplitude increases with time (in practice a linear dependence was observed at long time, *i.e.* $f_A = \bar{f_A} t/T$). However they should not depend on the convergence parameter d. The order 8 of convergence should disappear for small enough d whatever t/T. In that case, the error on period is negligible compared to the error amplitude: this results from the presence of the cosine, which elevates the error to the power 2. However, if the error in amplitude f_A increases in time slower than $t^2 f_T^2$, there exists a time after which the error on the wave period will play a major role (order 8 will be predominant). This effect is thought to explain the seemingly high order of convergence of the $L_2(\eta)$ error in figure (3.3b). A shows detailed comparisons at $t/T = 100$ with values of f_A and f_T extracted from our results.

3.2.4 Convergence with spatial discretization

The convergence with spatial refinement (*i.e.* as a function of $N_x = \lambda/\delta x$) is analyzed in the same way and shown in figure (3.5). The order of convergence in space is again 4. Due to the choice of the set of HP including polynomials up to order 4, and the fact that finite difference scheme of order 4 are used to compute the derivatives of free surface variables, this order 4 was the expected order of convergence.

At long time the convergence rate exhibits the same behavior as shown in the convergence with time resolution. The latter comments concerning the long time evolution of the error still holds (with, here, $d \equiv \delta x$), and is still thought to explain the increasing order of convergence of the total error $L_2(\eta)$ with time. Of course the values of the corresponding constants f_A and f_T are different. Another effect also occurs: when the C_o number increases, so does the order of convergence. This small yet clear effect at time



(a) $t/T = 1$ (b) $t/T = 100$

Figure 3.5: Convergence of the L_2 error on η in mesh refinement, with temporal discretization fixed. Solid lines correspond to the power regression of the error.

$t/T = 100$ is not completely understood. It would mean that the value of f_T increases faster with the C_o number than the value of f_A does.

3.2.5 Summary of numerical convergence study

After a comprehensive numerical study of the HPC method on a challenging nonlinear standing wave case in deep water conditions, the efficiency and accuracy of the Immersed Boundary modeling of the free surface applied on a fixed underlying spatial mesh was demonstrated (again without using filtering nor smoothing of the free surface). Optimal ranges of spatial and temporal discretization parameters were determined:

- the spatial discretization δx should be chosen to have $N_x = \lambda/\delta x$ between 40 and 90 nodes per wavelength, which is a reasonable range of values for practical applications.
- the temporal discretization δt should be best selected to have a CFL number $1.5 \leq C_o \leq 3.5$, meaning that the number of time steps per period $N_t = T/\delta t$, also given by N_x/C_o, is then comprised between $N_x/3.5$ and $N_x/1.5$. This is highly beneficial as it authorizes rather large time-steps for practical applications.
- the present implementation of the HPC shows an algebraic convergence rate with the spatial resolution of order greater than 4.
- it also shows an algebraic convergence rate with the temporal resolution of order comprised between 4 and 5 (for long time simulations).

3.3 Validation and limits of wave-body interaction against flume experiments

In order to validate the method presented earlier (and in particular the body fitted overlapping grid method described in section 2.4), three experimental test cases were selected. The first one is chosen so as to verify the boundary-fitted overlapping grid method selected to include a fully immersed body: a fully submerged horizontal cylinder of circular cross-section against literature results. Afterwards (section 3.3.2), a free surface piercing case is selected: a rectangular barge. Finally, in section 3.3.3, a focus will be made on a rectangular shaped horizontal cylinder of small dimension, exhibiting sharp corners, in order to analyze some of the limits intrinsic to the summoned potential assumptions.

3.3.1 Fixed horizontal submerged cylinder

Chaplin (1984) studied in detail a fixed horizontal cylinder, with a low submergence below the SWL, in regular waves of period $T = 1$ s. Accurate experimental results about the nonlinearities of wave loads on the cylinder were given and are often used in order to validate NWTs (*e.g.* Guerber, 2011). The total water depth is $h = 0.85$ m which, together with the period, imposes a wavelength of approximately $\lambda = 1.56$ m (slightly varying with the wave height). The cylinder of diameter $D = 0.102$ m is immersed with its center located at $z_c = -D = -0.102$ m below the SWL.

This problem is numerically difficult to solve for volume field methods as the cylinder is close to the free surface, such that the fluid domain right above the cylinder is reduced to a small water gap of height $D/2 \approx \lambda/30$ (when the water is at rest). This water gap needs to be meshed and resolved with the HPC method. Thus, a spatial discretization of $\lambda/\delta x \in [40, 90]$ would yield a discretization of this gap with only $\sim 2 - 3$ nodes.

The nonlinear regular incident waves are generated using the so-called stream function theory (Fenton, 1988). To avoid reflection on both the inlet wave maker side and the outlet Neumann wall and to impose the target incident wave field at the inlet, relaxations zones are introduced to enforce the requested values over a distance chosen as $L_{relax} = \lambda$ trough a commonly used exponential weighting function (*e.g.* Jacobsen *et al.*, 2011). Note that other techniques of waves generation and absorption are possible. For example, Clamond *et al.* (2005) introduced a damping term in the Bernoulli equation in order to modify the DFSBC: the wave elevation is smoothly driven to the SWL. Figure (3.6) shows at scale the computational domain used for the numerical simulations including the two relaxation zones.

We focus our attention on the vertical force exerted on the cylinder once the periodic wave motion is established in the NWT. A Fourier analysis is applied to the computed times-series of vertical force. The normalized amplitudes of the harmonics of the vertical force and the mean vertical force (drift force) are plotted in Figure (3.7) as a function of the Keulegan-Carpenter number. On this figure, the results from the present NWT are compared with the experimental values from Chaplin (1984) and the numerical results from Guerber (2011). The linear theory results from Ogilvie (1963) are added for comparison of the amplitude of the first harmonic. The definition and discussions about the KC number are conducted in section 1.4.

All harmonics amplitudes up to third order are in relative good agreement with the

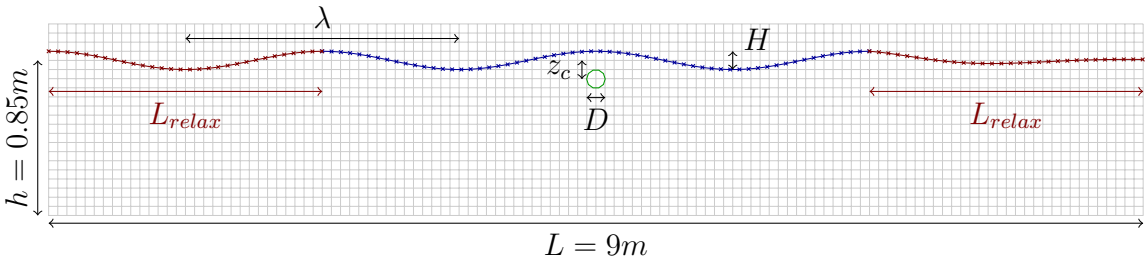


Figure 3.6: Schematic representation of the numerical set-up inspired from Chaplin (1984). Note that the mesh fitted to the cylinder is not represented in this figure.

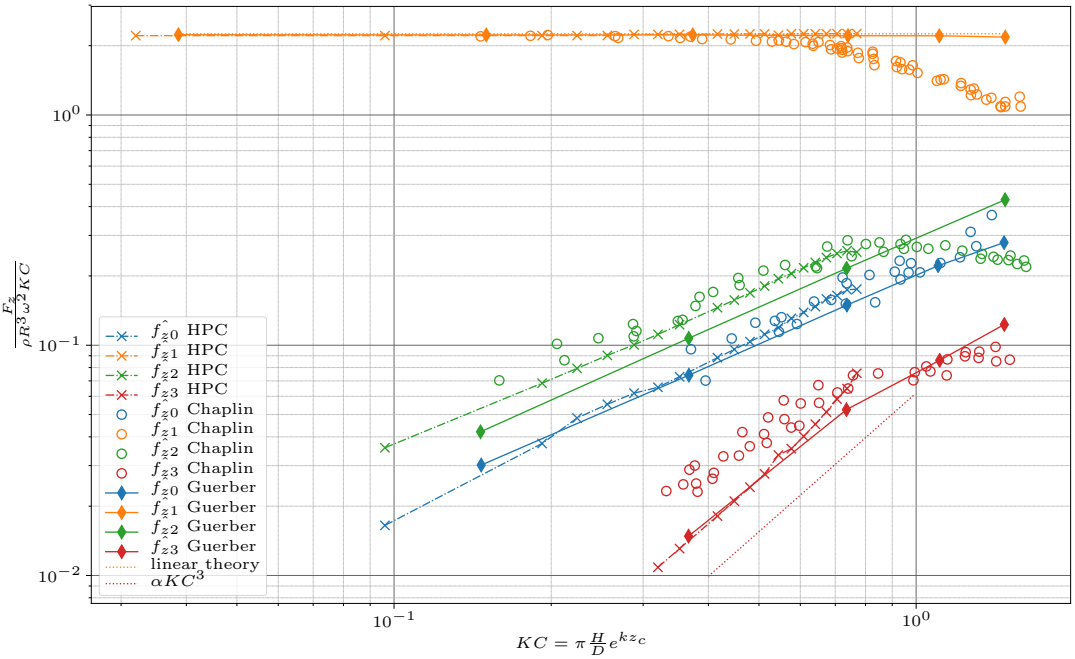


Figure 3.7: Amplitudes of the various harmonic components of the vertical load on the horizontal circular cylinder. Current results (crosses) compared to numerical simulations from Guerber (2011) (lines with diamonds) and experiments from Chaplin (1984) (empty circles). The amplitude of the first order harmonic based on linear prediction is added as well as a third order model line, to compare with the evolution of the amplitude of the third order harmonic.

BEM simulations from Guerber (2011), although some discrepancies can be denoted. The mean value and first order are difficult to distinguish between the two sets of numerical results, whereas the amplitude of the second order harmonics is closer from the experimental data with the current HPC method. For the third harmonic, which is of very small relative amplitude, is is difficult to identify the most accurate method.

Moreover, the behavior of the different harmonics amplitudes seems to agree with the expected theoretical results: an increase as KC^1 of the first amplitude of the harmonic (*i.e.* $\hat{f_{z1}}/KC$ is constant), as KC^2 of the drift force and second harmonic amplitude, and as KC^3 of the third harmonic amplitude. regarding the latter, the HPC method

reproduces more closely an order 3 in KC than both Guerber (2011) and the experimental results. Of course, limitations in the comparison with the experiments can be observed, in particular for larger wave heights. This is mainly due to the viscous effects (not considered here, nor in the simulations of Guerber (2011)) as this stalling is retrieved in the viscous computations of Tavassoli and Kim (2001).

However, a difficulty arises with the HPC method when KC increases (*i.e.* for larger incident wave heights). The Laplace equation is solved in the different cells inside the fluid volume. Thus, there should always be at least a fluid point in the volume above the cylinder top. In waves of large amplitude, the cylinder is very close to the free surface, in particular when a wave trough passes over the cylinder. In this situation, it is not possible to keep the number of point per wavelength N_x in the previously selected range $[40, 90]$ and keep square cells. Saw-tooth instabilities appear as KC exceeds 0.80 approximately. No filter were used in this work, as the main objective is to emphasis the limits of the method itself. Above a wave steepness of $H/\lambda = 2.6\%$, (*i.e.* $KC = 0.86$), no computation could remain stable after two or three wave periods.

As a conclusion, even if a limitation in wave height is met, those results, correct up to third order, give us confidence on a case which is particularly challenging for volume field methods.

3.3.2 Free surface piercing body, experiments and numerical comparison

In order to validate the HPC method with a (fixed) free surface piercing body and sharp corners, dedicated experiments were conducted in a wave flume at Centrale Marseille with a body of rectangular cross-section.

3.3.2.1 Experimental setup

The wave flume is 17 m long and 0.65 m wide. The water depth was set to $d = 0.509$ m. The body is a rectangular barge of draft 0.10 m for a length of 0.30 m (in the longitudinal direction of the flume) mounted on a 6-axis load cell measuring device. The width of the barge spans the width of the flume minus a small water gap of about 2 mm on both sides between the barge and the flume walls. A perforated metallic beach is placed at the end of the wave flume to dissipate the energy of the transmitted waves, and so to avoid reflection. The waves are generated with a flap type wave maker. The body is placed such that its front face is located at $x_b = 11.52$ m from the wave maker. 13 wave gauges are installed all along the wave flume. Unfortunately, the wave gauge dedicated to measure the run-up on the front face of the barge was found *a posteriori* to be defective. For some cases, a video of the experiment was recorded in the vicinity of the body, allowing to extract free surface profile and run-up at the front and rear faces of the barge.

Approximately 40 cases were tested in regular wave conditions, with varying wave period and height. A focus is made here on two periods: $T = 1.1$ s and $T = 1.5$ s. For the latter one, however, high wave steepness yielded high wave heights, resulting in dewetting and breaking, with a lot of turbulent effects, recirculations, and air entrainment. Thus, for this period, few relevant comparisons can be made with the potential model (which neglects all those effects).

Figure 3.8: Photograph of the experimental setup of the rectangular barge in the wave flume close to the body.

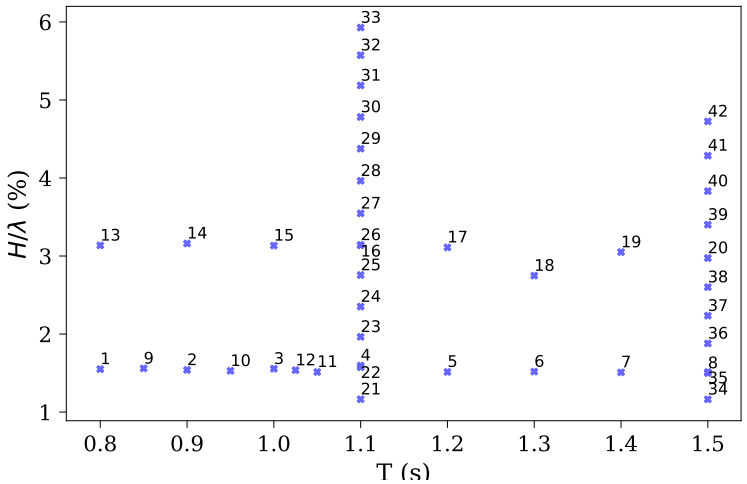


Figure 3.9: Overview of the conducted experiments in the $(T, H/\lambda)$ plane.

3.3.2.2 Numerical setup

The computational domain is automatically generated and mesh by the NWT depending on the case: its length is chosen as 8λ and relaxation zones are set in the inlet and outlet, with an exponential function over a length of 2λ. A second mesh is fitted to the body in order to ensure a precise computation of the flow dynamics in its vicinity. The mesh fitting the body is of breadth 0.30 m on each side (*i.e.* a total extent of 0.90 m). Both meshes communicate trough previously described interpolation boundaries. Their free surface curves communicate trough relaxation zones of length 0.14 m and constant function of unitary weight. In order to interpolate the free surface from the other one, a 1D B-spline interpolation is used. Different discretizations of the fitted mesh were tested, without significant difference.

3.3.2.3 Numerical results and comparisons with experiments

For the wave period $T = 1.1$ s, the steepest cases (31-33) were found to numerically break down. This is due to an important run down which leads to a dewetting at the bottom left corner of the barge. During the experiments, recirculations and turbulent effects were clearly visible on those steeper cases.

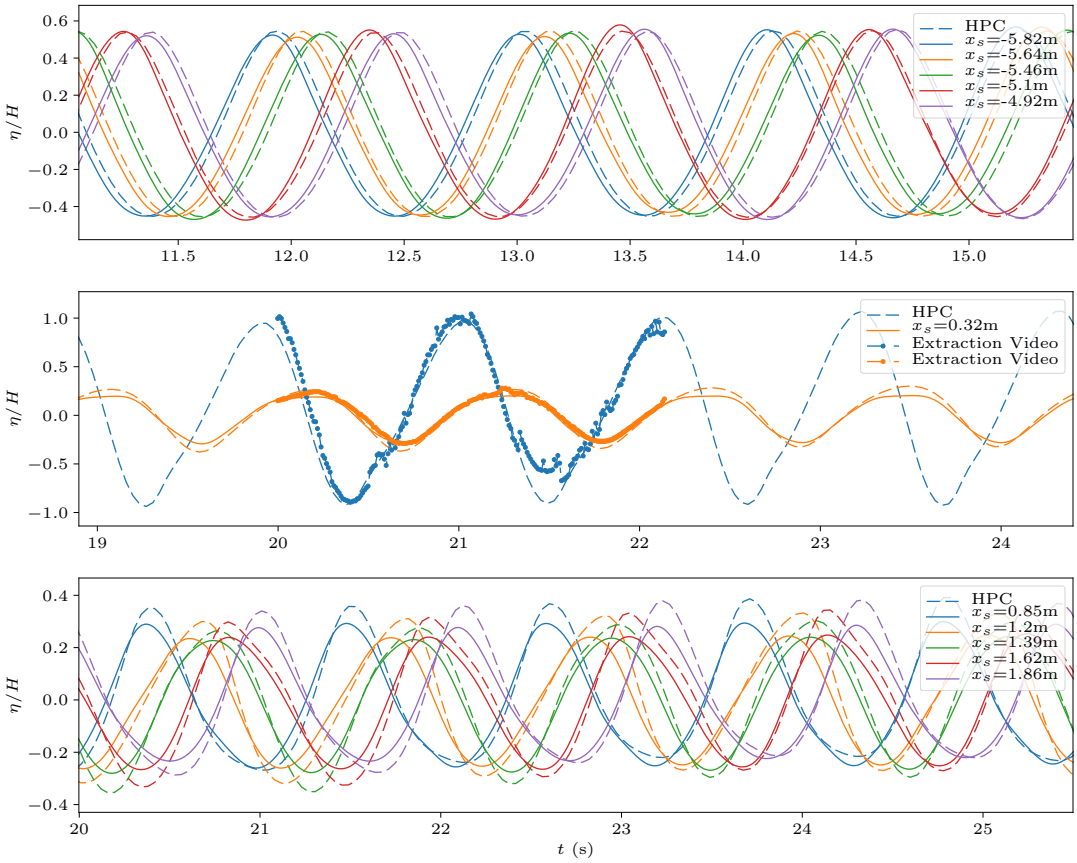


Figure 3.10: Comparisons of the free surface elevations recorded at different positions: HPC (dashed lines), experimental wave gauges (solid lines) and extracted elevation from the experimental video (circle markers). Waves gauges are divided in three groups (corresponding to subfigures): Incident waves upstream (subfig 1), Front and rear run-up (subfig 2), transmitted waves (downstream waves, subfig 3). Case 30: $T = 1.1$ s, $H/\lambda = 4.8\%$.

On figures (3.10 and 3.11), wave gauge measurements are compared to the free surface elevation time series from the HPC method for cases 30 and 21 respectively. Only the waves gauges appearing in both domains are shown (the computational domain is shorter than the experimental one). Note the time synchronisation between measurements and simulations was done on the first gauge only, and the determined phase shift was then applied to all the remaining gauges. This means that relative phases of the wave elevations in the numerical simulations are represented adequately in these figures. More generally, a very good agreement is found on both wave amplitudes and phases, at the various locations along the wave flume. However, some discrepancies of the run-up at the front and rear faces of the barge as well as concerning the downstream wave elevations can be observed. Those discrepancies become more marked when the steepness increases. It can be observed

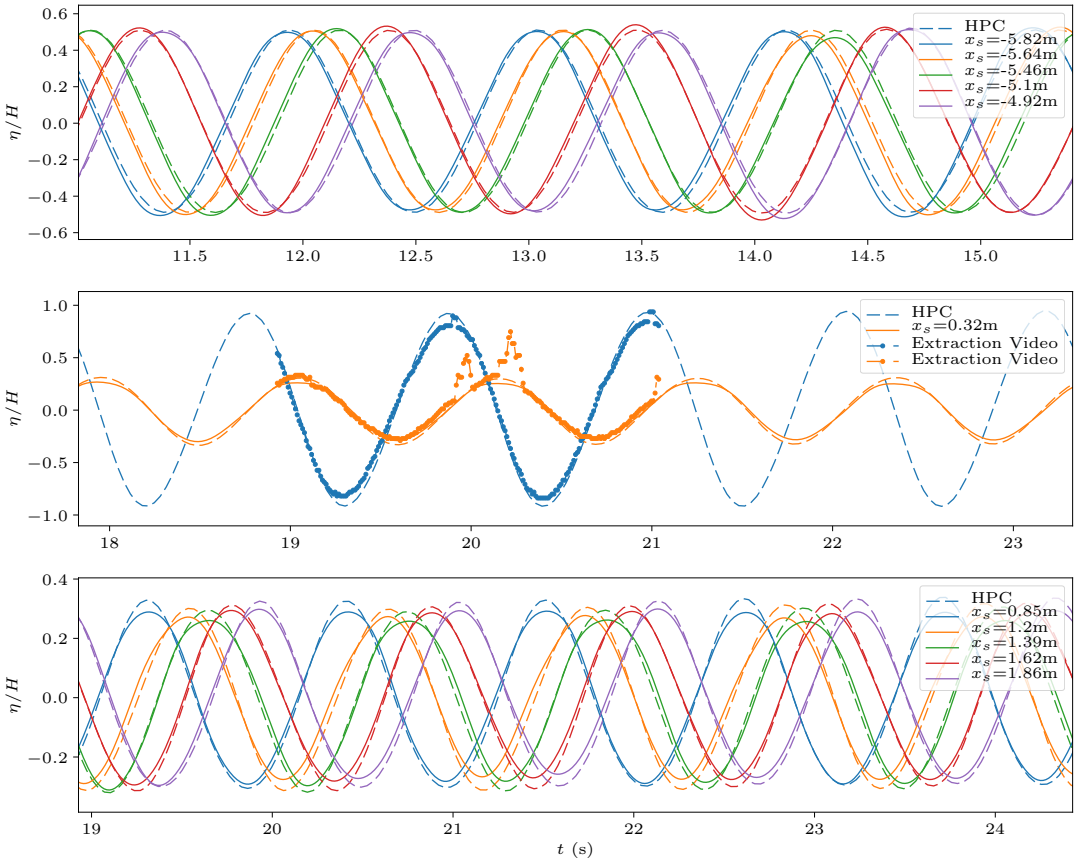


Figure 3.11: Comparisons of the free surface elevations recorded at different positions: HPC (dashed lines), experimental wave gauges (solid lines) and extracted elevation from the experimental video (circle markers). Waves gauges are divided in three groups (corresponding to subfigures): Incident waves upstream (subfig 1), Front and rear run-up (subfig 2), transmitted waves (downstream waves, subfig 3). Case 21: $T = 1.1$ s, $H/\lambda = 1.2\%$.

that the HPC potential model tends to overestimate larger run-up events, and transmitted wave heights. It is thought that the main cause of those discrepancies is related to the mathematical model itself. It is well known that non-dissipative models cannot correctly describe the flow in the vicinity of sharp angles even for linear incident waves. Thus, the present potential model is not perfectly appropriate to model the behavior of this type of flow. In practice, this effect is often counteracted by introducing a numerical lid in the vicinity of the body to numerically dissipate some energy (so trying to mimic dissipation due to viscosity). With such a method, the length and strength of the lid need to be tuned to match the expected result. This option was not tested here.

Obviously, those differences are expected to impact the loads exerted on the barge, mainly through the difference of run-up on the front and rear faces, impacting for example the elevation at which the pressure is set to the atmospheric pressure (dynamic boundary condition at the free surface).

Figure (3.12) shows at different time instants the free surface elevation in front of the body obtained from the HPC computation superimposed on snapshots from the experiments. On these pictures, we clearly denote aspects which cannot be taken into account

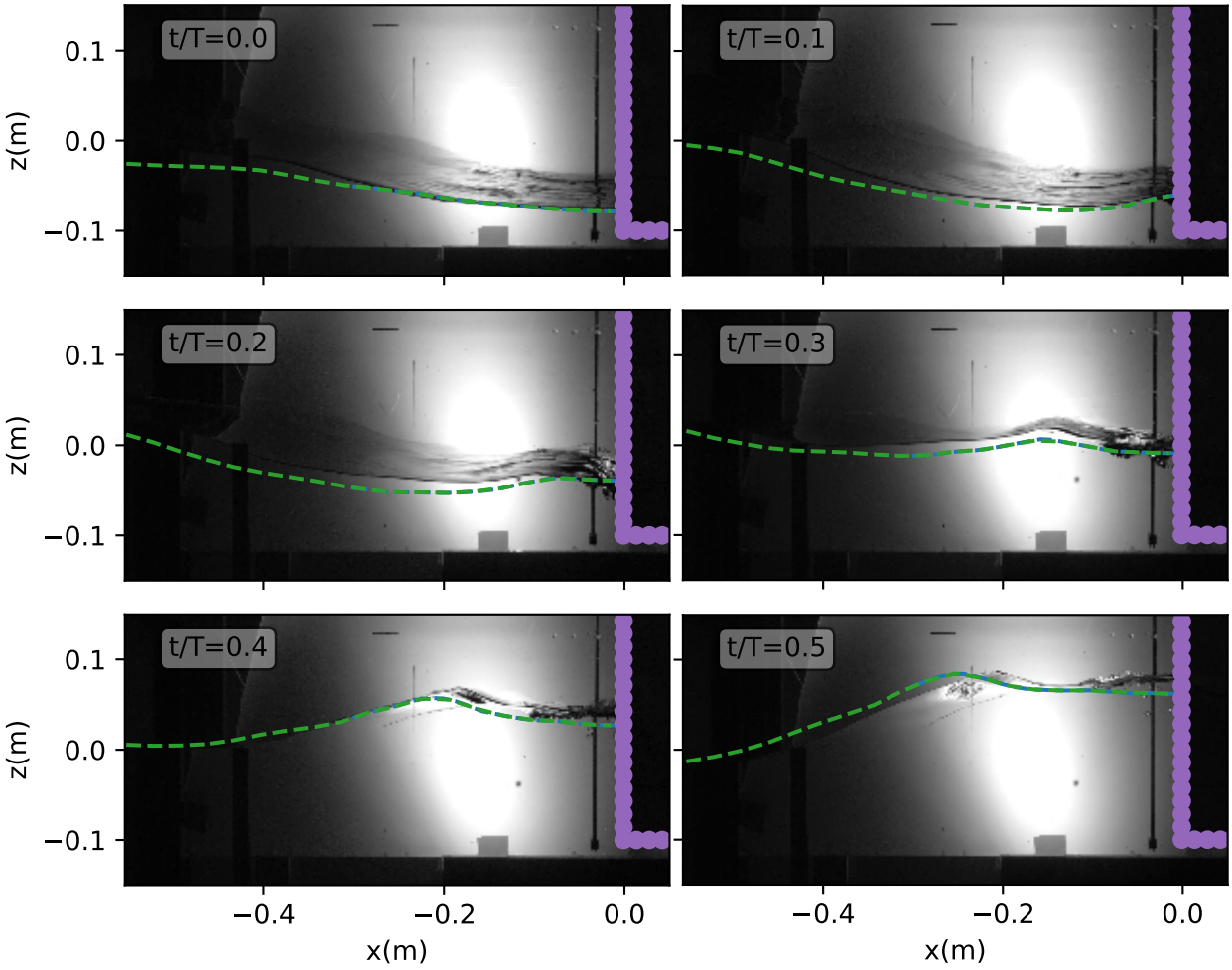
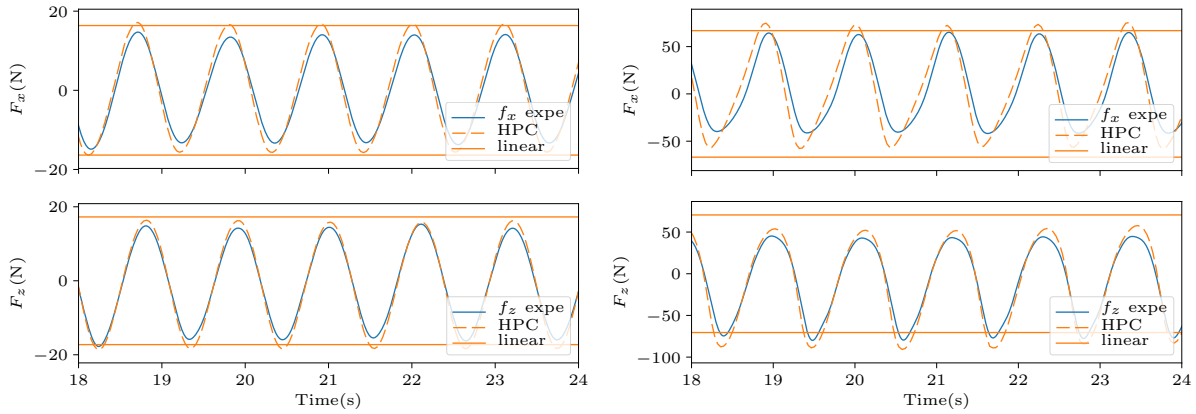


Figure 3.12: Case 30 $T = 1.1$ s, $H/\lambda = 4.8\%$. Comparison of the computed free surface elevation in front of the rectangular barge with the experiment at six different time instants. Incident waves come from the left.

in the potential model: during the rise of the water level, air entrainment and wave breaking take place, leading to important complex turbulent effects. At this stage, it is expected that the viscous effect play an important role. Although those dissipative effects are neglected in the HPC model, a relative fair agreement can be seen on the figure. The run-up is approximately correctly captured, as well as the reflected wave emerging during the elevation of the run-up. Moreover, the numerically computed reflected wave seems to be slightly faster than the experimental one. The difference can again be attributed to viscous effects, delaying the apparition of the reflected wave.

As the objective is to compare loads applied to the barge, time series of horizontal (F_x) and vertical (F_z) force components on the body are depicted on figure (3.13) for the cases 21 and 30, respectively the most linear and nonlinear cases with $T = 1.1$ s. The horizontal orange lines (symmetric with respect to the zero-force line) represent the amplitude of linear predictions of these loads. When the incoming waves are close to be linear (case 21), a good agreement is found between the linear and HPC models: the amplitudes are almost equal though the mean value computed with the HPC model slightly moves the



(a) Case 21 (b) Case 30

Figure 3.13: Case 21 and 30 $T = 1.1s$, $H/\lambda = 1.2\%$ and $H/\lambda = 4.8\%$. Time series of the hydrodynamic loads on the barge: experiments, linear results and HPC simulations.

vertical load extrema from the symmetric horizontal lines of the linear model result. Note that the effective (local) steepness, which determines the degree of nonlinearity, close to the body is approximately twice the incident steepness H/λ: only a small part of the energy is transmitted and thus the reflected wave adds to the incident wave locally. For a quantitative analysis, the amplitudes of incident and transmitted waves elevation can be compared in figure (3.11).

When compared with the experimental data, the computed loads are of approximately 15% higher magnitude. Note that, from an engineering point of view, the model is conservative in this case, and as expected, the differences in terms of run-up on the front and rear sides of the body lead to a slight over-prediction of both the horizontal and vertical loads. This over-prediction is seen for all cases, and consistently with approximately the same relative value. For steep incoming waves, the discrepancies between the linear and nonlinear models increase, as one could expect. In particular, the mean value (drift force) is no longer null. For the case 30, one can note that, although the loads are over predicted in magnitude, the shape of the time series of the loads (and thus the magnitude of the higher order harmonics of the loads) seems to be in good agreement with the experiment.

In order to compare the computed loads with the experimental ones more precisely, a Fourier decomposition of the time series is performed. The Fourier coefficients of this decomposition are shown for all the cases performed at $T = 1.1$ s on figure (3.14). As expected the amplitudes of the first harmonic exhibit an over-estimation compared to the experimental data, whereas the mean values ($\hat{f}^0_{xz}$) are in correct agreement. However, the evolution of all the harmonics with the wave height is found to be exactly the same as measured during the experiments (experimental and computed lines are parallel). For the vertical loads, all harmonics are roughly consistent with the measurements. An important difference can be noticed for the third order harmonic of the vertical load for small steepness. First, it can be denoted that this harmonic is 2 orders of magnitude lower than the first order harmonic, and even 3 orders magnitude lower than the hydrostatic force. Thus, an accurate computation as well as an accurate measurement of this small contribution is difficult to achieve. Moreover the behavior of the potential values seems to be more easily fitted with an order 3 model in steepness, which, together with the fact

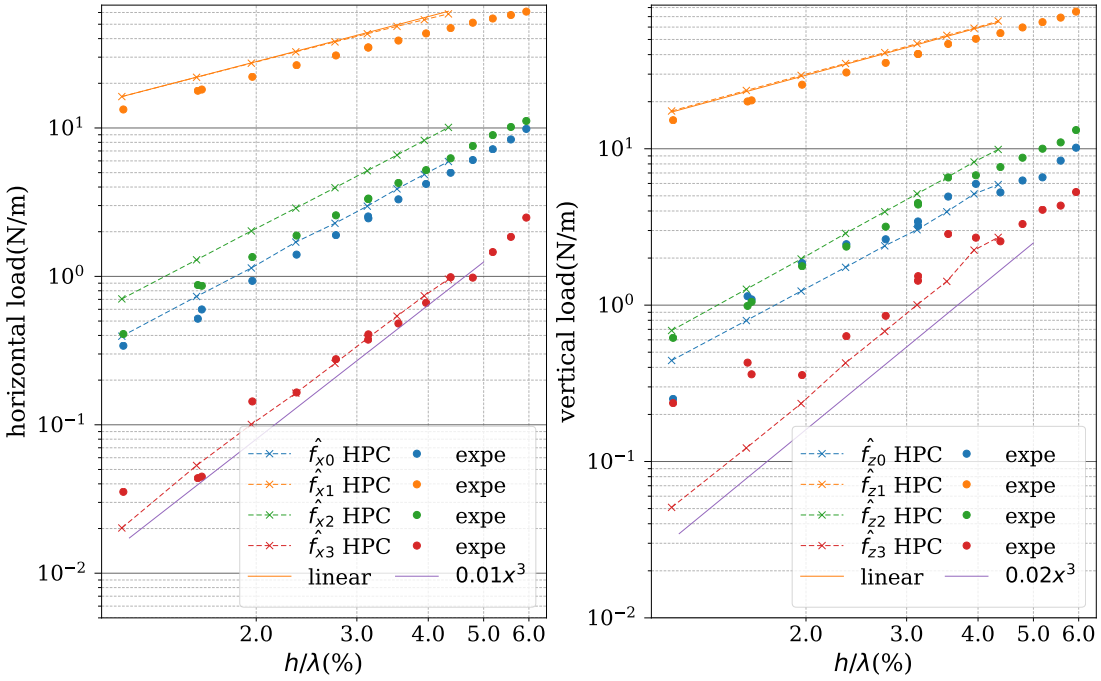


Figure 3.14: $T = 1.1$ s, Amplitudes of harmonics of the vertical and horizontal loads on the barge. HPC models (dashed lines), experimental results (dots) and linear transfer functions (solid orange line). A power function of order 3 is added to compare the increase rate (solid purple line).

that numerical and experimental third order harmonics match for higher steepness, tends to validate the numerical results. It is thought that either measurements is not accurate enough to capture such a small contribution, or that a physical effect giving energy to the third harmonic plays a role not properly taken into account by the model.

For the horizontal loads, more discrepancies are visible. First order and drift force show an overestimation of about 15%. The amplitude of the third order harmonic seems to be well captured, but the amplitude of the second order one exhibits a larger overestimation by about 70%. This effect is still under investigation, but the impact on the free surface of the viscous effects seems to be significant in this case. Note that the hydrostatic contribution originating from instantaneous wet free surface is one of the main contributions to the second order component. In the present setup, this term will play a role only on the horizontal load as the wet free surface changes only on the vertical walls of the barge. Thus, an important difference between the experiments and the numerical model is expected on this particular second order harmonic of the horizontal load: it was shown previously that our computations do not capture exactly the effective run-up elevations, probably due to the viscous effects not taken into account in the HPC model.

3.3.3 Analysis of the limits of the potential approach: rectangular horizontal cylinder

It was shown that the HPC method was not able to capture all physical effects happening in case of a free surface piercing barge. In order to separate some of the mentioned

effects, a test case where sharp corners are present without piercing the free surface is selected. This test case will also be used used as the main case to develop and validate models capable of treating rotational, viscous and turbulent effects that take place here and largely influence the results.

3.3.3.1 Description of the simulation

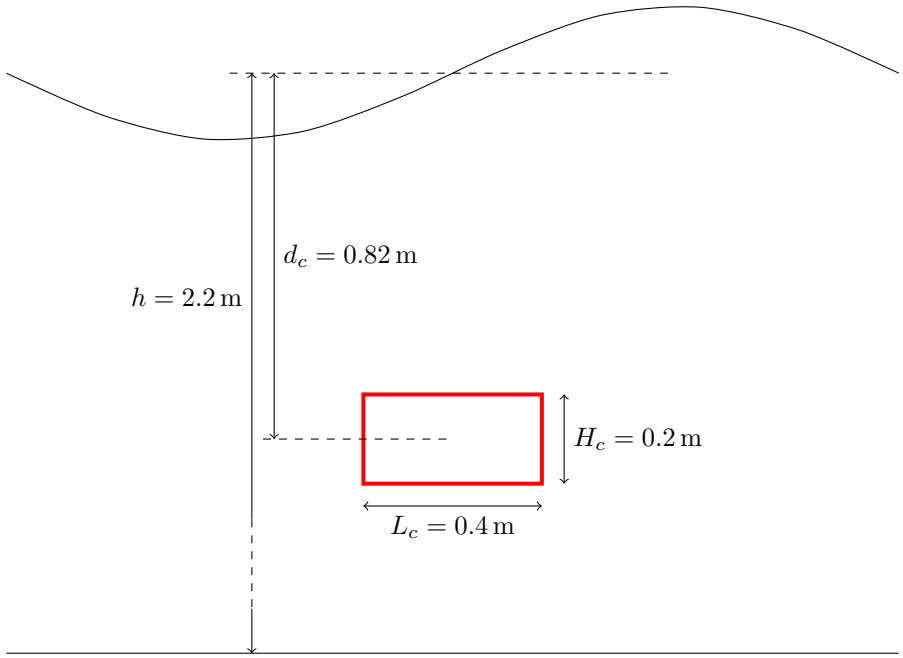


Figure 3.15: Geometric representation of the test case. Note that only the wavelength, wave elevation and the water depth are not represented at scale

The selected model has been studied experimentally by Arai (1995), Venugopal (2002), and Venugopal *et al.* (2006). A numerical investigation was performed with a RANS-VoF method combined with a k-ϵ turbulence model by Li and Lin (2010).

A geometrical schematic representation of the case is shown in fig. 3.15. The case consists in a 2D horizontal cylinder fully immersed at a relative depth of $h/d_c = 2.68$. The cylinder cross-section is of rectangular shape with its longest dimension parallel to the wave number vector. Its dimensions are given on fig. 3.15. Different wave periods, and thus the wavelengths will be simulated in this study. For a wave period of $T = 2\,\mathrm{s}$, the cylinder horizontal length relative to the wavelength is $\pi L_c/\lambda \approx 0.2$ which locates the case in the validity range of the Morison equations ($\pi L_c/\lambda \leq 0.4$, left panels in fig. 1.6), although some discrepancies might be expected.

The HPC simulation is run with a background mesh of $N_x = 60$ - following the drawn conclusion from the convergence study of section 3.2 - spanning over 8 wavelengths: the horizontal span of the mesh and its discretization thus depend on the selected wavelength (*i.e.* on the selected period). Two relaxation zones are set to respectively generate and absorb the waves, each one being applied over a two wavelength extent. The waves are imposed at the inlet boundary (and its vicinity with the relaxation zone) following the stream function theory (Fenton, 1988).

In addition, a mesh centered on - and fitted to - the cylinder is defined, with a horizontal width of 1 m and total height of 0.8 m. Square shaped cells are used, of size 0.02 m×0.02 m. Because this mesh refinement does fulfill the convergence criteria ($N_x > 90$), it will remain fixed throughout this study. The computational time steps are enforced trough the CFL number based on the phase velocity of the wave and relative to the background - propagating the waves - mesh discretization. This CFL number is chosen as $C_o = 2$ for all computations.

The models are compared using the KC number as the main parameter of the wave flow. It has been showed (*e.g.* Venugopal, 2002), that a modification of either the depth submergence or the frequency parameter (which controls the wave period) does not influence significantly the obtained hydrodynamic coefficients as long as KC remains constant.

3.3.3.2 Computation of the hydrodynamic coefficients

Literature comparisons are done on the hydrodynamic coefficients (the inertia coefficient $C_{Mx,z}$ and drag coefficient $C_{Dx,z}$) that model the forces applied on the body following the so-called Morison equation (Morison *et al.* (1950)):

$$F_{xm} = 1/2\rho C_{Dx} H_c u_x \sqrt{u_x^2 + u_z^2} + \rho A C_{Mx} \dot{u}_x \tag{3.3a}$$

$$F_{zm} = 1/2\rho C_{Dz} L_c u_z \sqrt{u_x^2 + u_z^2} + \rho A C_{Mx} \dot{u}_z \tag{3.3b}$$

where F_{xm} and F_{zm} are the modeled (by the Morison formula) loads applied on the object. For simplicity those forces will be called the Morison loads. u_x, u_z are respectively the horizontal and vertical velocity of the flow, and $\dot{u}_x$, $\dot{u}_z$ are the particle acceleration in the corresponding direction. Note that the selection of those kinematic values represents a first significant decision. The Morison equations were originally developed to model the force applied by a uniform oscillatory flow on an object of small relative dimensions. The associated assumptions is that the body is subjected to a flow it has a limited impact on. In this case, the choice of the associated kinematic is straightforward: one should select the kinematics of the unperturbed flow. By extension, the imposed wave model at the inlet (here a stream function theory) applied at the location of the center of the cylinder. In our 2D framework, the loads are expressed in N m^{-1}: H_c and L_c correspond respectively to the height and width of the cylinder, in turn defining the area of the rectangular cylinder $A = H_c L_c$.

Given a temporal load series, many methods exist so as to access the "corresponding" hydrodynamic coefficients, *i.e.* the hydrodynamic coefficients that yield Morison loads as close as possible from the HPC predicted loads. One of the most common method is to minimize the square root of the error between the modeled loads and the real ones. At a given time steps i the error is given by $e_{x,z}(i) = F_{x,z}(i) - F_{xm,zm}(i)$. Indexes x, z represent the horizontal and vertical errors respectively. Thus, one could minimize the resulting mean square over time of the previously defined error. Following Venugopal (2002), the coefficients that minimize such error over a certain number of time step N can be analytically derived as:

$$C_{Dx,z} = \frac{2}{\rho D} \frac{f_1 f_2 - f_3 f_4}{f_2 f_5 - f_4^2} \tag{3.4a}$$

$$C_{Mx,z} = \frac{2}{\rho A} \frac{f_3 f_5 - f_1 f_4}{f_2 f_5 - f_4^2} \tag{3.4b}$$

where D is either H_c or L_c for the horizontal and vertical coefficients respectively.

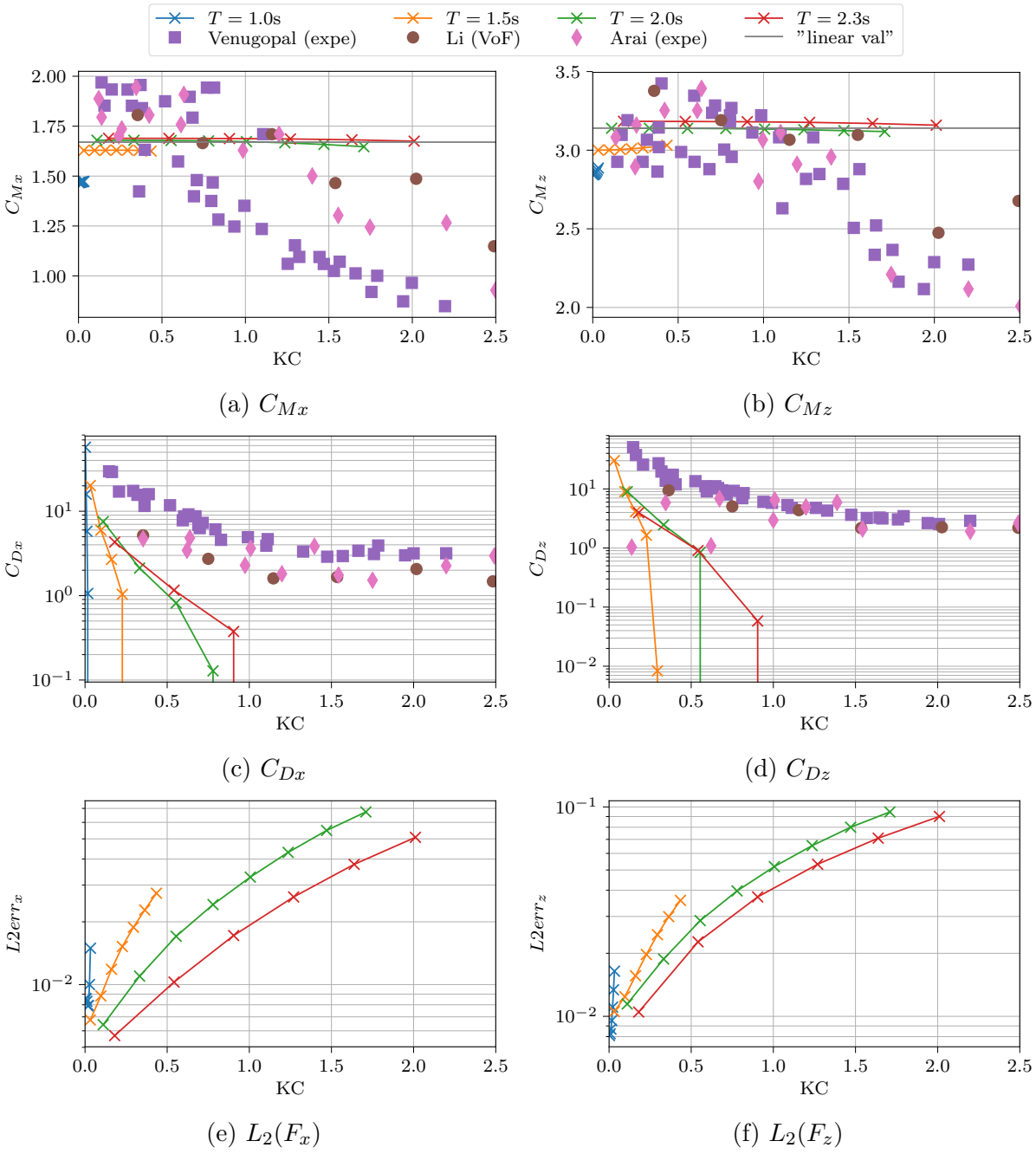
$$\begin{aligned} f_1 &= \sum_i^N F_{x,z}(i) u_{x,z}(i) \left\| \mathbf{u}(i) \right\| \\ f_2 &= \sum_i^N \dot{u}_{x,z}^2 \\ f_3 &= \sum_i^N F_{x,z}(i) \dot{u}_{x,z} \\ f_4 &= \sum_i^N u_{x,z}(i) \left\| \mathbf{u}(i) \right\| \dot{u}_{x,z} \\ f_5 &= \sum_i^N u_{x,z}(i)^4 \end{aligned} \tag{3.5}$$

Remark. *Note that another method is presented in Arai (1995): it consists in expanding in terms of Fourier series the kinematic variables - based on their analytic expressions - to obtain the Fourier decomposition of the Morison loads. Then, a numerical harmonic decomposition is also performed on the obtained loads. Finally, both decompositions are equalized, and the corresponding terms are identified. With this method, the coefficients are obtained in terms of the Fourier harmonics that constitute the load series. This method was also implemented in this work, with no major differences in terms of obtained harmonic coefficients. The results will thus not be presented here.*

3.3.3.3 Results and interpretations

On fig. 3.16 all results in terms of hydrodynamic coefficients are depicted: figs. 3.16a and 3.16b show respectively the obtained horizontal and vertical inertia coefficients, figs. 3.16c and 3.16d the obtained drag coefficients, and figs. 3.16e and 3.16f the errors in the reconstruction of the obtained loads *via* the Morison formula (*i.e.* the L_2 norm of the error $e_{x,z}(i)$, $L_2(F_{x,z}) = \sqrt{\sum_i e_{x,z}^2} / \max(F_{x,z})$). We would like to stress that this error does not represent a discrepancy between the obtained values and the experimental ones, but rather the consistence in modeling the loads with the Morison eqs. (3.3a) and (3.3b). Note that, in order to emphasize discrepancies, the figure limits are chosen so as to fit the range of the presented data.

The HPC results show that the inertia coefficients are dependent on the wave period (spanning over $\sim$ 10%), but are almost constant with respect to KC. In a potential framework, this could be expected: the wave loads are divided by the kinematics to obtain the inertia coefficients. Hence, because of the absence of energy dissipation, the velocity and accelerations increase with the wave height in the same manner as the loads, and thus inertia coefficients are practically constant. This, of course, is different from the physical expectations: viscous effects will increase with the wave height and drastically modify the coefficients. Such a behavior is obtained both experimentally and with the FVM-VoF method applied by Li and Lin (2010). However, our method being potential, it is obvious that this will not be correctly captured. Note that this shortcoming is however not restricted to the HPC method, but will manifest with any potential model.



(a) C_{Mx} (b) C_{Mz}

(c) C_{Dx} (d) C_{Dz}

(e) $L_2(F_x)$ (f) $L_2(F_z)$

Figure 3.16: Inertia and drag coefficient in the two directions of the Loads obtained with the HPC method with different wave periods. The L_2 norm of the error between the Morison model and the real loads is also shown.

Another interesting point is the slight change of value obtained for a modification of the wave period. It is thought it can be explained by the cylinder submergence relative to the wavelength which is different when the wave period changes. Thus in some cases, the cylinder will be more impacted by the surface flow dynamics - and retroactively impact the free surface -, while in some others we will encounter mostly an oscillatory flow less dependent on surface effects. We think that it was not denoted during the experimental studies, mainly because of its small influence on the hydrodynamic coefficients compared

T (s)	H (m)	H/λ	Re	KC	β
1.0	0.01	0.50%	362	0.002	160000
	0.02	1.50%	1093	0.007	
	0.04	2.49%	1842	0.012	
	0.05	3.46%	2622	0.016	
	0.07	4.42%	3445	0.022	
	0.09	5.36%	4321	0.027	
	0.10	6.28%	5262	0.033	
1.5	0.02	0.50%	3415	0.032	106667
	0.05	1.50%	10267	0.096	
	0.09	2.49%	17187	0.161	
	0.12	3.47%	24216	0.227	
	0.16	4.44%	31397	0.294	
	0.19	5.38%	38766	0.363	
	0.23	6.31%	46359	0.435	
2.0	0.03	0.50%	8844	0.111	80000
	0.06	1.00%	17695	0.221	
	0.09	1.50%	26562	0.332	
	0.15	2.49%	44368	0.555	
	0.21	3.48%	62323	0.779	
	0.28	4.45%	80482	1.006	
	0.34	5.41%	98903	1.236	
	0.40	6.36%	117641	1.471	
	0.46	7.28%	136752	1.709	
2.3	0.04	0.50%	12551	0.180	69565
	0.12	1.50%	37687	0.542	
	0.19	2.49%	62921	0.904	
	0.27	3.48%	88322	1.270	
	0.35	4.46%	113957	1.638	
	0.43	5.43%	139900	2.011	

Table 3.1: Waves parameters for the simulated cases. KC and Re are based on the horizontal width of the object and the maximum horizontal velocity at the cylinder submergence depth predicted by the wave theory, see section 1.4. $\beta = Re/KC$.

to the increase in KC number.

Concerning the drag coefficients - also represented in figs. 3.16c and 3.16d -, they are not expected to be captured at all with a potential model, they still are shown to emphasize that the method predicts very small drag (at least 5 times lower than the experimental ones), but that the best fit with a Morison model have some small C_D values. This also tends to give confidence in the obtained result and the associated Morison model: only a very small part of the load is in phase with the particle velocity squared, which was to be expected in a potential framework.

The obtained L_2 errors that are made by modeling the loads with a Morison equation is adimentionalized by the maximum of the load of concern and shown in figs. 3.16e and 3.16f. It is possible to note that, especially for small values of KC, the error is

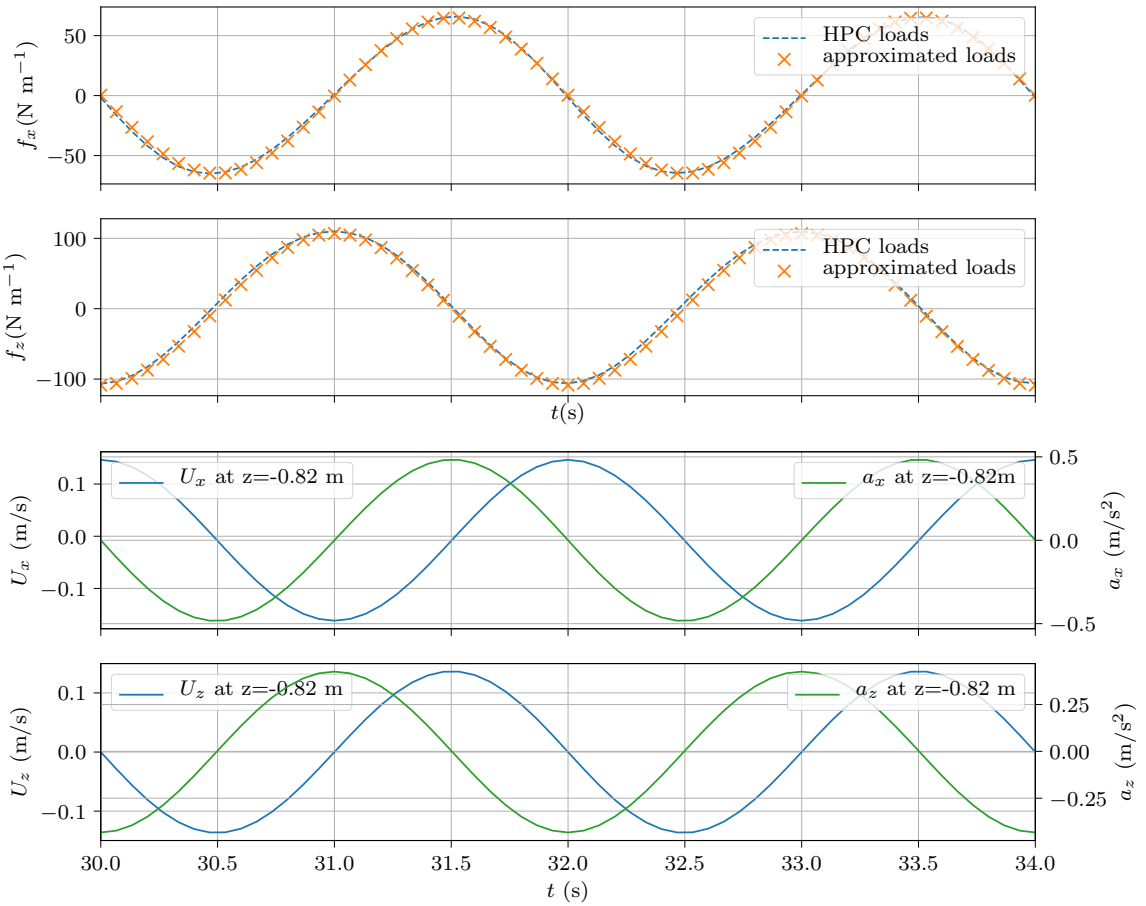


Figure 3.17: Case $H/\lambda = 3.5\%$, $T = 2\,\mathrm{s}$. Horizontal and vertical loads predicted by HPC, and their reconstruction with a Morison model. Kinematics used for reconstruction, *i.e.* predicted by the wave model (stream function) at $z_c = -0.82\mathrm{m}$. Associated reconstruction errors $(L_2(F_x), L_2(F_z)) = (2.4\%, 3.9\%)$.

relatively low. This means that the Morison equation is appropriate in this range.

For a constant KC, increasing the period means lowering the value of $\pi D/\lambda$ (see section 1.4) and lowering the value of H/D. Together with the wave-body interaction regimes depicted in fig. 1.6 (inertial panel I), the Morison equations should be valid for small period values. It is effectively denoted from figs. 3.16e and 3.16f that, for a constant KC, decreasing the period improves the Morison modeling.

On the other hand, when KC increases at constant period, significant reconstruction errors are obtained (up to 9%). Note that a relative error of 9% corresponds to loads that are relatively different and thus only a relative confidence can be granted in the Morison model in such case (see figs. 3.17 and 3.18, for which $(L_2(F_x), L_2(F_z)) = (2.4\%, 3.9\%)$ and $(5.0\%, 9\%)$). Different reasons might explain this behavior. First in when H/D is large, the viscous drag should be dominant, but is however not taken into account in the potential model. Then, for small KC, the object is located deeper, relative to the wave height, and the flow thus behaves more like an oscillatory flow that can be accurately modeled with a Morison approach. Finally, it is possible that waves of significant non-linearity parameter H/λ might be less correctly captured with either the potential model

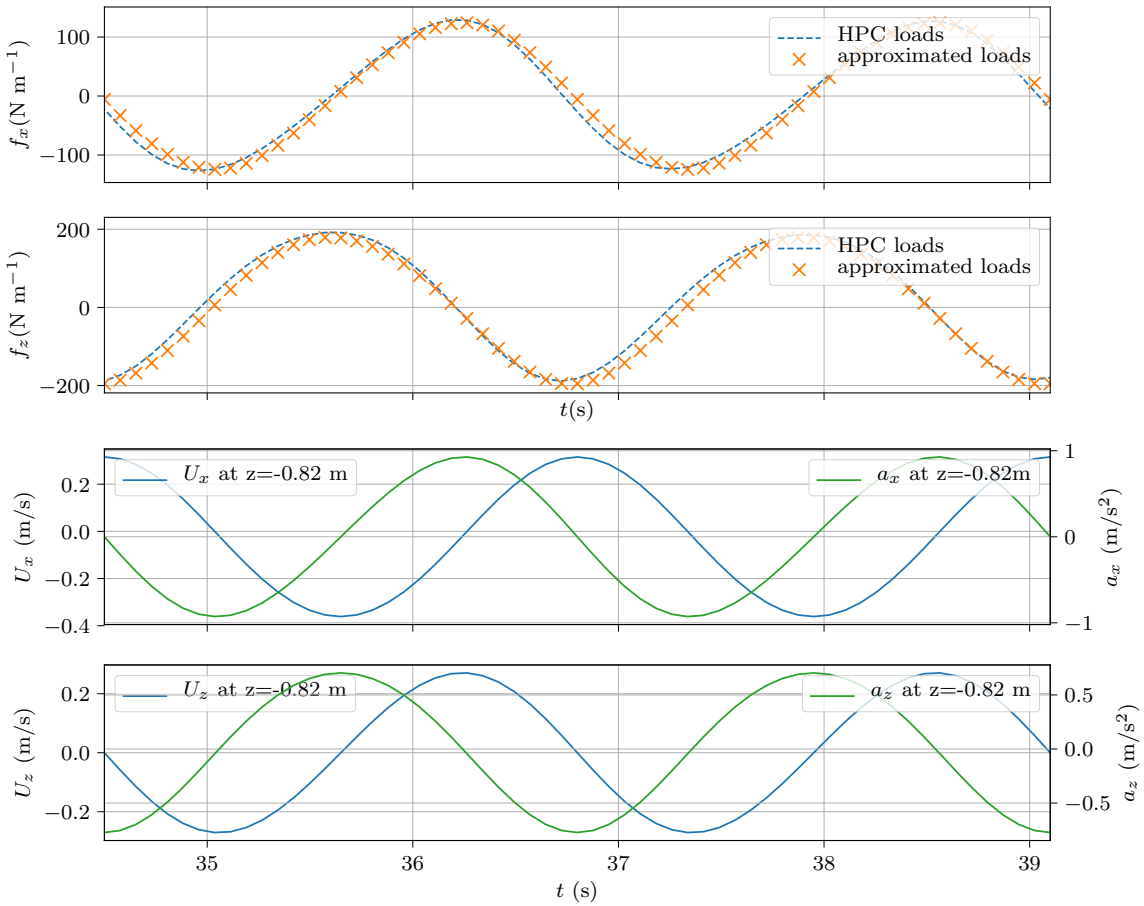


Figure 3.18: Case $H/\lambda = 5.5\%$, $T = 2.3$s. Horizontal and vertical loads predicted by HPC, and their reconstruction with a Morison model. Kinematics used for reconstruction, *i.e.* predicted by the wave model (stream function) at $z_c = -0.82$m. Associated reconstruction errors $(L_2(F_x), L_2(F_z)) = (5.0\%, 9.0\%)$.

or the Morison reconstruction. In conclusion, while our results exhibit significant differences with the conducted experiments and FVM-VoF method at larger KC number (up to $KC = 2.0$), they still seem to be consistent with the potential assumptions. We clearly denote a good agreement with the "linear value" extracted from Venugopal *et al.* (2006). However, a limit of the model in simulating physical wave fields and wave-structure interaction is clearly met: the fact that the viscous and turbulent effects cannot be taken into account represents a significant shortcoming when sharp corners are involved. This type of geometry is known to drastically impact the flow, especially at high Reynolds number (Tamura and Miyagi, 1999). The Reynolds number in our case relative to the maximum horizontal velocity and the height of the cylinder in the vertical direction is of order of $Re = 10^4$ (and varies with the KC number, see table 3.1). Thus, it is obvious that a potential model cannot capture several of the significant effects of such flow.

3.4 Fitted mesh method: results and limitations

All the above presented results make use of the IBM to model the free surface. The associated method was presented in section 2.3.4.2. However, the first natural approach when developing such a NWT is to allow for a mesh deformation to fit the free surface topology at any given time. This method is described in section 2.3.4.1. In this section we consider rapidly some of the cases described during this chapter (namely the standing wave and the circular cylinder), and recompute the potential solutions obtained with a mesh deformation method: the free surface condition is simply imposed as a Dirichlet condition for the BVP problem. It is the occasion of explaining the reasons that led to reconsider this approach.

3.4.1 Standing wave case

First of all, the standing wave case - described in section 3.2 - is selected. The deformation method applied on the volume mesh is a simple linearity between the bottom and the free surface. First, after a topological modification of the free surface, nodes located on the left and right boundaries are moved vertically taking into account the position of the intersection of the free surface with the given boundary. Note that, the nodes on the left and right Neumann boundaries remains evenly distributed at all times.

Afterwards, a 2D Laplace smoothing algorithm is applied on each fluid node (macro-cell center). A given Laplace iteration moves a point toward the barycentric center of its surrounding nodes. A strength parameter controls how much the point is moved in this direction (with a strength= 1 the end point will perfectly match the computed center, and with a strength= 0 the point will not be moved at all). 10 iterations are used, with a constant strength set to 0.5. Note that this is not a smoothing in the sense of reducing instabilities or dissipating energy. It is just a method to ensure an evenly distribution of nodes in the domain. No free surface node location or value is modified during this process.

Finally, the BVP can be resolved on a good quality mesh. While the main advantage of the fitted mesh method is its simplicity, one should remember that within this framework, one need to invert the local matrices at every time step, as the geometry of the cells might - and with the selected deformation method will all - change. No comparison in CPU times are shown because both methods have yet to be optimized.

Figure 3.19 shows with the same color scale and layout as fig. 3.2 the errors that are obtained after 10 periods - top subfigures - and after 100 periods - bottom subfigures -. Left subfigures recall the IBM results that can also be seen in the corresponding section (section 3.2). The maximum of the color scale corresponds to the maximum error obtained with the IBM method after 100 periods. On this figure, we can see that a greater stability over very long time is achieved with the fitted mesh method, and the associated errors seem qualitatively to be of lower magnitude.

Figure 3.20 depicts the convergence of the error with a decrease of the CFL number. The IBM results are also shown for comparison. Firstly, the achieved minimal error is lower with the fitted mesh method: once fully converged, the IBM yields errors approximately 5 to 10 times higher than the fitted mesh method. However, both errors match as long as the CFL is relatively large, *i.e.* during the convergence process. The convergence

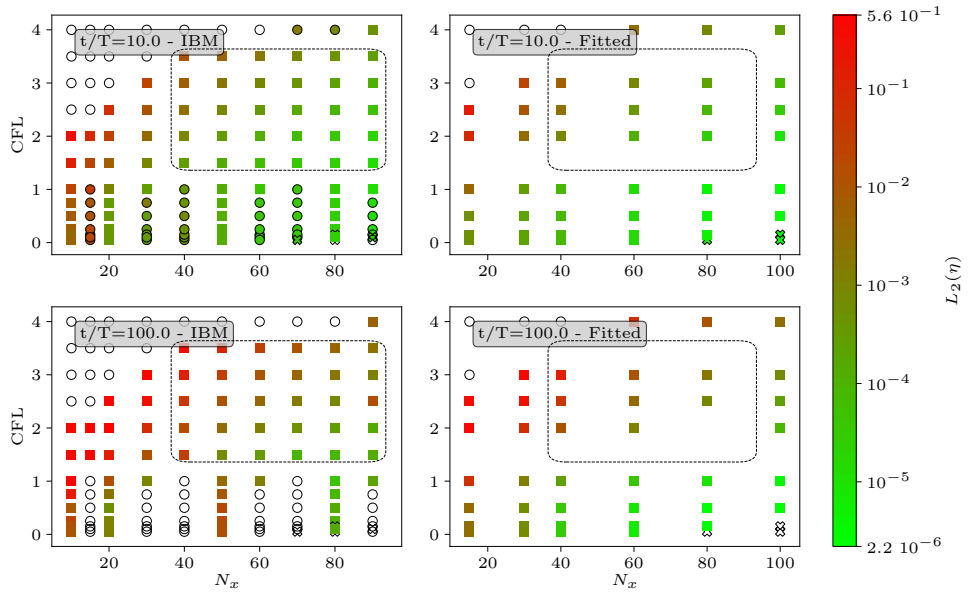
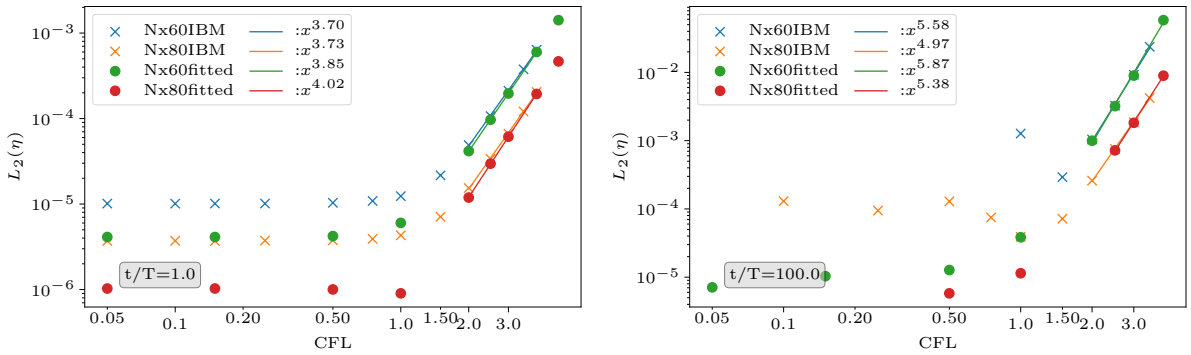


Figure 3.19: Comparison of the obtained error $L_2(\eta)$ (see eq. (3.1)) with the fitted mesh method (right panels) and the IBM free surface (left panels, recalling fig. 3.2) on the standing wave. $H/\lambda = 0.1$. Top and bottom panels are the errors after $t/T = 10$ and $t/T = 100$ respectively.



(a) $t/T = 1$ (b) $t/T = 100$

Figure 3.20: Comparison of the convergence with CFL number of the $L_{2\eta}$ error obtained with the fitted mesh and immersed boundary methods, for two different discretizations ($N_x = 60$ and 80)

rate of the fitted mesh method is slightly higher than the IBM convergence rate but of comparable magnitude (order 4 is retrieved).

At this stage, it is possible to state that the fitted mesh method yields - in addition to being simpler and stable across a wider range of parameters - more accurate results on this standing wave case that the implemented IBM.

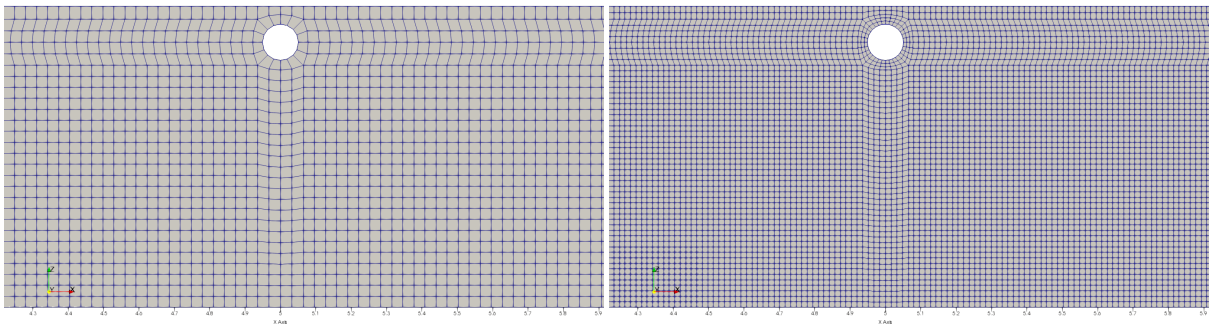
3.4.2 Circular horizontal cylinder case

3.4.2.1 Original case: submitted to regular waves

The case that is very challenging and made us reconsider the fitted mesh method is the fixed horizontal cylinder of Chaplin (1984). In the fitted mesh method, a small gap has to be meshed above the body at all times. A lot of meshes and deformation methods were implemented to try to stabilize this case with unsuccessful results. Amongst the numerous tested modifications, here is a non-exhaustive list of the evaluated changes:

- The mesh generation method: number of blocks, their positions and spans, the usage of an initial Laplace smoothing with its strength and number of iterations, *etc.*
- Its deformation method: linear, quadratic - and other powers higher and lower than 1 -, above a certain depth only, Laplace smoothing at each time steps, *etc.*
- And of course also its discretization: tests were mainly computed approximately in the range $N_x = 64$ to 130 points per wavelength.

One of the problems was to maintain a general discretization of around $N_x = 90$ - value at which the propagation process is converged - while maintaining a good quality discretization in the body vicinity and contour. For example, fig. 3.21 depicts two meshes generated with the same method with two different discretizations. The entire depth is represented on these figures. The screenshots are taken at the initial time step, and thus the top boundaries (*i.e.* free surfaces) are flat.



(a) $N_x \sim 64$ (b) $N_x \sim 130$

Figure 3.21: Screenshots of the coarsest fitted mesh tested as well as one of the finest in the vicinity of the body

However, a issue that we think is intrinsic to the HPC method raised: at some arbitrary time steps, the loads computed on the cylinder was reaching really high values. This did not yield to any significant perturbation of the wave or flow field. An example of the obtained load is shown in fig. 3.22.

After investigations, it was found that at those time steps a given macro-cell matrix (local) - often close to the body - exhibits a large condition number. In other words, its geometrical shape leads to a matrix that is nearly singular and thus difficult to invert. The problem is that, in this case, the faulty cell does not seem to be excessively stretched, nor exhibits a particular, easily identifiable, shape. In practice this cell matrix is still invertible but the coefficients of the inverted matrix are orders of magnitude greater that

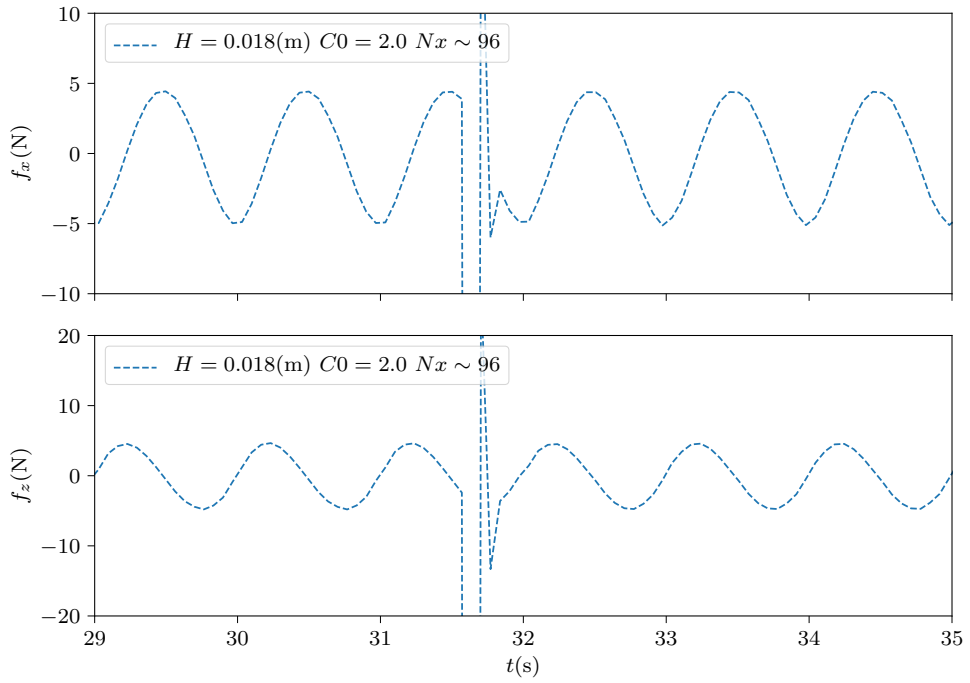


Figure 3.22: Example of instability of the predicted load at a given time step

the coefficients of the inverted matrices of the neighboring cells (usually 3 to 4 orders of magnitudes higher). Thus when filling the global matrix, a particular row will contain values orders of magnitude greater that the values in other rows. This leads in turn to a bad resolution of the potential in this particular cell. Even tough this effect is very limited in both time and space, the pressure at this instant is then not accurate whether it was computed by a finite difference method or by solving a Laplace problem on ϕ_t (which would have the same flaws as the global matrices of the problems on ϕ and ϕ_t are equal).

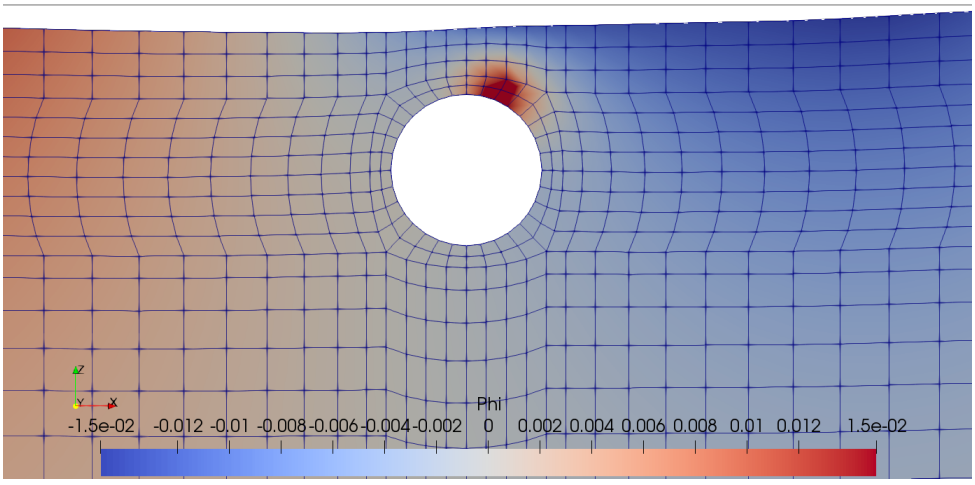


Figure 3.23: Example of obtained potential field at $t = 31.6366$ s

Figure 3.23 shows the potential field obtained at the problematic time instant exhibited by the load behavior in fig. 3.22(on the same case). The spread peak value around the faulty node is mostly due to the representation effect (paraview): even the direct neighbor nodes values are almost not affected while the node value of the potential rises.

3.4.2.2 Simplified case: resolution of a BVP

In order to push the investigation further, a simplified BVP was defined. The mesh topology is extracted at the faulty time instant, and all the external boundaries ("free surface", bottom, inlet and outlet) are replaced by Dirichlet conditions for which the potential is enforced following the analytic expression of the potential of an uniform horizontal flow around a cylinder:

$$\phi = v_0 r \cos\theta \left(1 + \frac{R^2}{r^2}\right) \tag{3.6}$$

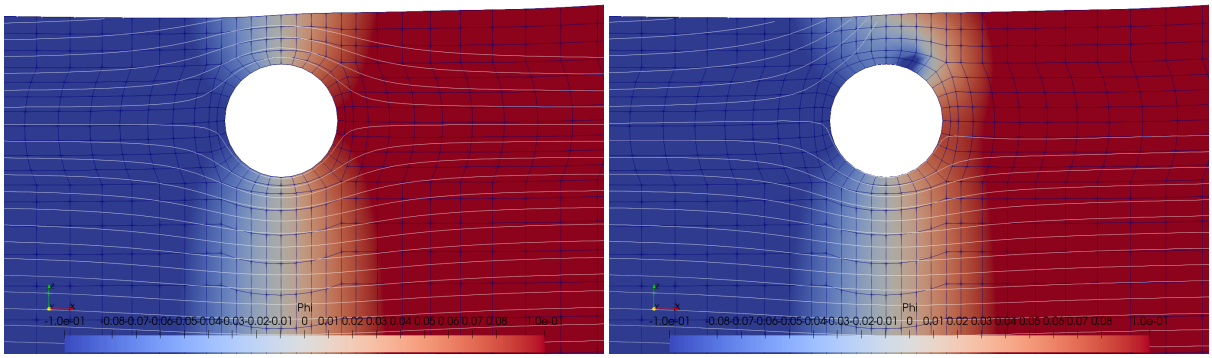
where $(r,\ \theta)$ are the polar coordinates of the boundary node in the cylinder reference frame, and R is the radius of the cylinder. The hope is that the BVP yields in the volume, values that correspond to the latter expression.

First, the cylinder outer boundary is also imposed as a Dirichlet condition, using the above-presented formula. In that case, the problem is "stabilized" and correct results are obtained. However if we release a degree of freedom by imposing a Neumann condition on the body boundary, by deriving eq. (3.6) along the normal - *i.e.* with respect to r - the BVP solution exhibits once again a significant local disturbance. The two obtained solutions of those BVPs are shown in fig. 3.24. Note that is was verified that the local cell inversions were correct, further confirmed by the good quality results obtained with the full Dirichlet BVP.

However, the important values of the inverted local matrix does greatly modify our iterative inversion when a Neumann condition is used at the body boundary. It should, in that case, be possible to retrieve correct results by further decreasing the expected residual during the matrix inversion. They were however, already fixed at $0.5{\cdot}10^{-8}$ and any further decrease would in turn significantly increase the computational cost. Because only the global residual is controlled, this tolerance represents a mean error value. Therefore, as the faulty cell vicinity is harder to properly resolve, it is possible that a tolerance reduction would not improve the results in this zone.

Even if this problem was investigated at a particular time step on a particular mesh, the same behavior proved to occur in almost every computation on this circular cylinder on which the mesh is allowed to deform. Note also that the faulty cell is not always the same, but almost always in the close vicinity of the body.

Remark. *When the mesh is completely fixed, and only the very free surface nodes are allowed to move, stable results could be obtained. In this case, the wave height is limited by the discretization (the wave amplitude cannot be larger that the cell size): only very coarse mesh could compute large wave height cases, which appears as an obvious limitation. With this method only - of course very limited in terms of achievable wave height - it was possible to obtain consistent results across cases.*



(a) Cylinder boundary: Dirichlet (b) Cylinder boundary: Neumann

Figure 3.24: Resulting velocity potential ϕ, solution of the BVPs. The outer boundaries are imposed as Dirichlet condition respecting: eq. (3.6). The cylinder condition is either directly applyied as a the latter equation (Dirichlet, left) or as a Neumann condition which value is derived from the same equation (right).

3.4.3 Conclusion on the fitted mesh method

The fitted mesh proved to yield very precise and valuable results as well as great convergence properties on the standing wave case. Every exhibited properties - for example stability, convergence order, achieved accuracy after convergence - are either of equal quality or better than with the IBM.

However, on the horizontal cylinder case, despite many efforts, it was not possible to obtain non perturbed results by the phenomenon explored above. At the end, the fitted mesh method was abandoned, mostly because of the difficulty to establish a clear criterion in terms of cell distortion that triggers this instability.

3.5 Summary of conclusions

In this part of the work, an implementation of the Harmonic Polynomial Cell (HPC) method for solving the fully nonlinear potential flow problem in combination with an Immersed Boundary Method (IBM) to track the free surface was achieved and validated. The newly developed NWT allows to simulate a wide range of situations in ocean and coastal engineering applications, involving nonlinear waves in uniform or variable water depth as well as wave-body interaction. In this version, the bodies in interaction with waves can be immersed or surface piercing, but have to be fixed. The numerical techniques of this NWT have been described, in particular the implications of choosing to work with non-deforming grids which has led to the development of an accurate and robust IBM variant.

After investigating the convergence properties of the numerical methods at long time with respect to a refinement in space and time on a freely evolving standing wave, it was shown that the NWT exhibits a high level of accuracy and other interesting features. First, a large range of time steps are shown to lead to stable computations over the target duration of 100 wave periods, although for small CFL C_o numbers the results tend to become unstable. This difference can however mostly be explained by the fact that no filtering nor smoothing were used during this work. Most importantly, the recommended range of CFL for this HPC implementation was shown to be $C_o \in [1.5, 3.5]$, which is of great benefit as these large values permit to use large time steps, and so to reduce the computational burden. The refinement in space discretization should lie in the range $[40, 90]$ for the number of nodes per wavelength. On this case, results are found to be accurate and converging with an order of convergence comprised between 4 and 5, when refining either the spatial or temporal discretization.

A second BVP is introduced and solved on the time derivative of the potential in order to obtain an accurate estimation of the pressure in the fluid domain through the Bernoulli equation. This method ensures a precise computation of loads on bodies, which will further be needed to compute the movements of a freely floating body.

An multi overlapping-mesh method was also developed as a way to compute the flow precisely in the vicinity of an object subjected to incoming regular waves with a finer local grid. In case the body pierces the free surface, a second free surface is added close to the body and evolves in the body fitted mesh. Specific strategies to couple the two (here fixed) grids and associated (time varying) free surface curves were proposed. This double mesh technique ensures a precise computation of complex flow patterns which can arise due to the presence of the body.

Coupled with relaxation zones to both generate and absorb waves at extremities of the NWT, the method is shown to be accurate for several 2D cases. In particular, the double mesh approach is shown to give accurate results on a fully submerged horizontal cylinder located very close to the free surface (flume experiments by Chaplin, 1984). We have shown that the mean (drift) vertical force on the cylinder and the amplitudes of the three first harmonic of this force were properly estimated by the NWT for a range of Keulegan-Carpenter number. The nonlinear capabilities of the model are thus confirmed.

As a final validation test, the NWT is used to simulate a series of dedicated experiments performed in the wave flume at Centrale Marseille with a surface piercing rectangular barge. As the present model does not consider viscous effects, this last case proved to be

really challenging, in particular in high wave conditions and due to the presence of sharp corners at the bottom of the barge. In these conditions, indeed, dissipative terms cannot be neglected if one wants an accurate prediction of the physical values. Nevertheless, the NWT showed very good results for cases with low steepness incident waves (loads of the barge, run-up on the barge's vertical sides, transmitted and reflected waves). As the wave steepness increases, the simulations still reproduce the time evolution of free surface elevation, loads and run-up signals, but the differences in amplitude become more marked. In these conditions, errors versus the experiments can be up to 15% on the amplitude of the first harmonic of the loads. Higher harmonics tend also to be overestimated compared to the experimental results. We however like to insist of the fact that some of the cases simulated here were extreme in the sense that waves started to break on the barge and we almost reached the dewetting of the front face of the barge (as shown in the pictures of the experiment on figure 3.12). This being considered, the ability of the NWT to run on such cases and to deliver a correct (though slightly overestimated) order of magnitude of loads and run-up elevations is regarded as a valuable outcome of this study.

A fully immersed horizontal cylinder of rectangular cross section which characteristic dimension is small compared to the wave dimensions is also tested. It was selected in order to separate some of the effects observed on the free surface piercing barge. The obtained hydrodynamic coefficients are compared to literature result, to exhibit that, while their orders of magnitude are correctly retrieved, their evolution with the KC number cannot be correctly captured due to the potential assumptions: viscous and turbulent effect are expected to play a major role, as soon as the KC increases slightly.

Lastly, a rapid investigation on the problem encountered with the approach of deforming the mesh to fit the free surface are conducted. It is shown that while there is no problem theoretically and that the cell matrices are correctly inverted, some macro-cell topologies yield associated local matrices whose inversions are difficult. In these cases, the resulting coefficients of the local inverse matrix reach high values. In turn, while the iterative solver manages to reach a global residual below the specified threshold, potential values are locally very far from being correctly computed.

The rest of this work aims at coupling this potential NWT with local viscous models in the vicinity of bodies. Most of the discrepancies observed with this model are consequence of the potential hypothesis. Namely, turbulent, viscous and rotational effects being neglected, important aspects of the physics are ignored, as was shown here for the case of the rectangular barge in large wave conditions, but also for the horizontal rectangular-based cylinder.

Chapter 4

Investigations on the volume of fluid method with OpenFOAM®

Contents

4.1 Introduction 68
4.2 Models and equations 71
4.3 Numerical methods 79
4.4 Description of the test case and base parameters of the simulations . 84
4.5 Comparison of the different turbulence models 89
4.6 Sensitivity and independence studies 99
4.7 Application and comparison with experimental results 110
4.8 Conclusions on OpenFOAM® as a NWT. 114

4.1 Introduction

4.1.1 Limitations of the potential approach

The potential HPC model presented, developed and validated in chapters 2 and 3 exhibited interesting results: nonlinearities are correctly captured for a relative range of parameters. However this HPC method - and more generally all potential models - suffer from inaccuracies when pushed outside the limits of their assumptions. The two main hypotheses of such models are the irrotationality and invicid properties of the flow. Those assumptions are relevant for most of the studied cases and parameters, which makes the HPC model suitable for a broad range of studies.

However, for some particular cases - for example harsh wave conditions, which are also key aspects of the design process (see *e.g.* Fedele *et al.*, 2017) - both those assumptions can be questioned. For example, any wave breaking event, would lead to vortical and turbulent effects. Other flow aspects, such as air entrapment or boundary layers are also impossible to capture with a potential model and may have significant consequences on the flow description.

In the previous chapters, such limits of the potential model were encountered, particularly on two test cases. The horizontal fully immersed cylinder described in section 3.3.3 possesses sharp corners. Vortices and turbulence are generated. In turn the hydrodynamic coefficients are reduced when the wave height increases (controlled by the Keulegan-Carpenter number). This effect was not retrieved under the potential assumptions. Circulation effects, around the cylinder are discussed in detail in Arai (1995) and briefly summarized here. When $KC = \pi$, the length of the path of a fluid particle is equal to the body breadth. At this stage, strong circulations are at play, which implies significant rotational of the velocity field. Asymptotically when KC increases further, the flow between the oscillations can be modeled as a constant unidirectional constant flow. In that case, the Reynolds number is decisive to understand the behavior of the flow past the body. Usually, high Reynolds numbers are at play, which means that vortices are shed, flow detachments occur, and a turbulent transition takes place.

The second test case described and discussed in section 3.3.2 is similar to the latter with additional effects originating from the free surface piercing by the floating barge. Air entrapment and wave breaking are at play which is expected to result in a large energy dissipation, especially for the steepest wave cases. The potential model being energetically conservative, its limits are exceeded.

In order to develop an accurate model that couples the previously computed potential flow with a CFD approach, the open source toolbox OpenFOAM® (open Field Operation And Manipulation, under the GPL v3 licence) is selected. Here we assess its capabilities to be used as an accurate NWT in an independent manner. Throughout this work, the considered version remains constant: OpenFOAM® v17.12. The main objective of this chapter is to obtain precise and consistent results that will serve as references for the following chapters. Another goal is to identify correct simulation parameters (mesh and time discretizations, spatial discretization schemes, turbulence model, *etc.*) that will be reused later without being questioned again.

4.1.2 OpenFOAM®, a widely used toolbox

OpenFOAM® is a powerful open source package that provides many routines to manipulate as well as operate on flow fields and associated meshes . Inside the package, several solvers are available based on these functions to numerically study many types of physical phenomena. Amongst them, a multiphase solver `interFoam` is available and was extensively tested over the years for example in the context of wave hydrodynamics, especially since the development of the well-known wave dedicated toolbox `waves2Foam` by Jacobsen *et al.* (2011). A specific solver denoted `waveFoam` is included in this external toolbox and will be used here.

For example, Chen *et al.* (2014) studied the `interFoam` capabilities with in-house wave generation and absorption programs in 3D. The numerical model was applied to a vertical cylinder with the laminar assumption. Results showed a good agreement with the experiments up to the 4^{th} harmonic of the surface elevation. Another study on extreme waves with the laminar assumption is conducted in Hu *et al.* (2016). Higuera *et al.* (2013) implemented new boundary conditions to mimic the behavior of realist wave makers.

While the cost of this approach is still relatively large compared to other types of model, practical engineering applications can be studied (see *e.g.* Islam and Soares, 2018).

The choice of OpenFOAM® in many examples in the literature - and in this work in future chapters - is driven by the openness of the source code. This allows for an easy implementation of new functions and routines. Most of the referenced papers here benefited of this freedom and open source feature: as an example, complicated numerical methods such as an immersed boundary method can be developed, see *e.g.* Constant *et al.* (2017) and Riahi *et al.* (2018).

4.1.3 Turbulence modeling

While many applications are made considering the laminar version of the NS equations, some works have also been done in studying wave flows with the Reynolds Averaged Navier-Stokes (RANS) equations coupled with different turbulence closure schemes. For example, Brown *et al.* (2016) assessed the capabilities of several turbulence models to capture the behavior of a spilling breaking wave.

However, it was shown by Mayer and Madsen (2001) that the k-ω model of Wilcox (1998) is unstable under potential waves. This conclusion was recently extended to most of the commonly used turbulence model by Larsen and Fuhrman (2018), who proved that under an Airy wave, several turbulence models are formally unstable, and derived the associated growth rate of the instability. This analysis was conducted neglecting the convection and diffusion terms of the closure equations. According to the authors, this instability is thought to explain the observed overestimation of the turbulent kinetic energy (TKE) in many studies that solve for the RANS equations to simulate wave flows (Brown *et al.*, 2016; Hsu *et al.*, 2015).

An *ad hoc* correction, proposed by Mayer and Madsen (2001) and also adopted in Jacobsen *et al.* (2011), consists in modifying the production term of the TKE using the mean rotation rather than the strain rate. Thus, no production will occur in a purely potential flow.

Recently, Devolder *et al.* (2017, 2018) developed and released a modified version of some turbulence models specifically for OpenFOAM®. They first included the necessary

buoyancy modification, but also added a buoyancy production term directly into the TKE equation.

However, Larsen and Fuhrman (2018) state that, while the above mentioned modifications should be included, the instability itself is not prevented. They in turn modified several of the existing models and made them available. Seemingly this model begins to be widely used for its effectiveness, especially in engineering problems (*e.g.* Hsiao *et al.*, 2020; Patil, 2019). However, even if the stabilization is achieved with the suggested corrected model, undertow TKE profiles still exhibit relative discrepancies when compared to experiments.

During this work, such instabilities are encountered and the different available models and corrections will be discussed in further details (section 4.2.4), and then afterwards extensively tested (section 4.5).

4.1.4 Outline of this chapter

Section 4.2 focuses on briefly recalling the underlying mathematical formalism associated with both the RANS equations and the volume of fluid (VoF) approach. This section also presents the available turbulence models, with a focus on the $k-\omega$ SST version, and their associated modification suggested by Devolder *et al.* (2017) and Larsen and Fuhrman (2018) in section 4.2.4. We also take the opportunity to propose further modifications to the latter. Afterwards, the numerical methods are rapidly described in section 4.3.

After defining the selected case and the associated numerical parameters (mesh, time step size, *etc.*) in section 4.4, a thorough investigation is conducted on the available turbulence models and their respective impacts on the local flow description (section 4.5). Then, in section 4.6 many investigations are performed. Amongst them, the independence both in the mesh and time step are assessed (section 4.6.2). Finally, a comparison with literature results of the obtained hydrodynamic coefficients for different wave heights and periods is performed in section 4.7.

4.2 Models and equations

In this section, the mathematical backgrounds associated with the different employed strategies and approaches are briefly recalled. We first focus on deriving the RANS equations within a multiphase VoF framework, and this framework is also recalled. Finally, the turbulence model $k-\omega$ SST will be presented.

4.2.1 Navier-Stokes equations

The tensor form of the Cauchy momentum equation, in its conservation form, is as follows:

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) = \nabla \cdot \boldsymbol{\sigma} + \mathbf{f} \tag{4.1}$$

where $\mathbf{f}$ is the external source force and $\boldsymbol{\sigma}$ is the Cauchy stress tensor (also denoted simply the stress tensor). $\mathbf{u}$ stands for the velocity vector while ρ is classically the density of the fluid. The operation denoted with $\otimes$ is the tensor product, defined such that $(\mathbf{a} \otimes \mathbf{b})_{ij} = \mathbf{a}_i \mathbf{b}_j$. Within the Newtonian fluid assumption, the stress tensor can be further developed into:

$$\begin{aligned} \boldsymbol{\sigma} &= -p\mathbb{1} + \mathbb{T} \\ &= -p\mathbb{1} + 2\mu\mathbb{D} - \frac{2}{3}\mu\nabla \cdot \mathbf{u}\ \mathbb{1} \\ &= -p\mathbb{1} + \mu\left[\nabla\mathbf{u} + \nabla\mathbf{u}^T\right] - \frac{2}{3}\mu\nabla \cdot \mathbf{u}\ \mathbb{1} \end{aligned} \tag{4.2}$$

where μ is the dynamic viscosity, $\mathbb{T}$ is the deviatoric (shear) stress tensor, $\mathbb{1}$ is the identity matrix, and $\mathbb{D} = \frac{1}{2}\left[\nabla\mathbf{u} + \nabla\mathbf{u}^T\right]$ is the strain rate tensor (or rate-of-strain tensor). Note that some assumptions have been made to derive this equation: i) the stress does not directly depends on the velocity $\mathbf{u}$, but on its derivative $\nabla\mathbf{u}$, and the dependence in the latter is linear, ii) the fluid is isotropic and iii) the bulk viscosity was neglected, thus the mechanical pressure is assumed to be equal to the thermodynamical pressure.

Equation (4.1) can be expressed in terms of the pressure and the shear stress tensor. Along with the conservation of mass, the Navier-Stokes equations are derived:

$$\begin{cases} \nabla \cdot \rho\mathbf{u} = 0 & \text{(4.3a)} \\ \dfrac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) = -\nabla p + \nabla \cdot \mathbb{T} + \mathbf{f} & \text{(4.3b)} \end{cases}$$

4.2.2 Reynolds Averaged Navier-Stokes (RANS) method

A brief summary of the RANS method and the obtained equations is given here. We first separate the velocity vector into a statistical mean component and a perturbation (also called fluctuation) component.

$$\mathbf{u} = \bar{\mathbf{u}} + \mathbf{u}' \tag{4.4}$$

$$\text{where } \bar{\mathbf{u}} = \frac{1}{\delta t}\int_t^{t+\delta t} \mathbf{u} dt \tag{4.5}$$

The Reynolds averaged versions of eqs. (4.3a) and (4.3b) must also be satisfied, and taken into account that most operations are linear, in particular the shear stress tensor is linear with respect to $\mathbf{u}$, thus $\overline{\mathbb{T}(\mathbf{u})} = \mathbb{T}(\bar{\mathbf{u}}) = \bar{\mathbb{T}}$, the RANS equations can be derived:

$$\begin{cases} \nabla \cdot \rho\bar{\mathbf{u}} = 0 & \text{(4.6a)} \\ \dfrac{\partial \rho\bar{\mathbf{u}}}{\partial t} + \nabla \cdot \rho\overline{\mathbf{u} \otimes \mathbf{u}} = -\nabla\bar{p} - \nabla \cdot \bar{\mathbb{T}} + \bar{\mathbf{f}} & \text{(4.6b)} \end{cases}$$

These new equations are very similar to the original NS eqs. (4.3a) and (4.3b), using $\bar{\mathbf{u}}$ in place of $\mathbf{u}$. However, the nonlinear advective terms are different: $\overline{\mathbf{u} \otimes \mathbf{u}} \neq \bar{\mathbf{u}} \otimes \bar{\mathbf{u}}$. Developing this terms further, applying the decomposition eq. (4.4), and taking advantage of the fact that $\overline{\mathbf{u}'} = 0$, yields:

$$\begin{aligned} \frac{\partial \rho\bar{\mathbf{u}}}{\partial t} + \nabla \cdot \rho\overline{(\bar{\mathbf{u}} + \mathbf{u}') \otimes (\bar{\mathbf{u}} + \mathbf{u}')} &= -\nabla\bar{p} - \nabla \cdot \bar{\mathbb{T}} + \bar{\mathbf{f}} \\ \frac{\partial \rho\bar{\mathbf{u}}}{\partial t} + \nabla \cdot \rho(\bar{\mathbf{u}} \otimes \bar{\mathbf{u}} + \overline{\mathbf{u}' \otimes \mathbf{u}'}) &= -\nabla\bar{p} - \nabla \cdot \bar{\mathbb{T}} + \bar{\mathbf{f}} \end{aligned} \tag{4.7}$$

From the nonlinearity of the convection term, a perturbation terms arises, known as the Reynolds Stress tensor, that represents the convection of the perturbation by itself $\nabla \cdot \rho\overline{\mathbf{u}' \otimes \mathbf{u}'}$. If one wants to capture numerically this tensor and thus resolve the problem down to the scale of the turbulence, the needed cell size is of the same order of magnitude as the smallest turbulent scale (*i.e.* Kolmogorov scale). In that case, the model would be similar to a direct numerical simulation (DNS). This is out of reach in terms of computational cost for our particular purpose: we need to simulate large domains with relatively high Reynolds numbers and be able to explore different physical parameters, *e.g.* wave height, period. The associated computational cost would be prohibitive. Thus, this term is modeled: we assume a local linear relationship of the Reynolds Stress tensor with respect to the shear stress tensor. This is the Boussinesq assumption:

$$\begin{aligned} -\rho\overline{\mathbf{u}' \otimes \mathbf{u}'} &= \frac{\mu_t}{\mu}\bar{\mathbb{T}} - \frac{2}{3}\rho k\mathbb{1} \\ &= \mu_t\left[\nabla\bar{\mathbf{u}} + \nabla\bar{\mathbf{u}}^T\right] - \frac{2}{3}\mu_t\nabla \cdot \bar{\mathbf{u}}\ \mathbb{1} - \frac{2}{3}\rho k\mathbb{1} \end{aligned} \tag{4.8}$$

where μ_t is an unknown coefficient that is not constant. This variable is denoted the turbulent dynamic viscosity due to its similarity with the dynamic viscosity μ. k is the turbulent kinetic energy $k = \dfrac{1}{2}\overline{\mathbf{u}' \cdot \mathbf{u}'}$. Thus, the resulting system of equations, also named the RANS equations is:

$$\begin{cases} \nabla \cdot \rho\bar{\mathbf{u}} = 0 & \text{(4.9a)} \\ \dfrac{\partial \rho\bar{\mathbf{u}}}{\partial t} + \nabla \cdot \rho\bar{\mathbf{u}} \otimes \bar{\mathbf{u}} = -\nabla\bar{p} - \nabla \cdot \underbrace{\left(\mu_{eff}\left[\nabla\bar{\mathbf{u}} + \nabla\bar{\mathbf{u}}^T\right] - \dfrac{2}{3}\mu_{eff}\nabla \cdot \bar{\mathbf{u}}\mathbb{1} - \dfrac{2}{3}\rho k\mathbb{1}\right)}_{\mathbb{T}_{eff}(\bar{\mathbf{u}})} + \bar{\mathbf{f}} & \text{(4.9b)} \end{cases}$$

where the effective viscosity μ_{eff} is defined as $\mu_{eff} = \mu + \mu_t$. In that framework, two new variables have been incorporated into our problem (k and μ_t). Thus, closure(s) equation(s)

are required for this system, they are referred to as turbulence models. For example, $k-\epsilon$ or $k-\omega$ are two equations closure models while Spalart-Allmaras (Spalart and Allmaras, 1992) is a one-equation closure model. In this study, different versions of the $k-\omega$ SST of Menter (1994) will be used. A description of the corresponding variations of this closure model is done in section 4.2.4.

4.2.3 Volume of Fluid (VoF) method

Many methods exist to capture and simulate the evolution of the interface between two phases of a given flow. One of them, that does no require any mesh deformation, is the Volume of Fluid method introduced by Hirt and Nichols (1981). The RANS modeling within a VoF background is for example discussed in detail in Fan and Anglart (2019). In this section, the approach and associated equations are summarized.

With this method, the equations are represented in a continuous way across the interface: a scalar function that represents the volume fraction of a given fluid in a given cell is defined. Thus, the two immiscible fluids are represented and solved as one with varying physical properties. In OpenFOAM®, the volume fraction is denoted α; it is a function of t and $\mathbf{x}$. Thus when this scalar equals unity in a given cell, this cell is full of fluid 1, *i.e.* water in our case. If it is zero, this cell is full of fluid 2, *i.e.* air in our case. Any value in between means that the cell free surface crosses this cell and gives the relative fraction of water inside this cell. Density and mixture viscosity are computed as

$$\rho = \alpha\rho_1 + (1-\alpha)\rho_2 \tag{4.10}$$

$$\mu = \alpha\mu_1 + (1-\alpha)\mu_2 \tag{4.11}$$

where ρ_1, ρ_2 are respectively the density of the water and air, and μ_1 and μ_2 their dynamic viscosities. The following paragraphs describe the different equations of the two-phase system.

4.2.3.1 Incompressibility

As the phases are assumed to be incompressible and immiscible, the continuity equation in each phase remains:

$$\nabla \cdot \mathbf{u} = 0 \tag{4.12}$$

4.2.3.2 Transport of the volume fraction

The general continuity equation for any flow of variable density is given by:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \rho\mathbf{u} = 0 \tag{4.13}$$

Plugging the VoF decomposition eq. (4.10) that parametrizes the evolution of the density between ρ_1 and ρ_2 yields:

$$\frac{\partial \alpha}{\partial t}\rho_1 - \frac{\partial \alpha}{\partial t}\rho_2 + \rho_1 \nabla \cdot \alpha\mathbf{u} - \rho_2 \nabla \cdot \alpha\mathbf{u} = 0 \tag{4.14}$$

It is possible to simplify this equation by dividing by the density difference $\rho_1 - \rho_2$ in order to derive the transport equation of the volume fraction:

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot \alpha \mathbf{u} = 0 \tag{4.15}$$

An equivalent form, taking into account the incompressibility eq. (4.12), valid for each fluid independly, reads:

$$\frac{\partial \alpha}{\partial t} + \mathbf{u} \nabla \alpha = 0 \tag{4.16}$$

Remark. *Note that the transport equation for α is just another form of the continuity equation, thus mass conservation is automatically satisfied in the VoF framework.*

4.2.4 Turbulence modeling

Due to the relatively high Reynolds numbers that are locally reached, a model of the Reynolds stress tensor in the vicinity of the body can be needed to model the Reynolds stress tensor. A laminar computation signifies fixing the turbulent viscosity $\nu_t = 0$. This assumption is equivalent to neglecting the Reynolds stress tensor, and thus also the advection of the fluctuating component of the velocity by itself.

This section focuses on the original $k-\omega$ SST model available in OpenFOAM® as well as the derived models suggested by Devolder *et al.* (2017) and Larsen and Fuhrman (2018). Suggestions to improve the behavior of the latter in the presence of vortex shedding are also given. Results and comparisons between these variations will be later presented in section 4.5.

4.2.4.1 $k-\omega$ SST, original model in OpenFOAM®

Withing the OpenFOAM® toolbox a lot of model are available. In this work, the $k-\omega$ SST model proposed by Menter (1994) is selected for its relative capability to correctly capture the turbulence at different scales: vortex shedding, flow detachment and turbulence generation in the wall vicinities but also the larger scale effects where the flow is mostly laminar. The corresponding closure equations for a two-phase fluid are:

$$\begin{cases} \dfrac{\partial \rho k}{\partial t} + \nabla \cdot \rho \mathbf{u} k = \rho P_k - \beta^* \rho \omega k + \nabla \cdot [\rho(\nu + \sigma_k \nu_t) \nabla k] & \text{(4.17a)} \\ \dfrac{\partial \rho \omega}{\partial t} + \nabla \cdot \rho \mathbf{u} \omega = \rho P_\omega - \beta \rho \omega^2 + \nabla \cdot [\rho(\nu + \sigma_\omega \nu_t) \nabla \omega] + 2(1 - F_1) \dfrac{\rho \sigma_{\omega_2}}{\omega} \nabla k \cdot \nabla \omega & \text{(4.17b)} \\ \nu_t = \dfrac{a_1 k}{\max(a_1 \omega, \sqrt{2\mathbb{D} : \mathbb{D}} F_2)} & \text{(4.17c)} \\ P_\omega = \dfrac{\gamma \omega}{k} \mathbb{G} & \text{(4.17d)} \\ P_k = \min(\mathbb{G}, 10 \beta^* k \omega) & \text{(4.17e)} \\ \mathbb{G} = 2 \nu_t \mathbb{D} : \nabla \mathbf{u} & \text{(4.17f)} \end{cases}$$

ϕ	σ_k	σ_ω	β	γ
ϕ_1	0.85034	0.5	0.075	0.5532
ϕ_2	1.0	0.85616	0.0828	0.4403

Table 4.1: Values of the different parameters of the $k-\omega$ SST model in OpenFOAM®. Table from Devolder *et al.* (2017).

We recall that $k = \frac{1}{2}\overline{\mathbf{u}' \cdot \mathbf{u}'}$ is the mean turbulent kinetic energy, ω denotes the specific dissipation rate, and P_k and P_ω are their respective production terms. Note that the operator ":" is Frobenius inner product $\mathbb{A} : \mathbb{B} = A_{ij}B_{ij}$.

Remark. *This system is in reality a blending between the $k-\omega$ model (Wilcox, 1998) (in the vicinity of the walls) and the $k-\epsilon$ model (Launder and Sharma, 1974). The blending functions F_1 and F_2 evolve from 1 to 0 when moving away from the wall. Also note that the values of the coefficient σ_k, σ_ω, β and γ also evolve with respect to the distance to the closest wall as $\phi = F_1\phi_1 + (1-F_1)\phi_2$. Thus, the value of these parameters should be defined in each zone,* i.e. *indexed 1 and 2. In this work, the default values are used for those functions, and are recalled in table 4.1.*

However, an important note here is that the native OpenFOAM® $k-\omega$ SST model does not directly implement eq. (4.17) but instead its constant density equivalent: the ρ are taken out of the derivatives and simplified. In a VoF framework, the density however exhibits large gradients at the free surface. This will of course, greatly impact the results in this zone. In practice, a significant increase of the TKE in the free surface vicinity zone is observed, which leads to a large damping of the wave fields and kinematics.

4.2.4.2 Modification suggested by Devolder *et al.* (2017)

The above discussed shortcoming is corrected by Devolder *et al.* (2017). They proposed and released a model that implements the required correction for a multiphase flow (takes into account the modification of ρ). Still, an unphysical rise of the TKE is obtained. To counteract this effect, a buoyancy term is added directly in the RHS of the TKE equation:

$$G_b = -\frac{\mu_t}{\sigma_b}\mathbf{g} \cdot \nabla\rho \tag{4.18}$$

where $\mathbf{g}$ is the gravity acceleration vector and σ_b a coefficient set as default to $\sigma_b = 0.85$. With this term - treated implicitly in the proposed modified model - any large gradient of ρ in the direction of the gravity will instantly drive the turbulent viscosity to zero, switching to a laminar model close to the free surface.

Notice that the added/modified terms are different only when the density varies. Thus, when the density is constant, this model reduces to the native model, and the exact same behavior should be retrieved close to a body. Hereafter, this model will be referred to as the "buoyant" $k-\omega$ SST model.

4.2.4.3 Modification suggested by Larsen and Fuhrman (2018)

While the latter model greatly reduces the over generation of turbulence in the free surface zone, according to Larsen and Fuhrman (2018), the turbulent viscosity produced in the

area where the flow is essentially potential is still orders of magnitude higher than it should be. These authors have shown that in the case of a potential flow driven by potential water waves, the turbulence model is unconditionally unstable leading to high eddy viscosity values and thus a significant unphysical decay of the waves in the propagation zone. To counteract this shortcoming, they suggested a modification of the model. Their main analysis is focused on the $k-\omega$ model, but equivalent variations of several other models are also developed. For example, the $k-\omega$ SST model is stabilized by introducing a new limiter that applies to the eddy viscosity denominator:

$$\nu_t = \frac{a_1 k}{\max(a_1\omega, \sqrt{S_y}F_2, a_1\lambda_2\frac{\beta}{\beta^*\gamma}\frac{S_y}{S_k}\omega)} \tag{4.19}$$

where $S_k = 2\mathbb{D} : \mathbb{D}$ and $S_y = 2\Omega : \Omega$. Here, Ω is the mean rotation rate tensor (*i.e.* anti-symmetric part of the velocity gradient) $\Omega = 1/2(\nabla\mathbf{u} - \nabla\mathbf{u}^T)$. The aim of the new limiter is to increase the denominator whenever the rate of strain is large compared to the rotation rate, thus reducing the value of ν_t. Note that diminishing ν_t also has a retroacting effect on the production terms of k and ω trough the variable $\mathbb{G}$ (see eqs. (4.17d) to (4.17f)). Physically, the goal is to reduce the value of ν_t itself only in regions where the rate of strain tensor is high compared to the rotation rate tensor ($\mathbb{D} : \mathbb{D} \gg \Omega : \Omega$), for example in the almost potential zones beneath water waves), and retrieve the original model whenever rotation is at play. Note that a new parameter, λ_2 is introduced with this model which sets the threshold of activation of this limiter.

Also note that a buoyancy term, corresponding to eq. (4.18), is also taken into account and added in the TKE equation even tough its definition in the Larsen and Fuhrman (2018) model differs slightly.In practice, this model with $\lambda_2 = 0$ effectively yields the same results as the buoyant model.

For the rest of this work, this model will be denoted the stabilized $k-\omega$ SST model.

4.2.4.4 Additional discussions and suggestions

In section 4.5 it will be shown that, on the test case of interest, the stabilized model is intrusive in the body vicinity, even if it should not be activated in theory. Some suggestions to counteract this shortcoming are given in this §. Amongst them, the first two will be implemented and tested. The associated results and discussions are given in section 4.5.4.

Option a: limiting the decrease of the skew term. Because the stabilization method activates when $\Omega : \Omega = 0$, for example between two counter-rotative vortices, the first intuitive possibility would be to limit the minimum value of this term such that the stabilizer stays efficient in the potential regions but does not activate as soon as the velocity rotational cancels out. For example, one could redefine the limiting scheme of ν_t as

$$\nu_t = \frac{a_1 k}{\max(a_1\omega, \sqrt{S_y}F_2, a_1\lambda_2\frac{\beta}{\beta^*\gamma}\frac{S_y}{\max(S_k, \lambda_2\lambda_3 S_y)}\omega)} \tag{4.20}$$

with λ_3 a small constant coefficient. Note that, as stated by Larsen and Fuhrman (2018), the growth rate of the TKE of their stabilized SST model - obtained with simplifying

assumptions such as neglecting the diffusive and convecting part of the TKE and ω equations - is given by:

$$\Gamma_\infty = \frac{(S_k - \lambda_2 S_y)\beta^*}{\lambda_2}\sqrt{\frac{\alpha}{S_y\beta}} \tag{4.21}$$

The suggested modification would thus yield a formally stable model given that

$$\max(\lambda_2\lambda_3 S_y, S_k) \leq \lambda_2 S_y \tag{4.22}$$

While the second term of the max in eq. (4.20) reduces to the same stability condition as in Larsen and Fuhrman (2018) (*i.e.* $S_y/S_k \geq 0$), the newly introduced condition simply corresponds to $\lambda_3 \leq 1$. The aim of this modification is to shut down the ν_t damping as soon as the rate of rotation reaches a value large enough to be actually compared to the rate of strain. This suggestion was implemented and tested, and the corresponding results will be shown in section 4.5.4. However, no improved behavior of the turbulent viscosity field will be obtained with this modification.

Option b: smoothing the skew and symmetric variables. The main reasoning of Larsen and Fuhrman (2018) is valid: in potential zones, S_k is approximately null while S_y is not, while in turbulence/vortical driven zones S_y is *overall* of significant magnitude. The only flaw is that S_y can - and will - *locally* reach almost null value.

Thus, one could improve the method itself by computing the limiter ratio S_y/S_k in a different way, such that it is less dependent on the cell value itself. For example, the denominator (currently S_k) could be defined as a mean value of $2\Omega : \Omega$ within a certain distance of the considered cell (instead of the cell value of S_k itself). This would smooth out the sharp reached minima in-between vortices but still be triggered when the flow is on average driven by potential effects. Moreover, that would remove the high dependence on the cell size in between the vortices. Even if this exact smoothing function seems very attractive in principle, it has not been implemented and it is thought that its major drawback would be the required computational cost. Note however, that another smoothing method, described below, was implemented and tested. The general expression for the eddy viscosity, with a smoothing of the stabilized model, would read:

$$\nu_t = \frac{a_1 k}{\max(a_1\omega, \sqrt{S_y}F_2, a_1\lambda_2\frac{\beta}{\beta^*\gamma}\frac{s(S_y)}{s(S_k)}\omega)} \tag{4.23}$$

with s a smoothing function. In this OpenFOAM® version (17.12) a different smoothing function than the one suggested above is available that - trough multiple iterations - reduces the highest value of a given function not allowing to increase further than a given ratio times the neighboring cell values: for all c_a and c_b two neighbor cells, ϕ a given field and m_r a given parameter, if $\phi(c_a) > m_r\phi(c_b)$ then the c_a value is set to $\phi(c_a) = m_r\phi(c_b)$, and repeat until convergence.

Based, on this available function, the suggested stabilization modification, eq. (4.23), was implemented. A modification of the existing routine was implemented, that smooth out the minima instead: the new smoothing procedure ensures that a given cell value is not lower than $f_s(= 1/m_r)$ times any neighbor cell value. At $f_s = 0$ the smoothing function is thus not effective (and only costs a verification of algorithmic complexity

$O(n_{\text{cell}})$). When at $f_s = 1$ no two neighbor cells values can differ: the entire mesh would share the same value. In that case, the higher reached value is enforced for all cells. This basically suppresses entirely the damping suggested by Larsen and Fuhrman (2018), but with a large associated computational price ($O(n_{\text{cell}}^2)$).

The author is aware that this procedure is not optimal and should be regarded as a first step as - not depending on the gradients but on the cell values themselves - the smoothing will depend on the chosen spatial discretization. Tests are conducted in the corresponding section 4.5.4.

Option c: investigations on wall functions Lastly, another possibility in order to really suppress the stabilizer in the vortex evolution zone would be to include one of the wall distance function in the limiter so as to shut it off in the vicinity of the wall. This idea was however not tested during this work, and would be only applicable to few turbulence closure models.

4.3 Numerical methods

4.3.1 Finite Volume Method (FVM)

The equations to solve are discretized using the finite volume method (FVM). A complete mathematical description of the different FVM are available in several textbooks, for instance in Eymard *et al.* (2000). In addition, great details on the implementation of this method, in particular in OpenFOAM®, are given for example in Moukalled *et al.* (2016). More VoF focused explanations were found in the document by Damian (2012). Furthermore for a detailed description of the method and the OpenFOAM® implementation, the reader is referred to the *Ph.D.* thesis of Jasak (1996) which serves as a reference for many of the OpenFOAM® implementations. A brief summary and explanations of the main equations and numerical methods are recalled here.

4.3.1.1 Generic equation

Let's consider the standard form of an equation for the transport of a quantity q:

$$\frac{\partial \rho q}{\partial t} + \nabla \cdot (\rho \mathbf{u} q) - \nabla \cdot (\rho \Gamma \nabla q) = S(q) \tag{4.24}$$

where Γ is the diffusion parameter for q and $S(q)$ the source term. Note that this quantity q can be either a scalar or a vector (in which case, the operation between the velocity and the variable of interest is the outer product: $\mathbf{u}q \equiv \mathbf{u} \otimes q$).

For any volume V_P the integral form of eq. (4.24) should be satisfied at every instant t:

$$\int_t^{t+\Delta t} \left[\underbrace{\frac{\partial}{\partial t} \int_{V_P} \rho q dV}_{\text{Time derivative}} + \underbrace{\int_{V_P} \nabla \cdot \rho \mathbf{u} q dV}_{\text{Convection}} - \underbrace{\int_{V_P} \nabla \cdot (\rho \Gamma \nabla q) dV}_{\text{Diffusion}} \right] dt = \int_t^{t+\Delta t} \underbrace{\int_{V_p} S(q) dV}_{\text{Source}} dt \tag{4.25}$$

The different terms in this equation will be approximated by discretizing them both in time and space.

4.3.1.2 Discretization of the terms

The volume is discretized into cells, small enough so that the behavior of q inside a given cell is approximately linear in space. A focus will be briefly made on how each term of eq. (4.25) is discretized. Each volume integral will be projected onto the faces closing the cell volume V_P using the Gauss theorem.

Convection term A second-order accurate discretization of the divergence of a given vector field $\mathbf{a}$ is expressed as:

$$\int_{V_P} \nabla \cdot \mathbf{a} dV = \sum_{faces} (\mathbf{S}_f \cdot \mathbf{a}_f) \tag{4.26}$$

where $\mathbf{a}_f$ is the value at the center of the face and $\mathbf{S}_f$ its - pointing outward - area vector. Thus, the convective term of any quantity q can be expressed as:

$$\int_{V_P} \nabla \cdot (\rho \mathbf{u} q) dV = \sum_{faces} (\rho_f \mathbf{S}_f \cdot \mathbf{u}_f) q_f = \sum_{faces} F_f q_f \tag{4.27}$$

where F_f is the mass flux through the face f. At that stage an interpolation method needs to be selected in order to obtain the value q_f at the face from the values of q at the cell centers. Many discretizations schemes are available in OpenFOAM® and the selection is defined in the `fvScheme` file. The most notable and commonly used types of scheme are the Central Differencing, Upwind Differencing and Blended Differencing schemes. For more information the reader is referred to the literature on the FVM method given above. Note that this eq. (4.27) links, in an implicit way, the value of the quantity q in a given cell to the neighbor cell values of the same quantity *via* the faces values. However, as will be discussed in section 4.3.1.4, this equation is also applied explicitly, for example at the previous time step.

Given the density and velocity fields, the needed values of ρ_f and $\mathbf{u}_f$ are explicitly interpolated at the center of the face. In OpenFOAM®, the mass flux trough the faces F_f is stored as a "surface field", *i.e.* defined on the mesh faces, and denoted `rhoPhi`.

Remark. *When the variable q is the velocity $\mathbf{u}$ the convection term is nonlinear. In order to linearise the equation, the mass flux will be fixed during a given resolution, using the "old" values of the velocity field (explicit interpolations). Afterwards, a loop can be done until convergence,* i.e. *until F_f and $\mathbf{u}$ are consistent (`PIMPLE` algorithm). This will be further discussed in the coupled velocity-pressure scheme section 4.3.2*

Diffusion term Given a diffusion parameter Γ - which is not necessarily space independent - for the quantity q, and using again the Gauss theorem, the diffusion term is discretized as:

$$\int_{V_P} \nabla \cdot \rho \Gamma \nabla q dV = \sum_{faces} \mathbf{S} \rho_f \Gamma_f (\nabla q)_f \tag{4.28}$$

In that equation, an evaluation of the *gradient* of q is needed at the faces. Instead of trying to evaluate it directly, we focus on its product with the surface normal vector. The surface normal is decomposed into two components $\mathbf{S}_f = \mathbf{S}_{fn} + \mathbf{S}_{ft}$, where $\mathbf{S}_{fn}$ is collinear with vector $\mathbf{PN}_f$, with P the centroid of V_P and N_f the centroid of the neighbor cell trough the face f. Note that this decomposition is not unique, and a choice has thus to be made. The different existing decomposition methods are, however, not discussed here but can be found in the previously-mentioned references. First the orthogonal part of the product of $\mathbf{S}_f$ $(\nabla q)_f$ is calculated with:

$$\mathbf{S}_{fn} (\nabla q)_f = \|\mathbf{S}_{fn}\| \frac{q_N - q_P}{\|\mathbf{PN}_f\|} \tag{4.29}$$

Then the non-orthogonal part uses the Gauss theorem to compute the gradient at the center of the cell from the value of q at the faces:

$$(\nabla q)_P = \frac{1}{V_P} \sum_{faces} \mathbf{S_f} q_f \tag{4.30}$$

to further linearly interpolate ∇q onto the faces, using the classical equation:

$$(\nabla q)_f = f_x \nabla q_P + (1 - f_x)\nabla q_N \tag{4.31}$$

where f_x is the distance between P and the intersection of $\mathbf{PN}_f$ with the face. The latter equation is then multiplied by $\mathbf{S}_{ft}$ to compute the non-orthogonal correction.

Note that this method could be directly applied to the complete term. However the numerical stencil of this method is larger, as obtaining $(\nabla q)_f$ requires the gradient of q at both neighboring cells, which in turns requires the value of q at all their faces, which, again requires the value of q at all up to the second neighboring cells. Denoting $N^{(i)}$ the i^{th} neighbor of the volume V_P, it is possible to represent the requirements as:

$$\boxed{(\nabla q)_f} \xleftarrow[\text{on } \nabla q]{\text{eq. (4.31)}} \boxed{(\nabla q)_N \forall N^{(1)}} \xleftarrow{\text{eq. (4.30)}} \boxed{q_f \forall N^{(1)}} \xleftarrow[\text{on } q]{\text{eq. (4.31)}} \boxed{q_N \forall N^{(2)}}$$

Source term The source term is linearized, with respect to q, inside the cell. We obtain S_0 and S_1 such that $S(q) = S_0 + S_1 q$. Then the respective volume integral is given by:

$$\int_{V_P} S(q) dV = S_0 V_P + S_1 q_P V_P \tag{4.32}$$

4.3.1.3 Boundary Conditions

Boundary conditions are simply enforced whenever the corresponding face value is needed, making use of the assumption that the imposed value q_b (or gradient of the value in the case of a Neumann boundary condition $(\nabla q)_b$) is constant along the whole boundary face. For further detail, see Jasak (1996, p. 93-96).

4.3.1.4 Temporal discretization

When assuming that the mesh is constant in time, those spatial discretizations applied onto eq. (4.25) yield the "semi-discretized" equation:

$$\int_t^{t+\Delta t} \left[\left. \frac{\partial \rho q}{\partial t} \right|_P V_P + \sum_{faces} F_f q_f + \sum_{faces} \mathbf{S} \rho_f \Gamma_f (\nabla q)_f \right] dt = \int_t^{t+\Delta t} (S_0 V_p + S_1 V_p q_p) dt \tag{4.33}$$

In order to obtain the fully discretized form (*i.e.* in space and time), the time discretization assumes a linear behavior of the quantities in time i.e. $q^n = q^{n-1} + \Delta t \left(\frac{\partial q}{\partial t} \right)^{n-1}$ where the exponents represent the time step. Together with the face-to-cell interpolations presented above, the total equation, tough not written in its entire form here, can be expressed as:

$$\mathbb{A} q^n + \mathbb{H}(q^n) = \mathbf{R} \tag{4.34}$$

The matrix $\mathbb{A}$ is the diagonal part of the obtained linear system. Its terms mainly arise from the time derivative and the linear part of the source term, but also when implicit interpolations of the values at the faces q_f are requested as the value of the cell itself

is needed and maybe asked implicitly. The non-diagonal part $\mathbb{H}(q^n)$ contains all terms corresponding to the influence of others cells on the given cell. The source term $\mathbf{R}$ includes all contributions for which the new value q^n is not needed. Of course, source terms are contained in this RHS, but also all explicit values resulting from the different interpolations. Last time step values are also computed and added to $\mathbf{R}$.

The computation of eq. (4.33) to obtain the matrices of eq. (4.34) depends a lot on the spatial and temporal discretizations schemes. For example, the explicit temporal discretization scheme makes the assumption that the face values do not change significantly in time, and thus the previous time step values can be used whenever a face value is required. With this method, no term appear in $\mathbb{H}$ and the new field can be obtained explicitly without solving a system (only inverting a diagonal matrix).

For the spatial differencing schemes, the Upwind Differencing scheme uses the values of either q_P or q_{N_f} to compute the values at the faces depending on the sign of the flux at those faces, whereas the Central Differencing scheme uses both and combine them linearly.

In this work, the Euler time differencing scheme as been used for the time discretization. In that method which is first order accurate, the new faces values are implicitly expressed, *i.e.* the current unknown value is needed to compute q_f.

4.3.2 Coupled velocity-pressure system and its resolution

4.3.2.1 The system of semi-discretized equations

Given a set of equations, either eqs. (4.3a) and (4.3b) or eqs. (4.9a) and (4.9b), it is possible to obtain the "semi-discretized" form of the momentum equation, as presented in section 4.3.1:

$$\mathbb{A}\mathbf{u} = \mathbb{H}(\mathbf{u}) + \mathbf{R} \underbrace{-\nabla p + \rho \mathbf{g}}_{-\nabla p_d} \tag{4.35}$$

Note that the source term $\mathbf{R}$ could include both the dynamic and static pressure but as p is not known, a special treatment will be done in order to solve for it. Here, we define p_d the dynamic pressure as $p_d = p - \rho \mathbf{g} \cdot \mathbf{x}$.

Remark. *The gradient of the dynamic pressure might not be exactly $-\nabla p + \rho \mathbf{g}$. Particularly, it is not true when the volume mass ρ is not uniform. In our case, this remark applies close to the free surface, and a correcting term is implemented in the source term, accounting for the difference between the gradient of $\rho \mathbf{g} \mathbf{x}$ and $\rho \mathbf{g}$.*

Noticing that $\mathbb{A}$ is diagonal and thus can be easily inverted, the gradient of the pressure is expressed as a function of the rest. The divergence operator is then applied to the obtained equation to yield the Poisson equation for the pressure:

$$\nabla \cdot \left(\frac{1}{\mathbb{A}} \nabla p_d \right) = \nabla \cdot \frac{\mathbb{H}(\mathbf{u}) + \mathbf{R}}{\mathbb{A}} \tag{4.36}$$

It is also possible to derive an equation from eq. (4.35) that explicitly gives the velocity (the fact that in reality $\mathbf{u}$ is present in $\mathbb{H}(\mathbf{u})$ will be taken into account later):

$$\mathbf{u} = \frac{1}{\mathbb{A}} \left(\mathbb{H}(\mathbf{u}) + \mathbf{R} - \nabla p_d \right) \tag{4.37}$$

Together eqs. (4.36) and (4.37) form the coupled velocity-pressure system.

4.3.2.2 The `PISO` and `PIMPLE` loops

In order to find $(p, \mathbf{u})$ that are solutions to the above system of equations, a `PISO` loop (Issa, 1986) is used. First, we define $\mathbb{h}(\mathbf{u}_f)\mathbf{u}$ such that $\mathbb{H}(\mathbf{u}) = \mathbb{h}(\mathbf{u}_f)\mathbf{u}$ to represent the fact that $\mathbb{H}$ depends nonlinearly on $\mathbf{u}$.

The algorithm can be summarized as:

- $(p^0, \mathbf{u}^0)$ are the velocity and pressure at the beginning of the `PISO` algorithm
- The face values of the velocity $\mathbf{u}_f^0$ are computed
- The momentum equation is discretized, in order to obtain $\mathbb{A}$ and $\mathbb{h}(\mathbf{u}_f^0)$.
- if the momentum predictor is activated, solve the equation eq. (4.35) using the current pressure p^0. We obtain a predicted value of $\mathbf{u}$.
- `PISO` loop
  - Update the values of the non diagonal part $\mathbb{h}(\mathbf{u}_f^0)\mathbf{u}$.
  - The pressure equation eq. (4.36) is defined and solved. A new pressure field p is obtained.
  - The velocity explicitly is updated using eq. (4.37). A new velocity field $\mathbf{u}$ is obtained.

In eq. (4.37), the velocity is explicitly updated from a given pressure field $(-\nabla p_d/\mathbb{A})$ and a given velocity field $(-\mathbb{H}(\mathbf{u})/\mathbb{A})$. Its means that on a given `PISO` iteration the new pressure gradient has an impact on the newly computed velocity field. However, the velocity is explicitly updated: $\mathbf{u}$ is thus not strictly solution of eq. (4.37) (the RHS is computed from a fixed velocity field). This procedure needs to be repeated until convergence.

At the end of the `PISO` loop, the velocity and pressure fields are consistent and solution of the nonlinear system composed of eqs. (4.36) and (4.37). However $\mathbb{H} = \mathbb{h}(\mathbf{u}_f)\mathbf{u}$ also depends on the face flux values which could also be updated from the new values of the velocity. This is, however, not done and thus the modification of the face flux is supposed to have a smaller impact on the $\mathbb{H}(\mathbf{u})$ term than a modification of $\mathbf{u}$ itself. In other words, the nonlinear coupling is assumed to be of lesser importance than the velocity-pressure coupling. To take that into account anyway, the `SIMPLE` (`PIMPLE` when in conjunction with `PISO`) algorithm is available in OpenFOAM®. This new loop encapsulates the `PISO` algorithm. The overall algorithm can be summarized as:

- `PIMPLE` loop
  - Compute α from $\mathbf{u}$ and p.
  - Compute the face values of $\mathbf{u}_f$ from the velocity field and update $\mathbb{h}(\mathbf{u}_f)$.
  - Perform the previously explained algorithm, in particular, the `PISO` loop until convergence.
  - Solve for all other equations of our system. For example, if the flow is turbulent, compute the effective viscosity.

This algorithm is mainly used to allow for larger time step. However, in the literature, the `PIMPLE` loop seems to be rarely used, enforcing a number of iterations (`nOuterCorrector`) for the latter to 1.

4.4 Description of the test case and base parameters of the simulations

This section aims to describe the test case that will will be considered for the rest of this chapter. Numerical simulation parameters and schemes are also precised: they will be referred to as the "base" parameters, in the vicinity of which investigations will be later conducted.

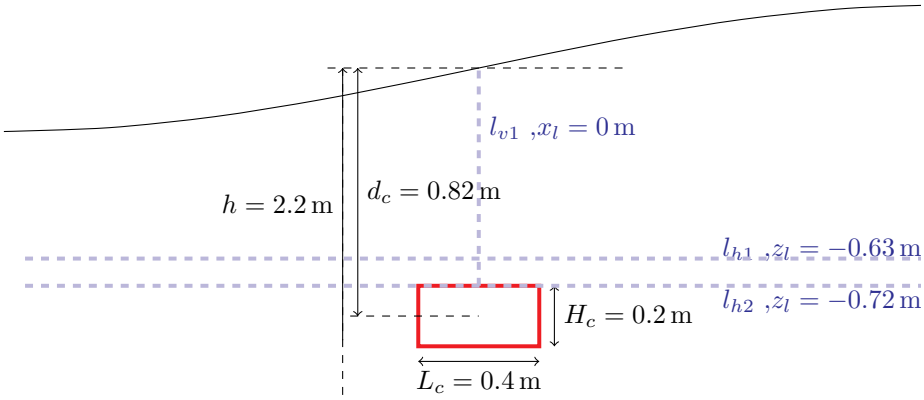


Figure 4.1: Schematic representation of the selected case. Note that the sea bottom is not represented, and everything is at scale, considering a wavelength $\lambda = 6.15\,\mathrm{m}$ (half of it represented) and a wave steepness $H/\lambda = 7.0\%$. Profiles in the following of this work will be sampled over the blue lines l_{v1}, l_{h1} and l_{h2}.

The fully immersed horizontal cylinder of aspect ratio 1/2 presented in section 3.3.3 is selected (see fig. 4.1). It was chosen for mostly two reasons:

- the object is below the free surface. Thus, it will be possible to implement coupling methods HPC/RANS and test it on this case, without having to develop a corresponding free surface coupling scheme.
- nonetheless it permits to show the limits of the previously presented potential model. Our goal in this chapter is to produce validation results that could serve as reference for the following, and in particular the potential-viscous coupling methods. Sharp corners are known to sensibly modify the flow, even when compared a slightly rounded corners (see *e.g.* Tamura and Miyagi, 1999). Hence we hope that the future developed (coupled) models will capture these effects although the latter are only resolved within a limited zone.

The total water depth is $h = 2.2\,\mathrm{m}$ and the relative submergence depth of the cylinder is set to $h/z_c = 2.68$, which corresponds to the center of the object being located at $(x_c, y_c, z_c) = (0, 0, -0.82)\mathrm{m}$.

4.4.1 Generated and absorbed waves

The focus is made on the interaction of the body with a regular wave. Most of the investigations will be conducted with a wave height H controlled by the KC number fixed at KC= 0.75. Later, this parameter will vary to allow a comparison with results from literature. The wave period also will remain, for most of the work, fixed, selecting $T = 2\mathrm{s}$.

Together with the above mentioned KC, this yields a wave steepness of $H/\lambda = 3.5\%$. The corresponding wavelength is $\lambda = 6.15$ m within a linear theory.

The toolbox `waves2Foam` (Jacobsen *et al.*, 2011) is employed to generate the incoming regular wave, modeled as a Stokes wave of order 5. This theory-based wave model is imposed over a length of $L_{ri} \approx 2\lambda$ trough a relaxation zone with the default waveFoam parameters. This relaxation zone technique ensures that both the wave model is correctly imposed and the reflected waves are absorbed. At the outlet, another relaxation zone of length $L_{ro} = 2\lambda$ is used to absorb the transmitted wave. Those values are the upper limit of the recommended length (1-2 wavelength) by the authors of the toolbox: Jacobsen *et al.* (2011) have studied in detail the reflections levels of such zones up to a length of 2λ and showed that in the worst case scenario ($H/h = 0.4$) this span is sufficient to reduce the reflection coefficient R down to less than 3%. As our case is computed with $H/h = 0.095$, *i.e.* lower than their minimum studied value, a reflection of less than 10^{-3} is expected. Actually, they showed that the reflection coefficients can be approximately modeled as a function of the sole parameter $L_r h/(\lambda H)$ that varied in their cases only in the range $[1, 10]$. In the present simulation this coefficient is equal to 20 and extrapolating their reflection model would yield $R < 10^{-4}$.

The mesh span is chosen as approximately 4 wavelengths on each side of the body, including the relaxation length.

4.4.2 Mesh generation and description

In this section, the generation and characteristics of the mesh, that will be referred to as the "base mesh"is detailed. The general advice from the literature is to use square cells (see *e.g.* Larsen *et al.*, 2019) and this recommendation is followed as closely as possible. However, the free surface itself is refined in the vertical direction. Note that square cells also help the matching between in the body corners vicinities, leading to a lower degree of non-orthogonality.

The mesh generation procedure will be maintained throughout this work, and is as follow. First, a background mesh is generated with the `blockMesh` utility on which 9 blocks are defined:

- 3 different vertical discretizations: the water, the free surface and the air. All of them spanning over the entire mesh length (*i.e.* 48 m)
- 3 different horizontal discretizations: the upstream and downstream propagation zones and a breadth close to the body

In order to obtain a correct match between those blocks, some of the discretizations are dependent from another: 3 degrees of freedom are available in each direction. Moreover, square cells will be enforced in the body vicinity, and a horizontal symmetry in discretization with respect to the body position is chosen. For practical reasons, the air discretization is equal to the background water one. This reduces the number of degrees of freedom to 4: 2 in the vertical direction and 2 in the horizontal discretization. Of course, relative spans of those blocks can also be considered as degrees of freedom, but some changes were tested during this work without significant impact on the results and will thus not be shown here.

The vertical span of the free surface zone was however modified later in this work to permit the capture of greater wave heights. For the main considered wave height

($H = 0.21$ m) a span of 0.6 m was selected, centered on the still water level.

Then, on this by-block defined mesh, the `snappyHexMesh` utility is used to:

- Snap the body out of the previously designed mesh and refine further in both directions in the vicinity of the body and in the free surface zone
- Refine by splitting the cells of the "center" block (body vicinity) into powers of 2 in each direction, controlled by a refinement level. A level of 1 means that each cell is split in 4, 2 in 16, *etc.*
- Discretize the boundary layer such that y^+ is limited to an overall maximum value of about 6. With this discretization of the boundary layer, it is not needed to use a wall model function for the turbulent variables in the vicinity of the body. This will be assessed in section 4.6.3
- Refine around the corners in both directions. The level of refinement close to a corner is always chosen as one higher than the body vicinity refinement level: a given cell is once again split into 4. Thus the discretization is twice as small in every direction close to a corner. This has been used in order to reduce non-orthogonality of the mesh at the location where the viscous boundary layers of the different walls match. Note that tests were conducted with this corner discretization deactivated in the mesh independence study section.

This method will be kept during the mesh independence study performed in section 4.6.2.1. The final discretizations of the base mesh are given in table 4.2.

mesh id	Background		Free surface		Body	nCells	name
	dx	dz	dx	dz	dx=dz		
1	0.045		0.045	0.01	0.0056	138592	bg045x045dx0056

Table 4.2: Table describing the base mesh densities (in m)

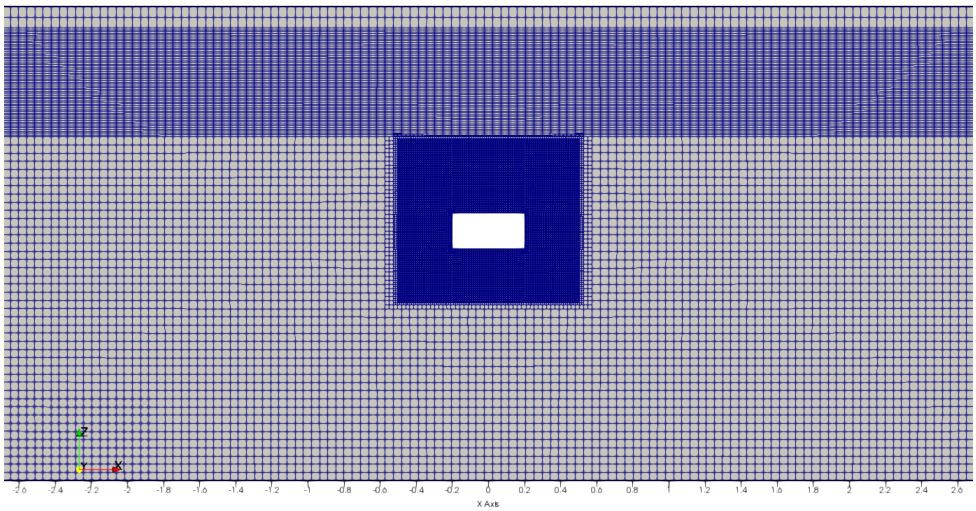


Figure 4.2: Example of mesh grid. (bg045x045dx0056)

A screenshot of this mesh is shown in fig. 4.2 ("bg045x045dx0056"). The final cell dimensions obtained in the free surface zones are $dz_{fs} = 0.01$ m, which corresponds to describing the wave height H with 21 cells, and $dx_{fs} = 0.045$ m which means that the wavelength is discretized with 136 cells. Those values are in the range of the commonly encountered discretizations in the literature. For details about the cells sizes the reader is referred to Larsen *et al.* (2019),

4.4.3 Numerical parameters

According to Larsen *et al.* (2019), lowering the time step always leads to a significant improvement in the capture of the free surface interface, but also the associated velocity profiles in the bulk of the water. A time independence study will be conducted, on which the controlling parameter is the time step size itself, instead of the maximum authorized CFL.

The reasoning behind this choice was the difficulty to obtain consistent and stable results across cases, particularly across meshes. Selecting a constant time step did not really improved this shortcoming, but rather made easier the comparison in terms of computational times for different meshes. Also note that the maximum CFL values given in this work are hardly comparable to the study focusing on the sole wave propagation, as the highest velocities and smallest cells are - in the current case - obtained in the vicinity of the body. The time step is then controlled by different effects, and way smaller than it would have been without body for the same maximum CFL.

Several different schemes and resolution methods (set *via* the files `fvSchemes` and `fvSolutions`) were also tested during this work, and a relative difficulty to maintain very consistent results was denoted. No modification of those parameters will be presented however in this chapter. The selected schemes and parameters are presented hereafter. A lot of details and recommendations are also given in Larsen *et al.* (2019). However, they stated themselves that the "diffusive balance" (in the sense of a balance between the diffusivity of the chosen scheme vs the time step size) they achieved is not universal. Furthermore, their study focused on propagating a given wave up to very long time (100 periods). In our case, the needed propagation was only up to about 2 to 4 wavelengths, and thus most long time effects on the wave shape (*e.g.* wiggles) were not encountered in this study.

The `Euler` time scheme is used, the gradient schemes are set to a `leastSquare` method which is a second order scheme. They are limited by a `cellMDLimited` set to 1.0.

The divergence schemes were all set to Gauss. While the turbulence associated ones use the upwind version, the advection of the momentum is set to `limitedLinearV` 1. Finally, the advection of the volume fraction is a `vanLeer01` scheme. In order to keep a sharp interface, an interface compression scheme is employed: a numerical term is added in the volume fraction transport equation eq. (4.16), that models the relative velocity at the crossing of the interface. This terms is classically discretized with the corresponding scheme `interfaceCompression` (and the corresponding `cAlpha` is set to 1). For further details about this method, the reader is referred to Deshpande *et al.* (2012). The discretization of the Laplacian is done with the classically used a `Gauss linear corrected` scheme. This scheme requires the keywords for both `snGrad` and

`interpolation` to be defined, that here, were respectively selected as `corrected` and `linear`.

Finally, the number of `PIMPLE` iterations was set, in this chapter, to a fixed values of `nOuterCorrector`= 1 (*i.e.* usage in `PISO` mode).

4.5 Comparison of the different turbulence models

As a first step, we aim to select a turbulence model that would permit to capture the turbulence effects close to our body. The $k-\omega$ SST model is used, this selection was made for its wide use in the wave-body interaction literature due to its capabilities to model both highly turbulent flows in the vicinity of the body and recover the stability of $k-\epsilon$ model further away. However, before that, the results of a laminar computation are discussed.

4.5.1 Laminar computation

As a first try, a laminar model is used to simulate the flow. Of course, this model is not expected to be adequate, particularly in the body vicinity. The Reynolds number, based on the vertical height of the body and the horizontal velocity amplitude, is $Re = 3.1 \cdot 10^4$.

The temporal load series predicted by this simulation are shown fig. 4.3, and compared with a modified version of $k-\omega$ SST presented in section 4.2.4 and tested in section 4.5.3.

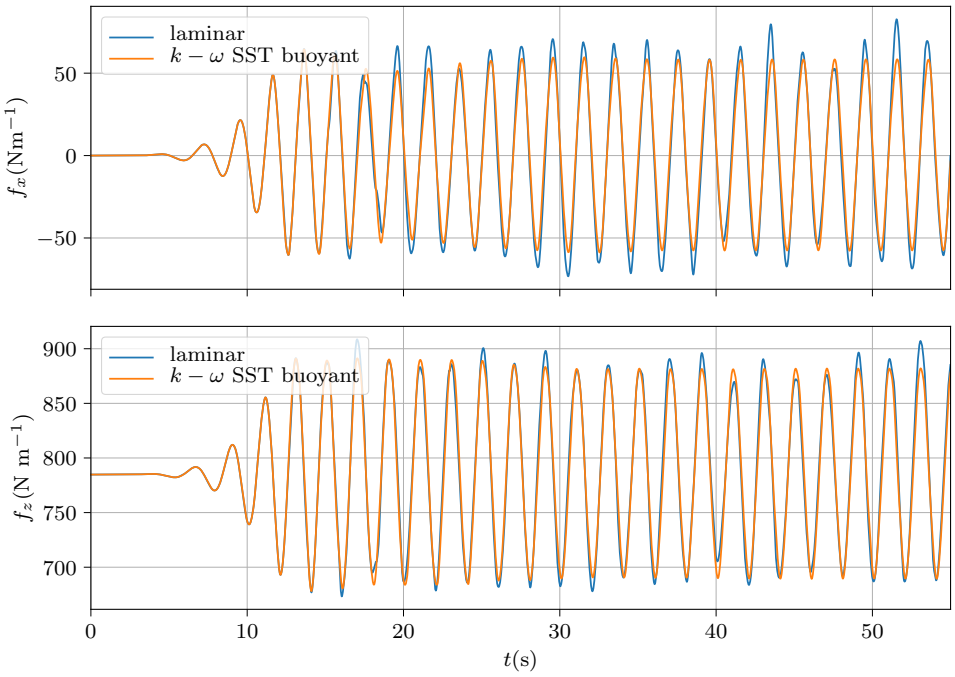
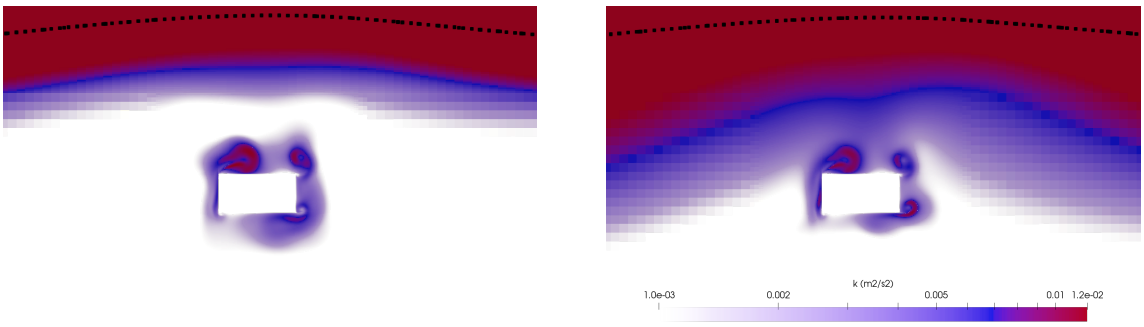


Figure 4.3: Loads on the cylinder with $H/\lambda = 3.5\%$, laminar vs turbulent ($k-\omega$ SST buoyant). Computations with OpenFOAM®.

The obtained load series are not completely out of bounds. A non periodic behavior is obtained, that advocates for the fact that the simulation parameters are not well suited. It is thought that the main problem is the model itself. However, many references indeed use a laminar model, even with sharp corners. Some effort were put in trying to stabilize this laminar computation, without much success. Combined with the aim of producing correct and consistent turbulence fields that will serve as a reference later in this work, the laminar model was thus dropped.

Subsequent results are obtained considering different versions of the $k-\omega$ SST model.

4.5.2 Native OpenFOAM® $k-\omega$ SST model



(a) $t/T = 15$ (b) $t/T = 25$

Figure 4.4: Turbulent kinetic energy field at two different physical times. Native $k-\omega$ SST model. Black dots mark the free surface position ($\alpha = 0.5$)

Figure 4.4 depicts the turbulent kinetic energy k (TKE) field at two different times obtained with the $k-\omega$ SST model native of OpenFOAM®. The generated TKE due to the density variation at the interface increases with time and diffuses in the bulk of water. In turn, the turbulent viscosity follows the same path. Thus, an large damping of the free surface elevation, increasing rapidly over time, occurs. After $t/T = 30$ this perturbation also impacts the body vicinity flow in a significant manner.

In practice, using the native model as is, reveals impossible because it modifies too much the free surface behavior and thus the associated velocity profiles in the water. For this reasons the native OpenFOAM® model is ruled out as it is clear that no good quality results will be obtained with this model.

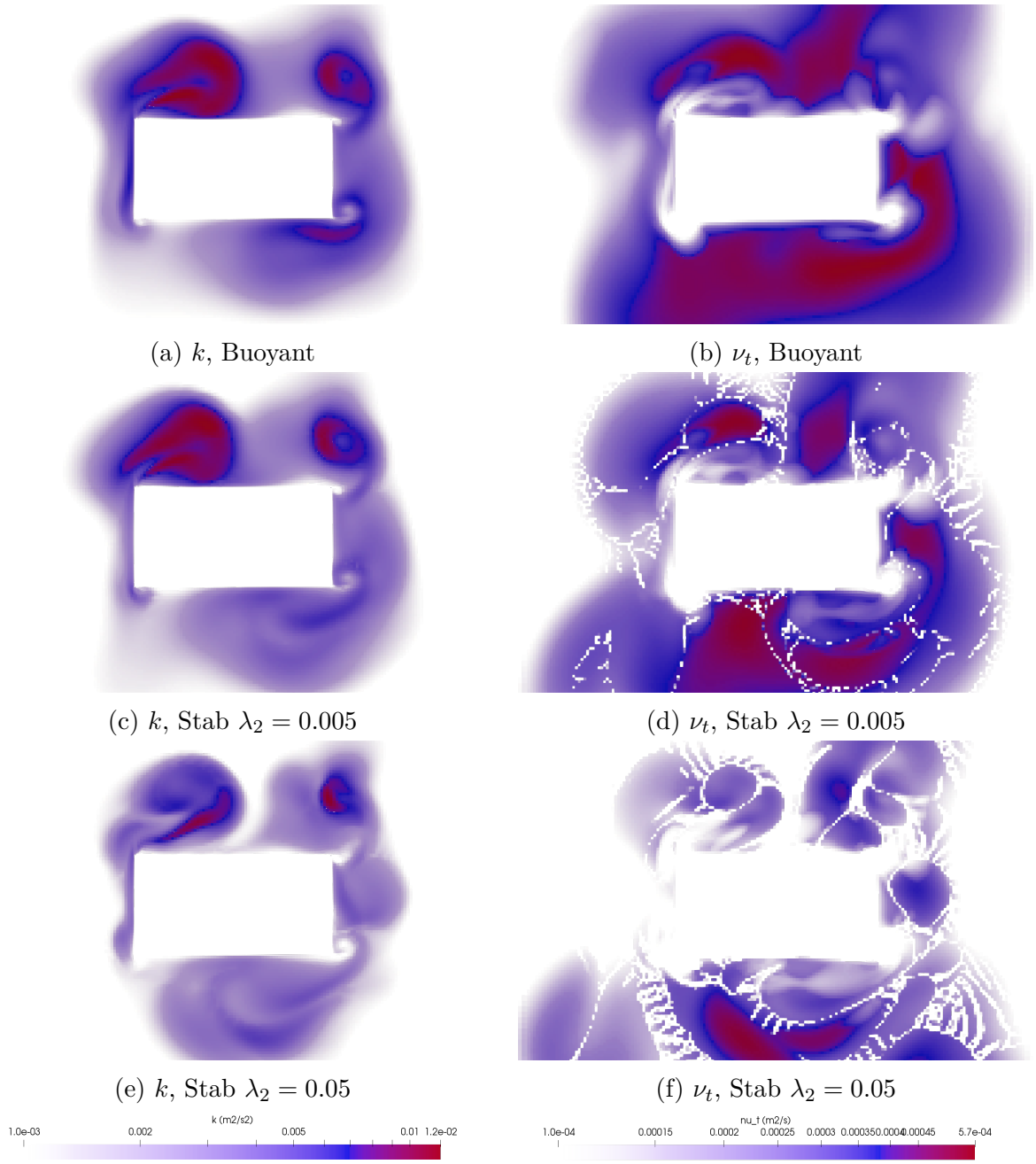
4.5.3 Modified $k-\omega$ SST models

To counteract this effect, two different modifications of the $k-\omega$ SST models developed in the literature, have been tested. Their associated methods and equations are described in section 4.2.4.

The first one was developed by Devolder *et al.* (2017) and adds two features to the original SST model: i) it takes into account the modification of the density across the interface, allowing a difference of viscosity for the two phases of the flow, and ii) it adds into the RHS of the turbulent kinetic energy equation a buoyancy term $G_b = -\frac{\mu_t}{\sigma_b}\mathbf{g}\cdot\nabla\rho$, so that any gradient of density in the direction of the gravity acceleration vector drives the TKE to 0.

The second implements the same modifications, but additionally sets up a new limiter during the computation of ν_t. It was developed by Larsen and Fuhrman (2018) and aims to damp the intrinsic instability of several turbulence models under a potential wave field.

Figure 4.5 shows a comparison of the TKE (left) and turbulent eddy viscosity (right) fields for the different variations of the model at $t/T = 15$. Notice that, while not shown here, the results obtained with the stabilized model with $\lambda_2 = 0$ and the buoyant model are in perfect agreement. An almost perfect agreement in the body vicinity is obtained also between these models and the native model, but only up to the point where the spurious turbulence generated begins to drastically perturb the body vicinity flow.



(a) k, Buoyant

(b) ν_t, Buoyant

(c) k, Stab $\lambda_2 = 0.005$

(d) ν_t, Stab $\lambda_2 = 0.005$

(e) k, Stab $\lambda_2 = 0.05$

(f) ν_t, Stab $\lambda_2 = 0.05$

Figure 4.5: Comparison of the modified turbulence models at $t/T = 15$: buoyant model from Devolder *et al.* (2017) and stabilized model from Larsen and Fuhrman (2018) with different values of the λ_2 parameter. Left column shows the TKE k and right column the turbulent viscosity ν_t. Note that $\lambda_2 = 0.05$ is the default and recommended value.

While a correct agreement in term of turbulent kinetic energy is obtained between the stabilized-$\lambda_2 = 0.005$ and the buoyant model, it is not true for the stabilized-$\lambda_2 = 0.05$. Moreover, whatever the value of λ_2 a strange phenomenon appears on the turbulent eddy viscosity field: lines of zero ν_t are noticeable. It is possible to deduce that, in at least some regions/cells close from the body, the rate of rotation is very small compared to the rate of strain thus triggering the newly introduced damping. This however, is not an expected behavior: the model should be non-intrusive in zones that exhibit non zero rotation. The introduced limiter should not be triggered in the vicinity of the body.

In order to understand the causes of the behavior of the stabilizer in this zone, the different terms of eq. (4.19) are computed and shown separately. Let's recall the definitions of $S_y = 2\mathbb{D} : \mathbb{D}$ and $S_k = 2\Omega : \Omega$ the symmetric and skew terms that appear in the stabilized model limiter. Those fields, obtained with the stabilized model-$\lambda_2 = 0.05$, are depicted at time $t/T = 15$ on fig. 4.6. From the stabilized ν_t equation, we may deduce that the limiter will trigger whenever the ratio S_y/S_k is of great magnitude. To analyze the effect of the stabilization model, the ratio of the new denominator of the ν_t expression over the classical $k-\omega$ SST one is computed and depicted on the figure. This ratio is thus expressed as:

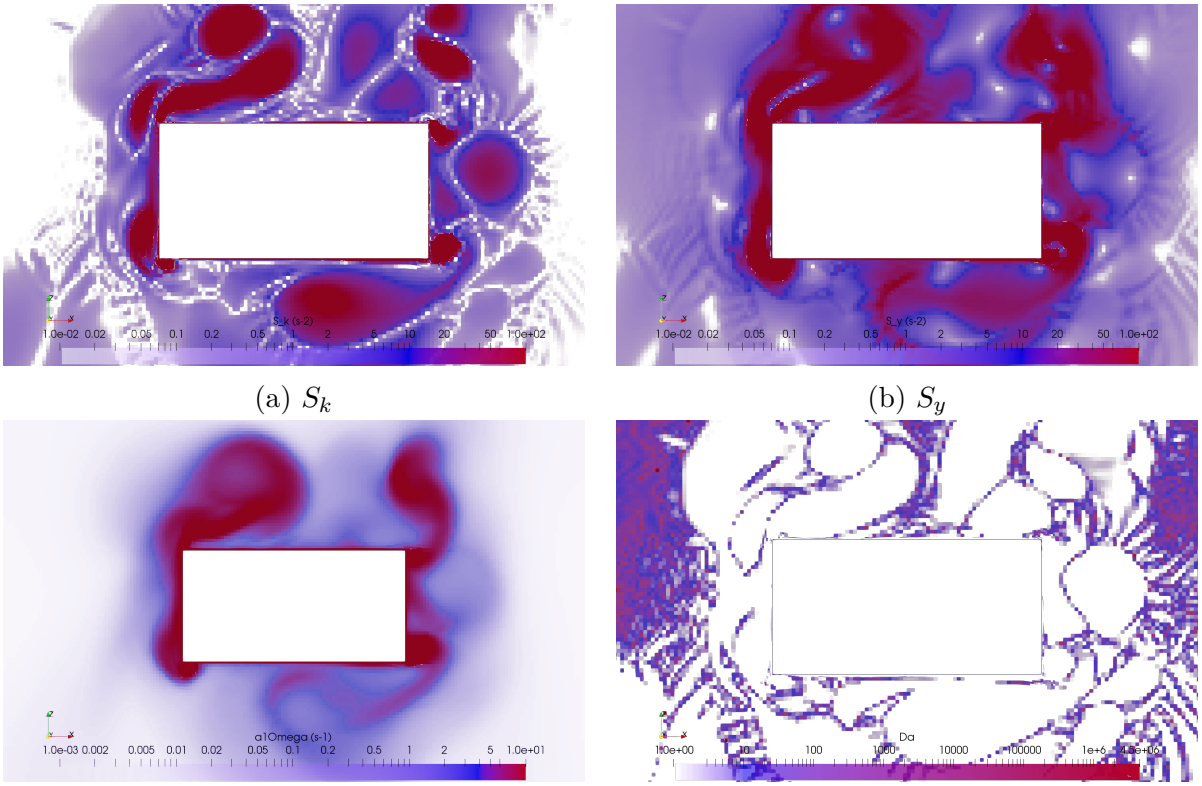
$$D_a = \frac{\nu_t \text{native}}{\nu_t \text{stab}} = \frac{\max(a_1\omega, \sqrt{S_y}F_2, a_1\lambda_2 \frac{\beta}{\beta^*\gamma}\frac{S_y}{S_k}\omega)}{\max(a_1\omega, \sqrt{S_y}F_2)} \tag{4.38}$$

Thus, whenever this "damping" ratio is equal to 1, the stabilization method shuts off, and the original $k-\omega$ SST is retrieved. However, any other value means that the turbulent viscosity is reduced by D_a compared to what the original model would have yielded. Notice also that a numerical small number ϵ_{num}, that prevents S_y/S_k to reach infinity is also added in the numerical computation of D_a, thus $S_y/(S_k+\epsilon_{\text{num}})$ is computed instead of directly the former.

One can clearly see from fig. 4.5 that the damping reaches very large values in some regions (way greater than 1). Those regions correspond to the lines of null isovalues of the skew field, as could be predicted by looking closely at eqs. (4.19) and (4.38). In turn the turbulent viscosity is highly affected on these lines for all tested values of λ_2 (see *e.g.* fig. 4.5d). This behavior, although predictable, seems undesired: null values of the skew field will always appear between counter-rotating vortices, during the change of sign of the vorticity. Thus, the stabilization process will always perturb the turbulent fields whenever vortices are emitted.

Notice than the mesh discretization will have an significant leverage on the obtained effect. A finer discretization will lead to a better capture of the zeros of the rotational, and thus higher values of the added damping. On the other hand, it will also reduce the "volume" on which this damping is applied (that will approach the exact mathematical null isovalues lines of S_k). A similar study was conducted on a coarser mesh, that showed that the effect is even stronger in this case and can even be seen directly on the TKE for $\lambda_2 = 0.005$.

The only new parameter of this stabilization model is λ_2, and one could try to reduce it below the smallest tested value in this work (0.0005, however not shown here). However, as - with our discretization - D_a reaches values higher than 10^6 with $\lambda_2 = 0.05$, it is expected that the stabilization would really shut off everywhere in the body vicinity when $\lambda_2 \sim 5 \cdot 10^{-8}$. Obvisouly, such a small value of λ_2 would almost completely shut off



(a) S_k (b) S_y

(c) $a_1\omega$ (d) D_a

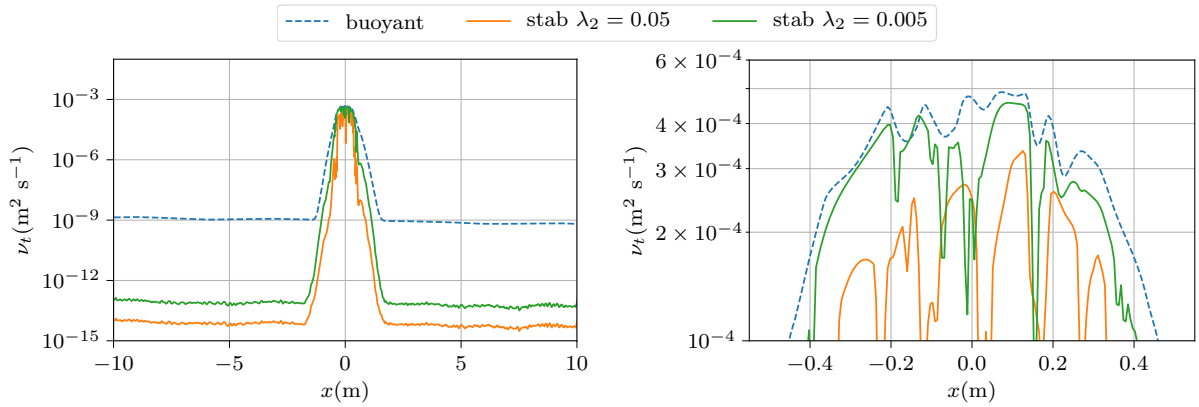
Figure 4.6: Fields obtained with stabilized model at $t/T = 15$. Note that by definition, the damping term D_a evolves in $[1, \sim \infty]$. In practice values up to $5 \cdot 10^6$ are obtained in this region. The complete range is shown for finer analysis. These fields are obtained with the stabilized model with $\lambda_2 = 0.05$.

the stabilisation even in the purely potential region. Furthermore, any given mesh would require a different value of λ_2 for this shut off to effectively happen.

In order to compare the stabilization effect on the potential part of the flow in a quantitative manner, a horizontal line is defined, on which the turbulent viscosity is sampled. Results are shown on fig. 4.7. The vertical position of the line is chosen as $z_l = -0.63$ m such that it crosses the vortices above the body. For reference, we recall that the body spans vertically from -0.72 m to -0.92 m and horizontally from -0.2 m to 0.2 m.

Note that a computation of the stabilized model with $\lambda_2 = 0$ was performed and yielded undistinguishable results from the buoyant model. It is thus not shown here for clarity reasons.

The stabilization clearly plays a significant and expected role away from the body. The turbulent eddy viscosity is several orders of magnitude lower than the non stabilized one (either $\lambda_2 = 0$ or the buoyant model) for the same boundary and initial conditions. However in the vicinity of the body the stabilization plays a non-negligible role for the reasons stated previously. In this zone, the recommended $\lambda_2 = 0.05$ reduces the turbulent viscosity by a factor ~ 2. These discrepancies can be reduced by decreasing λ_2 but never actually be suppressed without completely shutting off the stabilized model, as was demonstrated earlier. Note also that for a NWT the involved simulated times are small.



(a) (b) Zoom

Figure 4.7: ν_t profile along a horizontal line at time $t/T = 15$ just above the body $z_l = -0.63$ m for the different computed cases.

Thus, it can be expected that the exponential growth of k and ν_t shown by Larsen and Fuhrman (2018) does not actually play a significant role. It is at least not noticeable on this figure as the magnitude of the turbulent viscosity remains contained in the potential zone. However it will be shown later that an unstable growth still occurs which in turn, makes the model relatively dependent on the mesh and the time step size.

4.5.4 Suggested $k - \omega$ SST models

The main goals of a modification of the turbulence model are

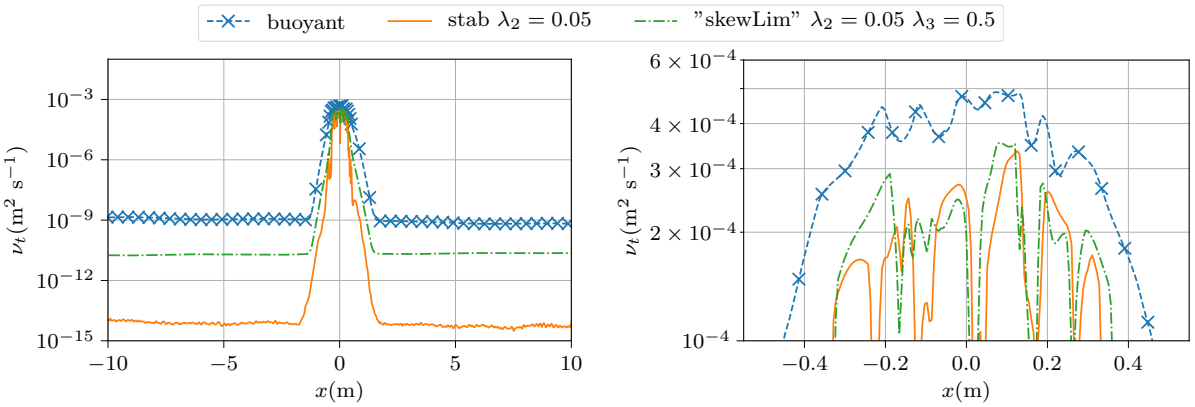
1. to reduce the overall value of ν_t inside the potential zone, *i.e.* to dampen the instability occurring in the propagations zones and
2. without damaging the obtained turbulence fields in the body vicinity, *i.e.* retrieving the obtained values of ν_t obtained with either $\lambda_2 = 0$ or with the buoyant model (but also with the native model prior to the perturbation).

It was shown that the stabilized model with $\lambda_2 \neq 0$ fails to fulfill the second requirement when the buoyancy model (or the stabilized model with $\lambda_2 = 0$) does not fulfill "1)".

The two following paragraphs aim to test respectively the first and second suggestions presented in section 4.2.4.4 and implemented during this work. They are respectively denoted "skewLimiting" and "smoothing" corrections.

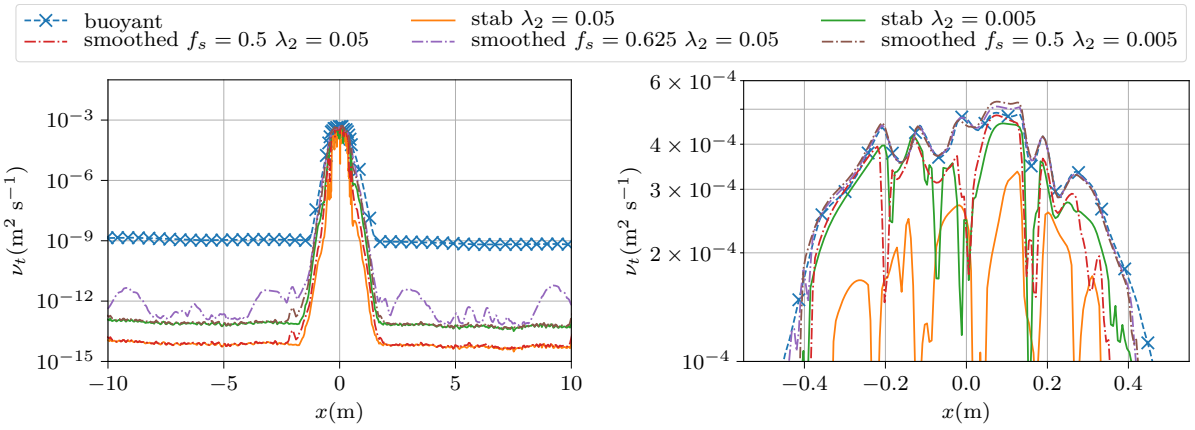
The skew limiting correction The results of the "correction" added to the Larsen and Fuhrman (2018) model are presented in fig. 4.8. However, when compared to the original stabilized model, the requirement "2)" is only very slightly improved: the minima of ν_t in the body vicinity are slightly higher, whereas the stabilization in the potential zones is less effective (requirement "1)").

While the deterioration of "1)" was expected, the hope was to obtain a better improvement of "2)" when compared to the stabilized model. Due to the small gain obtained, this modification is neither selected nor recommended.



(a) (b) Zoom

Figure 4.8: ν_t along a horizontal line at time $t/T = 15$ just above the body $z_l = -0.63\,\mathrm{m}$ for the different computed cases.



(a) (b) Zoom

Figure 4.9: ν_t along a horizontal line at time $t/T = 15$ just above the body $z_l = -0.63\,\mathrm{m}$ for the different computed cases.

Smoothing of the skew term Here, we focus on the modification of the stabilized model that consists in smoothing out the minima of S_k and S_y (not necessary for the latter but done for consistency) as presented in section 4.2.4.4. Results are presented on fig. 4.9. It can be denoted a consistent improvement of the requirement "2)" without modifying "1)" (with $f_s = 0.5$), when compared to the stabilized model.

Indeed, the smoothed model applied with $f_s = 0.5$ is always better than the stabilized model with the same λ_2 inside the turbulent zone. In the propagation zone, the stabilized values of ν_t are rapidly recovered.

For example, even with $\lambda_2 = 0.05$ the smoothed model is slightly more accurate than the stabilized one with $\lambda_2 = 0.005$ in the body vicinity ("2"). Furthermore, a way stronger damping occurs with the former in the potential zones ("1"). The obtained results and their implication on both aspects are qualitatively compared in table 4.3. The results are aggregated into a double-entry table taking advantage of the fact that $f_s = 0$ is equivalent to the stabilized model, and $\lambda_2 = 0$ or $f_s = 1$ is equivalent to the buoyant model.

f_s \ λ_2	0 ≡ buoyant	0.005	0.05
0 ≡ stab		+	++
0.5		+	++
0.625	−−		+
1 ≡ buoyant		−−	

0 ≡ buoyant	0.005	0.05
	+	−
	++	+
++		++
	++	

Table 4.3: Qualitative performance in fulfilling requirement "1)" (left table) and requirement "2)" (right table). See section 4.5.4 for definition of requirements

The same study was conducted using another coarser mesh, on which the same general conclusion could be drawn: adding the smoothing with a strength parameter of $f_s = 0.5$ always improves the results in the body vicinity zone, without degrading the stability in the potential zone. We recall that a value of $f_s = 0.5$ means that no two neighbor cells will, at a given time, have computed values differing of more than a factor 2.

Notice that this smoothing should not alter the stability condition exhibited by Larsen and Fuhrman (2018) (*i.e.* $\lambda_2 > S_y/S_k$), because it does not modify the fundamental equation, but only the scale over which it is computed and applied. A further step would be to base the smoothing function on the operand (for exampled on its gradient), so as not to be dependent on the mesh discretization.

As a conclusion, it is possible to state that while this modification of the Larsen and Fuhrman (2018) model is only a first step, it seems to greatly improve the non-intrusive property of the model in the presence of vortices and when turbulent effects should arise. Note that no generic recommendation can be given for the selection of the filter strength because the current implementation makes it dependent on the cell sizes. It seems however, that a low value can only increase the simulation quality. At $f_s = 0$ the model indeed proved to shut off the smoothing and to reduce almost exactly to the Larsen and Fuhrman (2018) model. However, the current implementation of the smoothing process can be computationally expensive, particularly for high f_s values. (+20% at $f_s = 0.625$).

4.5.5 Instability at long time of the turbulence models.

During this work, an increase of the turbulent eddy viscosity close to the body was denoted. This rise was appearing whatever the chosen turbulence model. Althought its magnitude and growth rate might differ depending on models, numerical parameters, mesh, *etc.*, its location is rather consistent across cases: the TKE stacks up at approximately $x = 0.5$ m to 1.5 m away from the body.

On fig. 4.10 results from three of the selected models are plotted (with consistent colors) over the sample line presented above ($z_l = -0.63$ m). The obtained profiles at $t/T = 15, 30, 50$ are shown. On this figure, a large growth is noticeable. This effect is present with all turbulence models, whether they are stabilized of not. The reached values seem to be unphysical: they are of large magnitude compared to the physical kinematic viscosity ν but more importantly, their magnitude grows past the overall value obtained in the zone generating the turbulence. It is thought that the generated turbulence feeds the instability at this location, where the velocity rotational is not small enough to activate the stabilizers. Further away, the stabilization - smoothed or not - manages to maintain

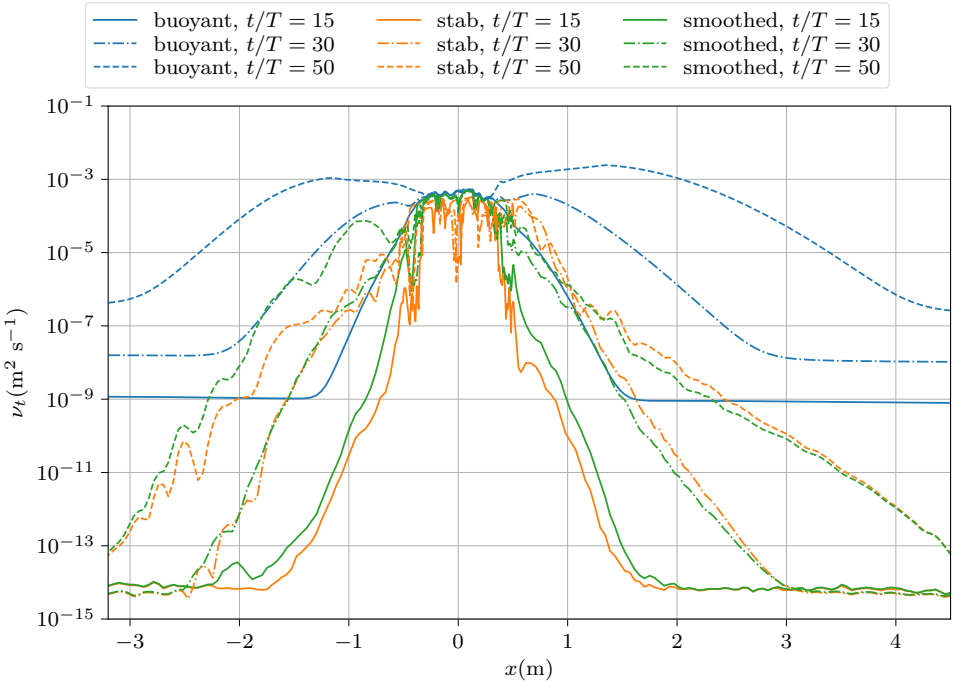


Figure 4.10: Turbulent viscosity just above the body ($z_l = -0.63\,\mathrm{m}$) at different physical times, for the different versions of the $k-\omega$ SST model (stabilized, buoyant and smoothed), $\lambda_2 = 0.05$, $f_s = 0.5$.

a low value of the turbulent viscosity, while the buoyant one shows a steady rise of ν_t.

4.5.6 Conclusion and selection of a turbulence model

The native $k-\omega$ SST model of OpenFOAM® cannot be used in this multiphase approach, because of the large associated generation of turbulent kinetic energy close to the free surface. The modifications suggested by Devolder *et al.* (2017) greatly help in that regard. The latter model can be validated against the former at short time, before the native model is affected by the free surface production of TKE. The model of Larsen and Fuhrman (2018) is also in agreement with those results when we select $\lambda_2 = 0$. Because it also implements the modifications suggested by Devolder *et al.* (2017), this conclusion was expected.

However, in the vortex shedding zone, the turbulent viscosity is damped as soon as the limiter parameter $\lambda_2 \neq 0$. While it is thought that this scheme should definitely be selected in an engineering application because the main physical fields (velocity and pressure) are seemingly not affected in a significant way, it has an intrusive behavior on the turbulent fields, which is very dependent on the selected λ_2 value.

In this work, our main objective is to produce good quality results in terms of local fields - including the turbulent eddy viscosity - in the body vicinity, in order to compare later with the coupling schemes. Thus, although the stabilized model yields better results in the potential zone, the non-stabilized model is chosen in this work for future computations, not to perturb turbulence mechanism in the vicinity of the body. Hence, we accept the drawback of not being able to compute long time evolutions of the wave

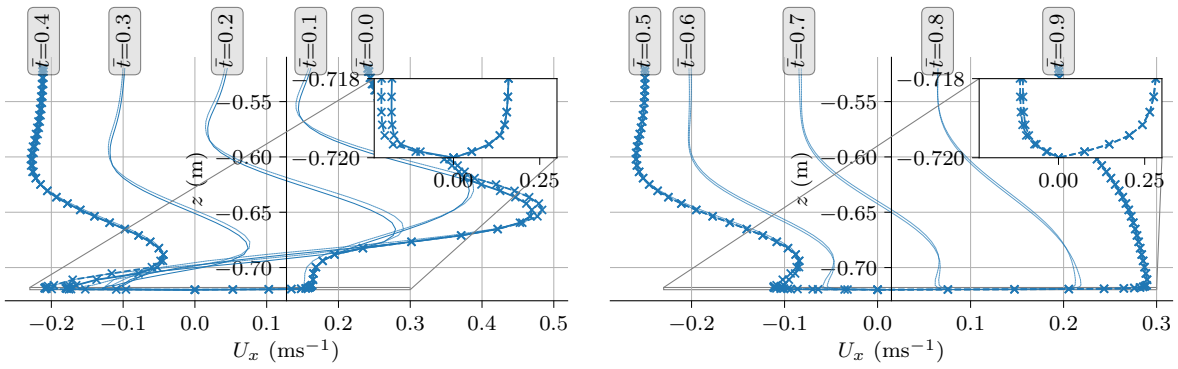
fields ($t/T > 50$).

Notice that the "smoothed" variant of the stabilized $k - \omega$ SST model was developed and implemented in the last period of this thesis and will only be considered furtively in the rest of this work. However, the results presented in this section did exhibit a really positive effect on the computation of the turbulent effects. For this reason, this type of modification is recommended, and would have been applied more frequently if it had been possible.

4.6 Sensitivity and independence studies

4.6.1 Local periodicity

It was shown that activating a turbulence model has a great effect on the periodicity of the predicted loads. Here, we assess the periodicity of the local variables on the selected base mesh (bg045x045dx0056) shown on fig. 4.2 and described in the same section (section 4.4.2). A vertical line is defined on the top center of the cylinder ($x_l = 0\,\mathrm{m}, z \in [-0.72, -0.52]\mathrm{m}$), on which the horizontal velocity profiles are sampled. The results at two adjacent periods are compared ($t/T = 15$ and $t/T = 16$): a scaled time is defined as $\bar{t} = t\%T$ where $\%$ represents the remainder of the Euclidean division. The velocities profiles at the same scaled $\bar{t}$ are superimposed and shown fig. 4.11. While only 5 profiles are shown of each half periods for clarity, 20 profiles per half period are actually computed to obtain a correct envelope. This outer envelopes over these half periods are marked with crosses (x), such that a marker appears at each face intersected face. With this layout, it seems that the boundary layer is correctly described with approximately 5 points within - or below - the buffer zone. A finer analysis will however be conducted in section 4.6.3.



(a) $\bar{t} < 0.5$ (b) $\bar{t} \geq 0.5$

Figure 4.11: Horizontal velocity profiles at the top-center of the cylinder. Two different periods are represented, namely $t//T = 15$ and $t//T = 16$. $\bar{t}$ is the adimentionalized remainder of $t//T$, thick lines represent the extrema of the horizontal velocities. Note that only those extrema are represented in the inset of each graph.

From this figure we clearly see a good periodicity behavior of the horizontal velocity - as the profiles overlap almost exactly -, even locally . Slight discrepancies are only visible at the outer left extrema of the first half period in the boundary layer. Thus, is it assumed that simulating 15 or 16 wave periods is sufficient to compare converged values.

4.6.2 A significant sensitivity to mesh size and time step

All the computations up to this point were performed on a given mesh, presented in section 4.4.2. In the latter section the method employed for the mesh generation was also presented. Hence, this section focuses on trying to evaluate the influence of a modification of the computational grid, and explain *a posteriori* why this mesh was selected in the previous analysis.

4.6.2.1 Mesh independence study

mesh id	Background		Free surface		Body	nCells	name
	dx	dz	dx	dz	dx=dz		
0	0.045		0.01		0.0228	112518	bg045x045dx0228
1					0.0113	117990	bg045x045dx0113
2					0.0056	136592	bg045x045dx0056
3					0.0028	218525	bg045x045dx0028
4	0.0225				0.0028	425068	bg023x023dx0028

Table 4.4: Table describing the tested mesh principal densities (in m), sorted by the number of cells

The selected meshes are described in table 4.4, in which the spatial steps are in meters. The names of the mesh are based on the cell dimensions in the "background" zone (`bg <dx>x<dz>` - *i.e.* in the bulk of the water, away from the body) and the final cell size - maintained of square shape - in body vicinity (`..dx<dxf>`). The last mesh is used to evaluate the influence of the bulk of the water discretization.

Note that the free surface is mostly described with a stream wise discretization of $dx = 0.045$ m and always with a cross stream discretization of $dz = 0.01$ m. Relative to the "base" wave height and wave period ($T = 2$ s; $H/\lambda = 3.5\%$), those values lead to a wavelength described with $\lambda/dx = 135$ cells and a wave height with $H/dz = 21$ cells. According to the literature, this is generally sufficient to correctly compute the free surface. For example, Musiedlak (2019) showed that past 15 cells per wave height, the results exhibit only minor differences. This author also states that finer discretization yields numerical instabilities of the free surface.

It is also possible to remark that the number of cells can be roughly separated in 100k in the background mesh+free surface and the rest serving for the finer discretization close to the body (except for the finest background were 400k are used prior to refining).

On fig. 4.12 the temporal series of the vertical and horizontal loads are shown. From this graph, it is possible to perceive discrepancies, but to state nonetheless that they will not be of large magnitude, especially on the vertical load. It means that except the coarsest grid, all others are capable of describing the loads series with a relative accuracy. In order to understand the differences in a finer, and more quantitative point of view, the averaged amplitudes of the load series are computed and shown on fig. 4.13, as a function of the average local refinement ("local dx" average characteristic length of the cells close to the cylinder). Thus, the abscissa represents the refinement level close to the body.

The background mesh does not seem to play an significant role in the obtained results: one can compare the amplitude of the loads from the inner discretization of 0.0028 m on a background mesh of 0.045×0.045 m (bg045x045dx0028) with the same inner discretization on a background mesh of 0.023×0.023 m (*i.e.* twice as fine, bg023x023dx0028) and find that they are nearly equal. However, the number of cells in the latter is almost 4 times larger. This conclusion led us to consider the finest cell characteristic length as a more correct convergence metric than the more classical total number of cells.

However, the discretization of the corner ("nc"), seems to play a significant role, especially on the vertical loads amplitude. Note that the corners discretization sizes are

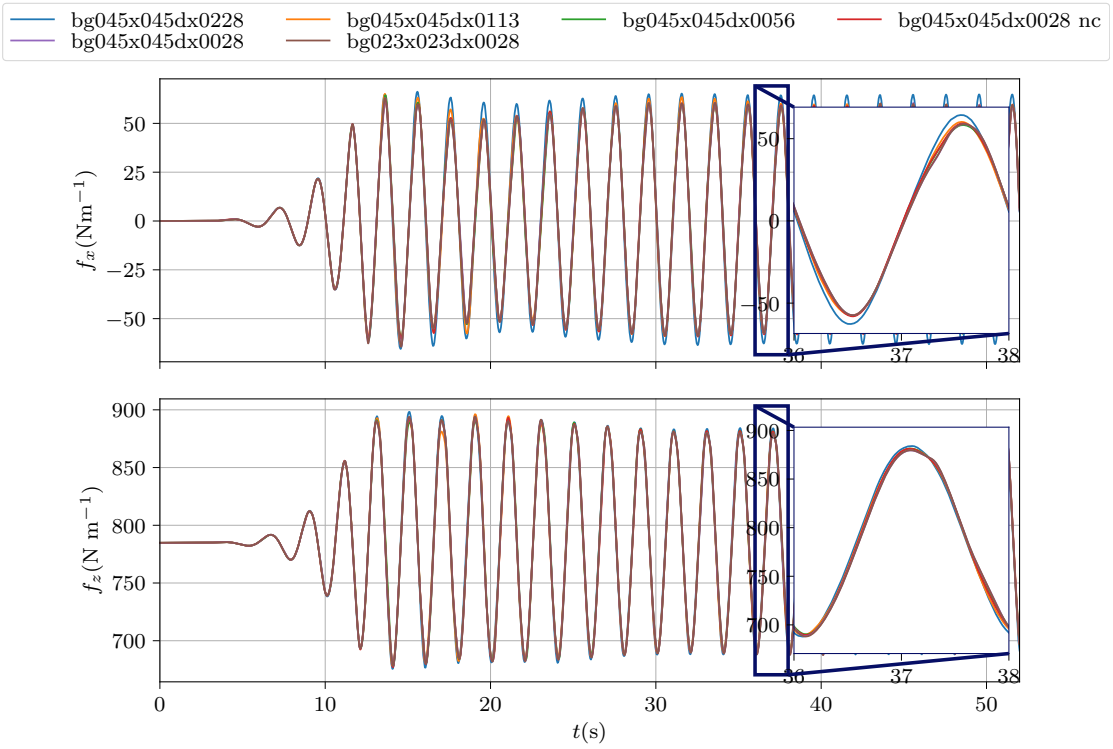
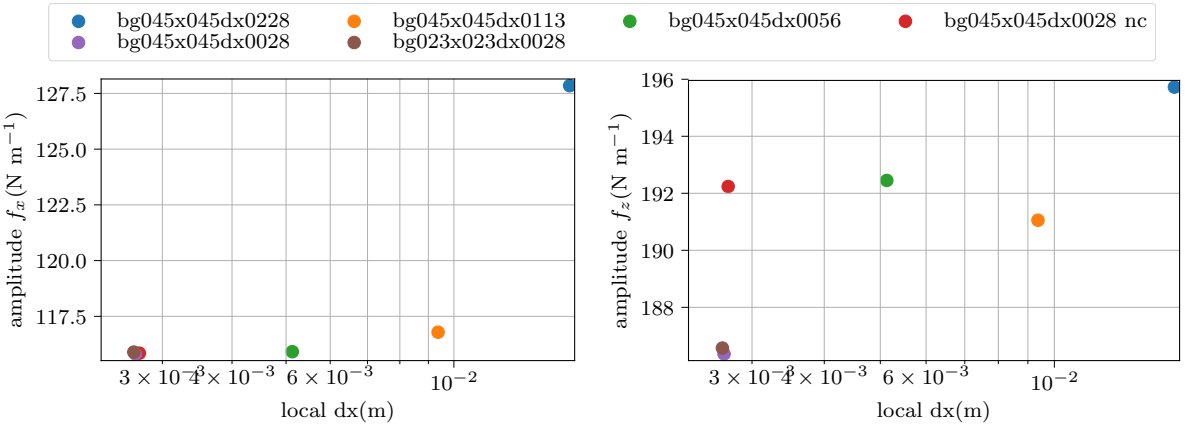


Figure 4.12: Temporal loads comparisons of the computation on the different meshes presented table 4.4."nc" stands for "no corners" which means that the corners were not additionnaly refined (see section 4.4.2 for details).



(a) f_x (b) f_z

Figure 4.13: Amplitude of the loads on the cylinder with respect to the average discretization in the vicinity of the cylinder for different meshes. "nc" stands for "no corners" which means that the corners where not additionnaly refined.

equal for "dx0028 nc" and "dx0056" because the latter has an additional refinement level in these zones. The obtained loads for those cases, both horizontal and vertical are almost of same amplitude. It is thus deduced that the discretization of the corners also has a significant leverage on the obtained results.

Nonetheless, note that the discrepancies are, overall, of relatively small amplitudes. If

the coarsest mesh is not considered, they are contained within a 4% range, for both the vertical and horizontal loads. It is supposed that all of the currently considered meshes (except "dx0228") are fine enough to be considered converged.

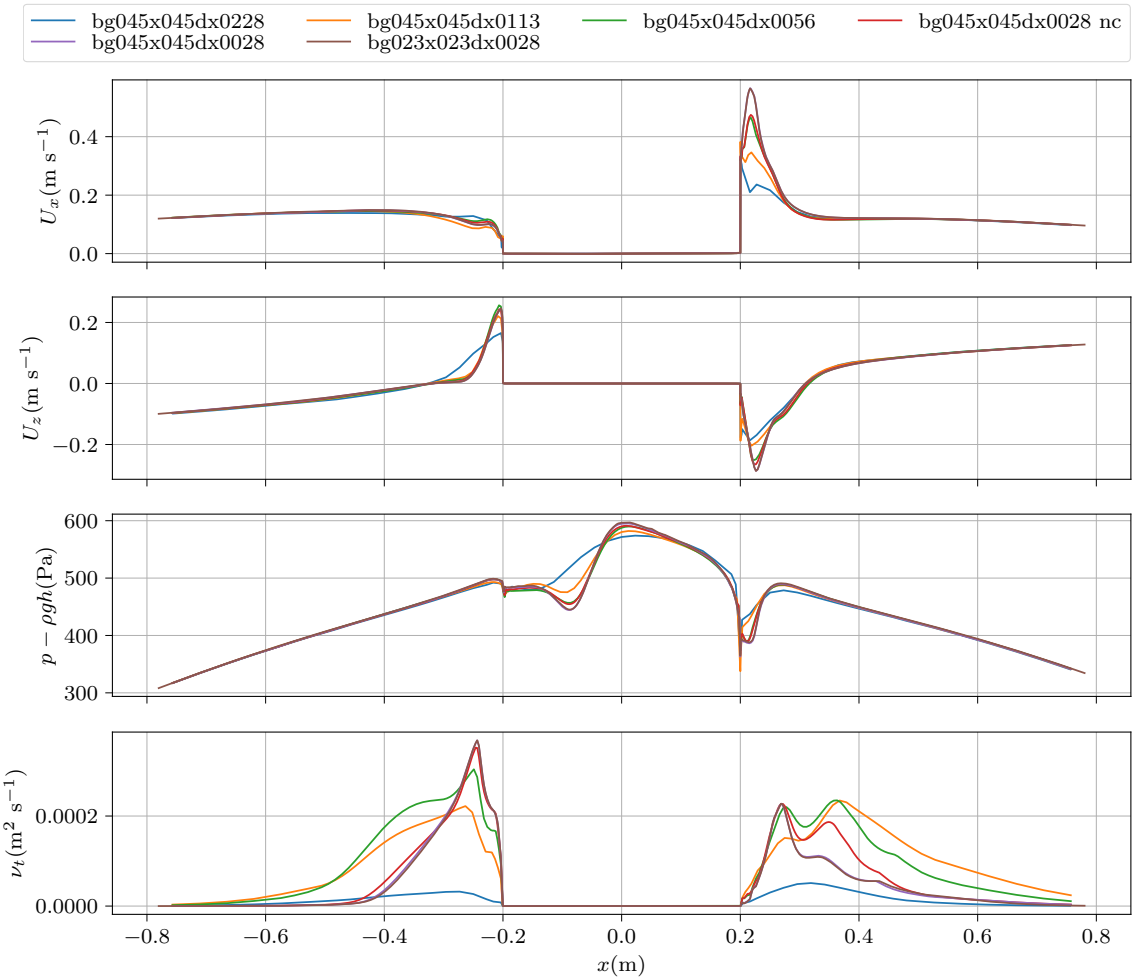


Figure 4.14: Profile of the computed fields on a horizontal line at $z_l = -0.72$ m (top of the cylinder) at time $t/T = 15$

However, comparing the local variables close to the cylinder yields results that are more difficult to interpret. Figure 4.14 depicts at a given time $t/T = 15$ the field profiles along a horizontal line that contains the top wall of the body ($z_l = -0.72$ m). This line was chosen as the integrated pressure on the top cylinder wall is the main responsible of the vertical load variations.

This figure emphasizes relatively large differences, especially concerning the turbulent viscosity. Note that the only three results sets that are in good agreement on ν_t are in fact computed with the same body vicinity discretization $dx = 0.0028$ m, and are in fact, just three variations, namely: original, fine background, no corner refinement. The latter seems however to have an impact on the top-right corners values of ν_t leaving only the first two in perfect agreement everywhere (and on every field). Thus, no clear convergence can be extracted basing on the turbulent eddy viscosity field.

The horizontal velocity (particularly, at this given time, the peak located at the top-right corner of the body, $x = 0.25$ m) is also sensitive to the spatial discretization. This

peak does not seem to have converged. Note that here, in a similar manner as for the loads, “dx0028 nc” and “dx0056” are almost equivalent. This denotes once again that the corner discretization seems to have a significant influence.

The rather large discrepancies in ν_t - but also in horizontal velocity in a lower extent - are thought to be the consequence of the previously encountered turbulent viscosity rise (see section 4.5.5). This non-convergence, even for very fine meshes compared to the literature clearly denotes an instability, or at least the fact that the periodic regime in ν_t is not yet established. In any case, simulating more than 100 periods for each test case is hardly feasible due to the associated computational costs.

Apart from that, the obtained results are - except the turbulent eddy viscosity - relatively converged for all meshes after the inner discretization of $dx = 0.0056\,\mathrm{m}$. The corresponding mesh (bg045x045dx0056) was selected as the base mesh for this reason.

4.6.2.2 Time step independence study

In this section, a focus is made on the modification of the time step size. The mesh “bg045x045dx0056” presented in table 4.4 is once again selected, even though the more expensive “bg045x045dx0028” mesh will also be subject to the current study for comparison.

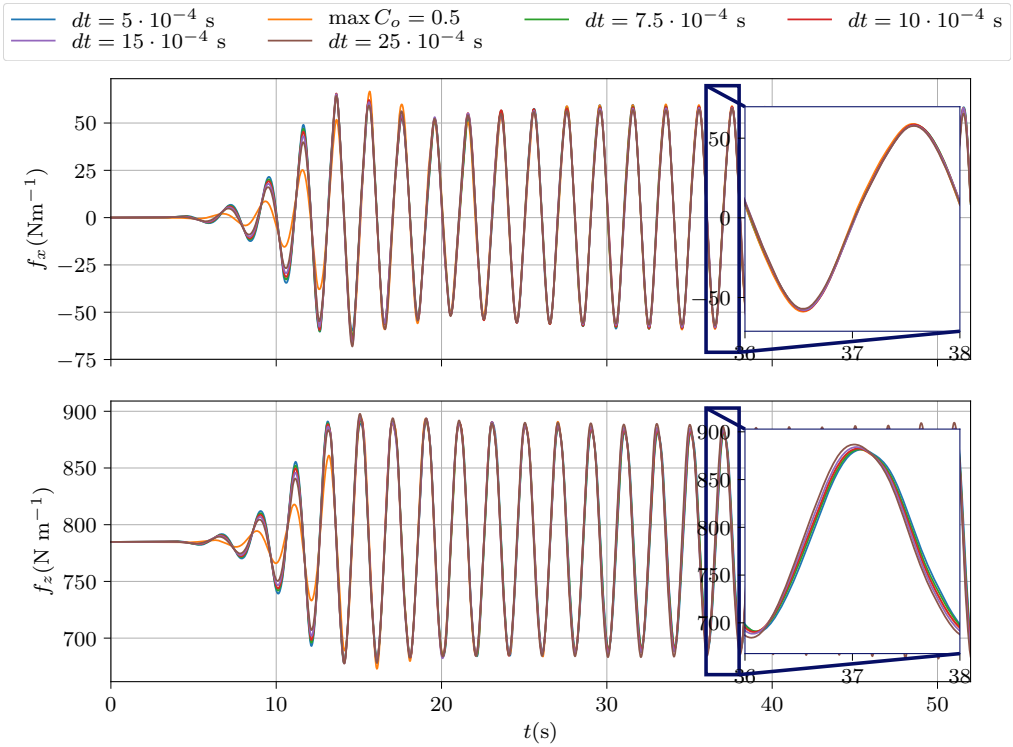
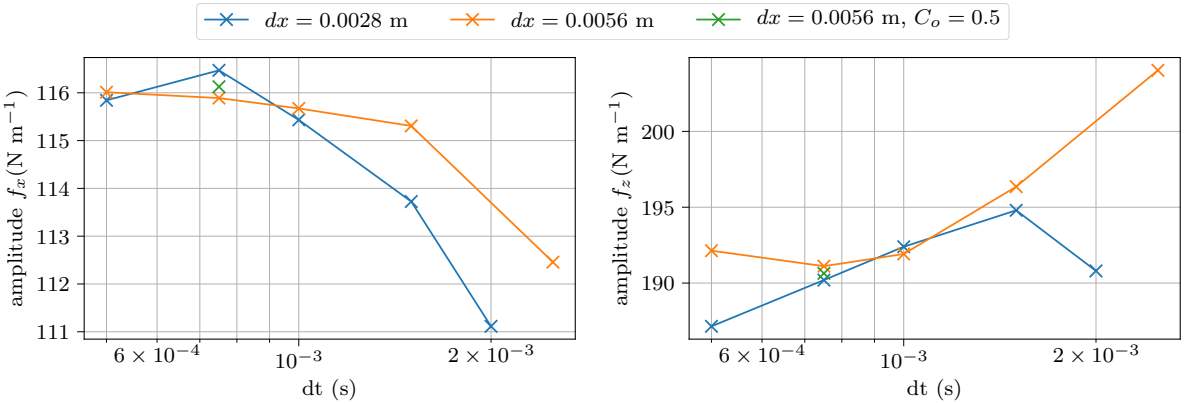


Figure 4.15: Temporal series of the loads applied on the cylinder for different time step sizes. A computation with the CFL number controlling the time step is also added, mesh:“bg045x045dx0056“

Figure 4.15 shows, for different time step sizes, the temporal series of the loads applied to the cylinder. A case controlled by a maximum local CFL number (set to 0.5) is also added. This CFL number enforces a time step of approximately 7.5×10^{-4} s to

10×10^{-4} s. This figure shows that even though the time step has an influence on the wave loads, the latter is limited, and almost only visible on the vertical load.



(a) f_x (b) f_z

Figure 4.16: Amplitude of the loads as a function of the time step size. A CFL controled case is added ($\max(C_o) = 0.5$ *i.e.* $dt \approx 7.5 \times 10^{-4}$ s to 10×10^{-4} s) with an arbitrary abscissa of $dt = 7.5 \times 10^{-4}$ s. Two different meshes are used: bg045x045dx0028 ($dx = 0.0028$ m) and bg045x045dx0056 ($dx = 0.0056$ m).

The amplitudes of the loads for those two meshes ("bg045x045dx0028" and "bg045x045dx0056") are depicted fig. 4.16 in order to quantitatively grasp the time step size influence more finely.

We retrieve on this figure the limited discrepancies in terms of wave loads amplitudes for the different time step size used. It also seems that a convergence has been achieved for the coarser mesh, approximately when $dt = 10 \times 10^{-4}$ s. The finer mesh behavior is however different. It is not possible to pinpoint a time step size for which it seems to have converged. Note however that amplitudes of loads are, overall, within a 6% and a 8% range for the horizontal and vertical loads respectively.

The maximum local CFL number is classically enforced to limit the time step size. This number compare the velocity multiplied by the time step to the cell size, which means that if the CFL is greater than unity, a given particle would be advected of more that one cell length in a given time step: first orders schemes would not be able to capture this. Because the two meshes are different ($dx = 0.0028$ m is twice as fine as $dx = 0.0056$ m), the associated CFL numbers differ by a factor two for the same time step size. It is thus thought that the convergence would be reached for a time step twice as small for the finer mesh (*i.e.* the finest computed value 5×10^{-4} s).

With the same layout as previously, the local fields are compared on fig. 4.17: their profiles over a horizontal line are shown at $t/T = 21$. Note that this time was selected because the transient regime duration was longer on certain computational cases, highly depending on the time step size, and that this time step can be divided by all the selected time step sizes.

Discrepancies are contained whenever dt is lower than $\approx 10 \times 10^{-4}$ s but some can still be seen and it is difficult to conclude on the correct convergence. Once again, the turbulent viscosity is different and varies greatly with the time step size selection.

This behavior of the turbulent viscosity was somewhat expected. Larsen and Fuhrman

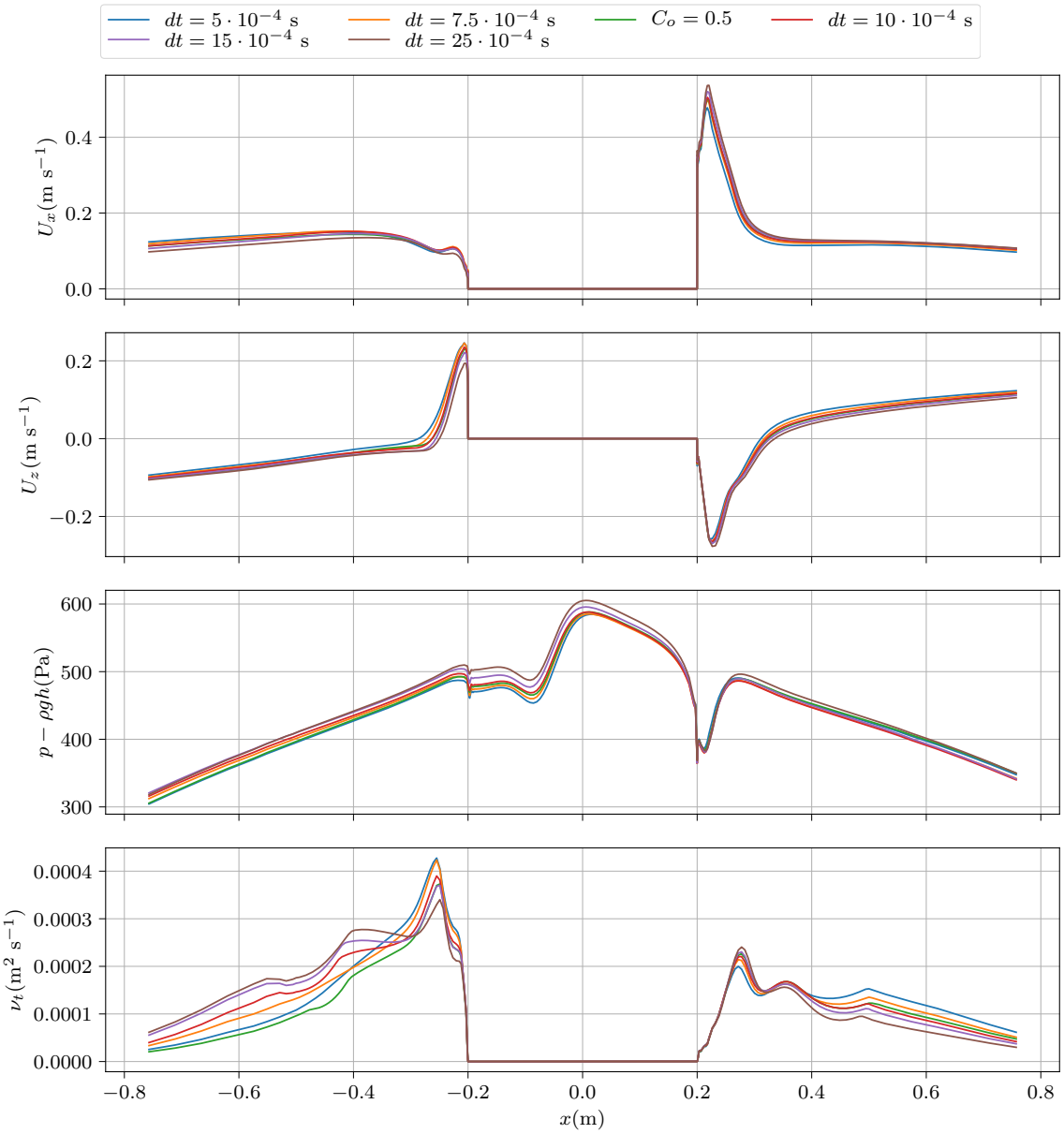


Figure 4.17: Fields sampled over a vertical line at the top most altitude of the cylinder ($z_l = -0.72$ m), at $t/T = 21$ for the coarser mesh "bg045x045dx0056"

(2018) have shown that the turbulent fields were unstable and would follow an exponential growth under potential waves. It was confirmed here, by the study of the evolution of ν_t over time in section 4.5. Thus, we could expect that a modification of the CFL number would impact this growth in a significant manner.

For the rest of this work, and any other computation shown outside this particular paragraph, the time step is fixed at $dt = 5 \times 10^{-4}$s. With this selection, converged results are expected, at least on the horizontal load for the mesh denoted "dx045x045dx0056". It should also *in theory* yield converged results with the finer mesh. In terms of CFL number, this time step is approximately obtained with $\max(C_o) \approx 0.3$ on the coarsest of the two meshes. We would like to recall that this time step is however lower than the one obtained at $C_o = 0.5$ without body (and without the associated discretization).

BC set	**u**	$p - \rho g h$	$k(\mathrm{m^2\,s^{-2}})$	ω $(\mathrm{s^{-1}})$	ν_t
wall1	fV **0**	zG	fV 10^{-10}	fV:10^{10}	`calculated`
wall2	fV **0**	zG	fV 10^{-10}	`omegaWallFunction`	`calculated`
wall3	fV **0**	zG	`kqRWallFunction`	`omegaWallFunction`	`nutkWallFunction`

Table 4.5: sets of wall boundary conditions. zG stands for `zeroGradient` while fV for `fixedValue` and their associated values are precised.

4.6.3 Boundary layer and wall functions.

This section focus on the impact of the boundary layer discretization and the wall function associated with the turbulent variables boundary conditions.

In the near wall region, the distance to the wall, denoted y, is adimentionalized:

$$y^+ = \frac{y u_\tau}{\nu} \tag{4.39}$$

where u_τ is the friction velocity defined using the wall shear stress $\tau_w = \mu \left.\frac{\partial u}{\partial y}\right|_{y=0}$:

$$u_\tau = \sqrt{\frac{\tau_w}{\rho}} \tag{4.40}$$

Then, the near-wall region can be divided into three part:

- The viscous sub-layer ($y^+ < 5$) where the molecular viscosity dominates over the turbulent viscosity. In this region the tangential velocity is linear in y^+.
- The buffer layer ($5 < y^+ < 30$). In this region the velocity is not well modeled and this region aim is mainly a transition layer between the two other layers.
- the logarithmic region ($30 < y^+ < 200$) where the turbulent viscosity dominates over the molecular viscosity. In this region, the velocity follows a log law of y^+.

The main purpose of wall functions is to be able model the above-mentioned behavior without having to discretize and solve the viscous layer region. Models that can handle a first cell located in the logarithmic regions are denoted High Reynolds models, whereas when the first cell should be located in the viscous sub-layer, we employ the term Low Reynolds model.

Here we verify the correct computation on the previous mesh in the body vicinity by comparing the results obtained with different wall modeled boundary conditions. The mesh denoted “bg045x045dx0056“ is selected. A similar mesh is defined, on which the boundary layer is not refined. This zone treatment is the only difference between the two meshes.

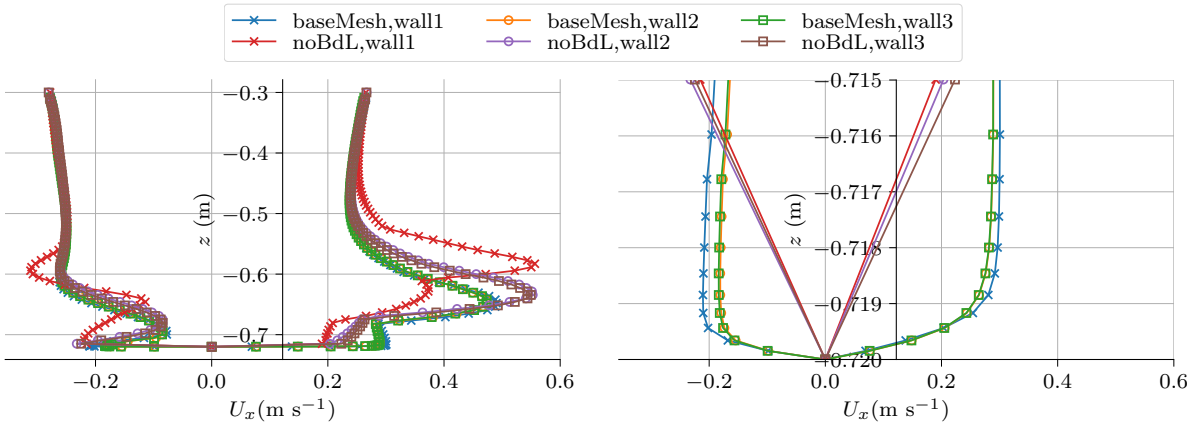
For the basic mesh, the fist cell is located at $y^+ = 6.09$. Note that this value corresponds to the maximum obtained, over all faces and over all times. The averaged over time and space is $\overline{y^+} = 1.55$. When the boundary layer is not discretized however, $\max y^+ = 117.7$ and $\overline{y^+} = 55$. It is thus expected that a wall model is necessary for the second, on which the first cell falls into the logarithmic region, while no model should be required for the original mesh.

The tested sets of boundary conditions are given in table 4.5. The set denoted “wall2“ was used for all previous computations. Note that the `zeroGradient` is used to impose

a Neumann boundary condition of a null value, and `fixedValue` imposes a Dirichlet condition. A complete description of several available wall functions along with their implementations details and analysis is given in the document by Liu (2016). We recall here the main properties of used wall models:

- `kqRWallFunction` is a simple `zeroGradient`, and thus should be used to model a high Reynolds flow in the vicinity of the wall. An equivalent choice, specifically for low Reynold simulations (viscous sub-layer) is `kLowReWallFunction`.
- `omegaWallFunction` is a blending, depending on the computed value of y^+, that is useable for both high Reynolds and low Reynolds simulations.
- `nutkWallFunction` sets the value in the first cell depending on y^+ and the TKE. The first cell should be located in the logarithmic area (Liu, 2016).

Thus, for the mesh on which the boundary layer is discretized, we expect all boundary conditions to yield relatively close results. On the other hand, it is expected that only "wall3" would be relevant when the boundary layer is not solved ($\max(y^+) = 117$).



(a) (b) Zoom

Figure 4.18: Comparisons of the obtained envelopes of the horizontal velocity (computed over 40 time steps in a period) along a vertical line ($x_l = 0$) obtained with different wall models and two meshes ("baseMesh" and "noBdL") differing by their boundary layer discretizations (see text for details). (b) is a zoom on the boundary layer region.

The three sets combined with the two different meshes ("bg045x045dx0056" with and without boundary layer) are computed. Their horizontal velocity over a vertical line centered on the cylinder are sampled at 40 time steps and the resulting envelopes are shown on fig. 4.18.

Firstly, the two different sets of modeled walls (wall 2 and 3) always yield the exact same results. Thus, it is deduced that the ω wall model has a significant effect on the computation. It is also noticeable that the three envelopes of the horizontal velocity on baseMesh are grouped together, with discrepancies only noticeable on the zoomed fig. 4.18b. This tends to indicate that we do not need to invoke a wall model, and that the resolution of this zone is well captured and accurate: a fine enough discretization is used. The same is not true for the computation without boundary layer (noBdL): when "wall1" (tentative to resolve the boundary layer) is selected as a boundary condition set, the prediction of velocity is relatively different from those predicted either with "wall2"

or “wall3” (wall models activated).

Nevertheless, the two different groups (“baseMesh” and “noBndL”), when compared together, exhibit discrepancies. Is was shown that many small differences - for example the corners discretizations in section 4.6.2.1 - can have a relative impact on the obtained profiles, of the same orders of magnitude as observed here. In our case, the body vicinity discretizations - in particular the corner discretizations - are different between the two meshes. It is thought to be the source of these observed discrepancies.

However, if one should trust one of the obtained field profiles, it would be the baseMesh results. Even if powerful model can be set up, the quality of a boundary layer resolution is difficult to match. With the increase in computational powers in the recent years and due to the marginal added cost of such discretization (when the Reynolds number is not too high), those effects are mostly solved nowadays.

In this work, and also for every previously shown computations, the “wall2” set was selected, mostly because this particular study was done at a later date. However, for consistency, either the “wall3” or “wall1” is recommended in this case. In any case, obtained results were done with the boundary layer discretized, and thus no major differences is expected between those three sets.

4.6.4 Conclusions on the investigations studies.

It is relatively difficult at this point to select a “best” mesh and “best” time step value. Both have leverage on the obtained loads and variable fields. While the global variables seem to stay consistent across changes in the mesh density and time step size - as well as changes in the used schemes, not presented here - the local fields values, especially the turbulent viscosity field is sensitive to several of those parameters. One of the issue seems to be the production of turbulent kinetic energy at the sharp corner which are not really converging.

However, the major culprit seems to be the instability of the turbulence model pointed by Larsen and Fuhrman (2018) fed by the generation of TKE at the wall. While every computation exhibits a good almost periodic behavior, the turbulent viscosity field slowly grows over time. The effect of this rise on the other variables is however only visible over long durations ($\sim$ 50 to 100 periods) and thus, good comparison are obtained. At this stage the mesh “bg045x045dx0056” presented in table 4.4 is selected for its contained size and relative good convergence behavior that could be observed on the time step size independence study.

The time step size is selected as a fixed value of 5×10^{-4} s. This value was proved to yield accurate and converged results, at least concerning the physical fields, as well as concerning the predicted wave loads (on which the convergence is achieved earlier). Notice also that the same results are obtained when the time step is controlled by the maximum CFL value of $C_o = 0.5$ which is a common value from the literature.

However if the computationalal resources were more limited - or a good capture of the global variables were the only requirement -, for example during a topological optimisation of the structure, coarser mesh and the associated coarser time step could be selected without losing much in terms of precision on the load predictions. For example, it would be advantageous to select “dx00112”, and an associated time step of 2×10^{-3} s, corresponding to twice the needed time step on our baseMesh to converge the loads am-

plitudes. Savings of 20% in the problem size and of a factor 4 on time step would be achieved. While of lower impact on the savings, the same conclusion can be drawn on the presence of a computed boundary layer: We here chose to maintain the boundary layer discretization, for a cost of approximately 5×10^3 cells ($\sim 5\%$ of the simulation with the dx0056 mesh), but this cost could be lowered by modeling the first cells values without affecting the global results in a significant manner.

4.7 Application and comparison with experimental results

Selecting the mesh "bg045x045dx0056", the wave height is modified so as to simulate different KC number conditions. The main goal is to compare in terms of hydrodynamic coefficients (C_M and C_D) with the HPC method (section 3.3.3), experiments (Venugopal *et al.*, 2006; Arai, 1995) and another numerical VoF simulation (Li and Lin, 2010).

4.7.1 Errors during the wave propagation

In order to compute the hydrodynamic coefficients and compare then with literature results, the same method presented earlier will be used: the minimization of the least square error between the Morison modeled loads and the computed loads. This method was presented in section 3.3.3.2 and is not recalled here. Note that the resulting equations require the knowledge of the kinematic fields ($\mathbf{u}$ and $\dot{\mathbf{u}}$) at the location of the body. Those kinematic fields are of course not directly available and must be determined in another way. The classical approach is to use the same theory as the one that served to generate the waves - that is denoted the wave model - and deduce what would be the kinematic fields at the locations of the body in the absence of the latter. The underlying assumption is that the body presence does not perturb the flow in a significant manner.

Directly applying the method presented in section 3.3.3.2 to the loads predicted by OpenFOAM®, unphysical results were obtained: the drag coefficients were for example negative for large wave heights. In the literature, many numerical errors are pointed out for happening during the wave propagation, such as for example the intrinsic dissipation of some employed numerical schemes, or the unphysical rise of the turbulent viscosity field. For this reason, investigations with OpenFOAM® were conducted on the same mesh without the body to obtain the real kinematic fields the object is subjected to.

Figure 4.19 depicts the evolution of the free surface elevation over time of such computation. The shown results are obtained with the steepest studied case ($H/\lambda = 8.9\%$). A contained error - mostly concerning the amplitude - is done when the main case ($H/\lambda = 3.5\%$ is selected), and is thus not shown here.

Relatively high discrepancies can be denoted, mostly on the wave amplitude. These discrepancies are thought to be the results of the selection of the numerical parameters: numerical schemes, convergence algorithm, *etc.*, because they were chosen mostly to correctly capture the body vicinity effects, and thus differ from the recommendation of Larsen *et al.* (2019) to correctly capture the propagation of the wave. However, a wave amplitude difference - with the same damping for all kinematic fields - would not be much of a problem as the hydrodynamic coefficients would be modified proportionally. In the hydrodynamic coefficient figure, we would just expect consistently lower results.

Nonetheless, any phase shift in kinematics proved to greatly modify the decomposition of the load into its inertial and drag components. From fig. 4.19, the steepest wave surface elevation exhibits a temporal phase-shift of approximately $\delta t/T = 0.03$ at $x = 0$ (*i.e.* after propagating for 24 m) and $t/T = 15$, even though it was negligible when $H/\lambda = 3.5\%$. Thus a phase-shift linear interpolation was selected and applied to the

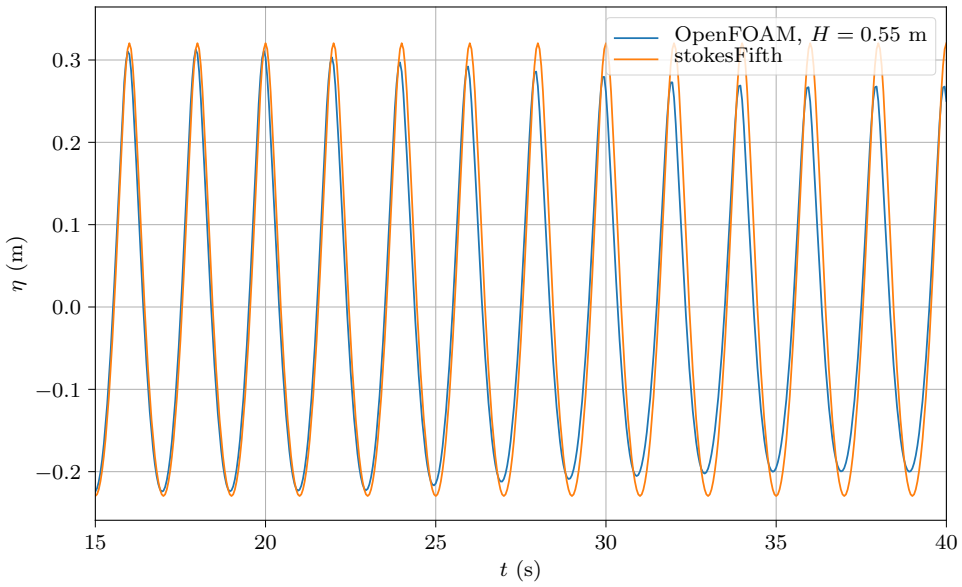


Figure 4.19: Free surface elevation over time at $x = 0$ predicted by OpenFOAM® without body for the steepest wave case ($H/h = 8.9\%$) - Comparison between OpenFOAM® and a Stokes 5^{th} order theory.

theoretical kinematics to render for the OpenFOAM® errors:

$$\delta t(H) = 0.03 \frac{H - H_{min}}{H_{max} - H_{min}} \tag{4.41}$$

H_{min} and H_{max} are the wave heights of respectively the smallest and largest tested waves ($H_{min}/\lambda = 3.5\%$ and $H_{max}/\lambda = 8.9\%$). This maximum value of 3% of the period might not seem significant but it largely affects the resulting hydrodynamic coefficients. Because it models more closely the real fields the object is subjected to, it seems fair to apply this *ad hoc* correction model.

Note that those values were extracted from computations with $T = 2$s but also used, as is, on the computations with $T = 2.5$s. Also note that no correction is done in amplitude, due mainly to the fact that an amplitude error could be interpreted on the figure.

However, those corrections, based solely on the surface elevations at $x = 0$ are not perfect. When comparing kinematic themselves (wave model vs computed kinematics) on the body-free case, the discrepancies (in particular the phase shift) are somewhat different depending on the considered kinetic field (u_x, u_z).

Another possibility would be to compute, with OpenFOAM®, for all cases, the body-free equivalent and extract the kinematic fields from this computation (at every time step). This is not done due to large associated numerical cost and data storage requirement.

Finally, we would like to stress that no such *ad hoc* correction was performed on the HPC results, and more generally, only the `waveFoam` solver with the current parameters requires such a correction: future developed solvers will also not invoke any phase shift correction.

4.7.2 Results and comments

Figure 4.20 shows the different hydrodynamic coefficients computed in both direction compared with literature results. The kinematics on which they are based are corrected from the wave model by a translation in time (see previous §). Note that the other VoF method authors (Li and Lin, 2010) state that a k-ϵ model was used to compute the turbulence, but without further details.

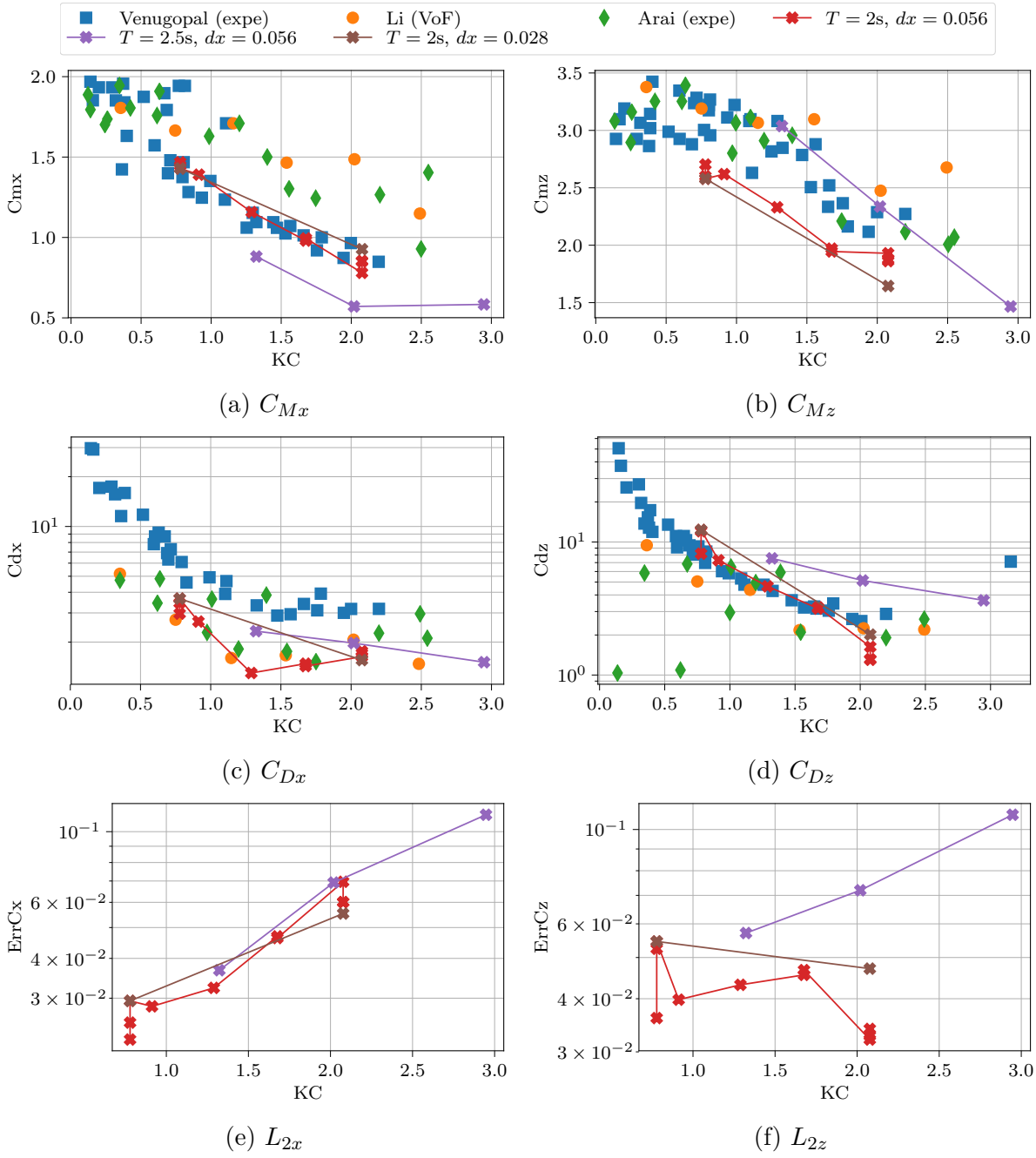
Note that the reconstruction errors (figs. 4.20e and 4.20f) show that the loads computed with the period $T = 2.5$s are not correctly modeled with the Morison decomposition. Thus, no confidence can be granted to the hydrodynamic coefficients results, especially concerning the vertical loads at high KC numbers. Note that, changing the incoming waves period modifies their wavelength, and the mesh (cell sizes, propagation and relaxation zones length) as well as the time step should be modified accordingly. This was however not done which also increase the lack of confidence in those results. In others words, neither the computation of the loads itself nor their conversion into hydrodynamic coefficients are trustworthy, and would require further investigations, for this particular period $T = 2.5$ s with this particular selected mesh (span and discretization).

However, despite these negative aspects, some interesting results are still obtained. First, obtained values are within the expected range for all hydrodynamic coefficients. If we focus on the period for which the mesh was designed, and the time step was selected (*i.e.* $T = 2$ s), horizontal inertia coefficients (fig. 4.20a) compare well with Venugopal *et al.* (2006) and are of lower magnitude than both Arai (1995) and Li and Lin (2010). If the 20% underprediction of the wave elevation were taken into account, results would be closer from the other VoF method.

The vertical coefficients (fig. 4.20b) at the same period of 2 s are underpredicted, but a 20% correction would once again gives us a correct prediction with a good agreement with the literature results. Concerning the drag (figs. 4.20c and 4.20d), the visual impact of such correction would be of lesser importance, since they are shown in a logarithmic scale. Anyway, good quality results are obtain for a relative range of wave nonlinearity ($H/\lambda \in [3.5\%, 8.9\%]$).

Altogether, the underestimation of the wave amplitude seems to be the major culprit in the underprediction of vertical mass coefficients (which is the only coefficient that can be clearly denoted as underpredicted). In other words, given the real kinematics the cylinder is subjected to, valuable results in terms of loads are obtain. A further confirmation of this conclusion is the correct behavior of the coefficients when KC increase.

Finally, from all these observations, the current method is considered as a valuable improvement over the HPC potential method (see fig. 3.16), that was neither able to capture the drag coefficients, nor the evolution of the inertia coefficients with the KC number.



(a) C_{Mx} (b) C_{Mz}

(c) C_{Dx} (d) C_{Dz}

(e) L_{2x} (f) L_{2z}

Figure 4.20: Inertia and drag coefficients in the two directions obtained with OpenFOAM® with different wave periods. The L_2 norm of the error between the Morison model and the real loads is also shown, *i.e.* a large error means the Morison model is not well suited to represent the wave loads obtained with our RANS model.

4.8 Conclusions on OpenFOAM® as a NWT.

Using any code as a NWT would require it to be able to solve many different types of flow and interactions at the same time. First, at large scale, the propagation of the wave should be correctly captured with no significant damping. Using the k-ω SST model modified by Devolder *et al.* (2017, 2018) does not satisfy this first criterion (at least at long times). This would probably be improved with the stabilized technique suggested by Larsen and Fuhrman (2018).

Then the model should also offer a good quality and accuracy in capturing the turbulent and rotational effects close to body. It was proven in section 4.5 that the stabilized model (Larsen and Fuhrman, 2018) has too much influence on the eddy viscosity in this area where, on the contrary the buoyant model (Devolder *et al.*, 2017) is better suited to capture the turbulent effects.

A further modification of the stabilized version was suggested and tested. It was shown to combine the advantages of both mentioned turbulence model modifications, thus being considered as very promising. It consists in a spatial smoothing of some terms of the limiter introduced by Larsen and Fuhrman (2018) (see sections 4.2.4.4 and 4.5.4).

The selection of numerical parameters, numerical schemes and algorithm that would fulfill those two requirements is tricky. If we wish that this selection can be extended to other physical parameters (wave period, wave height, *etc.*) the task becomes challenging. We managed to exhibit a set of parameter (mesh, time step, schemes) that deliver good quality results on the base studied case ($H/\lambda = 3.5\%$). However, applying this set to higher wave heights and larger wave periods was shown to yield a significant surface elevation damping. Overall, the hydrodynamic coefficients are nonetheless correctly captured, especially is one takes into account the real flow kinematics the body is subjected to.

In a design workflow, the selection should however be different: an accurate capture the wave propagation is needed so as to subject the object to the requested wave, even if is it done at the expense of some precision in the local fields description. An *ad hoc* correction consisting in increasing the imposed wave height at the inlet could also be applyied to counteract the emphasized damping.

Not much effort were put into trying to obtain a perfectly suited selection of parameters for all aspects of the flow and all studied wave height. We instead keep in mind that we succeeded in producing relatively good quality results in the body vicinity for the wave stepness of 3.5%, that will serve as reference for the next chapters. In particular, if the mesh stays constant, most of the other parameters only have a relatively limited influence on the body vicinity flow.

CPU costs, orders of magnitude In table 4.6 are presented some of the observed execution time observed during this chapter. They are however indicative and can greatly vary from period to period and run to run. Note that the computations were run on a 40 core Intel® Xeon® CPU E5-2650 v3 @ 2.30GHz. The multi-threading possibility was largely used. In particular most of the expensive computations were run in parallel, using 4 cores: in this case, a star (*) is added next to the associated execution times. Note that the main studied computational case (bg045x045dx0056) was run both in serial and parallel, and the two corresponding execution times are reported in the table 4.6. When

compared, a very effective parallelism is denoted: 4 cores reduce the computation cost for one period by a factor 3.7.

Note that, the first 15 periods are very expensive for the two finest mesh (namely g045x045dx0028 and bg023x023dx0028), on which the parallelism seems effective, as can be seen from the one period execution time. This behavior is not fully understood yet.

mesh	variation	n_{cell}	dt	execution time for $t/T = 0 \rightarrow 15$	execution time for $\Delta t/T = 1$
bg045x045dx0228		112518	5×10^{-4} s	10 h 38 min	0 h 53 min
bg045x045dx0113		117990		16 h 41 min	1 h 13 min
bg045x045dx0056	noBdL	136272		15 h 56 min	1 h 2 min
	base	138592		24 h 1 min	1 h 40 min
				6 h 11 min*	0 h 27 min*
			7.5×10^{-4} s	12 h 47 min	1 h 2 min
			10×10^{-4} s	9 h 42 min	0 h 49 min
			25×10^{-3} s	4 h 20 min	0 h 24 min
bg045x045dx0028		218525	5×10^{-4} s	47 h 33 min*	1 h 20 min*
bg023x023dx0028		425068		85 h 28 min*	2 h 44 min*

Table 4.6: Execution times on 4 processor (*) or 1 processor, for some of the presented meshes and variations. Note that the parallel decomposition seems to not have been correctly parametrized, especially for the coarsest meshes.

Chapter 5

Development of a one-way modular domain coupling method.

Contents

5.1 Introduction 118
5.2 Method, implementation and usage 121
5.3 Validation and comparisons of the different approaches 128
5.4 Sensitivity studies and parameters exploration 139
5.5 Comparison of the hydrodynamic coefficients with results from the literature 149
5.6 Conclusion 152

5.1 Introduction

5.1.1 Method presentation

When evaluating and validating the HPC method in chapter 3, it was shown that the potential hypothesis, while very performant to simulate the propagation of water waves, does encounter limits: for example sharp corners and high Reynolds number effects cannot be correctly captured. The HPC method is fast and accurate to model waves and wave-structure interaction up to high orders of nonlinearity, but its underlying assumptions, namely irrotationality and inviscid fluid, make it unfit to capture effects such as wave breaking, air entrapment, but also vortex shedding, viscous damping, boundary layer effects, *etc.* Some of those effects were shown to have a relative importance on the flow physics on bodies with sharps corners (sections 3.3.2 and 3.3.3).

On the other hand, RANS models, commonly used to solve such effects, were shown (chapter 4) to be hard to set up if the goal is to capture both the wave propagation phase and the small scale effects accurately and consistently. Finally using those approaches separately, yields to either not captured turbulent effect in the body zone (fully potential approach), or too large damping in the propagation zone (fully viscous CFD approach). Furthermore, the computational cost of RANS models is still large, when compared to any potential model.

In conclusion the HPC potential method is better suited to capture the propagation of the waves and simulating most of the (potentials) effects, while a RANS approach is necessary to compute local vortex shedding and fluid-structure interaction at high Reynolds numbers.

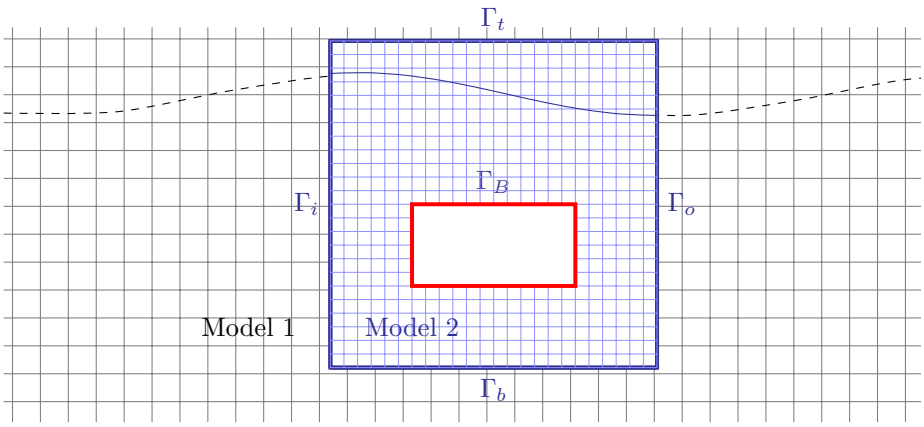


Figure 5.1: Generic schematic of a domain decomposition (DD) solved by different models.

Based on this analysis, this chapter focuses on developing a coupled algorithm, with the aim to benefit from each model where it best performs. A first focus is done on the domain decomposition (DD) method. The general spatial layout is shown on fig. 5.1: two different models, denoted “model1” and “model2” are used to compute the flow in different manners (*i.e.* with different set of equations) in two different - non necessary overlapping - domains. Usually, the “model2”, close to a body, is a more complex model than “model1”. In this case, body induced effects, that cannot be captured by the outer model, are assumed to be of small magnitude in the associated domain. This approach

has been widely used for its relative ease of implementation: no modification of the solver themselves is required. Hereafter make reference to some of them, without any claim of completeness. A more detailed review can for example be found in the recent scoping study of Davidson and Costello (2020). While these authors focus on the applications to Waves Energy Converters, a complete description of most of the current DD implementations used in the recent years is given.

5.1.2 Literature review

Such DD methods were first applied in the aerodynamic field (Lock and Williams, 1987) and were later introduced in the context of ship seakeeping more than two decades ago by Campana and Di Macio (1994) and Campana *et al.* (1995). An early attempt to use a DD method in the context of moored ship is presented in Bingham (2000). Quéméré *et al.* (2001) suggested a coupling approach for highly different block discretizations and applied it to a RANS/LES coupling in Quéméré and Sagaut (2002).

Many types of models can be interfaced. For example, wave theory based models can be used in conjunction with potential or RANS models. See *e.g.* Wei *et al.* (2017) or Christensen *et al.* (2009) for applications of DD, coupling a Boussinesq model and a RANS solver.

Another widely used combination is the matching of a potential solver with the laminar Navier-Stokes equations or RANS equations. The potential equations can for example be solved with a HOS approach (Choi *et al.*, 2018) or a BEM resolution (Colicchio *et al.*, 2006). Recently Hanssen (2019) and Siddiqui *et al.* (2018) coupled NS equations with a HPC method. Siddiqui *et al.* (2018) studied the wave interaction and hydrodynamic of a damaged ships, for which they later released two experimental studies (Siddiqui *et al.*, 2019, 2020).

In Sriram *et al.* (2014), the FNPT, solved with a BEM, is strongly coupled with a FEM NS resolution, and applied to a wave breaking problem. This numerical model is later applied in Kumar *et al.* (2020) to the estimation of long wave run-up.

A solver denoted "qaleFOAM" was also developed, coupling a FNPT using a Quasi Lagrangian Eulerian FEM with OpenFOAM®. The method is presented and applied in Li *et al.* (2018a,b), Yan *et al.* (2019), and Wang *et al.* (2020).

Another DD, very innovative, presented in Kristiansen and Faltinsen (2012) and Kristiansen *et al.* (2013), employs a potential viscous model as the external model. The coupling is then made with a NS based model and both are solved with the finite volume method.

Once again, many more literature references can be found in Davidson and Costello (2020, section 4).

5.1.3 Presentation and objectives of the current approach

The final objective is to implement and evaluate a velocity decomposition method (see chapter 6). Because a DD approach represents a straight-forward intermediate step in the development of the latter, we present in this chapter the obtained solver and results. Thus, a one-way domain coupling approach is selected.

As a first step, the free surface coupling is not tested. The coupling is programmed in C++ in the OpenFOAM® framework. A plan is to release a public version of the code under the GPL v3 license.

Hanssen (2019) developed and tested a numerical scheme that couple a HPC method, with a level-set NS solver originally developed by Colicchio *et al.* (2006). While Hanssen (2019) tried to use OpenFOAM® as the CFD solver, his attempts were unsuccessful and exhibited a lot of drawbacks that are discussed in the cited document. The author explains those difficulties and the final choice by stating that:

- The volume of fluid method yields a less sharp interface, and thus lower precision is achieved when passing information to the HPC method.
- The FVM computes the values at the cell centers, while the FDM method he finally used works with node centered values, resulting in a better consistency with HPC.
- No modification of the source code was done to impose the wanted solution at boundary. He states that the pre-existing `externalCoupled` boundary were found to yield strange results, that made him question the validity of the underlying implementation.

While the first point is not investigated in this work, because the solver is coupled in an unidirectional manner as a first step, the rest of the problematics are tackled here and the last one will be discussed in detail. A new set of boundary conditions along with a new solver, were developed during this work. Note that the term "solver" simply follows the OpenFOAM® notations. More precisely, this is just a new `C++` file, in which most of the calls to methods and equations were copied and adapted from the multiphase VoF solver `interFoam`. A new file is needed mostly for new definitions and calls to the newly implemented toolbox dedicated to coupling methods.

5.1.4 Organisation of this chapter

First, we present the methods that are used and implemented in section 5.2. The opportunity is taken here to briefly present a case setup. While the potential code used in this work is developed solely in 2D, thus only allowing 2D computations, the coupling is relatively modular, such that other external models could be selected without too much difficulty.

In the following, the method is validated in depth in section 5.3 on the same horizontal rectangular based cylinder studied in section 3.3.3 and chapter 4. The new solution, while using the HPC results at the boundary condition, is confronted to the OpenFOAM® solution, used in an independent manner. The latter computation will be denoted `waveFoam` to avoir any ambiguity as the newly developed solver also uses the OpenFOAM® package.

After that, studies will be conducted on the sensitivity of the computation to several parameters (section 5.4). For example, we evaluate the capability of the method to recover if initialized at a latter time ("hotstart" computation, section 5.4.1). It is meant to decrease the computational cost. We also focus on the effect of changes in boundary conditions (section 5.4.3) or in the CFD mesh breadth (section 5.4.2).

Afterwards, in section 5.5, the hydrodynamic coefficients will be computed and extracted for varying wave heights and compared with literature results. Finally, in section 5.6, a summary of the main results will be given and a conclusion on this approach will be drawn.

5.2 Method, implementation and usage

In this section, we describe the numerical methods that are used in the current implementation of the DD solver. The usage of the solver as well as the set-up of a corresponding case is also discussed.

The new solver is developed inside the OpenFOAM® framework. It mimics the implementation of `interFoam`, the 3D multiphase-VoF solver, but is based on a multi-region decomposition available in the OpenFOAM® toolbox. Thus, two types of regions are defined: `external` and `fluid`, and stored into the corresponding list `externalRegions` and `fluidRegions`. The first region type is intended for the importation of results from any external case (also in an OpenFOAM® supported format) while CFD equations are only solved on the second type region(s).

This approach was selected in order to maintain as much modularity as possible: it would be possible in the solver to solve for an equation concerning the `external` regions as well. In this work, the `external` regions are computed *a priori*, and imported onto the `fluid` regions: an unidirectional coupling is considered. Altogether, the presented coupling method is able to use any other OpenFOAM® case as external regions, and tests were conducted coupling a laminar solution of `waveFoam` with a local RANS domain. The associated results will however not be shown here.Although implemented and briefly tested, no coupling in terms of free surface variables is shown in this work. Thus, the body is fully immersed and the CFD domain fully encompasses the structure.

Remark. *The underlying assumption behind a one-way coupling approach is that the additions provided by the internal model (model 2, here CFD), do not have a significant feedback in the outer zone (model 1, here potential). This needs to be respected if one wants to keep the spatial RANS domain of limited extent and still obtain adequate results. The potential model inherently conserves energy, thus trying to match boundaries yields valuable results only if the energy dissipation due to the RANS model is negligible. This is particularly true if one wants to allow for a free surface coupling at two different boundaries,* e.g. Γ_i *and* Γ_o*, see fig. 5.1.*

It would, however, be possible to extend the current coupling to free surface piercing bodies with few different approaches/conditions:

- *Use a CFD domain that encompass only the immersed part of the body, in the same manner as presented in Kristiansen and Faltinsen (2012).*
- *If the case has an inner free surface that is wall-bounded, for example the case of a moonpool free surface studied by Kristiansen* et al. *(2013), this free surface can be treated independently by a VoF method for example.*
- *Imposing a matching of the CFD free surface to the potential free surface through relaxation zones to avoid nonphysical reflection at the CFD domain boundaries (Zhang, 2018).*

5.2.1 Workflow, usage and setup of a domain coupled case

Here is presented the workflow that is followed to prepare and run a case with the current implementation of the domain coupling solver:

1. Compute the HPC simulation of the case (with the body included in the simulation),

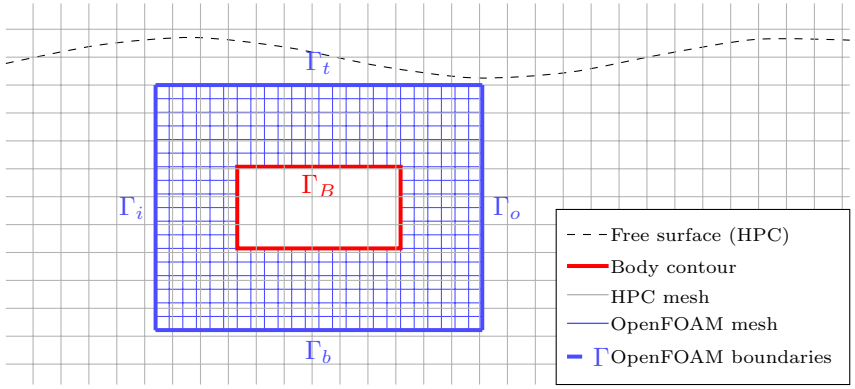


Figure 5.2: Schematic representation of the domain coupling method applied here. Note that only one "external" and one "internal" mesh are represented here. In our application, two external meshes will be used, corresponding to the different HPC grids, namely background and immersed.

2. Convert the HPC results into an OpenFOAM®-like case (mesh and fields in the OpenFOAM® format and locations)
3. Prepare the multi-region OpenFOAM® case:
   (a) In the corresponding dictionary define the names of the `fluidRegions` (to solve), and the `externalRegions` (to import)
   (b) for each `externalRegions`, the OpenFOAM® case with its results obtained in 2.) should be copied in the root directory. Its name should correspond to the one defined in (a). In practice, a symbolic link was set instead of a pure copy.
   (c) Create also a link to (or copy) the `externalRegion` mesh in the directory `constant/<externalRegionName>/polyMesh`
   (d) In the corresponding directories, place selected `fluidRegions` meshes, boundary conditions, *etc.*
4. Run the domain coupling solver

Note that, the algorithm provides the possibility to have multiple potential cases, each one being stored into a different `externalRegion`. This possibility was implemented and is extensively used to allow the importation of all the used meshes in the HPC method (background and immersed, see section 2.4 for further detail). In our application, the finest mesh, located in the vicinity of the body is granted the highest priority, *i.e.* the newly defined boundary conditions will try to use interpolations from this mesh first, if available.

5.2.2 The main algorithm

The one-way coupling method is summarized in pseudo-code in alg. 5.1. A spatial schematic representation of the coupling method is shown in fig. 5.2 with only one external and one internal RANS region.

Algorithm 5.1: Coupling algorithm

```
System Initialization
Define external regions
Define CFD regions

while not(endTime)
   for region in potentialRegions
      do nothing
   for region in CFDRegions
      Solve FVM equations
   t++
```

From that simple algorithm two aspects need to be precised First, the passing of HPC results to the FVM equations at the boundary conditions is done using Object Oriented Programming, trying to respect the OpenFOAM® philosophy. New boundary conditions are developed and used such that a call to one of them - automatically done when solving the FVM equations - will, when needed, trigger i) the reading of the framing (in time) fields values of the potential solution, ii) store them into the corresponding registry for further use and iii) interpolate them at the boundary themselves.

Secondly, the `do nothing` part is obviously not needed in the current method. However, if one wants to solve an equation in the external regions, or even send a signal to another external solver, this place would be adequate. A more advanced time coupling algorithm (see, *e.g.* Hanssen, 2019) would be for example implemented in this file.

A choice was made to perform every interpolation within OpenFOAM® solver. Another possibility could be to compute externally the values directly on the OpenFOAM® mesh. This technique would have a significant advantage as the interpolation could be done directly in HPC code through the interpolation eq. (2.13). This interpolation would then be very fast and accurate. The current approach, although not the most efficient, was selected for mostly three reasons:

- If the interpolation were to be done inside HPC, the OpenFOAM® mesh would be needed before the HPC computation and would behave as an input of the latter, *i.e.* each modification of the OpenFOAM® mesh would request a new HPC simulation.
- The usage of dynamic OpenFOAM® mesh would be very restricted and the implementation of such relatively complex.
- A modular behavior is obtained: we can input any type of precomputed fields, as long as it respects the OpenFOAM® format, and it is possible in the future to implement any potential computation directly in the solver. This would request changes inside the potential loop of alg. 5.1.

A project however is to develop the HPC method within the OpenFOAM® package. In that case, if the external solution would come from a HPC resolution, the corresponding interpolation would be used.

5.2.3 The CFD problem

The inner - or CFD - problem, referred as “model 2” on fig. 5.1, is only a slight variation of the original RANS problem defined in eqs. (4.9a) and (4.9b). The only differences

concern the outer boundary conditions, at which the potential solution is enforced:

$$\mathbf{u}^{(i)} = \mathbf{u}^{(e)} \text{ at } \Gamma_{t,b,i,o} \tag{5.1}$$

$$p^{(i)} = p^{(e)} \text{ at } \Gamma_{t,b,i,o} \tag{5.2}$$

$$\nu_t^{(i)} = 0 \text{ at } \Gamma_{t,b,i,o} \tag{5.3}$$

where the exponent represents the model employed to compute the given field. (i) stands for internal and (e) for external. In order to impose such conditions, we need to have access to the external variables at any time and at the boundary faces locations.

5.2.4 Spatial interpolation

Let's, for a given field $\mathbf{f}$, introduce an index, that stands for the mesh on which the field is defined. Together with the exponent defined above, $\mathbf{f}_e^{(e)}$ then represents the field computed by the external solver, defined on the external mesh. So, once interpolated on the CFD mesh (inner mesh) the resulting field is denoted $\mathbf{f}_i^{(e)}$. This field will, in our case, be both the velocity $\mathbf{v}$ and dynamic pressure $p - \rho gh$. With other coupling approaches it could also include the turbulent viscosity ν_t for a turbulent coupling (*i.e.* RANS / LES coupling).

Remark. *The HPC method computes and solves for fields defined at the nodes, by opposition to the FVM that works with cell centered fields. However, OpenFOAM® is able to read fields defined at nodes. Thus, a point-to-cell field interpolation is used and can be switched off if the external case is instantiated with an already cell centered field.*

Hence, we assume that $\mathbf{f}_\mathbf{e}^{(\mathbf{e})}$ is known and defined on the external mesh (output of HPC converted from point-to-cell but still on the HPC mesh). This field needs to be interpolated onto the CFD mesh in order to obtain $\mathbf{f}_\mathbf{i}^{(\mathbf{e})}$. For a domain coupling, the interpolation is needed at least at the outer boundary faces. Note that it is also needed everywhere if one wants to compute a hot start using the precomputed potential values as a starting point. Two different types of interpolation functions are currently considered and described hereafter.

Weighted method For any point $\mathbf{P_i}$ centroid of either a boundary face or a cell of the inner mesh, the objective is to find all points $\mathbf{P_e}$ of the external mesh having an influence on $\mathbf{P_i}$. An associated weight is computed for each of them. In this work the weighted method used and tested is based on the inverse distance function. In the implementation of this method, origin points $\mathbf{P_e}$ are searched within a given radius d_{max} from the location of interest ($\mathbf{P_i}$). Then their weights are computed as

$$w = \left[\frac{1}{d(\mathbf{P_i}, \mathbf{P_e})}\right]^p \tag{5.4}$$

and normalized afterwards. Here, $d(\cdot)$ is the 3D distance function. With this method, one needs to take care when selecting the power p: when the external point is farther than d_{max}, the added weight would be small enough to be neglected without significant loss, *i.e.*, once normalized $\overline{(1/d_{max})^p} \ll 1$. Note that both the identification process (at first

time step only if the meshes are fixed) and the computation of the interpolated fields (at each time step) are increasing with d_{max}. Due to significant increase in the computational cost without evident gain in the interpolation quality, this method is not used during this work, but still available in the developed software.

Direct methods First, we identify the cell c_e of the external mesh in which $\mathbf{P_i}$ is located. Afterwards, many methods are available. In this work, the `cellPoint` method is selected. First, the external field $\mathbf{f_e^{(e)}}$ is interpolated onto the cell nodes giving us access to both nodes and cell centers values. Then, the cell c_e is discretized in tetrahedrons pointing towards the cell center. Finally, the interpolation is done using the tetrahedron in which the $\mathbf{P}_i$ is situated: the weighting is obtained from its barycentric coordinates with respect to the tetrahedron vertex locations.

Note that whatever the method, a step of identification is necessary to compute the addressing (and weights if necessary), before interpolating the field. This process can be computationally expensive and is required at each time step only if the meshes are moving relative to each other. So, in our case, the interpolation is updated, when the external field changes, with the new values of $\mathbf{f_e^{(e)}}$ but uses addressing (and weights) that were computed at the first time step.

Remark. *During this work, both methods were re-coded locally, inspired from the source code of OpenFOAM®. This was necessary as the discretization of CFD and internal mesh were very different, and the existing routines could be improved in such conditions. Moreover, no precise interpolation onto the boundaries faces was found: the only one available seemed to interpolate on the closest cell and directly project its value from there (*i.e. *assume a null gradient along the face normal).*

5.2.5 Time interpolation

The above described procedure can only be done with the knowledge of $\mathbf{f_e^{(e)}}$. However, this knowledge is only guaranteed at the external time steps (from the results of the potential model). The HPC time discretization is however approximately two orders of magnitude higher than the expected CFD one.

Hence, we first interpolate in space these available fields, giving us the corresponding fields on the CFD mesh $\mathbf{f_i^{(e)}}$ at those time intants only. Afterwards, whenever requested (*i.e.* at each CFD time step), a time interpolation is performed from these fields. Currently the only implemented time interpolation is linear. Recalling that the HPC methods was working with a time discretization of about $T/30$, the linear interpolation could not be sufficient to obtain correct results. C++ objects that handle the time interpolations are defined such that it could be easily modifiable to higher order interpolations.

To sum up, in the current implementation, if we need a potential field on the CFD mesh at a given time t, we seek the closest potential output result times $t1$ and $t2$ framing t and we then interpolate spatially at these times (if not already done), as described in section 5.2.4. Finally, we compute $\mathbf{f}_i^{(e)}(t)$ linearly from the obtained fields $\mathbf{f}_i^{(e)}(t_1)$ and $\mathbf{f}_i^{(e)}(t_2)$.

5.2.6 Boundary and initial conditions

In the domain coupling framework, if the field computed with the external solver is defined on the CFD mesh $\mathbf{f_i^{(e)}}$, it is used to initialize and set the boundary conditions of $\mathbf{f_i^{(i)}}$. Thus,

- At $t = t_0$, $\mathbf{f_i^{(i)}} = \mathbf{f_i^{(e)}}$: initialization is made with the fields computed with the external solver. This is not necessary but permits to drastically approach the initial solution from the real solution. This way, the periodic behavior is reached within less iterations. If the starting time of the simulation is not zero - and thus, $\mathbf{f}_i^{(e)} \neq \mathbf{0}$ -, this computation is referred to as a "hotStart". Discussions and results are available on this method in section 5.4.1.
- At all times, the boundary conditions are set as equal:

$$\mathbf{f_i^{(i)}} = \mathbf{f_i^{(e)}} \quad \text{at} \quad \Gamma_{t,b,io} \tag{5.5}$$

Lets recall that $\mathbf{f}$ represents (in our case) both the velocity and the pressure fields. However, eq. (5.5) cannot be enforced directly: in practice, errors are made both during the computation of $\mathbf{u}_e^{(e)}$ (HPC model) and the interpolation from one mesh to the other to obtain $\mathbf{u}_i^{(e)}$. Thus, if all surrounding boundaries are of Dirichlet type, one would impose an unbalanced mass flux in the CFD domain. In our case, this excess mass relative order of magnitude is 10^{-7}, but it is still large enough for OpenFOAM® to trigger a mass unbalance error. Moreover, as energy is dissipated in the RANS domain, but conserved in the potential domain, such enforcement would probably lead to stability problems. To overcome these issues, a lot of type of conditions can be used. The main ones and their behavior are described below.

- `fixedValue`: the simple Dirichlet condition. It imposes directly the value at the face, that will later be used when operators (Laplacian, divergence, gradient, *etc.*) are discretized for resolution. See section 4.3 for details.
- `zeroGradient`: Neumann condition. When evaluated, the boundary face value is set equal to owner cell value. So a constraint is relieved there and both can evolve simultaneously. Thus, this boundary condition does not take any input value.
- `fixedGradient`: Neumann condition. Same as `zeroGradient` but imposes a user-entered value for the jump from the nearest cell center to the current face.
- `inletOutlet`: Mixed Dirichlet-Neumann condition. Switch from `fixedValue` to `zeroGradient` when the fluid flows outwards. The current face mass flux is used to determine the flow direction. Input values are required, but will only be enforced as Dirichlet when the fluid flow inwards.

Thus, depending of the boundary type, defining the keywords `value` or `inletValue` may be required.

To effectively set the velocity or pressure at the boundary location from an interpolated field - in other words automatically fill the required inputs presented above - , new boundary conditions types have been implemented that inherit from the one presented above, namely:

- `coupledPressure`: equivalent of `fixedValue`. The enforced value itself, however, is computed automatically from the existing `externalRegions` pressure field.
- `coupledVelocity`: same as `coupledPressure` but for the velocity field.

- `coupledVelocityInletOutet`: equivalent of `inletOutlet` but if the face flux is positive, the enforced values are computed from the potential field, and a `zeroGradient` otherwise. Only the velocity version has been implemented.

Note that those boundary conditions are independent and do not, in theory, require any input. However their C++ classes inheriting from the default ones presented above, the same parameters needs to be filled: the keywords are `inletValue` for `coupledVeloctyInletOutlet` and `value` for `coupledPressure` and `coupledVelocity`. Those values are mandatory but dummies and will be overwritten as soon as the boundary is evaluated. The relevant inputs (origin regions for example) are given trough global dictionaries to be able to access them from different parts of the code.

A lot of different combinations of those boundary types were tested with few of them yielding to a stable computation. Numerically, the combination of `coupledVelocityInletOutlet` for the velocity and `coupledPressure` for the pressure proved to be a good option, as it allows to impose the pressure as a Dirichlet condition and the velocity has, at any time, a degree of freedom that can be used to ensure the mass balance. With this set of outer conditions, every computations ran during this work proved to be stable.

Remarks.

- *With the `coupledVelocityInletOutlet` option, an inconsistency is present on the velocity when the flow goes outwards, as there is no physical reason for the flow to have a null gradient there. A better option would be to implement a boundary condition that switches at the same moment but can still impose a non null gradient (originating from the potential velocity field). Due mainly to a lack of time, this extension was not done during this work.*
- *Other types of boundaries were also developed and are available. They mainly concern either the volume fraction field α directly or boundaries that change depending on the volume fraction value (to behave differently in the air and in the water domain). They are, however, not presented here.*

5.3 Validation and comparisons of the different approaches

In this section, the method described above is applied on the horizontal rectangular-based fully-immersed cylinder case that was presented in section 3.3.3. Comparisons are made with the potential results alone and the full CFD ones presented in section 4.7. The latter were obtained employing the `waveFoam` solver (Jacobsen *et al.*, 2011). Thus, this solver name, "waveFoam", will be used to reference the associated results. In section 5.3.1 the selected parameters for the base computation are defined (boundary conditions, input and numerical parameters, mesh, *etc.*), and in section 5.3.2 an extensive presentation and discussion of the results are conducted.

5.3.1 Base parameters of the simulation

Because we aim to reproduce the obtained effects captured by `waveFoam`, an effort is made in keeping the parameters as similar as possible.

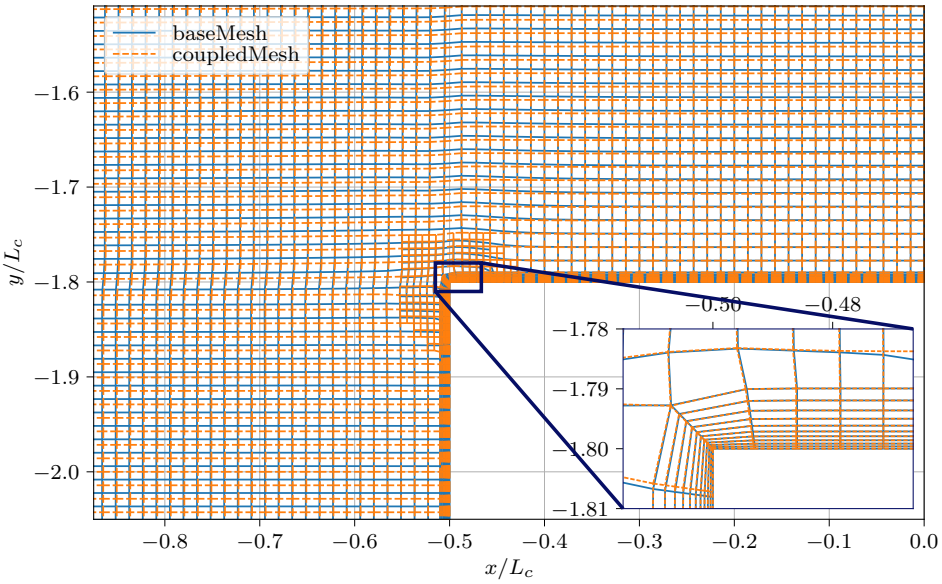


Figure 5.3: Coupled mesh used vs mesh used for the computation with waveFoam alone

CFD mesh The base mesh is inspired from the mesh denoted bg045x045dx0056 presented in section 4.4.2 and described in table 4.4. It is generated with the same procedure. The spatial extent, both vertical and horizontal, are the only major difference. Of course, no refinement is done close to the SWL (not in the domain).

A comparison of the resulting mesh with the original waveFoam one is shown in fig. 5.3. Some minor differences in the body vicinity can be noticed. It was shown (chapter 4) that many differences, that can be considered small, had influence on the results. Thus, one should keep in mind that even this almost inexistant difference in the body vicinity might affect the results.

BC set	boundary	$\mathbf{u}$	$p - \rho g h$	$k(\mathrm{m^2\,s^{-2}})$	ω $(\mathrm{s^{-1}})$
1	Γ_i	cIO	cfV	fV:10^{-10}	IO: 0.88
	Γ_t	cIO	cfV	fV:10^{-10}	IO: 0.88
	Γ_o	cIO	cfV	fV:10^{-10}	IO: 0.88
	Γ_b	cIO	cfV	fV:10^{-10}	IO: 0.88

Table 5.1: Table of the base set of boundary conditions. IO stands for inletOutlet, fV for fixedValue. A prefix "c" is added when their coupled counterparts are used.

The total horizontal breadth of the mesh is chosen initially as $B_m = 3\,\mathrm{m}$ (*i.e.* a scaled value of $B_m/\lambda = 0.48$ or $B_m/L_c = 7.5$) and its height $H_m = 1.5\,\mathrm{m}$ ($H_m/h = 0.68$). The breadth value has been chosen such that the mesh width covers the region where ν_t is large. From fig. 4.7, if we with to encompass all the region where $\nu_t \geq 2 \times 10^{-8}\,\mathrm{m^2\,s^{-1}}$, this zone should extend approximately from $-1.2\,\mathrm{m}$ to $1.3\,\mathrm{m}$ at $t/T = 15$. It was however shown later (section 4.5.5) that the zone of significant ν_t expands horizontally over time. Anyway, a study on the independence with respect to the horizontal width will be conducted in section 5.4.2.

Note that the resulting mesh has a horizontal span reduced by a factor 16, compared to the mesh used with waveFoam and a total height reduced by a factor ~ 1.5. However even if the area on which the NS equations are solved is thus reduced by a factor ~ 24, the relative refinement close to the body generates such a large number of cells that the total size of the system is only reduced by a factor ~ 4.5 (138k cells vs 31k). This remark holds because a relatively fine mesh was chosen as the "base" case: if one wants to compare coarser meshes (*e.g.* $dx = 0.0112\,\mathrm{m}$ in table 4.4), the relative decrease in terms of cell number would be larger.

Boundaries The four outer boundaries of the CFD mesh are set as `coupledVeloctyInletOutlet` for the velocity and `coupledPressure` for the pressure. This means that the pressure will be directly applied as a Dirichlet conditions while the velocity will swap between a null Neumann condition and a fixed imposed value depending on the sign of the face flux at the boundary face (more details in section 5.2.6). Except the gradient of the velocity, which is imposed as null when the fluid flows outwards, the values are set according to the spatial and temporal interpolations from the external mesh (as described in section 5.2).

The turbulent boundary conditions are set as `fixedValue` for the TKE k and as `inletOutlet` for the rate of dissipation ω. They are summarized in table 5.1. The goal is to enforce almost null values of the turbulent kinetic energy and a dissipation when the fluid flows towards the domain. Note that influence of variations of the turbulent boundary conditions are evaluated in section 5.4.3.

Numerical parameters Some of the schemes employed here are different from the ones presented in section 4.4.3. The differences are outlined here and their influence will be discussed in section 5.4.4.

All gradient discretizations are chosen as `Gauss linear`, which is also selected for the advection of the volume fraction by the compression flux. The discretization of the divergence of the velocity advection ($\nabla \cdot \rho \mathbf{u} \otimes \mathbf{u}$) is done by a `Gauss linearUpwind` scheme, which is an upwind approach corrected by the velocity gradient. However,

a mistake in those files was detected here, and is present in almost all computations discussed in this chapter: surface normal gradients and Laplacian schemes neglect the non-orthogonal part (*i.e.*, set as `orthogonal` instead of `corrected`). However, in the OpenFOAM® manual, this approach is only recommended when dealing with highly orthogonal meshes (<5° to 10°). However, it will be shown that its influence is not significant in the considered case.

Physical parameters & external results The results denoted here as "external" (potential HPC code) were presented in section 3.3.3, *i.e.* we use a model that already takes into account the body presence. The corresponding results are converted into an OpenFOAM® format and serve for the imposition of the outer boundary values.

In the same manner as in the previous chapter, the base case involves regular waves of steepness $H/\lambda = 3.5\%$ and period $T = 2$s. The corresponding HPC case is thus used here, in all the computations (except of course, for the varying wave height study performed later).

Finally both "fitted" and "background" meshes of the HPC case are imported, and the priority is given to the former: its cell density is larger and thus interpolations, when available, are expected to be more accurate than if they were done using the "background" values.

Time step, starting time, & others selected parameters to mimic waveFoam We here briefly sum-up some parameters that were selected in the previous chapter and will be maintained for our "base" DD simulation. The time step is fixed at $dt = 5 \times 10^{-4}$s, *i.e.* the value that was selected from the independence study with `waveFoam` (section 4.6.2.2). The starting time is also (in a first approach) set at $t_0/T = 0$. However, a significant gain can be achieved by modifying this value, a study will afterwards be conducted on this particular parameter (see section 5.4.1)

As for the turbulence model, we select the buoyant variation suggested by Devolder *et al.* (2017) of the $k - \omega$ SST model (see section 4.5.3 for details and section 4.5.3 for results and discussions).

5.3.2 Results

In this section, we aim to validate the current model with the parameters presented above. The three different models that were computed on this test case, are then compared, namely:

- The HPC method, presented in chapter 2 for which the results on this test case are presented in section 3.3.3. Note that this case also serves as the "external" model in the current approach.
- OpenFOAM® alone, denoted here "waveFoam" to avoid confusion, on which a focus is made on chapter 4. The corresponding predictions were investigated and discussed in section 4.7, then compared with experiments in section 4.6.
- The present domain coupling solver.

Loads of the cylinder The loads predicted by those three methods are compared in fig. 5.4. They are correctly predicted by the domain coupling method, especially the

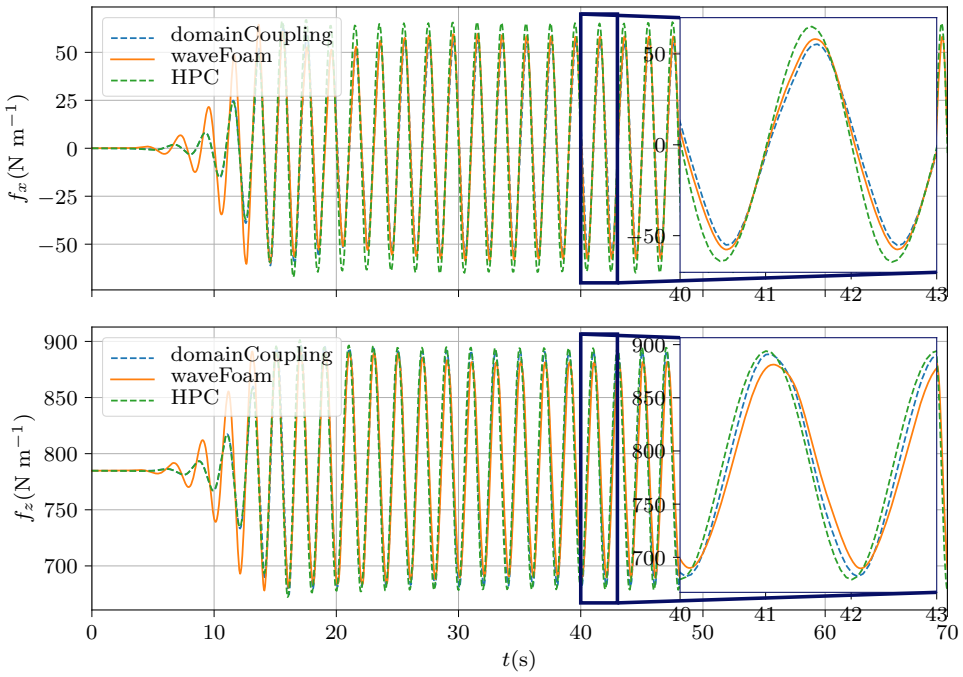
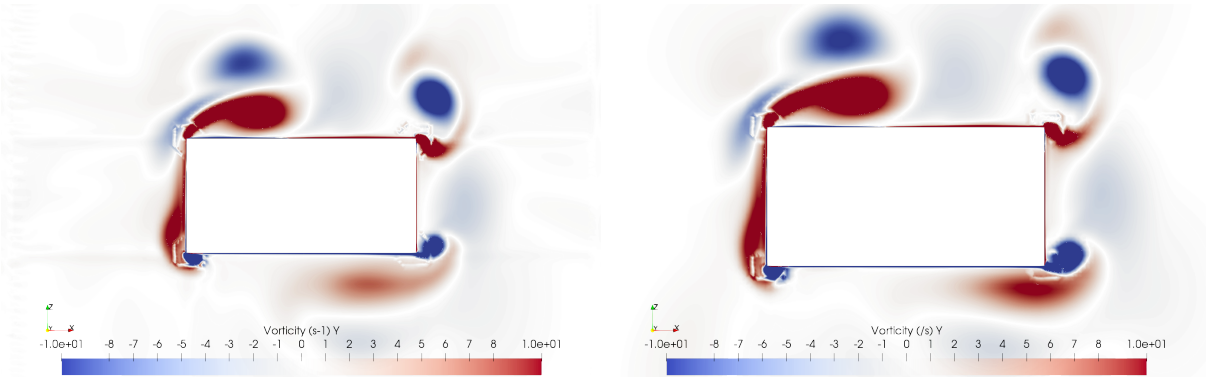


Figure 5.4: Horizontal and vertical loads obtained from the coupled algorithm compared with the waveFoam results.

horizontal load. However, the vertical load is less impacted by the presence of the CFD zone: the potential one is almost retrieved. Remember that, during the waveFoam study, the vertical load was difficult to capture in a precise manner. It exhibited a higher sensitivity to many changes than its horizontal counterpart (see *e.g.* section 4.6.2.1).

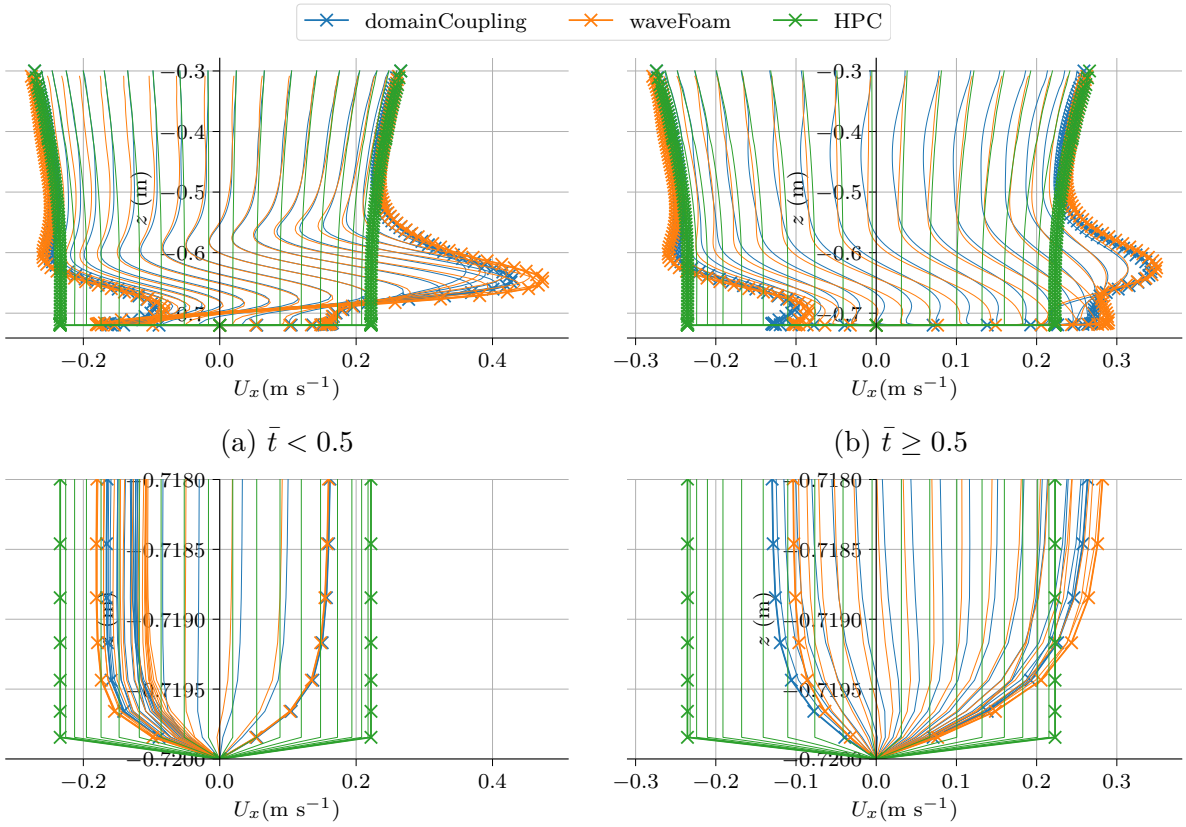
Moreover, compared to experiments (section 4.7 and fig. 4.20), the vertical inertia coefficients were shown to be under predicted with waveFoam. It was thought to be mainly the result of the kinematic fields damping, happening during the wave propagation. Here, we rely on HPC for this task, thus, the resulting propagation error expected to be almost negligible (compared to the one made with waveFoam). On the contrary, at this KC number of 0.78, HPC was shown to yield an excellent prediction of the vertical inertia coefficient when compared to experiments (see fig. 3.16b vs fig. 4.20b). Retrieving the vertical load predicted by HPC is therefore preferable.

Vorticity field The vorticity field is shown, at the usual time $t/T = 15$, on fig. 5.5. Some remarks can be done on the quality and regularity of the vorticity field, particularly close to the corners. We recall that an added refinement zone was set in the vicinity of each corner. The perimeters of these zones are noticeable on the figure due to the relatively large perturbations that are present. Note that the computation and display of the vorticity is done by Paraview (and its dedicated "filter"), which might be a source of error. However, a qualitatively good agreement is deduced from this side-by-side comparison, that shows that our domain coupling model is able to retrieve rotational effects even with a small domain which boundary conditions are set as irrotational. This also gives confidence in the size of the CFD mesh: it is sufficient to capture the effects not taken into account (and thus not imposed) by the potential model.



(a) domain coupling (b) waveFoam

Figure 5.5: Comparaison of the vorticity fields predicted by the domain coupled model and waveFoam at $t/T = 15$



(a) $\bar{t} < 0.5$ (b) $\bar{t} \geq 0.5$

(c) $\bar{t} < 0.5$ (d) $\bar{t} \geq 0.5$

Figure 5.6: Horizontal velocity profiles predicted by the three models, at different time steps (40 time steps per period) separated into half periods, along a vertical line ($x_l = 0\,\text{m}$ $z = -0.72\,\text{m}$ to $-0.3\,\text{m}$). Envelopes computed on these time steps are also added for comparison. Bottom subfigures are zooms on the boundary layer part. $t/T \in [15, 16]$.

Horizontal velocity profiles To have a more quantitative insight, the horizontal velocity profiles, at different time steps, predicted by waveFoam and by the domain coupling model are shown on fig. 5.6. Fields are sampled over a vertical line located at $x_l = 0\,\text{m}$,

from $z = -0.72\,\mathrm{m}$ to $-0.3\,\mathrm{m}$ (whole coupled CFD domain height). The left figs. 5.6a and 5.6c regroup all profiles of the first half period ($\bar{t} < 0.5$, where $\bar{t}$ is the scaled remainder of the Euclidian division of t by T), while the right hand side figs. 5.6b and 5.6d regroup the others ($\bar{t} \geq 0.5$). Bottom figs. 5.6c and 5.6d are zooms of the data in the boundary layer zone. In addition the maximum and minimum values of the horizontal velocity are computed at each location for all the available time steps within the half period of interest (envelopes, marked with crosses x at each face encountered) .

Note that, following the previous notations (introduced in section 5.2.4), the depicted HPC results are not exactly $\mathbf{u}_e^{(e)}$ (*i.e.* the HPC fields on the HPC mesh), but rather $\mathbf{u}_i^{(e)}$ (*i.e.* the corresponding fields, but interpolated on the inner mesh). Hence, they are directly the fields used to impose the BC: if the flux flow inwards, these HPC fields and the domain coupling ones should exactly match at the boundary location (here at $z = -0.3\,\mathrm{m}$). It will be shown later (vertical velocity profiles, fig. 5.7) that an inward flow happens approximately in the first half period (left figures). This enforcement can indeed be noticed on the left fig. 5.6a for all thin lines. A slight phase shift compared to waveFoam can also be denoted, that our domain coupling method does not recover.

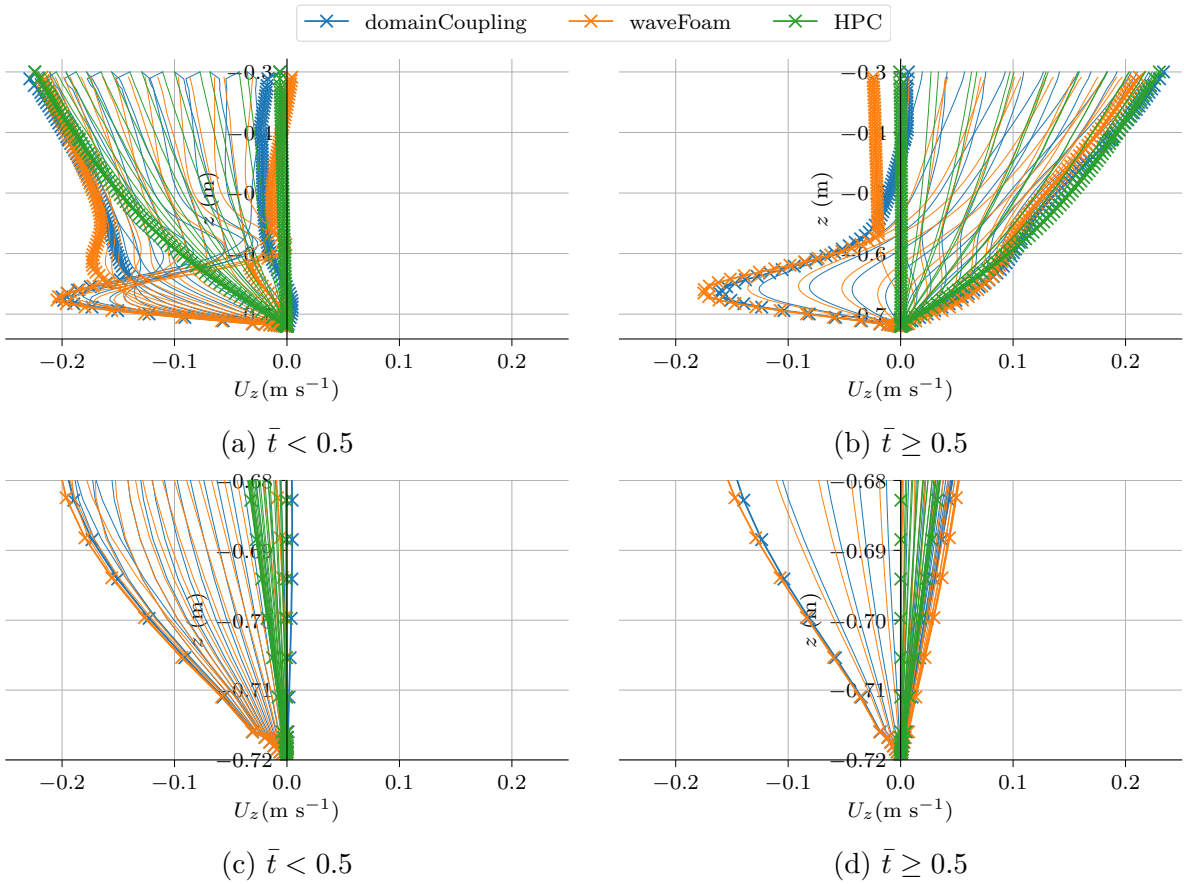
Remark. *On the body wall, no interpolation was requested (not needed in this framework),* $\mathbf{u}_i^{(e)}$ *is thus set as default to zero on* Γ_B*. However, this does not mean that HPC yields a null value there (*$\mathbf{u}_e^{(e)}(\Gamma_B) \neq 0$*). This will be taken into account when needed in the next chapter (see for example fig. 6.7a).*

A really good agreement can be denoted between the horizontal velocity field obtained with the FVM-VoF method alone (waveFoam) and the domain coupling model. Apart for some small discrepancies in the boundary layer zone, the domain coupling velocity is correctly computed and resemble a lot more the waveFoam field than the HPC field. This further emphasizes, this time in a quantitative manner, the capabilities of the current approach to recover the sought effects (turbulent, vortical), despite the limited CFD domain span.

Vertical velocity profiles The vertical velocity profiles, plotted in fig. 5.7, exhibit larger discrepancies. Note that the zoom is not performed on the same zone as the boundary layer zone does not exhibit a particular interest for this variable. The profiles are slightly different, seemingly due to the relative differences between the HPC and waveFoam model at the top boundary location, *i.e.* $z = -0.3\,\mathrm{m}$. The domain coupling solver tries to follow the HPC values (imposed at that location) and thus cannot exactly recover the waveFoam results.

In the first half period, the difference in behavior might be due to phase shift: only 40 time steps are used, which proved not to be enough when large time derivatives are at play, to accurately capture the envelopes. A good indicator for that is the relative spacing of the thin lines. For example, no line is present between the envelopes of the different models on the left of fig. 5.7b, which shows that a small phase shift might be the reason of the envelope discrepancies.

The effect of the `coupledVelocityInletOutlet` boundary condition (explained in section 5.2.6) is clearly noticeable on this figure. Indeed, the velocity values are enforced as Dirichlet conditions when the fluid flows inwards. It means that whenever U_z is negative at $z = -0.3\,\mathrm{m}$, the domain coupled velocity field needs to match the HPC predicted one



(a) $\bar{t} < 0.5$ (b) $\bar{t} \geq 0.5$

(c) $\bar{t} < 0.5$ (d) $\bar{t} \geq 0.5$

Figure 5.7: Vertical velocity profiles predicted by the three models, at different time steps (40 time steps per period) separated into half periods, along a vertical line ($x_l = 0$ m $z = -0.72$ m to -0.3 m). Envelopes computed on these time steps are also added for comparison. Bottom subfigures are zooms on the bottom part. $t/T \in [15, 16]$.

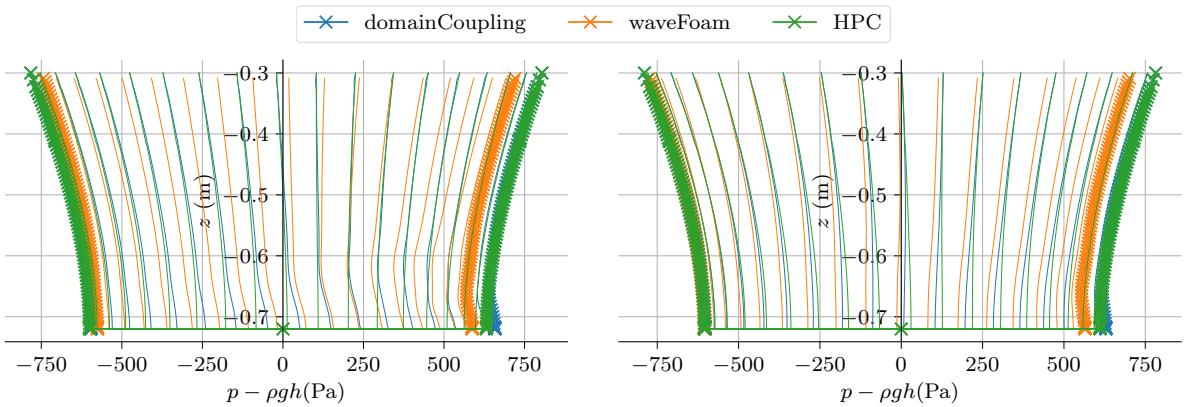
at the top. Thus, on the left fig. 5.7a, the vertical velocity is negative and we clearly see a jump between the face values at $z_l = -0.3$ m and the next values just below that. This jumps denotes a difficulty for the solver to impose the given values, and retain a smoothed matching with the rest of the domain.

However the behavior is smoothed when $U_z > 0$ (most lines of fig. 5.7b) and we note that the obtained velocities indeed differ from the HPC values at the boundary location. This same effect could also be denoted, as was previously pointed out, on the horizontal velocity in fig. 5.6. However the matching yielded smoother results on the horizontal velocity than on the vertical one with less difficulty to enforce the boundary condition when a Dirichlet condition was imposed.

Let's recall that the domain coupling method does not compute its own free surface - and associated kinematics - but indirectly summons the HPC free surface *via* the CFD domain boundary conditions. Thus, if waveFoam and HPC fields differ, it is logical that this difference will be retrieved when comparing waveFoam and the domain coupled solver.

Remark. *It would however be possible to couple the free surface to try to counteract some of these discrepancies. However, the dissipation due to the wave propagation that happens over long distance with waveFoam will not be taken into account with such a small mesh*

*breadth. It is possible that in our case, no significant improvement would be obtainable. This coupling in terms of the volume fraction variable α with the discrete HPC free surface was done during this work. However, for the moment, stability issues were encountered, and this coupling would require deeper investigations: it is not easy to enforce a potential free surface (*i.e. *no energy dissipation) at both side of the CFD mesh while some energy is dissipated in the RANS-VoF inner region. Here, a limit of the unidirectional coupling is obviously met.*



(a) $\bar{t} < 0.5$ (b) $\bar{t} \geq 0.5$

Figure 5.8: Dynamic pressure profiles predicted by the three models, at different time steps (40 time steps per period) separated into half periods, along a vertical line ($x_l = 0$ m $z = -0.72$ m to -0.3 m). Envelopes computed on these time steps are also added for comparison. $t/T \in [15, 16]$.

Pressure profiles The pressure that is yielded by HPC is computed from the Bernoulli equation on which the time derivative of the potential is obtained by a resolution of a dedicated Laplace problem. The obtained pressure is however slightly not periodic (as a consequence, the HPC horizontal and vertical loads amplitudes vary respectively in a 1.5% and 1.1% range). Moreover, the HPC and waveFoam pressures differ at the boundary location.

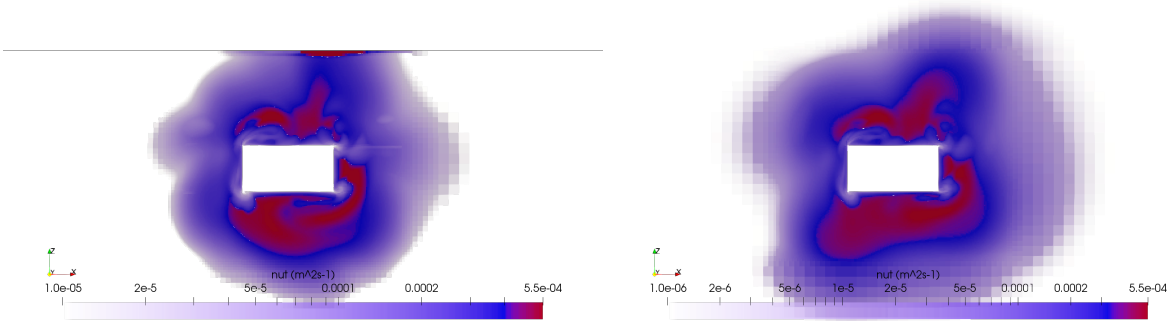
Thus, discrepancies are expected on this variable. Those discrepancies can easily be observed on fig. 5.8 on which the envelopes differ. This is not only related to a phase-shift problem as both HPC envelopes are wider than the waveFoam results (and a time step marking line is present between the different model envelopes). On this variable, the domain coupling solver matches almost perfectly the HPC pressure with however small differences when approaching the body. Note that the obtained curve reproduce almost exactly the waveFoam pressure behavior: at the end, the waveFoam pressure behavior is obtained, with a shift, due to the imposed boundary value, that is propagated in the domain in a constant manner.

Turbulent variables fields Turbulence modeling is a challenge under potential-like waves. It was shown previously, on a waveFoam case alone, that the turbulent viscosity field tends to grow over time, never reaching a physical periodic behavior. Moreover, this instability, and its growth, are dependent on many - even small - numerical parameter

modifications (*e.g.* mesh size, corners discretization, time step size, *etc.*) For further details and discussions, the reader is referred to section 4.6.

With the aim to maintain the comparison as good as possible, we tried to mimic as closely as possible the waveFoam simulation: i) the mesh is almost the same as the one used in the waveFoam computation; ii) the simulation starts at $t = 0$ to allow the turbulent viscosity to grow in the same manner; iii) the selected and imposed fixed time step is the same in both cases.

However some discrepancies, that will play a role in the development of the turbulence instability are present, namely: 1) the meshes are not *exactly* the same, 2) once the turbulence development reaches the edge of the domain coupled inner region it will be shut off by null boundary conditions and 3) no turbulence will be generated and perturb our results by both the free surface and the fluid in the potential propagating domain. Note that the third aspect was found - when using the $k - \omega$ SST model modification suggested by Devolder *et al.* (2017) - not to be predominant: it was the turbulence generated at the body walls that was the main source of the TKE growth.



(a) domain coupling (b) waveFoam

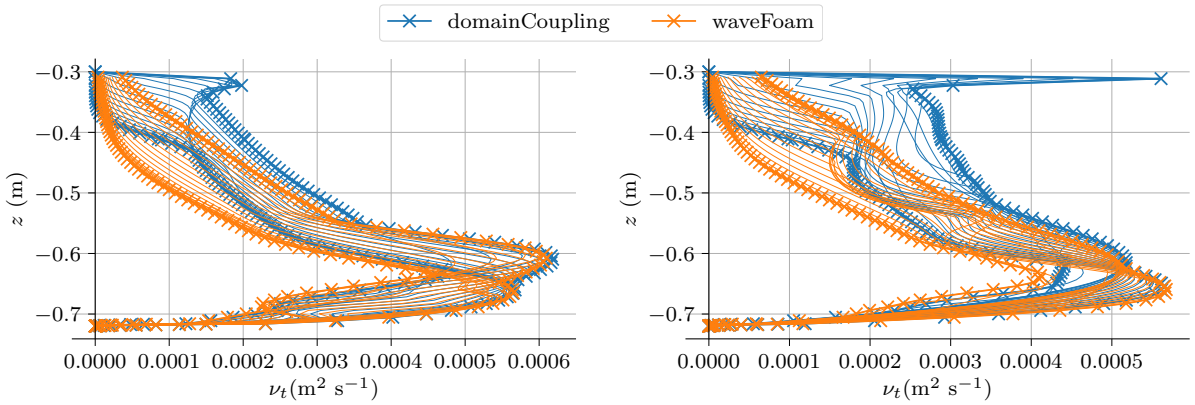
Figure 5.9: Comparaison of the turbulent viscosity field between the domain coupled case and the OpenFOAM® only case at $t/T = 15$. The horizontal black line on fig. 5.9a is the upper boundary of the CFD region, denoted Γ_t on fig. 5.2.

In fig. 5.9 side-by-side maps of the obtained ν_t field at $t/T = 15$ are shown. Results are in good agreement, especially compared to discrepancies observed within the waveFoam study, when modifications of parameters were tested. The same values are reached in terms of amplitude, and the spatial patterns are also relatively consistent. Some differences can be spotted, particularly at the bottom left and top right of the cylinder.

Also note that an instability seems to appear at the top boundary condition of the CFD domain, represented by a black line on fig. 5.9a. However, the latter does not really grow in time, but rather follows the behavior of the ν_t field in its vicinity: when large ν_t values are computed close to the boundary with nothing to reduce them, the enforcement of a null ν_t at the boundary faces is difficult, leading to the first few cells reaching high values.

However the generation of turbulent viscosity exhibits the exact same behavior as with OpenFOAM® alone (section 4.5.5). Eddy viscosity is generated close to the body and increases progressively in the nearby area reaching nonphysical values over time, even at locations where the turbulent fields should be negligible. However, it is thought not to be

a problem specific to the DD, because the exact same issue was observed with waveFoam.



(a) $\bar{t} < 0.5$ (b) $\bar{t} \geq 0.5$

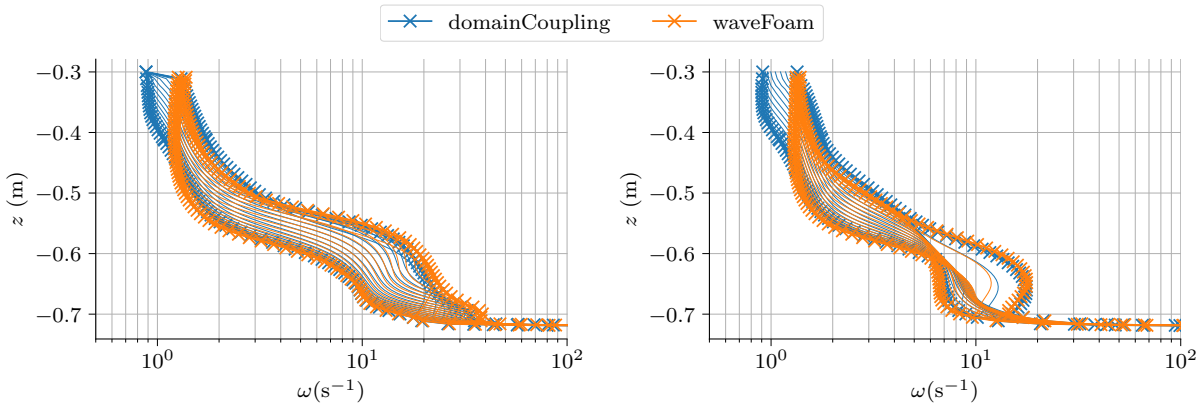
Figure 5.10: Turbulent viscosity profiles predicted by the two models, at different time steps (40 time steps per period) separated into half periods, along a vertical line ($x_l = 0\,\text{m}$ $z = -0.72\,\text{m}$ to $-0.3\,\text{m}$). Envelopes computed on these time steps are also added for comparison. $t/T \in [15, 16]$.

While from the fig. 5.9, we expect relatively large discrepancies, we present for the sake of completeness, the sampling of the eddy viscosity on the same vertical line in fig. 5.10 using the same layout as for the other variables.

Obviously, as HPC assumes an inviscid fluid, no corresponding profile is presented for this model. One could however add a zero curve at all times to represent it (this is the enforced value at the top boundary). We clearly notice on this figure the difficulty to enforce null values at the top boundary, especially at time where the ν_t field reach significant values in its vicinity. A limit of the approach is encountered: we assumed that the domain was wide enough to contain all turbulent effects, however waveFoam predicts non null eddy viscosity at the top boundary condition. Note that the time when waveFoam predicts the highest values are correlated with the "problematic" times of the domain coupling ν_t. However, do not forget that the obtained waveFoam results can be questioned in light of the underlying physics: ν_t reach 100 times the viscosity of water at that location. Furthermore, the predictions of waveFoam closer to the wall are matched with a great agreement, denoting that the eddy viscosity field manages to recover from the top boundary vicinity problem and yields expected values.

Note that the specific dissipation rate boundary condition can also play a major role in this instability: the dissipation rate is imposed as a fixed value only when the fluid flow inwards (`inletOutlet`), *i.e.* mainly during the first half periods, corresponding to the left subfigures. This behavior is noticeable on associated profiles shown on fig. 5.11. However, waveFoam predicts higher values of dissipation at this location. Together with the sharp velocity gradients (term which appears in the TKE production, see eqs. (4.17e) and (4.17f)), this can be the cause of the large rise of ν_t close to the boundary (over the 1-3 first cells).

Once again, despite the abrupt variations close to the top boundary, the domain coupling and waveFoam ν_t and ω match in a relatively close manner when approaching the body.



(a) $\bar{t} < 0.5$ (b) $\bar{t} \geq 0.5$

Figure 5.11: Turbulent specific dissipation rate profiles predicted by the two models, at different time steps (40 time steps per period) separated into half periods, along a vertical line ($x_l = 0\,\text{m}$ $z = -0.72\,\text{m}$ to $-0.3\,\text{m}$). Envelopes computed on these time steps are also added for comparison. $t/T \in [15, 16]$.

Also note that this effect, even though not impacting other variables, is seemingly resolvable by changing the top boundary condition on ω to artificially enforce null ν_t values. About this, results and discussions will be conducted in section 5.4.3.

5.4 Sensitivity studies and parameters exploration

5.4.1 Hotstart performance

In order to compare the domain coupling results with the waveFoam prediction, we previously ran a simulation starting at the exact same physical time ($t_0 = 0\,\mathrm{s}$). Note that the propagation distances are almost equal in waveFoam and in HPC (respectively 24 m and $4\lambda \approx 25.2\,\mathrm{m}$). However the current model, due to its contained domain size, is subjected to different phases:

- Waiting for the first waves to arrive. During the first few physical seconds, the wave is generated at the left HPC boundary condition (and relaxation zone) and the propagation begins. Thus, the CFD region receives almost null fields at its outer BC. The local CFD resolution is then dummy. However, if the time step were controlled by a maximum CFL number, this phase would be cheaper because the CFL would increase to counteract the almost null velocity values. This phase ($\sim t/T \in [0, 5]$) accounts for a large part of the computational cost ($\sim 32\%$ of the $t/T \in [0, 15]$ simulation), and can be regarded as completely useless.
- Accommodating and solving with non periodic/non converged values of HPC. In this part, the CFD solver receives and propagates information. This information is however not relevant and this part could also be avoided. Note also that in this phase, the involved velocities are of the order of magnitude as the final ones, thus the time step would not be small, even if the time step were controlled by a maximum CFL number. ($\sim t/T \in [5, 10]$, also 34% of total cost)
- Accommodating and solving with the periodic and converged values yielded by HPC. This is the only needed and useful part, and it is also expected to take a certain time to yield relevant and periodic results.

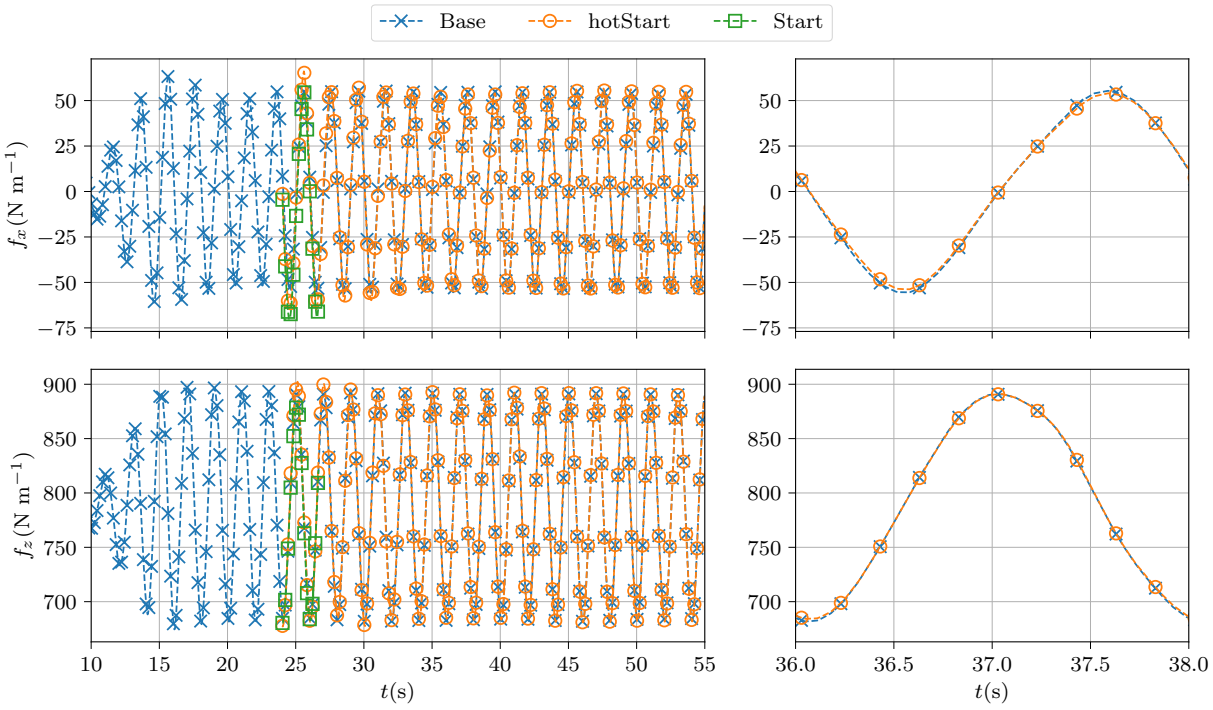
Thus, most of the computational time is spent doing calculations that will not serve, and could hopefully be avoided (in the case of a unidirectional coupling). Here, we focus on trying to quantify the time needed for the last phase, and thus on reducing as much a possible the simulated time. Two new computations, everything else being equal, are set up and run with a start time at $t_0/T = 12$. We will denote them with:

- "*start*": initializing all field values at a null value at the start time $\mathbf{f}_i^{(i)}(t_0) = \mathbf{0}$.
- "*hotstart*": initializing all field values with an interpolation from HPC at the start time, $\mathbf{f}_i^{(i)}(t_0) = \mathbf{f}_i^{(e)}(t_0)$. This method was briefly discussed in section 5.2.6.

Corresponding temporal load series are shown in fig. 5.12. Within 4-6 periods, all computations are in good agreement: maxima exhibit a relative difference of less than 3%. Note that recovering from a hotstart or from a zeroed initialized start does not seem to result in major differences. Both recover pretty easily and relatively fast, and the original case results are retrieved within a few periods.

Figure 5.13 shows the envelopes (only) of the turbulent viscosity field along a vertical line of abscissa $x_l = 0$. These envelopes are computed during different periods, namely $t/T \in [15, 16]$ for waveFoam, $t/T \in [15, 16]$ ($nT = 3$) and $t/T \in [16, 19]$ ($nT = 6$) for the domain coupled model.

Comparing the original domain coupled case with the OpenFOAM® one, the ν_t field had the same time to evolve and almost the same velocity/pressure values during its



(a) (b) Zoom

Figure 5.12: Temporal series of the loads when performing a "start" and "hotstart" a time $t_0/T = 12$. See text for details.

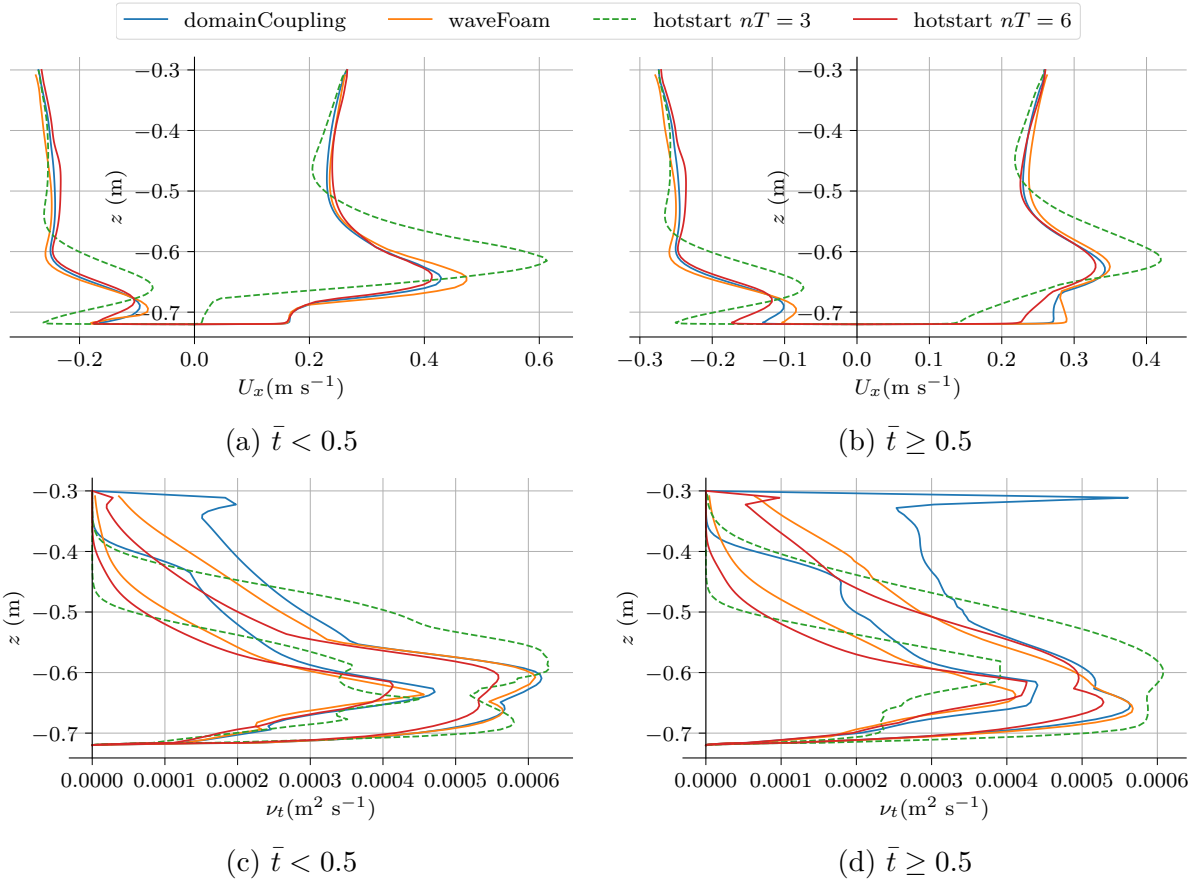
growth. Changing both the evolving time and the surrounding values during this growth will obviously lead to different fields, whether we face an unstable behavior or simply a long time convergence. This is what is observed on the figure, and we can note that after more periods ($nT = 6$) the behavior is closer from the base case which has evolved over $15T$. It will however never be perfectly equal. This does not imply that the "hotstart" approach cannot be used because it is difficult to judge which one is the most physical.

However, a significant gain in terms of computation cost is achieved. With the original start at $t_0 = 0$, the previous results were obtained by computing 16 periods (up to $t/T = 16$). Here, we managed to reduced it to 7 (up to $t/T = 19$). In practice, the first considered "periodic" full period is obtained with a CPU cost of 47% of the original one.

For this reason, all future computations will make use of this improvement and start at $t_0/T = 12$. Thus, results will now be shown at $t/T = 18$ (*i.e.* 6 simulated periods) and compared with the original waveFoam results at $t/T = 15$.

5.4.2 Sensitivity to CFD mesh

In chapter 4, it was shown that the mesh discretization has a significant effect on the obtained results. Thus, we try to keep this discretization intact in order to facilitate comparisons with the waveFoam results. However, it is still possible to evaluate the effect of a modification of the CFD mesh horizontal breadth. A mesh of contained span would be computationally faster. Hence, the main goals here are i) to validate that the current breadth yield converged results and ii) to investigate on its lower limits to evaluate how much computational power could be saved.



(a) $\bar{t} < 0.5$ (b) $\bar{t} \geq 0.5$

(c) $\bar{t} < 0.5$ (d) $\bar{t} \geq 0.5$

Figure 5.13: Horizontal velocity (figs. 5.13a and 5.13b) and turbulent kinematic viscosity (figs. 5.13c and 5.13d) envelopes, computed from different 20 time steps values per half wave period. For waveFoam and the base domainCoupling $t/T \in [15, 16]$. For the "hotstart" case - begining at $t_0/T = 12$, the used values are extracted from times $t/T \in [12 + nT, 13 + nT]$.

Let's recall that - in the same manner as with OpenFOAM® alone - the mesh is refined close to the body, more precisely over the range $x \in [-0.5, 0.5]$m. Thus reducing the horizontal domain breadth down to $B_m = 1$ m will not largely modify the problem size (from 31.1×10^3 cells to 27.3×10^3 cells for respectively $B_m = 3$ m and 1 m). Below this value however, a significant decrease in the number of cell is achieved (13.3×10^3 cells at $B_m = 0.5$ m).

The maxima of the loads are shown on fig. 5.14. We note that while some differences are exhibited, they are of small amplitudes as soon as $B_m \geq 0.75$ m, which already yields valuable results. Further reducing the mesh width (*i.e.* $B_m = 0.5$ m) leads to large discrepancies. Note that the horizontal width of the object is $L_c = 0.4$ m, thus, a 0.5 m domain width means that the extent of the CFD domain, past the body sides, is only of 0.05 m. Let's define this horizontal meshed length on each side of the cylinder $\delta l_m = (B_m - L_c)/2$. Thus relative to the incoming wavelength, the meshed part on each side is $\lambda/\delta l_m = 123$ with the most contained domain. With a mesh breadth of 0.75 m, the same non-dimenionnal parameter is 35 and reach 20 for the $B_m = 1$ m mesh. This means that meshing $1/35^{\text{th}}$ $1/20^{\text{th}}$ of the wavelength on each side of the cylinder is sufficient to

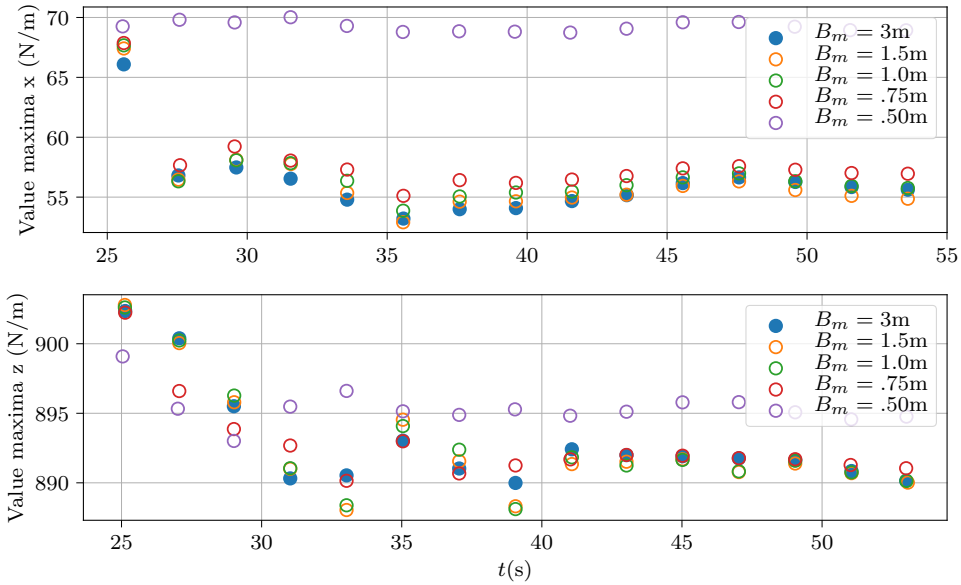


Figure 5.14: Extracted maxima of the loads obtained with different CFD mesh breadths

BC set	boundary	**u**	$p - \rho gh$	k(m^2 s^{-2})	ω (s^{-1})
1	$\Gamma_{i,t,o,b}$	cIO	cfV	fV:10^{-10}	IO: 0.88
2	$\Gamma_{i,t,o,b}$				fV: 0.88
3	$\Gamma_{i,t,o,b}$				fV: 100

Table 5.2: Table of the tested sets of boundary conditions. IO stands for `inletOutlet`, fV for `fixedValue`. A prefix "c" is added when their coupled counterparts are used. For details about these boundaries, see section 5.2.6.

capture most of the turbulent and vorticity effects in the vicinity of the body. To remain conservative and because the computation cost is not much higher, we retain the $B_m = 3$m CFD mesh for future computations (meshing a length of $1/5^{\text{th}}$ of the wavelength on each side of the body).

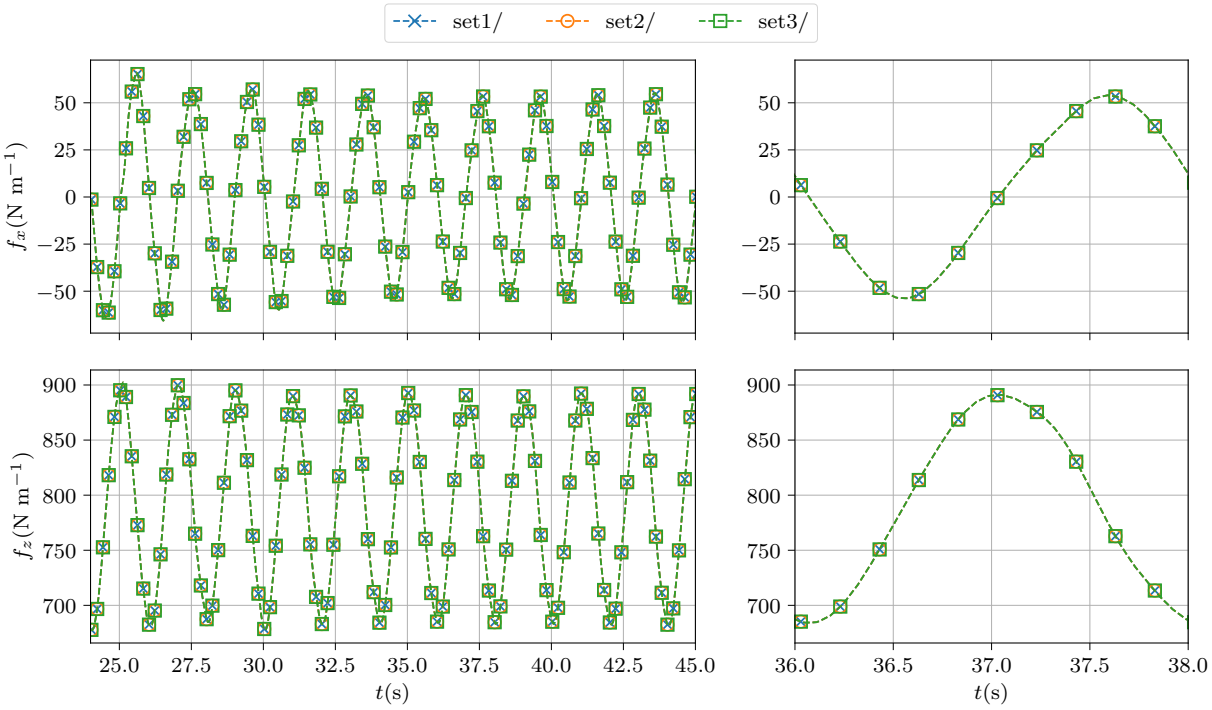
5.4.3 Boundary conditions

Variations of the boundary conditions of the specific dissipation rate of turbulence ω are tested. The main objective is to reduce the ν_t peak issue close to the top boundary (see section 5.3.2). In fig. 5.13d it was emphasized that "hotstart" simulation is also subjected to this behavior.

Note that, the original boundary value (0.88 s^{-1}) is not exactly the value that is found by waveFoam at this location, even if it is relatively close: at $\mathbf{x} = (0, 0, -0.3)$m waveFoam predicts a value of approximately 1.5 s^{-1} (which is almost constant over time), as could be denoted on fig. 5.11. However the first intuitive idea, of imposing the waveFoam predicted value was not selected. First, the dissipation rate is not constant along the boundary length and at the other boundary locations. Thus selecting the values at this particular location would be somewhat arbitrary. Moreover, the goal of the presented method is to be able to avoid a calculation with waveFoam: expensive and not perfect,

especially in terms of turbulence variables. Thus, if the stability of the presented method were dependent on a previous waveFoam run, it would lost most of its interest.

In addition, high gradients of the velocity were shown to appear in top boundary vicinity. They were noticeable when the fluid flows inwards, *i.e.* when the velocity was enforced as a Dirichlet condition. This is expected to increase the production of turbulence. Thus, to respect a correct value of ν_t higher values of dissipation rates than the one predicted by waveFoam are needed. For this reason we evaluate the effect of larger `fixedValue` with the hope to obtain a matching at the boundary (*i.e.* with HPC results) in a smoother way. However, the main objective remains to maintain the correct matching with the waveFoam prediction closer from the body.



(a) (b) Zoom

Figure 5.15: Loads series obtained with the different BC sets, see table 5.2

Loads series obtained with the three different sets, represented in fig. 5.16, exhibit no noticeable differences. The local fields are shown in fig. 5.16, at $t/T = 18$, *i.e.* after a simulated time of $nT = 6$ (all of them are "hotstart" computations with $t_0/T = 12$).

From these figures, it seems that it is numerically relevant to impose a large value of dissipation rate of turbulence at the outer boundaries of the CFD mesh. We coupled with a potential model, which is fundamentally different from the employed RANS model: for this reason, whatever the value, it is difficult to state that any turbulent boundary condition is more physically correct than another. The main assumption of the current DD approach is that turbulent, viscous, and vortical effects are negligible or vanish at the outer boundaries. One might as well impose them as such - even with numerical artifacts -, even more so if it does not modify the body vicinity descriptions, as can be noticed on fig. 5.16.

Thus, the BC set 3 is recommended, and selected. It should be applied in every

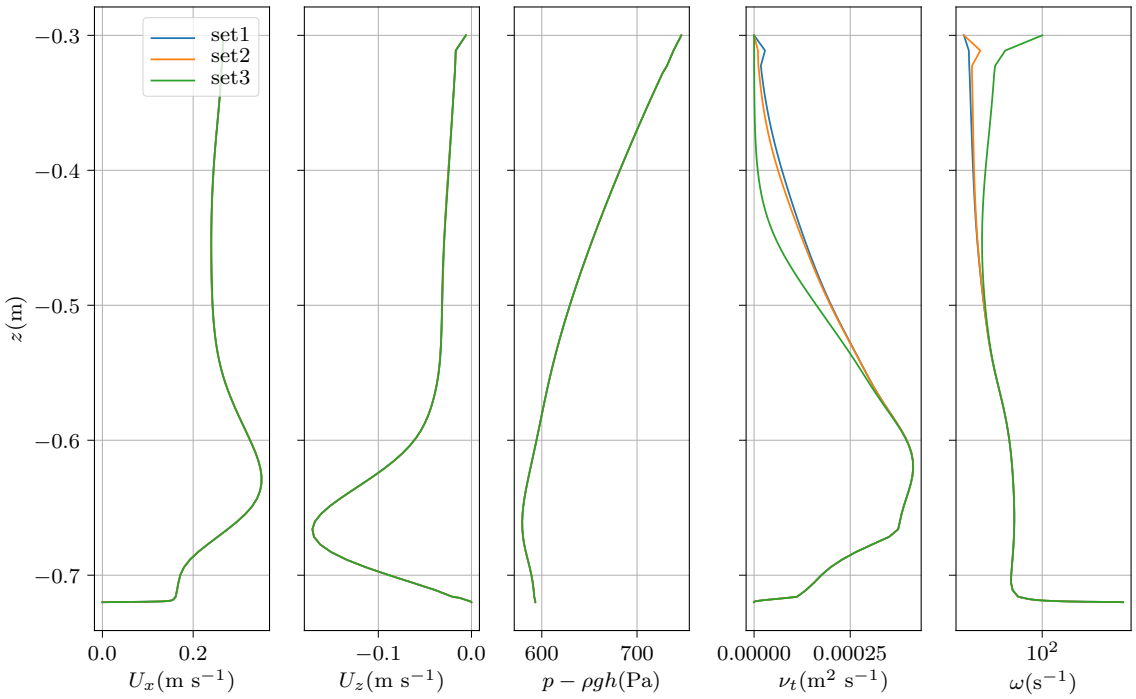


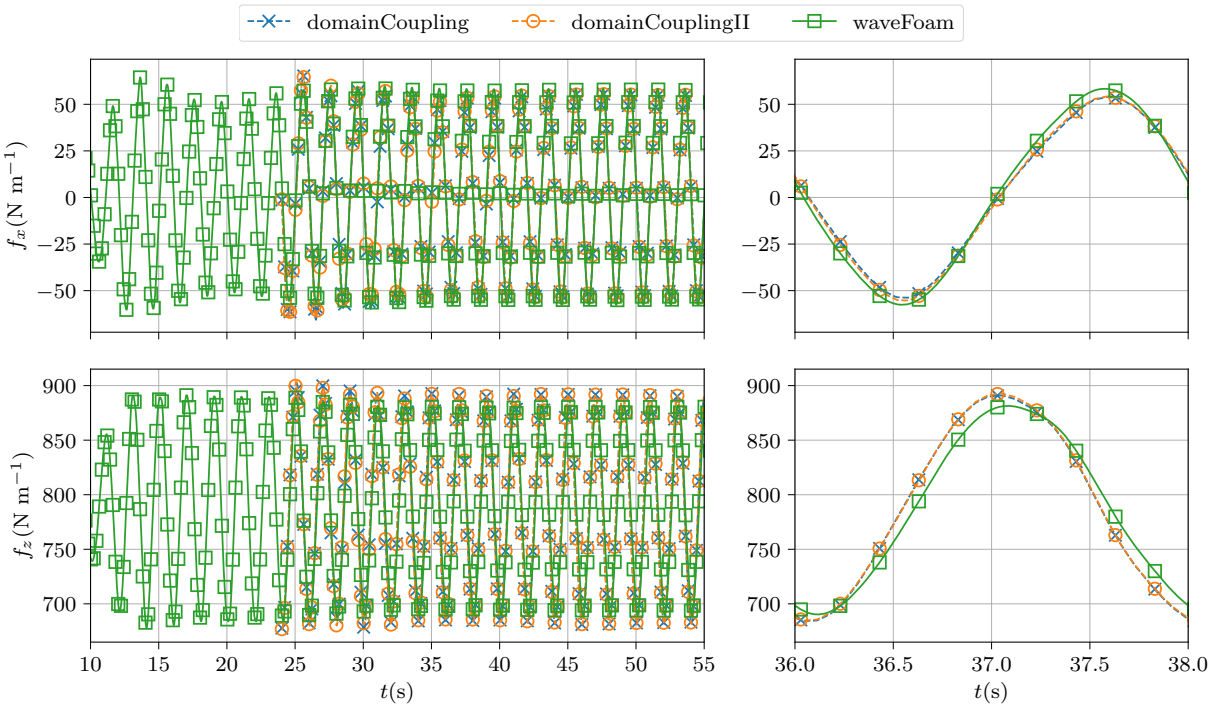
Figure 5.16: Fields profile over a vertical line located at $x_l = 0.$ at time $t/T = 18$ (6 simulated periods) obtained with different boundary condition sets, see table 5.2.

following domain coupled cases. However, a lot of computations having been carried out prior to this BC investigation, most will still use the boundary condition set 1 instead. This is not considered as an issue, because it does not fundamentally change the results of interest in the vicinity of the structure as was demonstrated here.

5.4.4 Numerical parameters, algorithm & schemes

One might notice that the employed schemes were not exactly the one selected during the waveFoam study (comparing section 5.3.1 with section 4.4.3). However, using the exact same parameters and schemes as during the waveFoam focused study (more precisely the contents of `fvSchemes` and `fvSolutions`) yields a very contained difference. The current domain coupling model using the waveFoam parameters and schemes (see section 4.4.3) is labeled "domainCouplingII" on figs. 5.17 and 5.18 depicting the obtained load series and fields profiles at $t/T = 18$ respectively. The transient results are relatively different (up to 4% in load magnitude) but after a few periods (~ 5), loads are nearly equal. The computation cost is however way lower with the waveFoam selection of parameters, mostly because only one `PIMPLE` iteration is imposed (*i.e.* usage in `PISO` mode).

A study was conducted to understand the small obtained discrepancies. The problem is that the process is nonlinear: modifying a parameter might not have the same effect, depending on the state of an other one. For this reason a lot of tests had to be performed. Most of the numerical parameters seemed to have a very limited impact. Amongst them, the various iterative solver setups, but also the number of outer corrector steps were found not to impact the results. For the latter, it means that the used time step value is small enough, such that only one `PIMPLE` iteration is required. In other words, with



(a) (b) Zoom

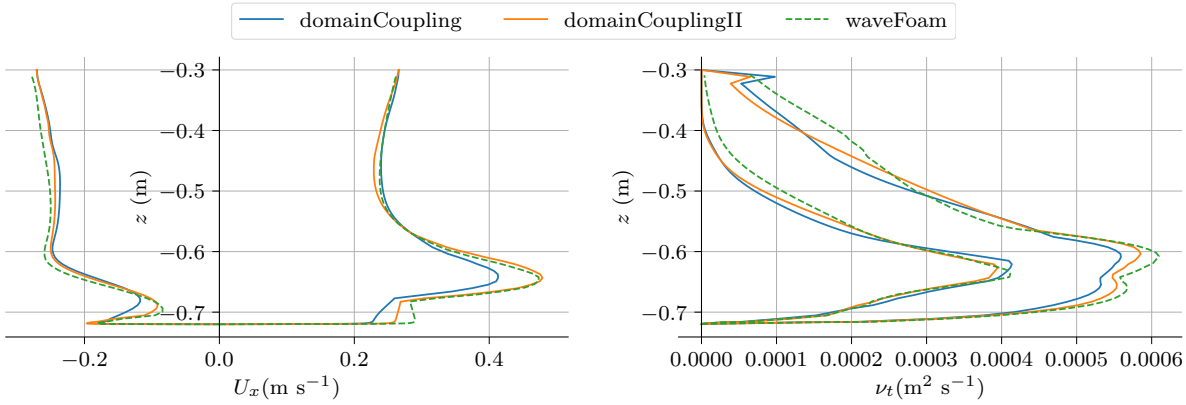
Figure 5.17: Temporal loads series with two different sets of finite volumes schemes and parameters (see text for details).

the previously applied correction (2 outer correction steps), it would be possible to use a larger time step. A specific study could be interesting, but was not conducted here.

At the end the two main sources of discrepancies could be traced back to the gradient and divergence schemes:

- In waveFoam, the gradient schemes are selected as `cellMDLimited leastSquares 1.0`, while we here used a simpler `linear` scheme.
- In waveFoam, the main divergence scheme, *i.e.* the advection of the velocity (`div(rhoPhi,U)`) is selected as `limitedLinearV 1`, while we here used a `linearUpwind grad(U)` scheme.

While it is not possible to perfectly assess the influence of each parameter and scheme, we can observe that this modification does only impact the obtained loads in the first few wave periods (see fig. 5.17). If anything, the obtained local flow fields agreement with waveFoam is improved, with the "domainCouplingII" simulation (see fig. 5.18). This could be expected, as the different parameters used for this simulation are closer to parameters used for the waveFoam computation. Even though the rest of the computations were done at an earlier date with the set of parameters presented in section 5.3.1, we do not expect a noticeable difference on the loads, but some local discrepancies can be explained by this parameter selection.



(a) (b)

Figure 5.18: Envelope over one period with two different sets of finite volume schemes and parameters (see text for details).

5.4.5 Turbulence modeling

The last study conducted on this domain coupled model concerns the influence of this new approach on the turbulence results at long time. It was shown that, with OpenFOAM® used in an independent manner (waveFoam), the turbulent fields (ν_t and k) raised up to values considered as non physical after a long time evolution (see section 4.5.5 and fig. 4.10).

In fig. 5.19 are shown the variable profiles over a horizontal line ($z_l = -0.63$ m), obtained with the domain coupled model with different variations of the $k-\omega$ SST model at different times. The corresponding models are presented in section 4.2.4 and the investigation on the obtained results with waveFoam is conducted in section 4.5. The buoyant curve obtained with waveFoam (earlier shown in fig. 4.10), is also added in fig. 5.19.

Note that, the buoyant as well as the stabilized-smoothed variation models results are in good agreement with waveFoam, on every variable. However, once again, a discrepancy in terms of pressure is noticeable, which seems to emerge from the boundary condition values (Γ_i and Γ_o at $x = \pm 1.5$ m). Phase shift and damping in the waveFoam propagation process, denoted in the previous chapter, seem to be the main reason of these differences.

During the study previously conducted on those model variations, we stated that the obtained loads were within a contained range. This observation still holds in the current context. However, we mainly focused on the ν_t profiles, showing that the stabilization suggested by Larsen and Fuhrman (2018) had an intrusive influence on this variable in the vortex shedding zone. Here, it is noticeable that this influence is not only restricted to this variable: other local fields are also affected by the stabilization. Its smoothing is sufficient to recover the prediction made by the buoyant model variation (Devolder *et al.*, 2017).

However, the domain coupling does not seem to dampen the behavior of the eddy viscosity in the relative vicinity of the body, even tough a very small value is successfully enforced at the boundary conditions. The fact that in this region neither the stabilized method nor its smoothed counterpart manage to damp the turbulent viscosity field rise indicates that the vorticity is not null. Thus, the instability pointed out by Larsen and

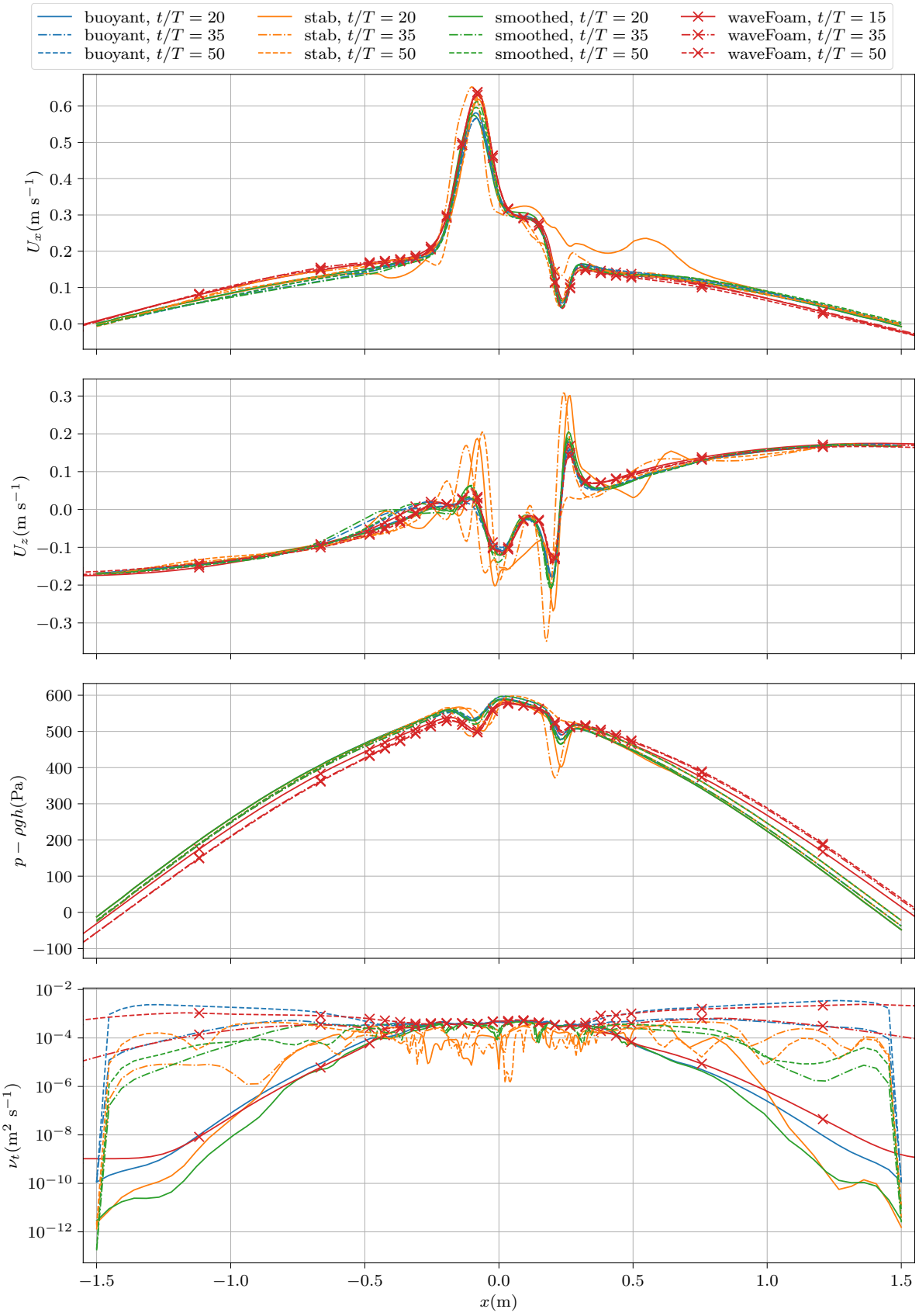


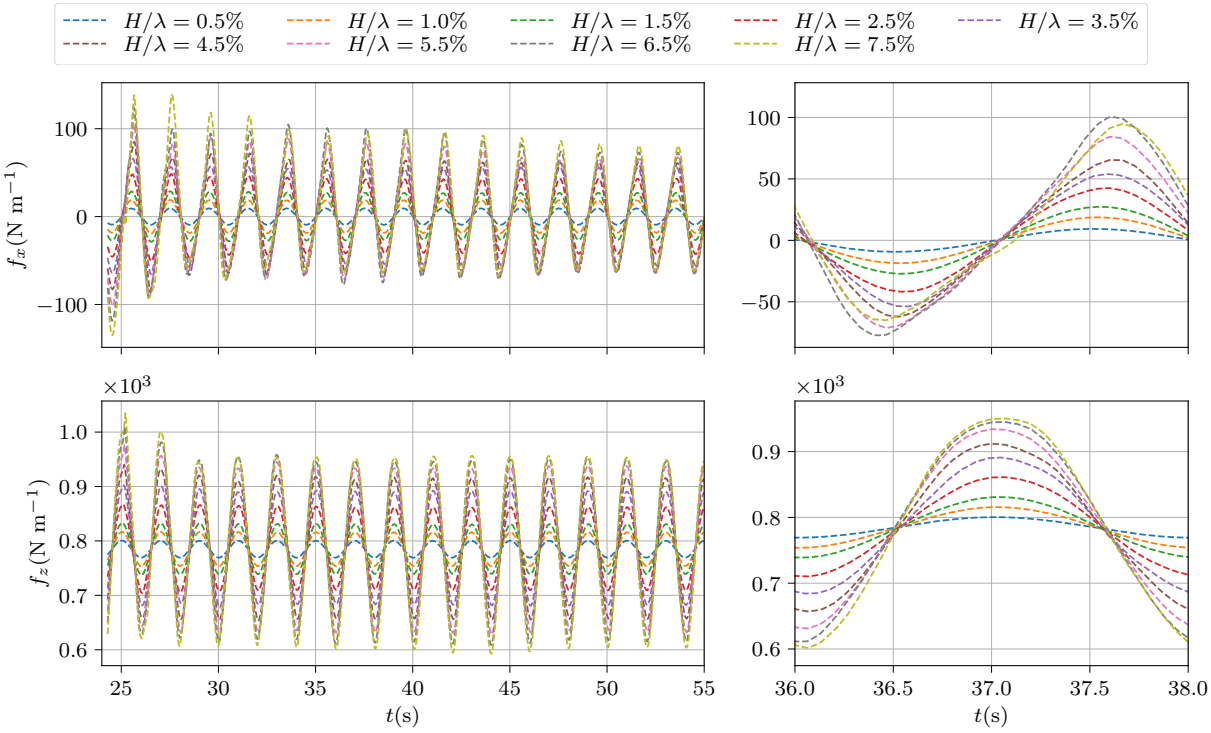
Figure 5.19: Domain coupled cases with different variations of the $k-\omega$ SST turbulence model, and waveFoam with the buoyant one. Fields sampled over a line at $z_l = -0.63$ m. $\lambda_2 = 0.05$, $f_s = 0.5$.

Fuhrman (2018) is fed without any possibility of damping.

Note that the original "set1" of boundary condition (*cf.* table 5.2) was used here, and the increase of ω suggested in "set3" could have an effect over a certain distance in damping the ν_t field.

5.5 Comparison of the hydrodynamic coefficients with results from the literature

Figure 5.20 depicts the obtained load series with different wave heights, varying the wave steepness from $H/\lambda = 0.5\%$ to 7.5%. We clearly denote that larger wave heights lead to dissipation, and a reduction over time of the load amplitudes, especially the horizontal one. The significant rise of the turbulent viscosity field is thought to be responsible of this behavior. The exact same effect was noticeable with the waveFoam solver, with an even larger impact.



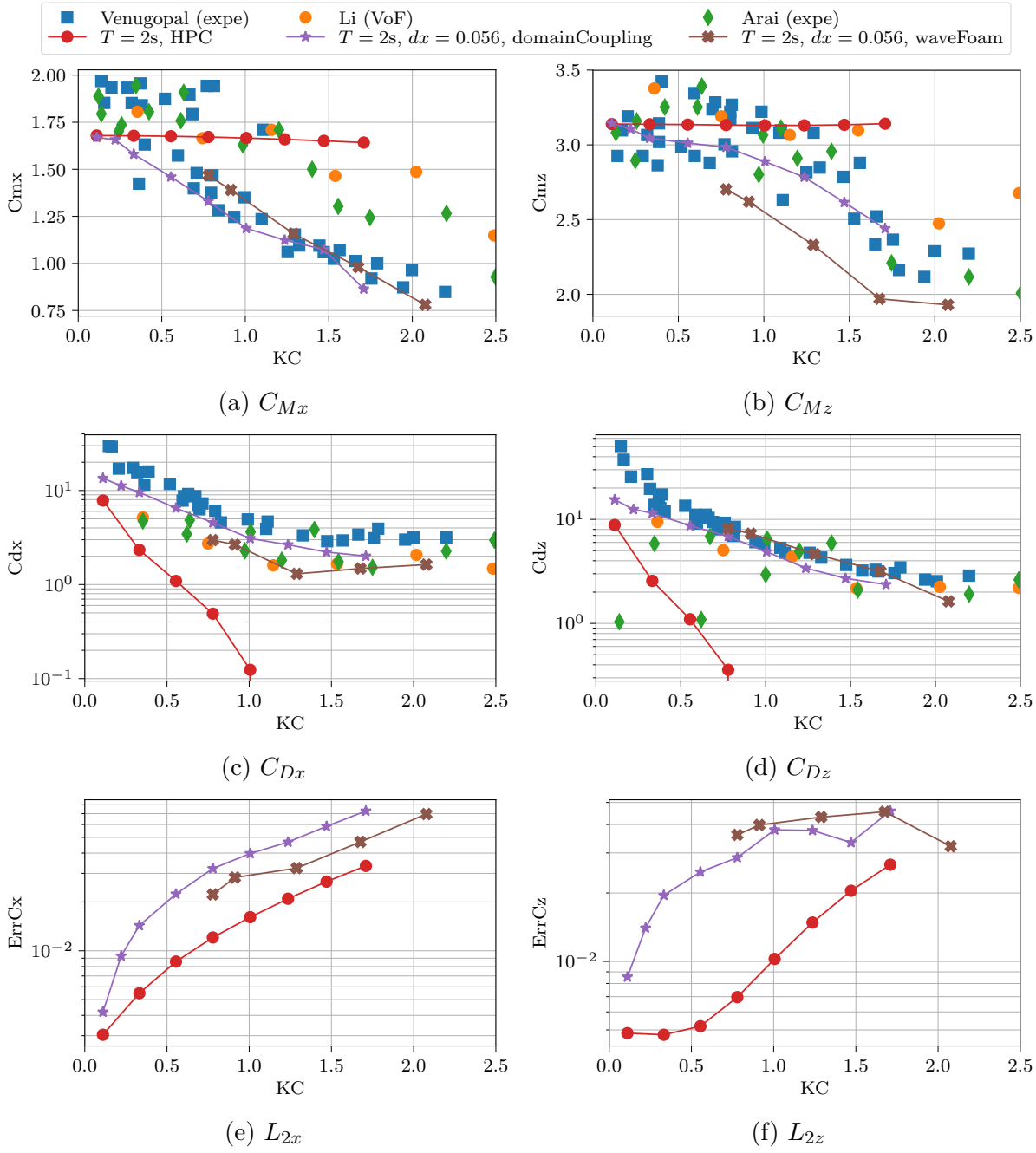
(a) (b) Zoom

Figure 5.20: Loads series for different wave heights, $T = 2$ s

Figure 5.21 depicts the hydrodynamic coefficients that are obtained with the DD method versus the literature available ones. Previously presented results, namely from the HPC method (that is also used to impose the boundary conditions of the current model) and from waveFoam alone, are also added for comparison.

First and foremost, it is noticed that the DD approach successfully captures a modification of the inertia coefficients when the wave height increases (controlled by the KC number). At low KC number, the results of the decomposition match the results of the FNLP model. When the KC number increases, the DD model leaves the HPC constant curve, even though the latter is used to impose the boundary conditions of the former. The decrease in C_m, in agreement with the literature, is encouraging. It is thought that the physical effects, inherent to the viscous and rotational model, are properly retrieved.

The drag coefficients are however different: domain coupling results do not seem to match the HPC drag at low KC number. However, let's recall that the drag predicted



(a) C_{Mx} (b) C_{Mz}

(c) C_{Dx} (d) C_{Dz}

(e) L_{2x} (f) L_{2z}

Figure 5.21: Inertia and drag coefficients in the two directions of the loads obtained with the DD method at different wave periods. The L_2 norm of the error between the Morison model and the real loads is also shown in the bottom panel.

by the potential approach only results from numerical errors (or abusive application of the Morison model). A good agreement is nonetheless found in terms of drag coefficients with our domain coupled model when compared to literature. Based on this finding, it is possible to further confirm the correct modeling of turbulent and vortical effects. In the end accurate estimates of the coefficients are found for a large range of wave heights: these KC numbers correspond to a wide range of nonlinearity parameter H/λ, namely a value of the latter from 0.5% for the most linear case (KC ≈ 0.1) up to $H/\lambda = 7.5\%$ for

the steepest case (KC ≈ 1.7).

Errors with Morison model The errors in modeling the loads with a Morison model are relatively large, up to almost 10%. At this value, it is not possible to state that the Morison formulation - and the calculated hydrodynamic coefficients - correctly represent the load series.

Morison equations first assumption is that the flow acceleration is approximately uniform in the body vicinity. This is assumption is verified when the body is of small dimension when compared to the wave length. This region corresponds to the left part of fig. 1.6,*i.e.* $piD/\lambda \leq 0.4$. In the current considered case, this non dimensional parameter is of value 0.2. Thus, those equations should be relatively valid even tough some discrepancies might be expected.

On the other hand, large KC numbers at fixed period imply large wave heights. With these wave heights, even though the cylinder is located relatively deep, significant effects on the free surface due to the presence of the cylinder may be expected (its influence, potential only, would have been taken into account in the HPC simulation in our case). The Morison model neglects the influence of the body on the flow. Moreover, Chaplin (1984) states that the Morison equations are not well suited to give a good temporal representation of orbital-flow. It is thought that these effects are the cause of the increasing error with the KC number.

5.6 Conclusion

A one-way modular domain coupling strategy was developed and implemented within the OpenFOAM® framework. It is developed with the requirement that the external results are available in the form of another OpenFOAM® case. Values are interpolated making use of locally developed routines especially designed to handle and to be optimized for large discretization differences (during the research of the cell in which a boundary face is located for example). Afterwards, enforcement of the obtained values is done trough dedicated boundary conditions at the outer perimeter of the CFD domain. This ensure a correct matching with potential results while allowing numerically some degree of freedom. A great stability degree and consistency with respect to parameter modifications is achieved.

A horizontal cylinder of rectangular cross-section at a significant submergence depth serves as a reference case, to extensively validate the model and its implementation. It is selected because the potential model - that is used as external model (chapters 2 and 3) - overlooks several aspects of the underlying physics, but also because a CFD domain, that fully encompasses the body could be defined with a relative span in all directions, while remaining entirely in the fluid region. While a coupling in free surface was developed and briefly tested, it is not shown in this work. Note that the underlying assumption is that turbulent, vortical and viscous effects should vanish when approaching the outer boundaries of the CFD domain.

It is shown that every variable exhibits the desired behavior: at the outer boundaries, they equal the imposed HPC values while matching the waveFoam values when approaching the cylinder walls. As a consequence, loads are seemingly correctly captured even if no external literature is directly available for comparison (the waveFoam results, obtained in chapter 4 were used for validation instead).

Several parameters investigations were performed to assess the capabilities of the newly implemented model. Amongst them, the possibility of starting the computation at a later time, *i.e.* using already periodic values yielded by HPC, was proven to greatly reduce the CPU cost required to reach a periodic behavior of the CFD variables. We have also shown that even a very limited mesh span ($B_m = 1\,\mathrm{m}$, *i.e.* a mesh extent of $\delta l_m/\lambda = 1/20$ past the body sides) was sufficient to yield valuable results.

Moreover, the associated hydrodynamic coefficients are in good agreement with results from the literature, for a large range of KC number. Note that this result is appreciable as the HPC inertia coefficients exhibited constant values during the variation of the KC number. The drag coefficients also land in the experimentally obtained ones, which, with HPC method used here was not possible due to its inherent potential assumptions.

However, difficulties in terms of turbulence modeling are once again encountered. In the same manner as with the waveFoam solver, the turbulent viscosity grows exponentially and perturbs our results at long time ($t/T \sim 50$ when $H/\lambda = 3.5\%$). The generation of turbulence close to the wall is thought to be responsible of the feeding of the instability pointed out by Larsen and Fuhrman (2018). The damped variation suggested by these authors were however not used, due to their intrusive behavior in the vortex shedding region. Smoothing this damping, however, consistently improves the variable field descriptions. In any case, no turbulence model variation was able to really reduce the overall rise of ν_t at long time.

Last but not least, the computational time of such DD method could drastically reduced the waveFoam requirement. On this case, waveFoam needs at least 15 periods to yield a periodic and stable behavior. With the used CPU, this requires an estimated execution time of about 24 h. On the contrary, the DD algorithm reaches a periodic behavior after 6 periods for a cost of about 3 h (40 min for each additional period). A simple model, that would estimate linearly the CPU savings ratio, would be:

$$\frac{eT_{wF}}{eT_{dC}} = \frac{\mathrm{nCell}_{wF}}{\mathrm{nCell}_{DD}} \frac{\mathrm{nT}_{wF}}{\mathrm{nT}_{DD}} \tag{5.6}$$

where eT represents the required execution time, nCell the number of cells of each approach (138×10^3 and 31×10^3 for respectively the CFD domain of the DD approach, and the waveFoam domain wF) and nT the required simulation period (respectively 6 and 15). This formula yields a cost reduction of a factor 11. However, the cost is "only" reduced by a factor 7.5. Note that during a certain time, waveFoam propagates the waves, and the problem resolution is faster (the wave did not reach the entire domain yet). Moreover, in the large used relaxation zones, a significant part is also relatively easy to solve.

When the flow is installed the associated cost could be estimated as

$$\frac{eT_{wF}}{eT_{DD}} = \frac{\mathrm{nCell}_{wF}}{\mathrm{nCell}_{DD}} \tag{5.7}$$

which corresponds to a cost reduced by a factor 4.45. In practice, this cost was reduced by a factor ~ 3. This difference might be due to a more difficult problem to solve, as we need to accomodate for large values at the boundaries. Note also that, on a coarser mesh, less refined in the body vicinity, these theoretical ratios - but also, probably, the ones obtained in practice - would be larger (higher ratio $\mathrm{nCell}_{wF}/\mathrm{nCell}_{DD}$).

Chapter 6

Restriction of the resolution to the complementary variables: a velocity decomposition approach

Contents

6.1 Introduction **156**
6.2 Theoretical formulation **159**
6.3 Numerical implementation **164**
6.4 Intrinsic differences with OpenFOAM **170**
6.5 Stability, convergence and various investigations **173**
6.6 Validation of the velocity coupling method **180**
6.7 Conclusions on the velocity decomposition scheme **186**

6.1 Introduction

6.1.1 The velocity decomposition method

The domain decomposition method presented in chapter 5 proved to be efficient in predicting the wave loads for a fraction of the computational price in comparison with the full CFD simulations. The main reason is the reduction of the physical time that needs to be simulated to achieve periodic results. Local field descriptions are obtained and show a good agreement with the experiments - even better than OpenFOAM alone - as far as the hydrodynamic coefficients are concerned.

However, while the FNLP fields are available over the entire domain, the domain coupling scheme only makes use of the values at the outer boundaries of the local grid. Yet, the potential model provides a potential field ϕ - and thus the velocity field - everywhere, and this also true for the (potential) pressure. This pressure was obtained *via* the Bernoulli equation that requires the time derivative of the potential. This variable ϕ_t was in turn obtained by the resolution of a dedicated BVP and thus a relative confidence can be granted to its accuracy. Together, the potential pressure and velocity are solution of the Euler equations. As we aim to find the velocity and pressure solution of the RANS equations, most of the objective has already been fulfilled and the resolution of the classical RANS equations can be considered as redundant for a large part.

An approach that takes advantage of the availability of the full potential solution consists in separating the velocity and pressure fields into two components, each one being solution of a different model, and deriving the respective equations so that the total velocity-pressure fields satisfy the total RANS equations. This approach is often referred to as a velocity decomposition, velocity coupling or functional decomposition (by opposition to the more classical spatial decomposition or domain coupling presented earlier).

Mathematically, two fields are defined, that are solutions of different models:

$$\mathbf{u}_t = \mathbf{u}_1 + \mathbf{u}_2 \tag{6.1}$$

where the indexes refer to the models that are employed for the hybrid coupling. The same decomposition may be performed on other fields of interest, for example the pressure, the volume fraction and even the turbulent variables. The aim is for the total fields to satisfy the most complete equations, here the RANS model. In this work, the model 1 will be the potential HPC model while the model 2 will refer to a modification of the RANS model. However, this decomposition is not unique and many approaches were employed in the literature. Some of the existing methods and developments are described hereafter.

6.1.2 Previous works

The acoustic and aerodynamic field of research was the first to consider such type of field decomposition (Morino, 1986, 1994; Morino *et al.*, 1999; Hafez *et al.*, 2006, 2007) mostly applied to solving the boundary layer problem where the viscosity plays a major role.

An actively studied and applied methodology in the last two decades was proposed by Ferrant *et al.* (2003), Gentaz *et al.* (2004), and Luquet *et al.* (2004). The velocity, pressure and surface elevation are decomposed into their incident and diffracted/radiated

components. A potential theory based wave model is used to explicitly obtain the incident field. Then a modified version of the RANS equations denoted the Spectral Wave Explicit Navier-Stokes Equations (SWENSE) are derived. Those equations require the explicit values of the incident fields (such as the potential wave elevation, potential velocities, …): given a grid at a particular time instant, the potential fields and kinematics are calculated. Afterwards, the newly derived SWENS equations are solved to yield the diffracted component. Notice that the potential-viscous coupling scheme is one-way in the sense that no feedback effect on the potential model is at play. This, however does not come with any hypothesis but instead with the drawback of having to mesh the fluid domain up to relatively far from the body of interest: the diffracted fields do not vanish with the distance to the object. Nevertheless, the main advantage of the method is to reduce the complexity of the boundary conditions for the SWENS equations, as no incident wave theory has to be imposed at the SWENSE BC. Thus, only a damping of the diffracted/radiated wave fields has to be set. The problem often encountered with RANS simulations addressing the propagation of incoming waves without significant damping is also avoided. Luquet *et al.* (2007a) described the methodology and the numerical implementation in great detail. They also introduced an irregular incident wave potential simulation, belonging to the family of HOS schemes. Over the years, numerous test cases were investigated with successful results. For example, Luquet *et al.* (2007b) focused on a tension leg platform, Alessandrini *et al.* (2008) and Monroy *et al.* (2009) applied the method for ship sea-keeping, and Li *et al.* (2017) solved the SWENS equations on the famous vertical cylinder piercing the free surface. A development of the SWENS equations with the OpenFOAM package employing a VoF method and thus a two-phase flow was recently done by Vukčević *et al.* (2016a,b) and Vukčević (2016). In a similar manner, while this method was originally developed within a single phase RANS solver, recent developments were conducted on the use of a fixed grid solver first with the use of a level-set tracking method (Reliquet *et al.*, 2013, 2019) to recently use the VoF method within a two-phase solver (Li *et al.*, 2018c,d, 2020).

Another approach was developed by Kim *et al.* (2005) in which the potential model is solved with the presence of the body. The complementary velocity is defined as $\mathbf{u}^* = \mathbf{u}_t - \mathbf{u}_p$ where $\mathbf{u}_t$ is the total velocity field that satisfies the Navier-Stokes equations and $\mathbf{u}_p$ is the potential one that satisfies the Laplace equation. The same decomposition is performed on the pressure field. Turbulent and rotational effects are thus comprised into these newly defined fields. In their implementation, a Rankine source type method is used to solve for the potential field. The studied cases mostly lie in the aerodynamic field of research.

With another approach, Edmund *et al.* (2011), Edmund (2012), and Edmund *et al.* (2013) also decompose the velocity field into a potential and rotational components. With the objective of allowing for a truncation of the NS domain, the potential is sought as a solution of the Laplace equation (without invoking the non-viscous assumption). The RANS sub-problem does not differ from the original one except at its outer boundary on which the potential velocity is imposed. Reciprocally, the potential model solves the Laplace equation with a special coupled boundary on the body surface and is thus impacted by the NS sub-problem. An iterative method is set to achieve convergence and consistence of the two sub-problems on the coupled boundaries. An important domain reduction is achieved on a steady flow over a NACA profile airfoil. This method was

extended to steady free surface flows in Rosemurgy *et al.* (2012). Recently, work has been conducted to develop the unsteady version of this approach by Chen *et al.* (2015), further extended to 3D and applied on the Wighley Hull by Chen and Maki (2017) .

Grilli *et al.* (2009) and Harris and Grilli (2010, 2012) developed a 2D one-way coupling scheme with a Large Eddy Simulation (LES) model in order to study the wave-induced transport of sediments near the sea bottom. A fully nonlinear potential wave tank is used to compute the overall wave field while perturbed NS-LES equations are solved in order to capture the fine scale viscous effects in the boundary layer zone. Later, a coupling between a FNTP model (BEM) and a NS Lattice-Boltzmann based on the same perturbation approach was developed (Janssen *et al.*, 2010; O'Reilly *et al.*, 2015).

In the context of vortex induced vibrations, Li (2017) recently adopted a velocity decomposition approach to couple a simplified NS solver (denoted Quasi-turbulent model) with another more complicated RANS model. Here, modifications are made on the equations of the second model, in a similar manner as Kim *et al.* (2005). Note that in their method, to the author understanding, only the turbulent viscosity needs iterations to match between both domains and the retroactive coupling (from model 2 to model 1) only occurs on this variable.

In a similar manner as Kim *et al.* (2005), Zhang (2018) decompose the total fields into a potential part, solved with a two-phase Euler solver, and complementary fields for which complementary RANS equations are derived. The coupling is done in a one-way manner but achieves a domain size reduction and stability by making use of transition zones (also called relaxation zones) to match the solution at the interface of the two domains.

The current work elaborates on the method introduced by Kim *et al.* (2005) in a similar fashion as the one presented in Zhang (2018). However, no transition zones are used in this work and thus the boundary conditions are applied in a direct way at the outer perimeter of the domain. As we wish to retain the domain reduction gain described in chapter 5, and keep a one-way coupling scheme in a first approach, results are not expected to be in perfect agreement with the RANS method applied in an independent manner (chapter 4). Thus, the method presented below is only applicable to cases where the viscous and turbulent effects do not perturb the far field flow in a significant way. Otherwise, stability issues should be expected at the boundary conditions with a difficulty to drive the complementary values (and the turbulent eddy viscosity) to zero. We guess that those problems led Zhang (2018) to consider transition zones at the boundaries of the complementary domain.

6.2 Theoretical formulation

Following Kim *et al.* (2005), the complementary counterpart q^* of a given variable q_p is defined so that:

$$q_t = q_p + q^* \tag{6.2}$$

Hereafter, for the sake of clarity, we index a total variable with a t. $\mathbf{u}_t, p_t$ are thus respectively the total velocity and pressure variables that are sought as solutions of the original Navier-Stokes equations. Note that q_p can be obtained from any solver, and in the following, the only requirement is for $(\mathbf{u}_p, p_p)$ to be solution of the Euler equations. Applied to the velocity $\mathbf{u}_t$, this decomposition with a velocity deriving from a potential, is the Helmholtz decomposition ($\mathbf{u}_t = \nabla \times \mathbf{a} + \nabla\psi$ where $\mathbf{a}$ is a vector field and ψ a scalar field). Thus $\mathbf{u}^*$ contains the rotational part of the total velocity. As stated earlier, the Helmholtz decomposition is not unique and we focus on solving for $\mathbf{u}^*$, p^* that complement our previously obtained potential velocity and pressure$(\mathbf{u}_p, p_p)$.

6.2.1 Complementary Navier-Stokes equations - Momentum and mass conservation

In order to derive the Navier-Stokes equations on the complementary velocity and pressure, it is noticed that the right hand side of the momentum eq. (4.3b) is linear in $\mathbf{u}$. The shear tensor is dependent on the used velocity. For a given velocity $\mathbf{u}$, this tensor is denoted $\mathbb{T}(\mathbf{u})$:

$$\mathbb{T}(\mathbf{u}) = \mu\nabla\mathbf{u} + \mu\nabla\mathbf{u}^T - 2/3\mu(\nabla\cdot\mathbf{u})\mathbb{1} \tag{6.3}$$

where $\mathbb{1}$ is the identity matrix. Thus, inserting the decomposition of $\mathbf{u}_t, p_t$ into the Navier-Stokes eqs. (4.3a) and (4.3b) and the advection eq. (4.16) for the volume fraction, yields:

$$\begin{cases} \nabla\cdot\mathbf{u}_p + \nabla\cdot\mathbf{u}^* = 0 & \text{(6.4a)} \\ \dfrac{\partial\rho\mathbf{u}_p}{\partial t} + \dfrac{\partial\rho\mathbf{u}^*}{\partial t} + \nabla\cdot\rho\mathbf{u}_p\otimes\mathbf{u}_p + \nabla\cdot\rho\mathbf{u}^*\otimes\mathbf{u}^* \\ \qquad + \nabla\cdot\rho\mathbf{u}_p\otimes\mathbf{u}^* + \nabla\cdot\rho\mathbf{u}^*\otimes\mathbf{u}_p \\ \qquad = -\nabla p^* - \nabla p_p - \nabla\cdot\mathbb{T}(\mathbf{u}_p) - \nabla\cdot\mathbb{T}(\mathbf{u}^*) + \mathbf{f} & \text{(6.4b)} \\ \dfrac{\partial\alpha}{\partial t} + (\mathbf{u}_p + \mathbf{u}^*)\nabla\alpha = 0 & \text{(6.4c)} \end{cases}$$

where $\mathbf{f}$ is the volume force field. The potential variables are chosen to be null above the free surface. Thus, the divergence of the potential velocity is zero whatever the region. Then, everywhere, the potential part of the continuity equation reduces to zero. This leads to the classical continuity equation on $\mathbf{u}^*$. The momentum equation greatly simplifies when stating that the potential velocity $\mathbf{u}_p$ derives from a scalar field solution of the Laplace equation (also true in the air as $\mathbf{u}_p = 0$). Together with the fact that the pressure is computed *a posteriori* from the Bernoulli equation, it is possible to state that the Euler momentum equation is verified, *i.e.*

$$\begin{cases} \nabla\cdot\mathbf{u}_p = 0 & \text{(6.5a)} \\ \dfrac{\partial\rho\mathbf{u}_p}{\partial t} + \nabla\cdot\rho\mathbf{u}_p\otimes\mathbf{u}_p = -\nabla p_p + \mathbf{f} & \text{(6.5b)} \end{cases}$$

The volume force reduces to $\mathbf{f} = \rho\mathbf{g}$ in our case. Thus, this source force terms will not appear anymore in the equations with the complementary variables, as it is already taken into account by the potential solution.

Notice that if the complementary RANS solver were used to describe the free surface, in a one-way coupling approach, the RANS free surface would have no reason to match the potential one. Thus, different mass density fields would exist, and the Euler eqs. (6.5a) and (6.5b) applied with the NS density would not be satisfied as is. Instead they would be verified using the potential mass density. As a consequence, the corresponding terms should be kept in the complementary equations and be explicitly discretized as done by Zhang (2018). It is however thought that, in the case of a free surface coupling, the underlying assumption that the body does not perturb the far-field in a significant way would reach its limit. In that case, developing a two-way coupling on which the free surfaces are matching at any time could be a more appropriate solution.

Remark. *An interpolation of the potential pressure from the potential grid to the CFD grid is needed. In order to minimize the interpolation error, the dynamic potential pressure is first computed on the potential grid by subtracting $\rho\mathbf{g}\cdot\mathbf{x}$ at any node of the potential grid. Only then, the interpolation of this dynamic pressure onto the CFD grid is performed. Afterwards the static pressure will be re-added when needed directly on the CFD grid points. This prevents from having to interpolate the vertical location of a point and thus allows to reduce the interpolation error. For further details on the interpolation methods - both in time and space -, the reader is referred to sections 5.2.4 and 5.2.5.*

The divergence of the strain rate tensor applied on the potential velocity field is null due to both the divergence free property of the $\mathbf{u}_p$ and the fact that μ is constant, such that $\nabla \cdot (\mu\nabla\mathbf{u}_p) = \mu\nabla^2\mathbf{u}_p = 0$. Note that, if μ were not constant, the latter equality would not be true. This will happen as we will invoke the Boussinesq assumption when applying the RANS method on the complementary NS equations.

Thus, the main system of equations on the complementary variables - also called the complementary NS equations for simplicity - can be expressed as:

$$\begin{cases} \nabla \cdot \mathbf{u}^* = 0 & \text{(6.6a)} \\ \dfrac{\partial \rho\mathbf{u}^*}{\partial t} + \nabla \cdot \rho\mathbf{u}^* \otimes \mathbf{u}^* + \boxed{\nabla \cdot \rho\mathbf{u}_p \otimes \mathbf{u}^*} + \boxed{\nabla \cdot \rho\mathbf{u}^* \otimes \mathbf{u}_p} = -\nabla p^* - \nabla \cdot \mathbb{T}(\mathbf{u}^*) & \text{(6.6b)} \\ \dfrac{\partial \alpha}{\partial t} + \mathbf{u}^*\nabla\alpha + \boxed{\mathbf{u}_p\nabla\alpha} = 0 & \text{(6.6c)} \end{cases}$$

together with the appropriate BC that will be described later in section 6.2.3. This system of equations presents similarities with the original NS eqs. (4.3a) and (4.3b) applied on the complementary variables. However, some differences - that are boxed in the above equations - appear on the latter. In the momentum equation new convection terms appear, namely the convection of the potential velocity by the complementary velocity and reciprocally the convection of the complementary velocity by the potential velocity. Convection of the volume fraction by the potential velocity is also noticeable on the α evolution equation. Convection terms of α will however be reunited such that the convection will be done by the total velocity $\mathbf{u}_t$: no modification are thus needed on the volume fraction evolution scheme.

6.2.2 Complementary RANS momentum equation

The procedure, presented in section 4.2.2 is repeated, applied this time to the complementary NS equations. The complementary velocity is decomposed into a time-averaged and fluctuation components, just as $\mathbf{u}_t$. Note that stating that the fluctuating part of the velocity is fully contained in the complementary velocity yields:

$$\begin{aligned} \mathbf{u}_t = \bar{\mathbf{u}} + \mathbf{u}' &= \overline{\mathbf{u}_p + \mathbf{u}^*} + (\mathbf{u}_p + \mathbf{u}^*)' \\ &= \mathbf{u}_p + \overline{\mathbf{u}^*} + \mathbf{u}^{*\prime} \end{aligned} \tag{6.7}$$

Thus, the fluctuating parts of the complementary and total velocities are equal $\mathbf{u}^{*\prime} = \mathbf{u}'$. Consequently, the Reynolds-Averaging method can be applied directly to the momentum eq. (6.6b) to yield:

$$\frac{\partial \rho \overline{\mathbf{u}^*}}{\partial t} + \nabla \cdot (\rho \overline{\mathbf{u}^*} \otimes \overline{\mathbf{u}^*} + \rho \overline{\mathbf{u}' \otimes \mathbf{u}'}) + \boxed{\nabla \cdot \rho \mathbf{u}_p \otimes \overline{\mathbf{u}^*}} + \boxed{\nabla \cdot \rho \overline{\mathbf{u}^*} \otimes \mathbf{u}_p} = -\nabla \bar{p} - \nabla \cdot \mathbb{T}(\bar{\mathbf{u}}^*) \tag{6.8}$$

The Boussinesq assumption is applied once again using eq. (4.8) as the Reynolds Stress tensor did not change. Note that this equation involves the total velocity $\bar{\mathbf{u}}_t$. Thus, one may apply once again the decomposition $\bar{\mathbf{u}}_t = \overline{\mathbf{u}_p} + \overline{\mathbf{u}^*}$.

$$\begin{aligned} \nabla \cdot (\rho \overline{\mathbf{u}' \otimes \mathbf{u}'}) = \nabla \cdot \mathbb{T}_t(\mathbf{u}) = &\nabla \cdot \mu_t([\nabla \bar{\mathbf{u}}^* + \nabla \bar{\mathbf{u}}^{*T}] - 2/3(\nabla \cdot \bar{\mathbf{u}}^*)\mathbb{1}) - 2/3\rho k \mathbb{1} \\ &+ \nabla \cdot \mu_t([\nabla \bar{\mathbf{u}}_p + \nabla \bar{\mathbf{u}}_p^T] - 2/3(\nabla \cdot \bar{\mathbf{u}}_p)\mathbb{1}) \end{aligned} \tag{6.9}$$

However, in that case - as μ_t is not constant - it is not possible to state that the divergence of the turbulent strain rate tensor reduces to zero when applied to the potential velocity. Thus, the RANS equations on the complementary variables - also called the complementary RANS equations - are expressed as:

$$\begin{cases} \nabla \cdot \bar{\mathbf{u}}^* = 0 & \text{(6.10a)} \\ \dfrac{\partial \rho \bar{\mathbf{u}}^*}{\partial t} + \nabla \cdot \rho \bar{\mathbf{u}}^* \otimes \bar{\mathbf{u}}^* + \boxed{\nabla \cdot \rho \mathbf{u}_p \otimes \bar{\mathbf{u}}^*} + \boxed{\nabla \cdot \rho \bar{\mathbf{u}}^* \otimes \mathbf{u}_p} & \\ \qquad = -\nabla p^* - \nabla \cdot \mathbb{T}_{eff}(\overline{\mathbf{u}^*}) - \boxed{\nabla \cdot \mathbb{T}_t(\mathbf{u}_p)} & \text{(6.10b)} \end{cases}$$

where $\mathbb{T}_{eff}$ is the sum of the shear stress tensor and the complementary part of the Boussinesq assumption of eq. (6.9). Its expression as a function of $\bar{\mathbf{u}}$ is given in eq. (4.9b). This system of equations is the one that will be implemented and solved.

Remark. *A given turbulence model which job is to compute the effective dynamic viscosity $\mu_{eff} = \mu + \mu_t$ does need the total velocity $\overline{\mathbf{u}_t}$ to do so. An error would be to feed it only with the complementary velocity that we solve for, because it is not the sole generator and convector of turbulence. Another important note is that trying to solve two turbulence models - one for the potential terms and one for the complementary terms then sum the two additional shear stress obtained - would be possible but risky as one needs to ensure the linearity of the turbulence model with respect to the velocity.*

Given the previous equations, the form of the semi-discretized eq. (4.35) remains intact. The difference are of course on how to compute diagonal and off-diagonal matrices of the momentum equation. More precisely, the different tensors are computed according to eqs. (6.10a) and (6.10b) in place of eqs. (4.9a) and (4.9b). However given an equation, OpenFOAM® is able to compute those terms. This implies that the pressure equation file does not need to be modified. Thus, the methodology described in section 4.3.2 is directly applied, and will not be repeated here.

6.2.3 Boundary conditions

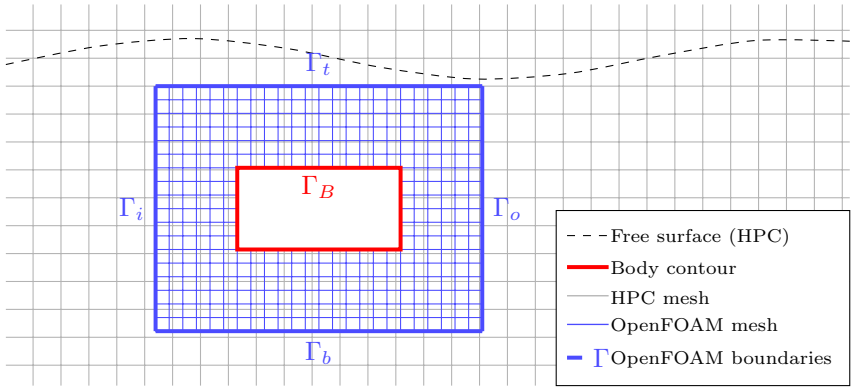


Figure 6.1: Schematic representation of the coupling method

In order to close the velocity-pressure problem, a set of boundary conditions has to be defined. While a domain coupling model has to enforce its outer boundaries according to the external solver values, it is not the case in the velocity coupling framework. Deriving a Dirichlet condition - imposing a total value $\mathbf{u}_D$ - in this framework yields:

$$\mathbf{u}^* = \mathbf{u}_D - \mathbf{u}_p \tag{6.11}$$

In particular, applied on a no-slip condition (sea bottom or body BC), we get:

$$\mathbf{u}^* = -\mathbf{u}_p \quad \text{at} \quad \Gamma_B \tag{6.12}$$

At the outer boundary, the same Dirichlet condition can be applied. It is assumed that the effect described by the complementary equations are restricted to the body vicinity and thus, the condition is reduced to:

$$\mathbf{u}^* = 0 \tag{6.13}$$

A Neumann condition on which the imposed spatial derivative of the total velocity is imposed as $\left.\frac{\partial \mathbf{u}}{\partial n}\right|_N$ would reduce to:

$$\frac{\partial \mathbf{u}^*}{\partial n} = \left.\frac{\partial \mathbf{u}}{\partial n}\right|_N - \frac{\partial \mathbf{u}_p}{\partial n} \tag{6.14}$$

Because of the one-way properties of the scheme, stability problems are expected when imposing eq. (6.13) at the outer boundaries of the RANS domain: the feedback of the turbulent and viscous part to the far-field flow cannot be exactly zero. Thus, in order to counteract this, mixed Neumann-Dirichlet conditions will be used, in the same manner as in chapter 5. Details on the numerical choices are given in section 6.3.5.

6.3 Numerical implementation

A focus is made on the description of the numerical implementation of the velocity coupling scheme developed in the OpenFOAM® framework. This section is devoted to underlining the differences between the original solver (waveFoam available with the waves2Foam toolbox) and our newly developed velocity decomposition solver (denoted velocityCoupling). For the sake of modularity, the potential results are still imported onto the potential mesh and then interpolated onto the CFD mesh. Thus, we assume that the potential fields (p_p, $\mathbf{u}_p$) are defined and available at any time instant on the CFD mesh. For further detail, the reader is referred to section 5.2.

6.3.1 PIMPLE loop

The PIMPLE loop, briefly described along with its purpose in section 4.3.2.2, is shown in alg. 6.1. It encapsulates the solveFluid.H method that defines the PISO loop. A PIMPLE iteration is also denoted an outer correction step. All developments were made in a multi-region paradigm in order to allow for multiple disjoint CFD zones. For example, the heave plates that are added at the base of each column of some semi-submersible platforms would each receive one CFD domain in their vicinity. This multi CFD zone possibility was however not tested in the present study. Except for this multi-region development - and the associated forAll loop - there are no difference compared to the original scheme.

Algorithm 6.1: Main solver

```
1
2     // PIMPLE LOOP
3     for (int oCorr=0; oCorr<nOuterCorr; ++oCorr)
4     {
5         const bool finalIterCorr = (oCorr == nOuterCorr-1) ;
6
7         forAll(fluidRegions, i)
8         {
9             l_pimple[i].loop();
10            #include "solveFluid.H"
11        }
12
13
14        #include "isFinalIter.H"
15        #include "correctTurbulenceIfFinalIter.H"
16
17
18
19        if(finalIter)
20        {
21            Info << "finalIter reached, end pimple outer correction
                  after "<< oCorr +1 << " iterations"<< endl;
22            break;
23        }
```

```
24
25      }
```

Remark. *Every external region is available and defined in memory. If one would want to perform time based computations it would be called in this file as a* `forAll( externalRegions,i)` *loop. This would, for instance, be the case for a strong coupling algorithm.*

6.3.2 `PISO` loop

The detail of the file `solveFluid.H` is shown in alg. 6.2. Notice that, for the sake of brevity, the part where the variables are defined as references to the storage are omitted (*e.g.* `volVectorField & Us = l_Us[i]` where `i` represents the index of the fluid region of interest). The following notations are used in the code: starred (*, *i.e.* complementary) variables are suffixed with an "`s`", total ones with a "`t`" and external ones (potential in our case) with a "`p`". Some of the original routines were not modified and need the total variables without any suffix to be defined. In this case, we define locally (*i.e.* sporadically during the time run) those variables by reference (*i.e.* pointing) to the total variables. This approach of changing the name of even the main variables greatly helps to verify that the velocity and pressure are not used and called without our awareness.

The first part of the algorithm (line 7-9) concerns the update of the volume fraction α using the previous time step values of the velocity and pressure. This part was kept for the original solver and the volume fraction is solved and allowed to evolve, even if no coupling in terms of free surface will be shown in this document. In practice all boundary conditions for α are set to a `fixedValue` of 1. Those include files require, as "input", the total velocity and pressure which thus need to be defined.

Remark. *It is also possible to modify the associated files and only solve for the complementary volume fraction α^* such that the total volume fraction (after summing the potential volume fraction) satisfies the corresponding eq.* (6.6)*. This would however require its own study and is outside the scope of this work.*

Note that the mass flux defined on the cell faces (denoted `rhoPhi`) is updated during the update of the volume fraction resolution. Thus, the complementary mass flux `rhoPhis` is updated by subtracting the potential mass flux (`line 12`).

Algorithm 6.2: solveFluid.H

```
1
2  const surfaceScalarField rhoPhip("rhoPhip",
       fvc::interpolate(rho)* l_phip[i]);
3
4
5  /* ----  solve for alpha.water            ---- */
6
7  #include "alphaControls.H"
8
9  #include "alphaEqnSubCycle.H"
```

```
10
11 // rho rhoPhi total was modified with the alpha computation, thus
      update rhoPhis
12 l_rhoPhis[i] = rhoPhi - rhoPhip;
13
14 l_mixture[i].correct();
15 if (l_pimple[i].frozenFlow())
16 {
17    continue;
18 }
19
20
21
22 /* ---- Define the Us equation, solve if momentumPredictor ---- */
23 #include "UsEqn.H"
24
25
26 /* ---- Piso loop, pressure-velocity coupling algorithm ---- */
27 label it=0;
28 while (l_pimple[i].correct()) //PISO LOOP
29 {
30    it++;
31    Info << "PISO LOOP, iteration" << it << endl;;
32    #include "pEqn.H"
33 }
```

6.3.3 Momentum equation

The momentum equation that rules the evolution of the complementary velocity $\mathbf{u}^*$ is defined in the file `UsEqn.H`. Most changes in that file are inside the equation definition itself. This file is shown in alg. 6.3, with comments meant to emphasize the newly added terms corresponding to the boxed terms in eq. (6.10b).

Algorithm 6.3: UsEqn.H

```
1     fvVectorMatrix UsEqn
2     (
3        fvm::ddt(rho, Us) +
4        fvm::div(rhoPhis, Us)
5      + fvm::div(rhoPhip, Us) // - New - convection of Us by the
          potential mass flux.
6      + fvc::div(rhoPhis, Up) // - New - convection of the
          potential velocity by Us.
7      + MRF.DDt(rho, Us)
8      + turbulence->divDevRhoReff(rho, Us)
9     /* - New - */
10      /* adding the part of divDevRhoReff which is not taken into
           account when using Us previous line */
```

```
11      - fvc::div(rho *turbulence->nut() *dev2(T(fvc::grad(Up))))
12      - fvc::laplacian(rho *turbulence.nut(), Up)
13      /* - end New - */
14       ==
15         fvOptions(rho, Us)
16      );
17
18
19      UsEqn.relax();
20
21      fvOptions.constrain(UsEqn);
22
23      if (pimple.momentumPredictor())
24      {
25          solve
26          (
27              UsEqn
28           ==
29              fvc::reconstruct
30              (
31                  (
32                      mixture.surfaceTensionForce()
33                    - ghf*fvc::snGrad(rho)
34                    - fvc::snGrad(p_rghs)
35                  ) * mesh.magSf()
36              )
37          );
38
39          fvOptions.correct(Us);
40          Ut=Us+Up;
41
42      }
```

If the momentum predictor is activated, the resolution is performed using the previous values of the pressure. In that case, the complementary velocity is updated. We thus update the total velocity (line 40).

The file `pEqn.H` is not shown here as it mainly remained intact from the original one apart for few minor changes of variable definitions and references. The original is however, shown and described in appendix B.

6.3.4 Turbulence object and equations

In alg. 6.3 a new turbulence term is added, corresponding to the last term in the RHS of eq. (6.10b). If one were using the base method `turbulence->divDevRhoReff` directly to the total velocity field, the computation would require the total viscosity instead of solely the turbulent viscosity. The last terms (lines 11-12) exactly correspond to the source code of `turbulence->divDevRhoReff` using ν_t instead of ν_{eff}. On the other hand, to ensure that the `turbulence` object correctly uses the total velocity field when its update

is required, one needs to take great care of its initialization. This initialization is shown in alg. 6.4 and performed once, only at the creation of the object. Of course, the turbulence object keeps a dynamic link (reference) to the given face flux and cell velocity fields.

Algorithm 6.4: Creation of the turbulence object

```
1  incompressible::turbulenceModel::New
2  (
3      l_U[i],
4      l_phi[i],
5      l_mixture[i]
6  ).ptr()
```

6.3.5 Boundary conditions

Outer BC of the local grid Most of the needed boundary conditions can be enforced by already existing code in OpenFOAM®. For example, the complementary pressure at the outer boundary is enforced as `fixedValue` with an imposed value of 0. The same is true for the velocities conditions: if one wants to impose the conditions as a Dirichlet condition, the `fixedValue` of **0** should be selected. However, we will show later (section 6.5.4) that this conditions is too restrictive and a mixed Neumann-Dirichlet (`inletOutlet`) leads to more stable and consistent results. With the latter, a **0** imposed value for both the gradient when the fluid flows outwards and the value itself when the opposite is true are fully adequate. Needless to say, those null values are only valid when the CFD mesh covers the entire area where the viscous and turbulence have significant effects.

Remark. *A zeroed* `inletOutlet` *in the velocity coupling framework is not strictly equivalent to a* `coupledVelocityInletOutlet` *developed in the domain coupling framework section 5.2.6 when the fluid flows outwards: the latter enforces a null gradient for the* complementary *part of the velocity, while the former does the same thing on the* total *velocity. Thus, the complementary* `inletOutlet` *would be equivalent to enforcing the gradient of the* potential *velocity in* `coupledVelocityInletOutet` *for an outward flow (instead of a null value). In the end, the usage of* `inletOutlet` **0** *for the complementary velocity is entirely justified and does not invoke any approximation.*

Also note that the `inletOutlet` implementation requires the user to specify the flux that is used to detect the flow direction. By default, a field of name `phi` is sought, which is however not defined anymore. Three face flux are available in our computation: the total face flux `phit`, the potential face flux `phip`, and the complementary face flux `phis`. Let's recall that we aim to impose null values of the complementary velocity at the outer boundaries, and thus expect `phis` to be almost null there. For this reason, it is expected that using `phit` or `phip` to determine the flow direction, would yield similar results (which moreover would mimic the domain coupling set up). However using the complementary face flux is conceivable: the complementary velocity would be more often imposed as null at the condition and would be allowed to flow outside only at time where the solver would have trouble otherwise. Tests will be conducted on these possible variations and results and recommendations will be given in section 6.5.4.

Body BC A particular no-slip condition must be implemented to ensure the enforcement of eq. (6.12). It was done during this work, but is not shown in this document for the sake of relative conciseness. The procedure interpolates the potential velocity at the boundary faces and imposes its negative counterpart as a fixed value. The name and keywords allocated to this boundary type is `minusCoupledVelocity` in the sense that it is exactly the `coupledVelocity` developed for the domain coupling scheme (*c.f.* section 5.2.6) only differing from a minus sign. Note that, as this condition inherits from the `fixedValue` boundary condition, the keyword `value` needs to be defined even tough it will not be used and immediately replaced by the adequate values from the potential results.

6.3.6 Various other contributions and implementations

A lot of other modifications were made in the solver. They are not described here for brevity reasons because they are mainly numerical and not of prime importance. For example, they relate to the fields generation, reading and instantiating but also the creation of list objects to simulate multi-region cases. Moreover at this stage an important part was already implemented and is reused, namely the interfacing and interpolations from regions to regions (potential to CFD in our particular case).

Note that while the domain coupling approach described in chapter 5 can be interfaced with any other model (linear potential, stream function theory, laminar CFD), as the total equations are solved, the current velocity coupling looses some generality as the external velocity and pressure are required to be solution of the Euler equations. However, the same analysis could be made with a laminar external flow with just few modifications of the above presented algorithms. Also note that this shortcoming could be avoided by discretizing the Euler equations in a similar manner as Zhang (2018). With this method, the residual of the potential Euler equations would serve as a source term in the complementary equations. In other words, instead of mathematically simplifying the adequate terms, we let OpenFOAM® do it for us.

6.4 Intrinsic differences with OpenFOAM

Some of the computational schemes and procedures of the original OpenFOAM® and the domain coupling approach cannot be reproduced exactly, and a great care should be taken in order to avoid nonphysical behavior or resolution.

6.4.1 Upwind schemes

The first one is the consequence of using the existing upwind schemes for the discretization of the divergence terms. When using this scheme, the resulting discretized terms depend on a given velocity field, that define the upwind direction, giving more or less importance depending on where the considered cell lies with respect to the flow velocity. As noted by Kim *et al.* (2005) the upwind scheme can be - and is commonly - summoned to discretize the divergence terms (for both implicit and explicit methods), and bases its selection of the weights on the same field that the one that is being discretized. In practice, a given mass flux defined on the cell faces `rhoPhi` is computed from the velocity field `U`. This field stays fixed during a `PIMPLE` iteration (see section 4.3.2.2 for further detail). A call is made to `div(rhoPhi, U)` that spatially discretizes the divergence term. For some discretization schemes - *e.g.* upwind schemes - the mass flux is used not only to compute the term itself, but also to allocate weights depending of the neighboring cell positions relative to the flow direction. However, calls are here made to `div(rhoPhis, Us)` which, on the opposite of a classical upwind approach, will compute the weights based on the complementary flow instead of the total one. Another way to see it is to state that the OpenFOAM function `div(a,b)` may not linear with respect to `a`, depending on the applied discretization scheme, while it mathematically should be.

In order to retrieve the same discretization as with waveFoam or domainCoupling, the advection terms of the complementary velocity could be summed to retrieve the classical upwind scheme (based on `rhoPhit`). It is however not possible for the external velocity on which only the convection by the complementary velocity remains. From a numerical point of view, this means that a choice has to be made between separating or aggregating the advection terms of $\mathbf{u}_p$.

Algorithm 6.5: Us and Up advection, separated

```
1  fvVectorMatrix UsEqn
2  (
3    ...
4     fvm::div(rhoPhis, Us)
5    + fvm::div(rhoPhip, Us) // - New - convection of Us by the
       *potential* mass flux.
6    + fvc::div(rhoPhis, Up) // - New - convection of the
       potential velocity by Us.
7    ...
8  )
```

Algorithm 6.6: Us and Up advection, Us aggregated

```
fvVectorMatrix UsEqn
(
  ...
  + fvm::div(rhoPhit, Us) // - New - convection of Us by the
     *total* mass flux.
  + fvc::div(rhoPhis, Up) // - New - convection of the
     potential velocity by Us.
  ...
)
```

The two possible solutions are shown on algs. 6.5 and 6.6 which correspond respectively to solving the advection of the complementary velocity in a separated manner:

$$\nabla \cdot \rho \mathbf{u}^* \otimes \mathbf{u}^* + \nabla \cdot \rho \mathbf{u}_p \otimes \mathbf{u}^* + \nabla \cdot \rho \mathbf{u}^* \otimes \mathbf{u}_p \tag{6.15}$$

or in an aggregated manner:

$$\nabla \cdot \rho (\mathbf{u}^* + \mathbf{u}_p) \otimes \mathbf{u}^* + \nabla \cdot \rho \mathbf{u}^* \otimes \mathbf{u}_p \tag{6.16}$$

Mathematically, those equations are equivalent, however numerically the discretization schemes employed can yield discrepancies. Note that this remark holds for both the complementary NS equations and the complementary RANS equations, which is why the averaging symbol $\overline{\cdot}$ was dropped in eqs. (6.15) and (6.16).

6.4.2 Residual threshold

Another difference is due to the computation of the residuals and the associated convergence criterion. In this framework, the solver does not solve for the total dynamic pressure but only for the complementary pressure. Orders of magnitude of p^* and p_t are very different. While not as important on velocities, difference in magnitudes are however still present. For most of iterative solving procedures, residual values are used as a convergence criterion. In OpenFOAM, the computation of the mean residual R of a given variable $\mathbf{x}$ that should satisfy $\mathbb{A}\mathbf{x} = \mathbf{b}$ uses the mean value of $\mathbf{x}$ (here denoted $\bar{\mathbf{x}}$) and is computed as:

$$R = \frac{|\mathbf{b} - \mathbb{A}\mathbf{x}|_1}{|\mathbb{A}\mathbf{x} - \mathbb{A}\bar{\mathbf{x}}|_1 + |\mathbf{b} - \mathbb{A}\bar{\mathbf{x}}|_1} \tag{6.17}$$

In eq. (6.17), the nominator is the dimensional residual. $|\cdot|_1$ is the L_1-norm, *i.e.* the mean value of the magnitude of the components. The denominator is a scaling factor that does not depend on the local value of $\mathbf{x}$ but on its mean magnitude. Thus, the dimensional error is divided by something of the same amplitude as the field itself.

Thus, when solving for a complementary variable (especially for p^* which is orders of magnitude lower than the original $p - \rho g h$), a given dimensional error will not yield the same residual as in the original approach. In other words, trying to solve with a tolerance of 10^{-7} means that the error relative to the *complementary* pressure of 10^{-7} has to be reached, while in the original approach it would be relative to the *total* pressure. If, for example, one wants a dimensional precision of 1 Pa, a target residual of $\sim 10^{-5}$ would be chosen with waveFoam or domainCoupling, while a value of $\sim 10^{-2}$ should be chosen

here. This residual target example depends of course on the relative magnitude of p^* and p_t, which is obviously case dependent.

A study on the impact of the target tolerances of the linear solvers will be conducted in section 6.5.3.

6.5 Stability, convergence and various investigations

This section aims to provide a correct set of numerical parameters and boundary conditions that ensure convergence and stability of the method. Investigations are conducted extensively on the parameters that inherently differ from the previous computational approaches, namely the domain coupling method and waveFoam.

6.5.1 Test case

The main test case is the same as the one presented and studied in the preceding chapters, *i.e.* the immersed horizontal rectangle-shaped cylinder, also studied for example by Venugopal *et al.* (2006). The main validation test case consists in regular waves of steepness $H/\lambda = 0.035$ and period $T = 2\,\mathrm{s}$ propagating and interacting with the cylinder. The objective is to retrieve the results presented in sections 4.6 and 5.3.2.

Apart from few intrinsic differences, the base parameters of the simulation remain constant compared to the description given in section 5.3.1. Amongst them, we try to maintain all numerical schemes identical (`PIMPLE` and `PISO` parameters, mesh, time step, discretization schemes and parameters, *etc.*). Thus, most of the parameters are not - or only briefly - recalled here.

With the domain coupling approach as the propagation distance is not simulated and is calculated *a priori* by the potential model, the computation could be started at a later time considering an already developed potential flow. In the same manner, the computation is started at $t_0/T = 12$ and the complementary values are initialized to zero at this time. It is equivalent to the "hotStart" (*i.e.* with values initialized as the potential ones) setup of domainCoupling solver. This is done hoping that the short domain that is used does not require as much simulation duration to reach a periodic behavior. A complete investigation of the benefits of this method was presented with the domain coupling algorithm in section 5.4.1.

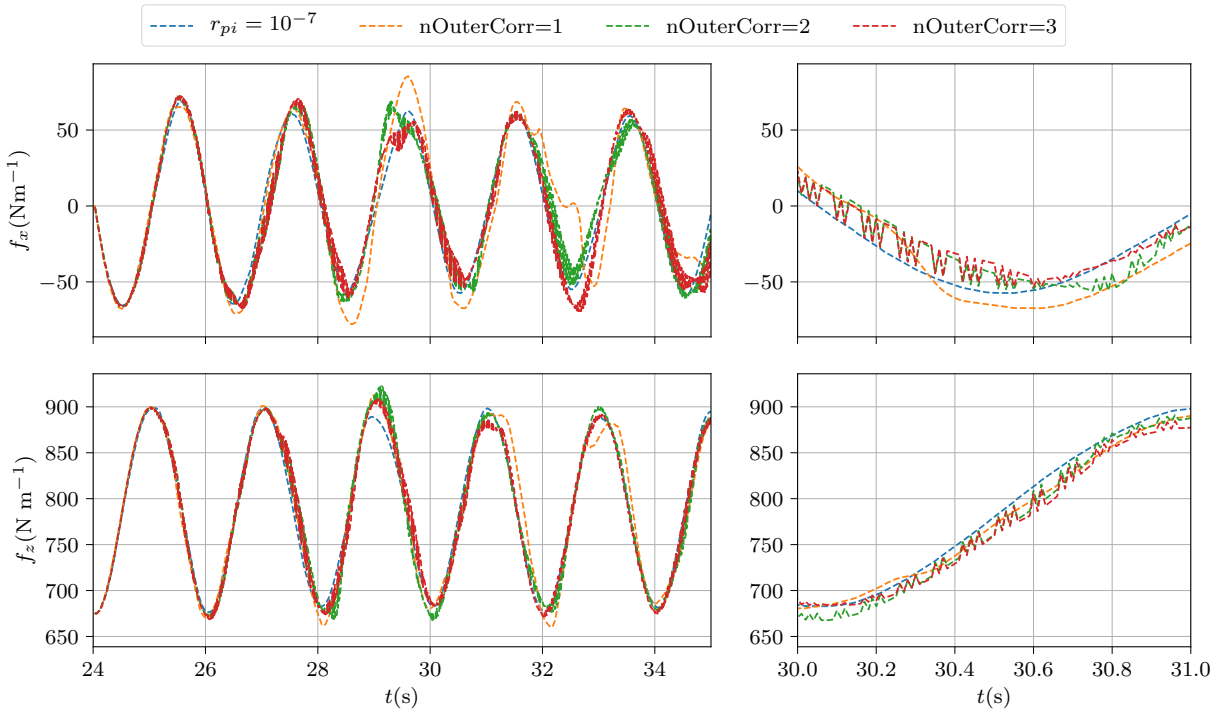
The mesh is also the same as the one presented in section 5.3.1. The mesh breadth in the horizontal direction is selected as $B_m = 3\,\mathrm{m}$ which represents 1/2 of the wavelength. This was proven to be sufficient in a domain coupling framework (see section 5.4.2) and will not be called into question here. The considered mesh spans vertically from $z = -1.8\,\mathrm{m}$ to $-0.3\,\mathrm{m}$ for a total water depth of $h = 2.2\,\mathrm{m}$.

The time step was kept constant and fixed at $dt = 5 \times 10^{-4}\,\mathrm{s}$. This value originates from the time independence study performed with the waveFoam solver presented in section 4.6.2.1.

6.5.2 Numerical scheme and parameters

When applying the parameters (defined in `fvSolution` and `fvSchemes` located in `system/<fluidRegionName>/`) directly imported from our domain coupled cases (but differing from our waveFoam simulations, see sections 5.3.1 and 5.4.4 for details), surprising outcomes are obtained in terms of loads (fig. 6.2). After investigations, a major parameter in this instability is the number of `PIMPLE` iterations (defined in `fvSolution`). On fig. 6.2, computed loads are shown, obtained with different exit conditions for the `PIMPLE` loop. Indeed it is possible to end after a given number of iteration (`nOuterCorr`) or by controlling the pressure residual at the beginning of the `PIMPLE` loop denoted r_{pi}, *i.e.* with

matrices that are obtained from an updated velocity flux. No computation except the residual controlled one is stable and yield correct results. This is relatively difficult to explain as the residual controlled simulation never performs more than two PIMPLE iterations, but instead alternates between one and two. Contrary to expectations, it seems that more iterations lead to even worse results in terms of stability (See the high frequency disturbance obtained with nOuterCorr= 3 in fig. 6.2).

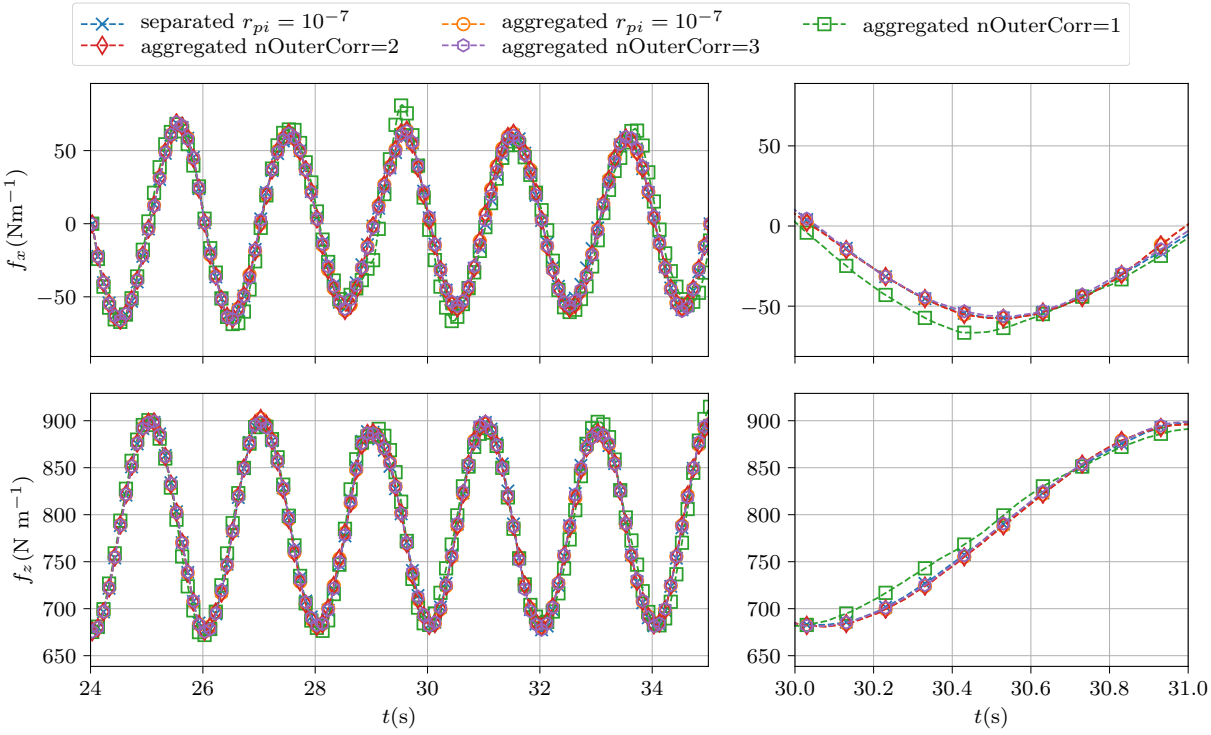


(a) (b) Zoom

Figure 6.2: Temporal series of the loads obtained with the velocity decomposition method with different exit conditions for the PIMPLE loop, either a fixed number of iterations (nOuterCorr) or controlled by the residual target. Boundary set 2. (b) is a focus on $t/T \in [15, 16]$.

In order to understand the origin of this phenomenon, we questioned every term inside the PIMPLE loop. As the PISO loop (described in section 4.3.2.2) control parameters did not affect the computations in a significant way, a focus was made on the steps carried out during the PIMPLE iteration but outside of the PISO loop. No influence of the momentum predictor on the quality of the results was denoted. Thus, an investigation was done on the terms discretization methods used in the complementary velocity equation. In section 6.4.1 a difference between the original waveFoam discretization (same as in the domain coupling framework) and the current method is described when an upwind discretization scheme is used. Thus, here are tested the differences between algs. 6.5 and 6.6 denoted respectively "separated" and "aggregated", recalling that the results shown in fig. 6.2 were obtained with the "separated" version.

While some differences can be spotted between the different numbers of outer corrector step we can see (fig. 6.3) that a great stabilization is achieved by reuniting the two advection terms of the complementary velocity. Note that for two or more outer corrector



(a) (b) Zoom

Figure 6.3: Temporal series of the loads obtained with the velocity decomposition method with different exit conditions for the PIMPLE loop, either a fixed number of iterations (nOuterCorr) or controlled by the residual target. See text for detail between "separated" and "aggregated". Boundary set 2.(b) is a focus on $t/T \in [15, 16]$.

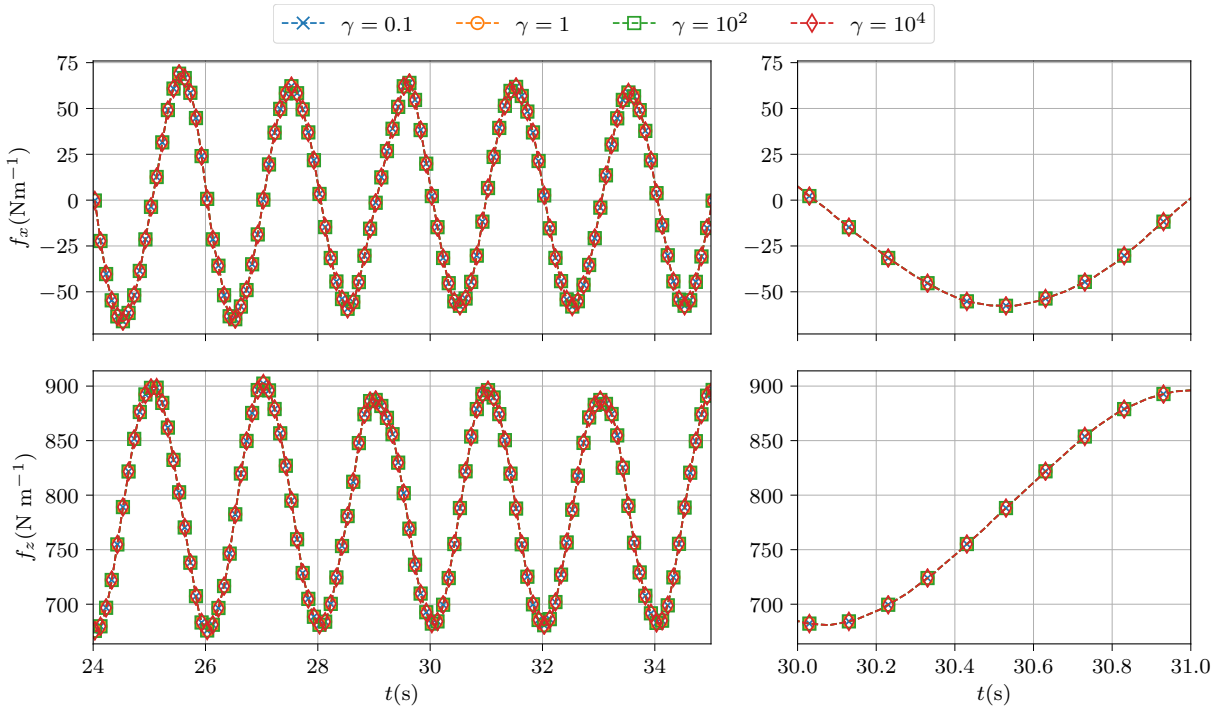
steps, we almost retrieve a converged computation that yields consistent results. However the residual controlled PIMPLE method seems to yield the most stable results and is thus selected for the rest of this work.

Also note that the other advection term $(\nabla \cdot \rho \mathbf{u}^* \otimes \mathbf{u}_p)$ also triggers a different version of the upwind scheme. However, the counterpart term is not present anymore and thus the resulting term cannot be aggregated.

6.5.3 Tolerances and residual controls

In this section, a focus is made on the modification of the tolerances that trigger both the end of the solvers and the end of the PIMPLE Loop. Because the numerical problems are intrinsically different, the residual cannot be directly compared to the one obtained with OpenFOAM® as was explained in section 6.4.2. In all previous computations, tolerance of the linear solvers, *i.e.* target residual that trigger the solver exit, were set to $r_{\mathbf{u}} = 10^{-9}$ for the velocity and to $r_p = 10^{-8}$ for the pressure. The residual threshold that allows the exit of the PIMPLE loop (tested at the beginning of an iteration, thus, with $\mathbb{H}$, p, $\mathbf{u}$ and $\mathbf{u}_f$ up-to-date and consistent), was set to $r_{pi} = 10^{-7}$. A scalar denoted γ that multiplies in the same manner all the previously listed tolerances is defined. $\gamma = 1$ corresponds the original tolerances (presented above), while with $\gamma = 100$ the tolerances are set to $r_{\mathbf{u}} = 10^{-7}$, $r_p = 10^{-6}$ and $r_{pi} = 10^{-5}$. Thus, we expect to obtain a convergence when γ

decrease along with an increase of the computation cost. Note that all tolerances are left unchanged for any other variables (volume fraction and turbulence).



(a) (b) Zoom

Figure 6.4: Temporal series of the loads obtained with the velocity decomposition method with different tolerance targets. γ parametrizes the restriction of the target residual values compared to the base case. See text for details. (b) is a focus on $t/T \in [15, 16]$.

Figure 6.4 shows the obtained results with different values of γ. The computation cost drastically decreases: the first full wave period costs approximately 35 min for $\gamma = 10^4$ and increase up to 165 min at $\gamma = 10^{-1}$. Note that the decrease is not linear: for example the first full period cost between the two finest cases only differs by a few minutes. Assuming that the periodic behavior is reached within 8 periods, a rapid extrapolation furnishes an estimation of the requested computational time to yield valuable results: ~ 5 h with $\gamma = 10^4$ and ~ 22 h with $\gamma = 1$ (monoproc, Intel® Xeon® CPU E5-2650 v3 @ 2.30GHz). However no difference can be seen on fig. 6.4 between the different computation cases. Thus, it is possible that - for the reasons detailed in section 6.4.2 - larger tolerances target are sufficient for the complementary RANS equations. However, as no such study was conducted on the domain coupling or waveFoam solver, it is not possible to securely guarantee that the same behavior would not be observed.

To ensure that the computations are indeed equivalent, a local analysis is performed at $t/T = 18$ (*i.e.* after 6 simulated periods): the fields are sampled along a vertical line located on the center top of the cylinder ($x_l = 0$, $z_l \in [-0.72, -0.3]$m). The obtained fields are depicted in fig. 6.5: an almost perfect agreement can be observed across all the tested range of control parameter γ.

The low amplitudes discrepancies observed on the complementary pressure are orders of magnitude below the expected errors due to others parameters (such as the mesh itself,

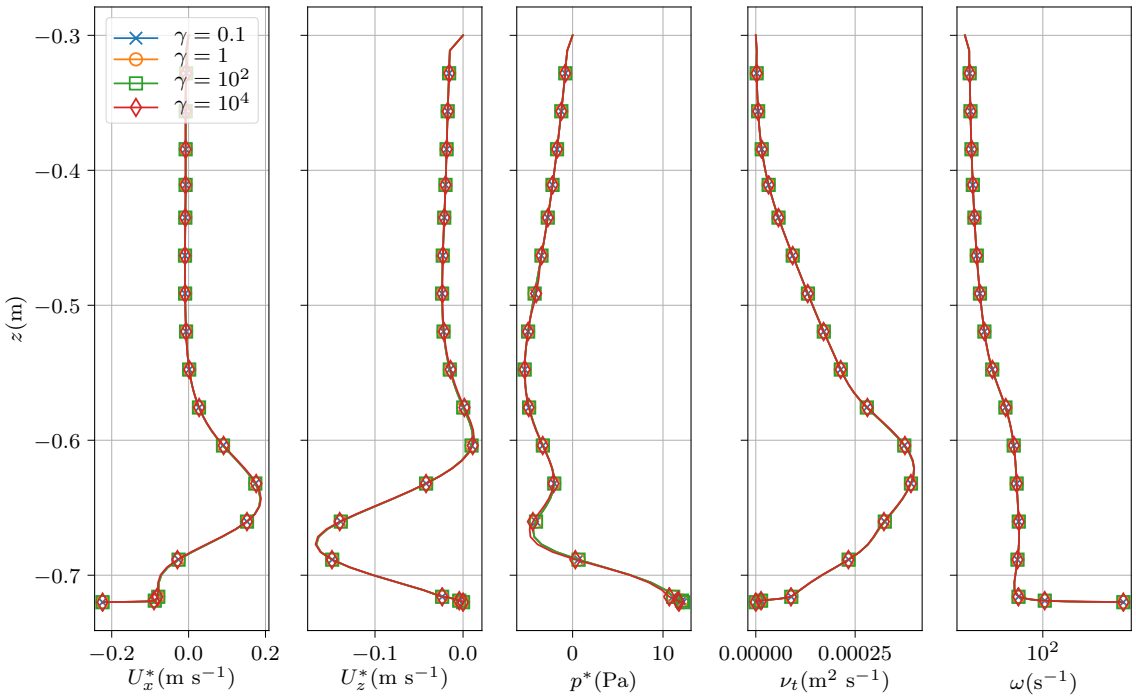


Figure 6.5: Fields at $t/T = 18$ (6 simulated periods) over a vertical line at $x_l = 0$ for different γ parameters. A marker is added every 5 faces encountered. ω variable is shown on a log-linear scale: it ranges from $1\,\mathrm{s}^{-1}$ to $1 \times 10^4\,\mathrm{s}^{-1}$

the time step value but mostly the turbulent rise). In a quantitative manner, defining the error with respect to the $\gamma = 0.1$ computation, the pressure at $z_l = -0.67$ m differs by $3.1\%, 0.06\%$ and 0.0005% for $\gamma = 10^4, 10^2$ and 1 respectively. Note that those differences are relative to the maximum observed *complementary* pressure at this particular time along the line, ≈12 Pa. It is recalled that the order of magnitude of the *total* dynamic pressure along this same line is 700 Pa (see *e.g.* fig. 5.16 or fig. 5.8) and the static pressure is even one order of magnitude higher at $z = -0.72$ m. Thus, the highest tested tolerances target ($\gamma = 10^4$) is completely acceptable, and a further increase of γ could even be possible. For the rest of this work, we select $\gamma = 100$ to ensure that no discrepancies originate from this selection.

As a conclusion, it is possible to state that accepting a seemingly important value of the residual does yield already converged computations. It is thought to be justified by the reason described in section 6.4.2. Hence, with $\gamma = 100$, values for target residuals are selected as $r_{\mathbf{u}^*} = 10^{-7}$ for the complementary velocity problem, $r_{p^*} = 10^{-6}$ for the complementary pressure and $r_{pi} = 10^{-5}$ for the `PIMPLE` loop.

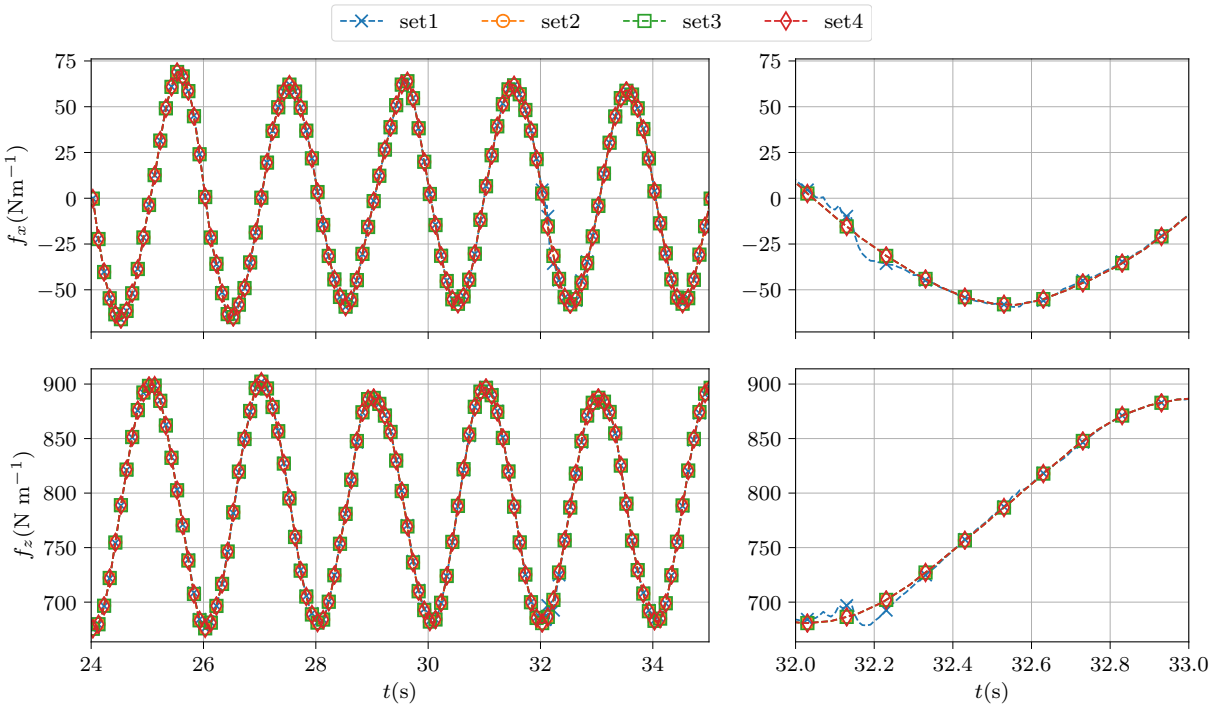
6.5.4 Boundary conditions

In order to solve for the complementary variables $\mathbf{u}^*$, p^*, a set of boundary conditions has to be selected for every physical boundary of the CFD domain represented on fig. 6.1. The natural choice would be, at the external boundaries Γ, to impose a null value for both $\mathbf{u}^*$ and p^* as those fields represent the additional (or complementary) fields compared to the main potential one and that are assumed to vanish away from the body. This set is

denoted set 1 (see table 6.1). A discussion about the available choices and their numerical implementation is conducted in section 6.3.5. Note that the body boundary conditions is selected and will be maintained as `minusCoupledVelocity`, described in the referenced section.

BC set	boundary	$\mathbf{u}^*$	p^*	$k(\mathrm{m}^2\,\mathrm{s}^{-2})$	ω (s^{-1})
1	$\Gamma_{i,o,t,b}$	fV: $\mathbf{0}$	fV: 0	fV:10^{-10}	IO: 0.88
2	$\Gamma_{i,o,t,b}$	IO: $\mathbf{0}$, u_f^*			
3	$\Gamma_{i,o,t,b}$	IO: $\mathbf{0}$, u_{ft}			
4	$\Gamma_{i,o,t,b}$	IO: $\mathbf{0}$, u_{fp}			

Table 6.1: Table of the base set of boundary conditions. IO stands for `inletOutlet`, fV for `fixedValue`. Values are precised when necessary (and not dummies). For IO conditions, we also indicate the face flux used to determine the direction of the flow.



(a) (b)

Figure 6.6: Obtained loads for different complementary velocity boundary conditions presented in table 6.1. For further details the reader is referred to section 6.3.5.

The selection of fully imposing a null fixed value at the outer boundaries for the complementary velocity (set 1) implies that no fluid can either enter or exit the domain except if explicitly prescribed by the potential model. While this is consistent with the recommended span of the mesh, which should be wide enough to encompass all pure CFD aspects of the flow, it is not possible in practice. This leads to instabilities even if the model manages to recover without a break-down of the computation. This problem can be resolved by simply giving the solver the freedom to have a non-null velocity whenever the face flow is not consistent with the velocity itself. Further allowing inwards and

outwards complementary flow whenever the potential flux points out of the domain also stabilizes the computation: that is, in essence, the solution adopted in the domain coupling framework. Figure 6.6 depicts those described effects on the load series.

For these reasons, the second set of boundary was selected for all the previous computations and will be maintained for the rest of this work (previous results also used this set).

6.5.5 Time step influence investigation

Another parameter that could drastically influence the results (and was of prime importance with waveFoam) but was kept fixed here, is the temporal discretization. Indeed, most of the former original NS equations are already solved and advanced in time (with the HPC solver). Thus, the remainder of this equation, the complementary NS equations, may not need the same time discretization. However, as the time step is mainly determined by the small scale effects, copying the time step that was emphasized in the independence study done with waveFoam might not be invalid. Moreover, the CFL number - which usually controls the time step - should remain below unity because otherwise a given particle (or scalar field) would be advected of more than one cell in at a given time instant. In our case, the advection velocity of greater magnitude is still $\mathbf{u}_t$, and still transports all variables except the potential velocity. This further supports the choice of keeping the same time step size.

Nevertheless, it would be interesting to conduct a complete time independence study (as well as a grid independence study) with this particular scheme. Due mainly to time constraint the hypothesis was made that, the influence of such modifications would be the same on this method as on the resolution with waveFoam. In all the previous and past computations, the time step size remains constant at $dt = 5 \times 10^{-4}$ s. This value is close to the minimum time step size obtained when the maximum CFL is set to 0.25 on the current mesh.

6.6 Validation of the velocity coupling method

6.6.1 Local comparisons with other models

This section focuses on the local fields over the same vertical line used before, *i.e.* located on the top of the cylinder at $x_l = 0$, from $z = -0.72$ m to -0.3 m (see line l_{v1} on fig. 4.1). We compare the velocity coupling results with the obtained results within the domain decomposition study (domainCoupling), and the independent OpenFOAM® results, denoted "waveFoam", the available solver part of the waves2Foam toolbox (Jacobsen *et al.*, 2011).

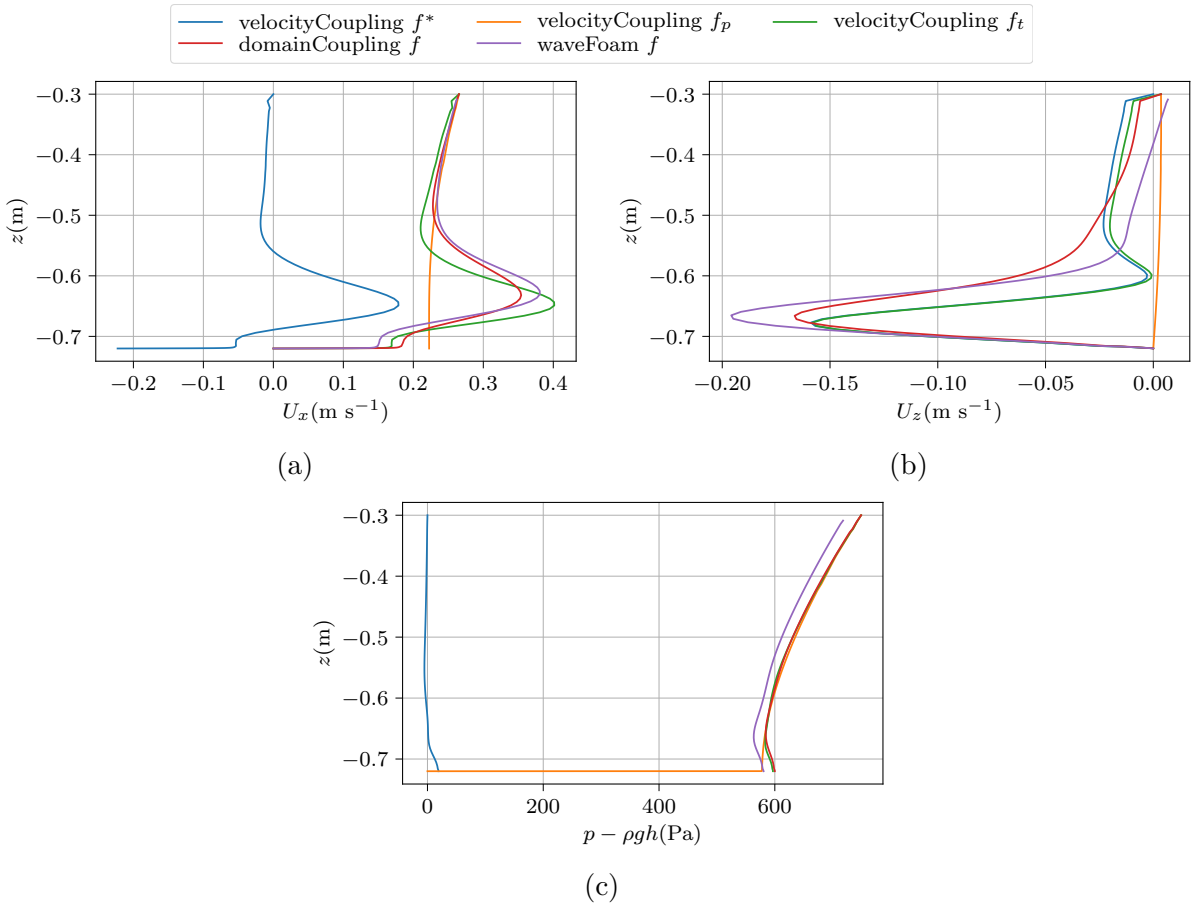


Figure 6.7: Comparison of the complementary fields, the potential ones and the resulting total ones with waveFoam and the domain decomposition solver at a particular time $t/T = 20$.

Figure 6.7 shows the different complementary fields along that line at a given physical time $t/T = 20$. It is recalled that the coupled algorithms (velocityCoupling and domainCoupling) are started at $t/T = 12$. Discrepancies are present on the figure on each variable in comparison with both waveFoam and the domain coupling solver. However the same general effects are captured, amongst them the vortex of center located approximately at $z_l = -0.65$ m that can be deduced from the velocity figures. It is also noted that the complementary pressure is of very low order of magnitude when compared to the dynamic pressure. This adds perspective to the error chasing that we performed on the previous

sections (*e.g.* section 6.5.3).

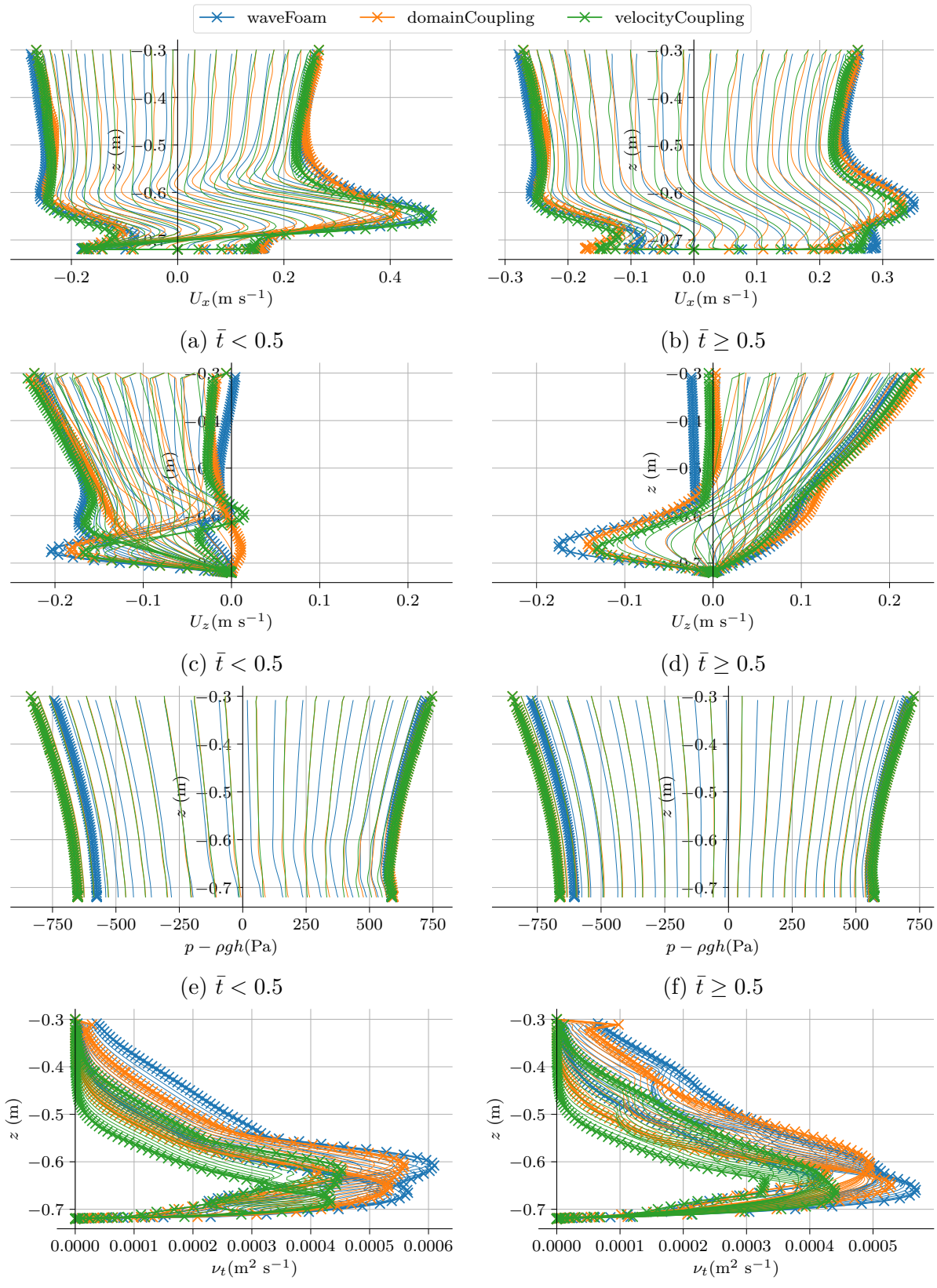
Though the agreement is not perfect, the reader is recalled that a null complementary pressure is imposed at $z = -0.3\,\mathrm{m}$, and thus, if waveFoam and HPC disagree on the value there, our coupled algorithm will not be able to recover the difference. It is illustrated on fig. 6.7c on which the pressure discrepancy propagates in an almost constant manner throughout the sample line: the profile of the velocityCoupling total pressure is almost exactly the same as the waveFoam pressure, with an added shift coming from the imposed value at the top.

Also notice that this figure depicts the sampled fields at a given time and could be affected by differences associated with phase-shift. In chapter 4, the occurrence of such phase-shift was demonstrated during the waveFoam propagation. For this reason, the rest of the figures depicting the local samplings of the fields will adopt the layout used when assessing the correct behavior of the domain coupling algorithm in section 5.3.2: multiple times will be shown on the figures, along with their envelopes at each vertical position. However, with this layout, it is not possible to depict the decomposed components ($\mathbf{u}^*$ and $\mathbf{u}_p$) and maintain readability. Thus, only the total velocity yield by the velocity coupling scheme will be shown and compared. However, as the potential fields will appear, the complementary variables can be retrieved by subtraction.

Figure 6.8 depicts those sampled fields obtained with the waveFoam solver, the domainCoupling solver and the velocityCoupling solver. Note that, while the relative time to period $\bar{t} = t\%T$ - where $\%$ denotes the remainder of the Euclidean division - is constant across cases, the absolute time itself is not. The coupled fields are extracted after 6 simulated periods ($t/T \in [18, 19]$) following the recommendations exhibited on the hotStart study section 5.4.1 while the OpenFOAM results are shown at the same time steps as presented in previous chapters, *i.e.* $t/T \in [15, 16]$. Thus, the turbulent viscosity instability might not have the same influence in both cases.

It should, however, have the same behavior between the velocity coupling case and the domain coupling one. This is not found to be true (figs. 6.8g and 6.8h): the amplitudes of ν_t predicted by the velocity coupling are approximately 20% lower than the domain coupling one (which in turn is relatively consistent with the waveFoam outputs). The shape and behavior seems however to be well captured and it is completely possible that the amplitudes would also be recovered after a longer evolution time of the velocity coupling algorithm. This effect is still under investigation. It could however be the result of any of the intrinsic differences presented above, or even of the necessary changes made to the discretization schemes and numerical loops: the turbulent viscosity instability growth proved to be very sensitive to any parameter change.

Other results however do not seem to be impacted in a significant manner by this turbulent viscosity difference. As far as the total velocity is concerned, it seems that some effects are even better captured with the velocity coupling when compared to the original waveFoam. Note that a period was separated into 40 time steps for the representation, and the envelopes were calculated based on the values at these time instants. However, $T/40$ is not a sufficient description to correctly capture extrema, especially when high gradients and important temporal derivatives are present. For example, focusing on the minima of U_z that are noticeable on fig. 6.8d, no other lines lie between the waveFoam and other model envelopes: the coarse time discretization, conjugated with a small time shift, might be the reason of this discrepancy.



(a) $\bar{t} < 0.5$ (b) $\bar{t} \geq 0.5$

(c) $\bar{t} < 0.5$ (d) $\bar{t} \geq 0.5$

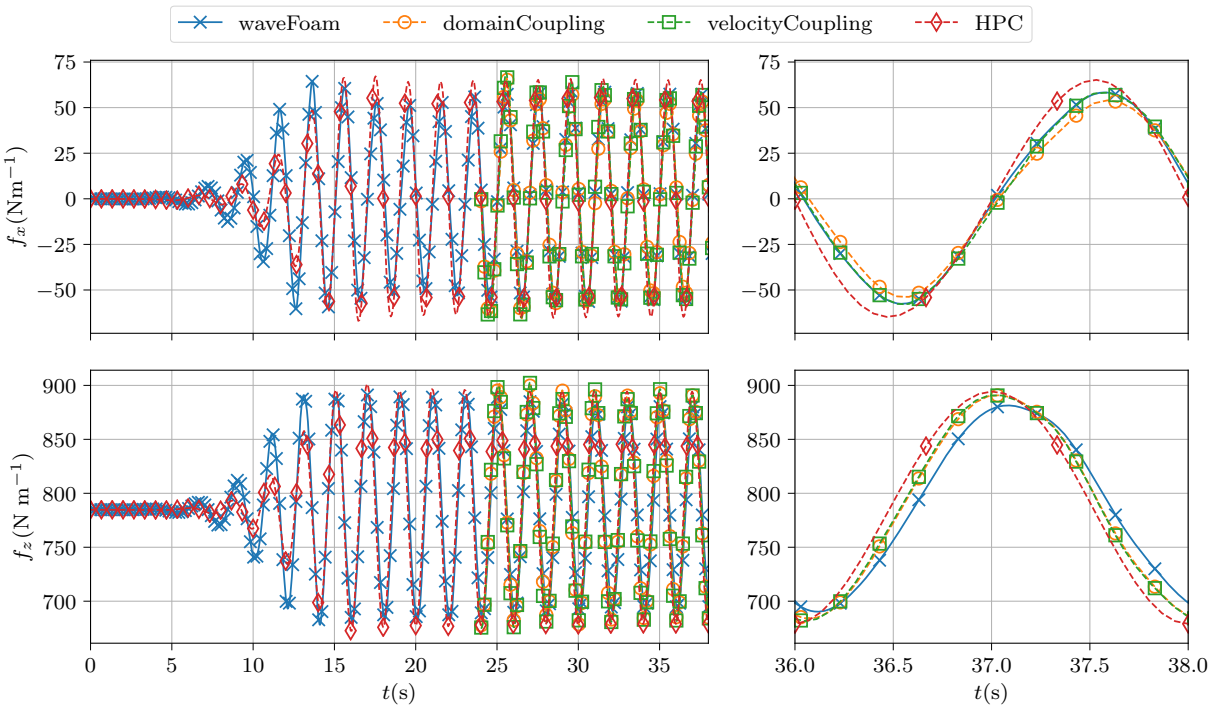
(e) $\bar{t} < 0.5$ (f) $\bar{t} \geq 0.5$

(g) $\bar{t} < 0.5$ (h) $\bar{t} \geq 0.5$

Figure 6.8: waveFoam, domain coupling and velocity coupling total fields sampled over a vertical line ($x_l = 0$m, $z = -0.72$ m to -0.3 m) at 40 time instants split in two half-periods (left and right subfigures).

At this stage it is possible to confirm the capabilities of the velocity coupling algorithm in recovering most of the turbulent and vortical effects. We thus expect correct comparisons between the experimental results and this scheme.

6.6.2 Obtained loads with the velocity decomposition method



(a) (b)

Figure 6.9: Temporal series of the loads obtained with the waveFoam solver, the domain decomposition method, the velocity decomposition method and HPC

The forces series are shown on fig. 6.9 which compares the results predicted by our four models of interest, namely the VoF-FVM RANS solver waveFoam, the potential-RANS domain decomposition solver, the potential-RANS velocity decomposition solver and the potential solver based on the HPC method.

The resulting loads exhibited by the velocity coupling solver matches in a really close manner the domain coupling results. When looking at the predicted pressure on figs. 6.8e and 6.8f, such a behavior could be expected as the domain and velocity coupling pressure are in close agreement.

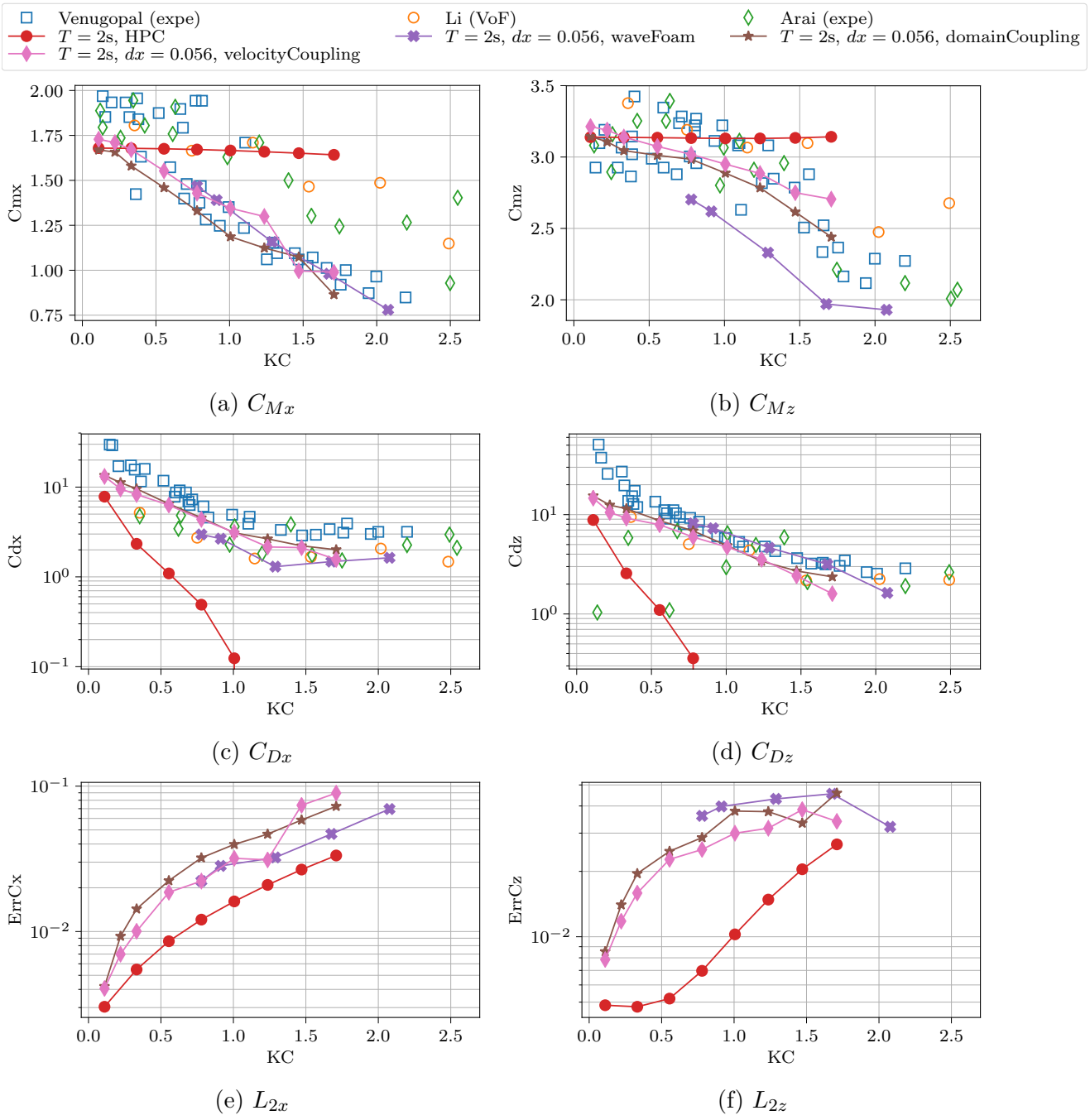
As this stage, it is possible to state that the proposed unidirectional velocity coupling yields valuables results for only a small portion of the computation cost of a complete RANS-VoF simulation. In our case, it leads to the possibility to test several parameter variations. With an industrial approach, it would allow to increase the number of design cases, for example varying the wave field conditions.

6.6.3 Comparisons with experiments

For the same case of horizontal cylinder with rectangular cross-section, different wave heights were already computed within the HPC framework (see section 3.3.3). It is then straightforward to run the newly implemented model with those cases. Here, we focus on the cases with incoming regular waves of period $T = 2\,\mathrm{s}$. Once again, the obtained loads can be reduced to four hydrodynamic coefficients. The calculation of the best (defined with respect to the L_2 norm) coefficients to model a given load series is presented in section 3.3.3.2. The obtained hydrodynamic coefficients are shown for different KC numbers on fig. 6.10. The error when fitting our loads with a Morison model (L_2 error we minimized) is also depicted on the last two subfigures. Thus, high errors that are noticeable for high nonlinearity parameters (*i.e.* high KC numbers) underline that the Morison equation is not adapted to model such loads, and *not* that the velocity coupling yields a high error compared to experimental results.

The obtained hydrodynamic coefficients are in good agreement with both the literature and the domain coupling approach, up to relatively high KC numbers. Once again, the decrease of the inertia coefficients - that could not be obtained with the potential model - is retrieved in a consistent manner. Discrepancies however arise for the steepest incoming waves ($H/\lambda = 7.5\%$), and limited to the vertical coefficients: an overestimation of the inertia coefficient and underestimation of the drag coefficients are observed.

Note that no other modification than the selection of the external results (*i.e.* HPC case used to obtain the potential variables) was done compared to the base wave height case previously studied in detail (KC=0.78): neither the mesh nor any of the numerical parameters (time step, schemes, tolerances, boundary conditions, *etc.*) were changed during the variation of the KC number.



(a) C_{Mx} (b) C_{Mz}

(c) C_{Dx} (d) C_{Dz}

(e) L_{2x} (f) L_{2z}

Figure 6.10: Inertia and drag coefficients in the horizontal (x) and vertical (z) directions obtained with the domain decomposition method, the velocity decomposition method, the VoF-FVM method (waveFoam) and the HPC method for different KC numbers. The L_2 error of the Morison fitting to the temporal loads series is also shown in the lower panels. Important error reveals that the Morison model is not well adapted to describe the temporal series.

6.7 Conclusions on the velocity decomposition scheme

In this chapter, a velocity coupling scheme was developed in the same manner as the one proposed by Kim *et al.* (2005). The mathematical methodology and numerical implementation were developed with keeping as a goal to allow for a modularity and evolution of the code. It was developed making use of the open source (GPL v3) OpenFOAM® library, version 17.12. The plan is to release the developed solver and the corresponding library publicly by the end of 2020.

The method is then applied and several numerical investigations are conducted. Some intrinsic differences with the original method were pointed out and their consequences were extensively assessed. It was shown, for example, that aggregating the two convection terms of the complementary velocity yields stable results for a wider range of parameters. However, when a control in terms of `PIMPLE` residual is chosen, both methods are in almost perfect agreement.

Another difference concerns the residual of the iterative solvers themselves: the newly defined problem (matrix, unknown vector and RHS) significantly changed compared to the original one corresponding to the original RANS equations. Thus a convergence in terms of target residuals and their effects was conducted, only to show that almost no influence can be identified on the results themselves even up to important tolerance values - and thus cheap - resolutions. In fact, choosing the highest tested value of target residual (corresponding to $\gamma = 10^4$) is completely possible. The maximal observed difference after 6 simulated periods is of about 0.4 Pa. Compared to the static and dynamic total pressure, this is completely negligible. For example, the discrepancy between the potential and the RANS pressures at the outer boundaries - which, on a one-way coupling, cannot be corrected - is of about 20 Pa. On the other hand, with this parameter value the computation cost decreases by a factor ~ 5 compared to the original tolerances.

Results were compared to the previously presented models and a good agreement was found both in most of the fields and variables, as well as in the load series predictions. It was possible to show that the major part of the vortical and turbulent effects were correctly captured and in agreement with the original VoF-FVM method.

A good agreement was found with experimentally obtained hydrodynamic coefficients, up to important degree of nonlinearity of the incoming waves ($H/\lambda = 7.5\%$). With OpenFOAM®, the phase shift of the load had to be corrected according to its propagation error in order to obtain decent results in terms of hydrodynamic coefficients. This correction was not needed with this approach, which relies on the HPC model to propagate the waves. On this regard any model that does not dissipate energy can be very accurate. Thus, no such correction had to be applied to the obtained load series to extract the presented hydrodynamic coefficients.

Note that however, some of the drawbacks are recovered: the turbulent viscosity field does not stabilize in a physical range, as could have been hoped for. It is thought to be explained by the exponential growth that will always happen under a potential wave-flow (Larsen and Fuhrman, 2018). In the vicinity of the body, turbulence is generated that feed the instability slightly further away from the body.

A first continuation of this work on the velocity coupling scheme would be not to simplify the Euler equations on the potential variables as stated in eq. (6.5), in the manner suggested by Zhang (2018). In a fully submerged case, this would imply that the error on

this equation (that can for example arise from the interpolations onto the inner domain) would behave as a source term in our modified RANS equation. This would also open the possibility to use the exact same routines with other external models (*i.e.* a potential viscous scheme as in Li (2017), a laminar solution, *etc.*), and give the numerical method the role of simplifying the adequate terms.

A more detailed description of the future potential works and developments are given in the general conclusion of this dissertation, the main reason being that most of the perspectives also apply to the domain coupling approach developed in chapter 5.

Chapter 7

General conclusions of the presented research

Contents

7.1 Summary of the main results **190**
7.2 Analysis of CPU costs **193**
7.3 Future works and further developments **195**

7.1 Summary of the main results

This thesis was devoted to the mathematical modeling and numerical simulation of interactions between nonlinear ocean waves and fixed marine structures of arbitrary shape. These structures can be either submerged or surface-piercing. Only 2D (x,z) configurations were considered here. Particular attention was paid to the prediction of nonlinear loads on the structure, with emphasis on the analysis of viscous and turbulent effects. In total, 4 modeling approaches have been considered and compared: a "fully potential" approach using a solver based on the HPC method developed from scratch during this work, a "fully viscous CFD" approach is the OpenFOAM® toolkit, and two potential-coupling strategies making use of the above mentioned solvers.

7.1.1 Fully potential approach (HPC)

First, a HPC method was implemented, giving very promising results against experiments. The obtained numerical wave tank was shown to have a great stability and good accuracy in capturing a highly nonlinear standing wave ($H/\lambda = 10\%$) freely evolving over long physical times (100 periods). The convergence orders are close to 4 in both time step size and space discretization, which was expected given the various applied numerical methods. Good quality results are obtained, on cases of practical interest including wave-body interaction, up to the third order of the predicted wave loads. Further applying the NWT on a free surface piercing body, but also sharp cornered objects, a fair agreement of results is shown compared to existing literature results, but also compared to the dedicated set of experiments performed during this work. No smoothing of any kind was used during this work in order to test for the inherent stability of the employed methods, particularly the immersed boundary free surface.

The limits of the potential theory are however encountered, and discrepancies with experimental data are found. This was expected, as turbulent and rotational effects were supposed to occur with some of the tested wave steepness and body geometry, such as air entrapment, vortex shedding, *etc.* Note that some of them were visually noticed during the experiments.

7.1.2 Fully viscous CFD approach

We have then shown, on the deeply immersed, rectangular-based, horizontal cylinder, that waveFoam is able to capture those effects, at the price of an increase in the CPU requirements. In any case a good comparison with experimental results was achieved concerning the loads series, especially if one takes into account the propagation discrepancies (in surface elevation amplitude and phase) to correct the hydrodynamic coefficients. However, setting up the model and its parameters in order to correctly capture all the required aspects of the flow has proved to be challenging. It is thought to be mainly the result of a growing instability of the turbulence closure scheme formally exhibited by Larsen and Fuhrman (2018), fed by the turbulence generation in the body vicinity in our case. In order to prevent this, these authors have suggested, and released, a modification of the turbulence model, in order to dampen the turbulence values whenever the vortical effects are small. In this work we have shown that this model has an intrusive impact within the vortex shedding region. A further modification was suggested and preliminary results

show good quality behavior. However, none of the tested models were able to correct entirely the supposedly unphysical growth of the turbulent viscosity.

In order to compute both the wave propagation and the body vicinity in an optimized manner, each zone would require its own turbulence model, numerical parameters and schemes, meshes, and time step size. For example, an interesting study in order to obtain an almost optimized propagation process was performed by Larsen *et al.* (2019). Those emphasized parameters are however probably not optimized for the body interaction flow. Because the most efficient resolution method of large scale wave propagation still is the potential approach, the last two chapters focus on developing and validate two coupling approaches, implemented making use of the OpenFOAM® package.

7.1.3 Potential-viscous coupling using a domain decomposition

The first one, a one-way domain decomposition method is implemented and largely evaluated on the same test case. The implementation done with the objective to be modular, in the sense that any other OpenFOAM® formatted set of results can serve as our "external" results (for example a laminar computation). In our tests, the HPC results were selected, after conversion into an OpenFOAM® format. The corresponding outer boundary conditions, that enforce the "external" model values, are developed. A really good agreement with waveFoam is obtained both in terms of global variables, but also down to local descriptions of flow and turbulent variables. Even the turbulent viscosity field is in agreement with the waveFoam results, despite its relatively high sensibility to many parameters, as was denoted in the previous chapter. Investigations on potential computational time savings were conducted, and will be detailed in the next section. The significant reduction of CPU costs allowed us to test several parameters configurations, to obtain a roughly optimized setup. Some discrepancies, between our domain coupled solver and waveFoam, are however still noticeable, but seem to be mostly attributed to the difference in the wave propagation process: a better confidence is granted to the HPC propagation with which the wave field is not subjected to any damping, or at least with which the associated numerical error is much lower. Note that while no results are shown in this study, the previously described waveFoam case also proved to work as the "external" model.

7.1.4 Potential-viscous coupling using a velocity decomposition

Finally, a one-way velocity coupling approach is developed. Potential velocity and pressure fields, as well as their derivatives, were already obtained throughout the whole domain, *via* the HPC resolution. Thus a large part of the NS equations is already satisfied everywhere. For this reason, the velocity and pressure are decomposed into their potential parts (already known by interpolation of the HPC results) and their complementary components. For theses last ones, a variation of the NS equations is derived. This method was implemented and also yielded good quality results when compared to the other methods on the same case. In particular we showed that it was possible to increase the requested residuals (increasing the tolerance of the solvers) without any loss of overall dimensional precision, and still obtain a significant decrease in computational cost. This solver, however, proved to be slightly less robust than the domain coupling one, necessitating the

PIMPLE algorithm to be activated and its exit condition controlled by a given tolerance. Note that this was found to be true with our set of parameters and boundary conditions. Compared to the domain coupling method, the results are almost equivalent, both in terms of loads and local fields descriptions, even if the horizontal loads were found to be of higher amplitudes (4%).

Several implemented features are not shown in this manuscript, such as the free surface coupling implemented in both methods, because the results are still preliminary and do not seem to exhibit the wished degree of stability.

7.2 Analysis of CPU costs

In this section, few CPU costs are compared on the main studied case considered in section 4.4. Almost all computations were run on a 40 core Intel® Xeon® CPU E5-2650 v3 @ 2.30GHz. We took advantage of the parallelism only in waveFoam computations.

Note that because limited efforts were dedicated to the optimization of the HPC implementation, no CPU times are given this model. For example, at 30×10^3 nodes, the time spent to associate matrix indexes to nodes is overwhelming compared to any other steps. However, the same process is almost instantaneous within OpenFOAM®, even up to 400×10^3 cells, denoting that this problem is solvable.

Table 7.1 shows some of the lower achieved CPU costs needed to obtain correct results with the 3 RANS methods. The mesh is the same in terms of discretization ($dx = 0.056$ m in the body vicinity) but its span, particularly in the stream-wise direction, is different, respectively 24 m for waveFoam and 3 m for the coupled cases. However, it is was shown that a reduction of this CFD breadth below 3 m was possible with the domain decomposition method, but, as this variation was not studied, CPU times for this larger breadth are shown instead, for a fair comparison between those models.

Moreover, the presented cost values for waveFoam are estimated (see "∼") by comparing a parallel run (run on 4 cores) and a sequential one over 1 period, and thus might not be perfectly accurate. Moreover, these values also proved to be prone to differences even when the same computation is performed (for example the writing results interval can be a source of variation) and trust can only be granted in their orders of magnitude, say ±10%. Lastly, no investigation of the requested tolerance was conducted for waveFoam or domainCoupling methods, and even the one concerning velocityCoupling could have been pushed further ($\gamma > 10^4$, see section 6.5.3).

Nevertheless, a significant gain is achieved with the coupling approaches without - or at least not noticeable here - loss of information. The chosen test case is well fitted for such coupling: deeply immersed, large wavelength compared to the body dimensions. However, it is thought that using such method as a further step after a potential computation is of great interest, for a relatively contained CPU cost. Moreover, performing a mesh/time step independence study with these methods could be valuable due to the separation of propagation and body vicinity physics but also the contained CPU cost.

	waveFoam	domainCoupling	velocityCoupling
periodic behavior after $\frac{t-t_0}{T} =$	15	6	6
number of cells	138×10^3	$<31 \times 10^3$	$<31 \times 10^3$
periodic behavior CPU time	∼24 h 4 min	3 h 11 min	3 h 0 min
cost of additional period	1 h 46 min	0 h 36 min	0 h 30 min

Table 7.1: CPU costs with the main employed mesh ($dx = 0.056$ m in the body vicinity) using one computational core. Note that those CPU time values might not be perfectly accurate (see text for details).

Remark. *Note that these results were drawn from cases with a mesh of $dx = 0.056$ m in the body vicinity, and coarser meshes were shown to also yield valuable results with waveFoam (see section 4.6.2.1). On these meshes (on which the background discretization*

is constant) a span reduction, e.g. *by our coupled approaches, would lead to a greater relative cell number reduction than with the mesh studied here. The relative CPU savings should be, in turn, more pronounced.*

7.3 Future works and further developments

This final section aims at discussing some of the possible follow-ups of the presented work. We discuss possibilities and tests that could be conducted rather directly in the continuation of this work, but also longer term ideas that would require a greater amount of time investment. The future works associated with HPC approach and the coupling methods are treated separately due to their independence in this regard.

7.3.1 HPC numerical wave tank

Future developments on this specific part of the work could encompass three main aspects, briefly outlined hereafter. First, the HPC model needs to be extended to 3D in order to simulate more realistic cases. Some developments were already done in this direction, but significant works concerning the IBM free surface (its evolution, but also identifications of the nodes types: ghosts, inactive and fluid, see section 2.3.4) and the general matrix construction are still required. However, none of the presented approaches are fundamentally restricted to 2D.

Second, we plan to implement the case of moving bodies in waves. In this direction the code has already been developed so as to allow a relative motion between the body fitted mesh and the background mesh. In fact, first tests were conducted but some developments are still required, particularly the complete computation of the derivative of the potential ϕ_t at the moving nodes where a Neumann condition is imposed.

Then, implementing a fully Lagrangian tracking method for the free surface nodes would be desirable. This is also of particular importance for nodes attached to body boundaries. The current semi-Lagrangian method can only describe a vertically moving marker. Many possible approaches exist to overcome this limitation. However, according to Hanssen (2019) using fully Lagrangian markers for the free surface increases the development of sawtooth instabilities (first discovered by Longuet-Higgins and Cokelet (1976)). In particular, they had to implement a five-point filter also used and described by Sun (2007) as their main 12^{th} Savitzky-Golay filter was not sufficient to maintain a good stability on some of the cases they studied.

Finally, even though it cannot be considered as an improvement of the method itself, the development of the latter within the OpenFOAM® framework would be interesting for multiple reasons. First, one could directly invoke the precise HPC interpolation - of spatial order 4 - during the coupling approaches. Then, it would ease the implementation of a strong coupling method. Last but not least, most of the mesh handling and utility functions for HPC were developed during this *Ph.D.*, but are far from equaling the optimizations that OpenFOAM® provide. This last reason is the root of why no CPU times concerning HPC were given in this study: the resolution of the BVP seems currently instantaneous compared to the mesh manipulations routine.

7.3.2 Domain and velocity decompositions

Several continuations of this project can be drawn and may require further investigations and developments. Most of them are common, between the domain coupling and velocity coupling approaches, and are thus discussed together hereafter.

Analysis of the effects of turbulence closure scheme and parameters First, it was here shown an overproduction of turbulence in the body vicinity. In turn, a reduction over - long - time of the predicted wave loads and kinematics is observed. It may be interesting to conduct a study focusing on the usage of others models, more adapted in close wall vicinity (for example $k - \omega$) to compare the build up of turbulent viscosity.

Assessment of maximal savings, modification of various parameters Another aspect would be to perform further tests by modifying other parameters, so as to evaluate the maximal gain that can be obtained. For example, mesh and time step independence studies could be interesting even though the extrapolation of the waveFoam results are thought to be adequate in this context: mesh and maximal CFL number were dependent on effects taking place in the body vicinity. Other parameters such as employed schemes or solvers can also be of prime interest in reducing the associated computational burden and find a more optimal setup, specifically for a coupled solver.

Extension to 3D cases OpenFOAM® is a 3 spatial dimensions (3D) toolbox. In order to perform a 2D simulation, one needs to define the 3D equivalent: the mesh is defined with only one cell "depth" (one cell in the suppressed spatial dimension), and the inactive faces are defined as `empty`. Thus, none of the developments presented in this thesis invoked the 2D hypothesis. While it was not tested in any case involving 3D effects, all the tested simulations were strictly speaking in 3D. As a matter of fact, in order to import the HPC results in an OpenFOAM® format (as an OpenFOAM® region), a 3D extrusion of our HPC results was implemented and performed.

With a 3D extension, one could wonder about the relative computational cost and the parallelization of the resolution. While confidence can be granted on the fact that the solving method itself would work with no modification in multi-threading, as only using similar functions and calls as the original OpenFOAM solver, the interpolation and communication part would probably require some work, especially if the potential results themselves were available as a parallel decomposition. It was neither done nor tested during this work and could represent a significant improvement for the application to 3D cases of practical interest.

Free surface simulation with RANS-VoF While not shown in the present manuscript, first tests of the coupling methods have been conducted to allow a free surface piercing body. In order to do so the free surface in the vicinity of the body is simulated by the VoF method available in the solver. The volume fraction α is imposed at the outer boundary following the obtained values in the potential domain and evolves inside the RANS domain in a classical manner. Corresponding boundary conditions were developed as well as modified boundaries for the complementary velocity and pressure that depends on the location (inside or outside water). However the method proved to be delicate and very sensitive to boundary conditions types. Because we are using a one-way coupling scheme, such issues could be expected, as a fundamental problem is present: a RANS model dissipate energy, thus directly matching the free surface elevation at both the inlet and outlet boundaries of the CFD mesh is possible but with only slight hope of yielding stable results. More complicated matching strategies could be setup to overcome this

difficulty. For example, Zhang (2018) applied relaxation techniques close to the outer boundaries of the viscous domain to allow for a smoother enforcement.

However, it is thought that this issue is inherent to one-way approaches and thus could only be fully resolved by extending to a two-way coupling.

Toward a two-way coupling With a two-way coupling approach, the turbulent flow also acts on the computation of the potential flow itself. This method would also ease the matching with the potential free surface, probably removing the need of invoking any smoothing. On the domain coupling framework, this is expected to be rather straightforward: the potential mesh could be reduced with its inner conditions being the outer conditions of the viscous solver. For the velocity coupling scheme, many solutions are possible. The first one is to use the method given by *e.g.* Edmund *et al.* (2011), and derive a more complex boundary condition on the body. Another possibility would be to match the free surfaces such that the BVP is resolved - not necessary at all time instants - using the RANS free surface topology and values.

Application with other external/internal models A turbulence modeling approach which seems to gain an always increasing attention in the recent years is the Large Eddy Simulation (LES) model. This method is already available in OpenFOAM® and one could use the routines developed during this work to interface the later with HPC external results. The interesting properties of LES, such as its good quality results and convergence in space behavior could resolve the turbulence problem altogether. This however, would come at the price of a significant increase of the associated computational burden.

Bibliography

Alessandrini, Bertrand, Pierre Ferrant, Lionel Gentaz, and Romain Luquet (2008). "Numerical Simulation of Ship Seakeeping by the SWENSE Approach". In: *Proceedings of the 10th International Ship Stability Workshop, Daejeon, Korea*, pp. 23–25.

Amores, Angel, Marta Marcos, Diego S. Carrió, and Lluís Gómez-Pujol (2020). "Coastal Impacts of Storm Gloria (January 2020) over the North-Western Mediterranean". In: *Natural Hazards and Earth System Sciences* 20.7, pp. 1955–1968.

Ansys, Aqwa (2013). "AQWA Theory Manual". In: *ed. Canonsburg, PA 15317, USA*.

Arai, Shinichi (1995). "Forces and Circulation of Horizontal Cylinders Submerged in Regular Waves". In: *The Fifth International Offshore and Polar Engineering Conference*. International Society of Offshore and Polar Engineers.

Bailey, Helen, Kate L. Brookes, and Paul M. Thompson (2014). "Assessing Environmental Impacts of Offshore Wind Farms: Lessons Learned and Recommendations for the Future". In: *Aquatic Biosystems* 10.1, p. 8. ISSN: 2046-9063. DOI: 10.1186/2046-9063-10-8.

Baker, Allison H., Elizabeth R. Jessup, and Tzanio V. Kolev (2009). "A Simple Strategy for Varying the Restart Parameter in GMRES(m)". en. In: *J. Comput. Appl. Math.* 230.2, pp. 751–761. ISSN: 0377-0427. DOI: 10.1016/j.cam.2009.01.009.

Bardazzi, Andrea, Claudio Lugni, Matteo Antuono, Giorgio Graziani, and Odd Magnus Faltinsen (2015). "Generalized HPC Method for the Poisson Equation". In: *J. Comput. Phys.* 299, pp. 630–648. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2015.07.026.

Bergdahl, Lars (2017). "Mooring Design for WECs". In: *Handbook of Ocean Wave Energy*, p. 159.

Bihs, Hans, Arun Kamath, Mayilvahanan Alagan Chella, Ankit Aggarwal, and Øivind A. Arntsen (2016). "A New Level Set Numerical Wave Tank with Improved Density Interpolation for Complex Wave Hydrodynamics". In: *Comput. Fluids* 140, pp. 191–208. ISSN: 0045-7930. DOI: 10.1016/j.compfluid.2016.09.012.

Bingham, Harry B. (2000). "A Hybrid Boussinesq-Panel Method for Predicting the Motion of a Moored Ship". In: *Coastal Engineering* 40.1, pp. 21–38.

Brown, Sheridan A., Deborah M. Greaves, Vanesa Magar, and Daniel C. Conley (2016). "Evaluation of Turbulence Closure Models under Spilling and Plunging Breakers in the Surf Zone". In: *Coastal Engineering* 114, pp. 177–193.

Campana, Emilio F. and Andrea Di Macio (1994). "Domain Decomposition in Free Surface Viscous Flows". In: *6th Intl Conf on Numerical Ship Hydrodynamics; 2-5 Aug 1993; Iowa, USA*.

Campana, Emilio F., Andrea Di Mascio, P.G. Esposito, and Francesco Lalli (1995). "Viscous-Inviscid Coupling in Free Surface Ship Flows". In: *Int. J. Numer. Meth. Fluids* 21.9, pp. 699–722. ISSN: 0271-2091, 1097-0363. DOI: 10.1002/fld.1650210902.

Chaplin, John R. (1984). "Nonlinear Forces on a Horizontal Cylinder beneath Waves". In: *J. Fluid Mech.* 147.-1, p. 449. ISSN: 0022-1120, 1469-7645. DOI: 10.1017/s0022112084002160.

Chen, L. F., Jun Zang, Andrew J. Hillis, Gerald CJ Morgan, and Andrew R. Plummer (2014). "Numerical Investigation of Wave–Structure Interaction Using OpenFOAM". In: *Ocean Engineering* 88, pp. 91–109.

Chen, Yang and Kevin J. Maki (2017). "A Velocity Decomposition Approach for Three-Dimensional Unsteady Flow". In: *European Journal of Mechanics-B/Fluids* 62, pp. 94–108.

Chen, Yang, Kevin J. Maki, and William J. Rosemurgy (2015). "A Velocity Decomposition Approach for Unsteady External Flow". In: *Volume 11: Prof. Robert f. Beck Honoring Symposium on Marine Hydrodynamics*. American Society of Mechanical Engineers. ISBN: 978-0-7918-5659-8. DOI: 10.1115/omae2015-42192.

Choi, Youngmyung, Benjamin Bouscasse, Sopheak Seng, Guillaume Ducrozet, Lionel Gentaz, and Pierre Ferrant (2018). "Generation of Regular and Irregular Waves in Navier-Stokes CFD Solvers by Matching with the Nonlinear Potential Wave Solution at the Boundaries". In: *ASME 2018 37th International Conference on Ocean, Offshore and Arctic Engineering*. American Society of Mechanical Engineers Digital Collection.

Christensen, Erik Damgaard, Henrik Bredmose, and Erik Asp Hansen (2009). "Transfer of Boussinesq Waves to a Navier-Stokes Solver: Application to Wave Loads on an Offshore Wind Turbine Foundation". In: *International Conference on Offshore Mechanics and Arctic Engineering*. Vol. 43444, pp. 917–926.

Clamond, Didier, Dorian Fructus, John Grue, and Øyvind Kristiansen (2005). "An Efficient Model for Three-Dimensional Surface Wave Simulations. Part II: Generation and Absorption". en. In: *J. Comput. Phys.* 205.2, pp. 686–705. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2004.11.038.

Colicchio, Giuseppina, Marilena Greco, and Odd Magnus Faltinsen (2006). "A BEM-Level Set Domain-Decomposition Strategy for Non-Linear and Fragmented Interfacial Flows". In: *International journal for numerical methods in engineering* 67.10, pp. 1385–1419.

Constant, Eddy, Julien Favier, Marcello Meldi, Philippe Meliga, and Eric Serre (2017). "An Immersed Boundary Method in OpenFOAM: Verification and Validation". In: *Computers & Fluids* 157, pp. 55–72.

Damian, S. Márquez (2012). *Description and Utilization of interFoam Multiphase Solver*. URL: http://infofich.unl.edu.ar/upload/3be0e16065026527477b4b948c4caa7523c8ea52.pdf.

Davidson, Josh and Ronan Costello (2020). "Efficient Nonlinear Hydrodynamic Models for Wave Energy Converter Design—A Scoping Study". In: *Journal of Marine Science and Engineering* 8.1, p. 35.

Dean, R.G. (1965). "Stream Function Representation of Nonlinear Ocean Waves". In: *J. Geophys. Res.* 70.18, pp. 4561–4572. ISSN: 0148-0227. DOI: 10.1029/jz070i018p04561.

Deshpande, Suraj S., Lakshman Anumolu, and Mario F. Trujillo (2012). "Evaluating the Performance of the Two-Phase Flow Solver interFoam". In: *Computational science & discovery* 5.1, p. 014016.

Devolder, Brecht, Pieter Rauwoens, and Peter Troch (2017). "Application of a Buoyancy-Modified k- SST Turbulence Model to Simulate Wave Run-up around a Monopile Subjected to Regular Waves Using OpenFOAM®". en. In: *Coastal Engineering* 125, pp. 81–94. ISSN: 0378-3839. DOI: 10.1016/j.coastaleng.2017.04.004.

Devolder, Brecht, Peter Troch, and Pieter Rauwoens (2018). "Performance of a Buoyancy-Modified k- and k- SST Turbulence Model for Simulating Wave Breaking under Regular Waves Using OpenFOAM®". en. In: *Coastal Engineering* 138, pp. 49–65. ISSN: 0378-3839. DOI: 10.1016/j.coastaleng.2018.04.011.

DNV GL, . (2018). *Offshore Standard DNVGL-OS-E301. Position Mooring*. Tech. rep. URL: https://scholar.google.com/scholar?oi=gsb40%5C&q=DNV%5C%20GL%5C%20(2018).%5C%20Offshore%5C%20Standard%5C%20DNVGL-OS-E301.%5C%20Position%5C%20Mooring.%5C&lookup=0%5C&hl=fr (visited on 08/27/2020).

Dommermuth, Douglas G, Eric A Schlageter, John C Talcott, Donald C Wyatt, and Evgeny A Novikov (1997). "Numerical Simulation of Bow Waves and Transom-Stern Flows". In: *APS Division of Fluid Dynamics Meeting Abstracts*. Vol. 1.

Ducrozet, Guillaume, Félicien Bonnefoy, David Le Touzé, and Pierre Ferrant (2012). "A Modified High-Order Spectral Method for Wavemaker Modeling in a Numerical Wave Tank". In: *Eur. J. Mech. B. Fluids* 34, pp. 19–34. ISSN: 0997-7546. DOI: 10.1016/j.euromechflu.2012.01.017.

Edmund, Deborah O, Kevin J Maki, and Robert F Beck (2011). "An Improved Viscous/Inviscid Velocity Decomposition Method". In: *26th International Workshop on Water Waves and Floating Bodies, Athens*. Citeseer.

Edmund, Deborah Osborn (2012). "A Velocity Decomposition Method for Efficient Numerical Computation of Steady External Flows". PhD thesis. The University of Michigan.

Edmund, Deborah Osborn, Kevin J. Maki, and Robert F. Beck (2013). "A Velocity-Decomposition Formulation for the Incompressible Navier–Stokes Equations". In: *Comput Mech* 52.3, pp. 669–680. ISSN: 0178-7675, 1432-0924. DOI: 10.1007/s00466-013-0839-6.

Eymard, Robert, Thierry Gallouët, and Raphaèle Herbin (2000). "Finite Volume Methods". en. In: *Handbook of Numerical Analysis*. Vol. 7. Solution of Equation in (Part 3), Techniques of Scientific Computing (Part 3). Elsevier, pp. 713–1018. DOI: 10.1016/S1570-8659(00)07005-8.

Fàbregas Flavià, Francesc, Cameron McNatt, François Rongère, Aurélien Babarit, and Alain H. Clément (2016). "Computation of the Diffraction Transfer Matrix and the Radiation Characteristics in the Open-Source BEM Code NEMOH". In: *Volume 6: Ocean Space Utilization; Ocean Renewable Energy*. American Society of Mechanical Engineers. ISBN: 978-0-7918-4997-2. DOI: 10.1115/omae2016-54130.

Fan, Wenyuan and Henryk Anglart (2019). "On the Closure Requirement for VOF Simulations with RANS Modeling". In: *arXiv:1911.09727 [physics]*. arXiv: 1911.09727 [physics]. URL: http://arxiv.org/abs/1911.09727 (visited on 08/06/2020).

Fedele, Francesco, Claudio Lugni, and Arun Chawla (2017). "The Sinking of the El Faro: Predicting Real World Rogue Waves during Hurricane Joaquin". In: *Sci Rep* 7.1, p. 11188. ISSN: 2045-2322. DOI: 10.1038/s41598-017-11505-5.

Feng, Xing and Wanqing Wu (2019). "Generation of Water Waves Using Momentum Source Wave-Maker Applied to a RANS Solver". In: *Mathematical Problems in Engineering* 2019, pp. 1–11. ISSN: 1024-123X, 1563-5147. DOI: 10.1155/2019/1308960.

Fenton, J.D. (1988). "The Numerical Solution of Steady Water Wave Problems". In: *Comput. Geosci.* 14.3, pp. 357–368. ISSN: 0098-3004. DOI: 10.1016/0098-3004(88)90066-0.

Fenton, John D. (1999). "Numerical Methods for Nonlinear Waves". In: *Advances in Coastal and Ocean Engineering*. WORLD SCIENTIFIC, pp. 241–324. ISBN: 978-981-02-3859-9. DOI: 10.1142/9789812797544_0005.

Ferrant, Pierre, Lionel Gentaz, Bertrand Alessandrini, and David Le Touzé (2003). "A Potential/RANSE Approach for Regular Water Wave Diffraction about 2-d Structures". In: *Ship Technology Research* 50.4, pp. 165–171. ISSN: 0937-7255. DOI: 10.1179/str.2003.50.4.004.

Fredsoe, Jorgen and B. Mutlu Sumer (1997). *Hydrodynamics around Cylindrical Structures*. Vol. 12. World Scientific.

Gentaz, Lionel, Romain Luquet, Bertrand Alessandrini, and Pierre Ferrant (2004). "Numerical Simulation of the 3D Viscous Flow around a Vertical Cylinder in Non-Linear Waves Using an Explicit Incident Wave Model". In: *23rd International Conference on Offshore Mechanics and Arctic Engineering, Volume 1, Parts a and b*. American Society of Mechanical Engineers. ASMEDC, pp. 157–163. ISBN: 0-7918-3743-2. DOI: 10.1115/omae2004-51098.

Gercken, Jens and Andreas Schmidt (2014). *Current Status of the European Oyster (Ostrea Edulis) and Possibilities for Restoration in the German North Sea 2014*. en. Tech. rep., p. 96.

Grilli, Stéphan T., Jeffrey C. Harris, and Nathanael Greene (2009). "Modeling of Wave-Induced Sediment Transport around Obstacles". In: *Coastal Engineering 2008: (In 5 Volumes)*. World Scientific, pp. 1638–1650.

Grilli, Stéphan T. and Juan Horrillo (1997). "Numerical Generation and Absorption of Fully Nonlinear Periodic Waves". In: *J. Eng. Mech.* 123.10, pp. 1060–1069. ISSN: 0733-9399, 1943-7889. DOI: 10.1061/(asce)0733-9399(1997)123:10(1060).

Guerber, E (2011). "Modélisation Numérique Des Interactions Non-Linéaires Entre Vagues et Structures Immergées, Appliquée à La Simulation de Systèmes Houlomoteurs". PhD thesis. Université Paris-Est.

Guerber, Etienne, Michel Benoit, Stephan T. Grilli, and Clément Buvat (2012). "A Fully Nonlinear Implicit Model for Wave Interactions with Submerged Structures in Forced or Free Motion". In: *Eng. Anal. Boundary Elem.* 36.7, pp. 1151–1163. ISSN: 0955-7997. DOI: 10.1016/j.enganabound.2012.02.005.

Hafez, Mohammed M., A. Shatalov, and M. Nakajima (2007). "Improved Numerical Simulations of Incompressible Flows Based on Viscous/Inviscid Interaction Procedures". In: *Comput. Fluids* 36.10, pp. 1588–1591. ISSN: 0045-7930. DOI: 10.1016/j.compfluid.2007.03.006.

Hafez, Mohammed M., A. Shatalov, and Essam M. Wahba (2006). "Numerical Simulations of Incompressible Aerodynamic Flows Using Viscous/Inviscid Interaction Procedures".

In: *Comput. Methods Appl. Mech. Eng.* 195.23-24, pp. 3110–3127. ISSN: 0045-7825. DOI: 10.1016/j.cma.2004.07.059.

Hague, C. H. and C. Swan (2009). "A Multiple Flux Boundary Element Method Applied to the Description of Surface Water Waves". en. In: *J. Comput. Phys.* 228.14, pp. 5111–5128. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2009.04.012.

Hanssen, Finn-Christian W. (2019). "Non-Linear Wave-Body Interaction in Severe Waves". PhD thesis. NTNU.

Hanssen, Finn-Christian W., Andrea Bardazzi, Claudio Lugni, and Marilena Greco (2017a). "Free-Surface Tracking in 2D with the Harmonic Polynomial Cell Method: Two Alternative Strategies". In: *Int J Numer Methods Eng* 113.2, pp. 311–351. ISSN: 0029-5981. DOI: 10.1002/nme.5615.

Hanssen, Finn-Christian W., Marilena Greco, and Odd Magnus Faltinsen (2017b). "Wave-Body Interaction with Overlapping Structured Grids in the HPC Method". In: *32nd IWWWFB, Dalian, China*.

Hanssen, Finn-Christian W., Marilena Greco, and Yanlin Shao (2015). "The Harmonic Polynomial Cell Method for Moving Bodies Immersed in a Cartesian Background Grid". In: *Volume 11: Prof. Robert f. Beck Honoring Symposium on Marine Hydrodynamics*. American Society of Mechanical Engineers. ISBN: 978-0-7918-5659-8. DOI: 10.1115/omae2015-41282.

Harris, Jeffrey C. and Stéphan T. Grilli (2010). "Coupling of NWT and Large-Eddy Simulation for Wave-Induced Sediment Transport". In: *The Twentieth International Offshore and Polar Engineering Conference*. International Society of Offshore and Polar Engineers.

— (2012). "A Perturbation Approach to Large Eddy Simulation of Wave-Induced Bottom Boundary Layer Flows". en. In: *Int. J. Numer. Meth. Fluids* 68.12, pp. 1574–1604. ISSN: 02712091. DOI: 10.1002/fld.2553.

Higuera, Pablo, Javier L. Lara, and Inigo J. Losada (2013). "Realistic Wave Generation and Active Wave Absorption for Navier–Stokes Models". In: *Coastal Eng.* 71, pp. 102–118. ISSN: 0378-3839. DOI: 10.1016/j.coastaleng.2012.07.002.

Hirt, C.W and B.D Nichols (1981). "Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries". In: *J. Comput. Phys.* 39.1, pp. 201–225. ISSN: 0021-9991. DOI: 10.1016/0021-9991(81)90145-5.

Hsiao, Yu, Chung-Liang Tsai, Yen-Lung Chen, Han-Lun Wu, and Shih-Chun Hsiao (2020). "Simulation of Wave-Current Interaction with a Sinusoidal Bottom Using OpenFOAM". In: *Appl. Ocean Res.* 94, p. 101998.

Hsu, Tai-Wen, Chih-Min Hsieh, Chin-Yen Tsai, and Shan-Hwei Ou (2015). "Coupling VOF/PLIC and Embedding Method for Simulating Wave Breaking on a Sloping Beach". In: *Journal of Marine Science and Technology* 23.4, pp. 498–507.

Hu, Zheng Zheng, Deborah Greaves, and Alison Raby (2016). "Numerical Wave Tank Study of Extreme Waves and Wave-Structure Interaction Using OpenFoam®". In: *Ocean Eng.* 126, pp. 329–342. ISSN: 0029-8018. DOI: 10.1016/j.oceaneng.2016.09.017.

Igwemezie, Victor, Ali Mehmanparast, and Athanasios Kolios (2019). "Current Trend in Offshore Wind Energy Sector and Material Requirements for Fatigue Resistance Improvement in Large Wind Turbine Support Structures–A Review". In: *Renewable and Sustainable Energy Reviews* 101, pp. 181–196.

Islam, H. and C. Guedes Soares (2018). "Prediction of Ship Resistance in Head Waves Using OpenFOAM". In: *Maritime Transportation and Harvesting of Sea Resources*. Taylor & Francis Group.

Issa, Raad I. (1986). "Solution of the Implicitly Discretised Fluid Flow Equations by Operator-Splitting". In: *Journal of computational physics* 62.1, pp. 40–65.

Jacobsen, Niels G., David R. Fuhrman, and Jørgen Fredsøe (2011). "A Wave Generation Toolbox for the Open-Source CFD Library: Openfoam®". In: *Int. J. Numer. Meth. Fluids* 70.9, pp. 1073–1088. ISSN: 0271-2091. DOI: 10.1002/fld.2726.

Janssen, Christian F., Manfred Krafczyk, and Stéphan T. Grilli (2010). "Modeling of Wave Breaking AndWave-Structure Interactions By Coupling of Fully Nonlinear Potential Flow And Lattice-Boltzmann Models". In: *The Twentieth International Offshore and Polar Engineering Conference*. International Society of Offshore and Polar Engineers.

Jasak, Hrvoje (1996). "Error Analysis and Estimation for the Finite Volume Method with Applications to Fluid Flows." PhD thesis. Imperial College London (University of London).

Jiao, Jialong and Songxing Huang (2020). "CFD Simulation of Ship Seakeeping Performance and Slamming Loads in Bi-Directional Cross Wave". In: *JMSE* 8.5, p. 312. ISSN: 2077-1312. DOI: 10.3390/jmse8050312.

Kim, CH, AH Clement, and K Tanizawa (1999). "Recent Research and Development of Numerical Wave Tanks-a Review". In: *Int. J. Offshore Polar Eng.* 9.04.

Kim, Kunho, Ana I. Sirviente, and Robert F. Beck (2005). "The Complementary RANS Equations for the Simulation of Viscous Flows". In: *Int. J. Numer. Meth. Fluids* 48.2, pp. 199–229. ISSN: 0271-2091, 1097-0363. DOI: 10.1002/fld.892.

Kim, Sung-Yong, Kyung-Mi Kim, Jong-Chun Park, Gyu-Mok Jeon, and Ho-Hwan Chun (2016). "Numerical Simulation of Wave and Current Interaction with a Fixed Offshore Substructure". In: *Int. J. Nav. Archit. Ocean Eng.* 8.2, pp. 188–197. ISSN: 2092-6782. DOI: 10.1016/j.ijnaoe.2016.02.002.

Kristiansen, Trygve and Odd Magnus Faltinsen (2012). "Gap Resonance Analyzed by a New Domain-Decomposition Method Combining Potential and Viscous Flow DRAFT". In: *Appl. Ocean Res.* 34, pp. 198–208.

Kristiansen, Trygve, Thomas Sauder, and Reza Firoozkoohi (2013). "Validation of a Hybrid Code Combining Potential and Viscous Flow with Application to 3D Moonpool". In: *International Conference on Offshore Mechanics and Arctic Engineering*. Vol. 55430. American Society of Mechanical Engineers, V009T12A029.

Kumar, G. Manoj, V. Sriram, and I. Didenkulova (2020). "A Hybrid Numerical Model Based on FNPT-NS for the Estimation of Long Wave Run-Up". In: *Ocean Engineering* 202, p. 107181.

Lacaze, Jean-Baptiste (2015). "Etude Expérimentale et Numérique Du Couplage Des Phénomènes Aérodynamiques et Hydrodynamiques Sur Une Éolienne Offshore Flottante". PhD thesis. Aix-Marseille.

Larsen, Bjarke Eltard and David R. Fuhrman (2018). "On the Over-Production of Turbulence beneath Surface Waves in Reynolds-Averaged Navier–Stokes Models". In: *J. Fluid Mech.* 853, pp. 419–460. ISSN: 0022-1120, 1469-7645. DOI: 10.1017/jfm.2018.577.

Larsen, Bjarke Eltard, David R. Fuhrman, and Johan Roenby (2019). "Performance of interFoam on the Simulation of Progressive Waves". In: *Coastal Engineering Journal* 61.3, pp. 380–400.

Launder, Brian Edward and B. I. Sharma (1974). "Application of the Energy-Dissipation Model of Turbulence to the Calculation of Flow near a Spinning Disc". In: *Letters in heat and mass transfer* 1.2, pp. 131–137.

Le Méhauté, Bernard (1976). "An Introduction to Water Waves". In: *An Introduction to Hydrodynamics and Water Waves*. Springer, pp. 197–211.

Lee, Chang-Ho (1995). *WAMIT Theory Manual*. Massachusetts Institute of Technology, Department of Ocean Engineering.

Li, Qian (2017). "A Hybrid Model Based on Functional Decomposition for Vortex Shedding Simulations". PhD Thesis. City, Universtiy of London.

Li, Qian, Jinghua Wang, Shiqiang Yan, Jiaye Gong, and Qingwei Ma (2018a). "A Zonal Hybrid Approach Coupling FNPT with OpenFOAM for Modelling Wave-Structure Interactions with Action of Current". In: *Ocean Systems Engineering* 8.4, pp. 381–407.

Li, Qian, Shiqiang Yan, Jinghua Wang, Qingwei Ma, Zhihua Xie, and V. Sriram (2018b). "Numerical Simulation of Focusing Wave Interaction with FPSO-like Structure Using FNPT-NS Solver". In: *The 28th International Ocean and Polar Engineering Conference*. International Society of Offshore and Polar Engineers.

Li, Yong and Mian Lin (2010). "Hydrodynamic Coefficients Induced by Waves and Currents for Submerged Circular Cylinder". In: *Procedia Engineering* 4, pp. 253–261.

Li, Zhaobin, Benjamin Bouscasse, Guillaume Ducrozet, Lionel Gentaz, and Pierre Ferrant (2018c). "Challenges in Developing a SWENSE Two-Phase CFD Solver for Complex Wave Conditions". In: *Proceedings of the 33rd International Workshop on Water Waves and Floating Bodies (IWWWFB), Guidel-Plages, France*, pp. 4–7.

Li, Zhaobin, Benjamin Bouscasse, Guillaume Ducrozet, Lionel Gentaz, David Le Touzé, and Pierre Ferrant (2020). "Spectral Wave Explicit Navier-Stokes Equations for Wave-Structure Interactions Using Two-Phase Computational Fluid Dynamics Solvers". In: *arXiv preprint arXiv:2005.12716*.

Li, Zhaobin, Benjamin Bouscasse, Lionel Gentaz, Guillaume Ducrozet, and Pierre Ferrant (2018d). "Progress in Coupling Potential Wave Models and Two-Phase Solvers with the SWENSE Methodology". In: *International Conference on Offshore Mechanics and Arctic Engineering*. Vol. 51302. American Society of Mechanical Engineers, V009T13A027.

Li, Zhaobin, Lionel Gentaz, Guillaume Ducrozet, and Pierre Ferrant (2017). "Calculation of High-Order Wave Loads on a Vertical Circular Cylinder Using the SWENSE Method". In: *32th International Workshop on Water Waves and Floating Bodies*.

Liang, Hui, Odd Magnus Faltinsen, and Yan-Lin Shao (2015). "Application of a 2D Harmonic Polynomial Cell (HPC) Method to Singular Flows and Lifting Problems". In: *Appl. Ocean Res.* 53, pp. 75–90. ISSN: 0141-1187. DOI: 10.1016/j.apor.2015.07.011.

Liang, Hui, Harrif Santo, Yanlin Shao, Yun Zhi Law, and Eng Soon Chan (2020). "Liquid Sloshing in an Upright Circular Tank under Periodic and Transient Excitations". In: *Physical Review Fluids* 5.8, p. 084801.

Lighthill, M. J. (1958). "On Displacement Thickness". In: *Journal of Fluid Mechanics* 4.4, pp. 383–392.

Liu, Fangqing (2016). "A Thorough Description of How Wall Functions Are Implemented in OpenFOAM". In: *Proceedings of CFD with OpenSource Software*, pp. 1–33.

Lock, R. C. and B. R. Williams (1987). "Viscous-Inviscid Interactions in External Aerodynamics". In: *Progress in Aerospace Sciences* 24.2, pp. 51–171.

Longuet-Higgins, Michael S and ED Cokelet (1976). "The Deformation of Steep Surface Waves on Water. I. A Numerical Method of Computation". In: *Proceedings of the Royal Society of London a: Mathematical, Physical and Engineering Sciences*. Vol. 350. The Royal Society, pp. 1–26.

Luquet, Romain, Guillaume Ducrozet, Lionel Gentaz, Pierre Ferrant, and Bertrand Alessandrini (2007a). "Applications of the SWENSE Method to Seakeeping Simulations in Irregular Waves". In: *Proc of the 9th Int. Conf. on Num. Ship Hydro*.

Luquet, Romain, Pierre Ferrant, Bertrand Alessandrini, Guillaume Ducrozet, and Lionel Gentaz (2007b). "Simulation of a TLP in Waves Using the SWENSE Scheme". In: *The Seventeenth International Offshore and Polar Engineering Conference*. International Society of Offshore and Polar Engineers.

Luquet, Romain, Lionel Gentaz, Pierre Ferrant, and Bertrand Alessandrini (2004). "Viscous Flow Simulation Past a Ship in Waves Using the SWENSE Approach". In: *Proceedings of the 25th Symposium on Naval Hydrodynamics*.

Ma, Q. W. and S. Yan (2006). "Quasi ALE Finite Element Method for Nonlinear Water Waves". en. In: *J. Comput. Phys.* 212.1, pp. 52–72. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2005.06.014.

Ma, Shaojun, Finn-Christian W. Hanssen, M.A. Siddiqui, Marilena Greco, and Odd Magnus Faltinsen (2017). "Local and Global Properties of the Harmonic Polynomial Cell Method: In-Depth Analysis in Two Dimensions". In: *Int J Numer Methods Eng* 113.4, pp. 681–718. ISSN: 0029-5981. DOI: 10.1002/nme.5631.

Mayer, Stefan and Per A. Madsen (2001). "Simulation of Breaking Waves in the Surf Zone Using a Navier-Stokes Solver". In: *Coastal Engineering 2000*, pp. 928–941.

Menter, Florian R. (1994). "Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications". In: *AIAA journal* 32.8, pp. 1598–1605.

Molin, B. and J. H. Ferziger (2003). *Hydrodynamique Des Structures Offshore.(French)*. American Society of Mechanical Engineers Digital Collection.

Monroy, Charles, Guillaume Ducrozet, P. Roux de Reilhac, Lionel Gentaz, Pierre Ferrant, and Bertrand Alessandrini (2009). "RANS Simulations of Ship Motions in Regular and Irregular Head Seas Using the SWENSE Method". In: *The Nineteenth International Offshore and Polar Engineering Conference*. International Society of Offshore and Polar Engineers.

Monti, Alessandro and Beatriz Martinez Romera (2020). "Fifty Shades of Binding: Appraising the Enforcement Toolkit for the EU's 2030 Renewable Energy Targets". In: *Review of European, Comparative & International Environmental Law*.

Morino, L. (1986). "Helmholtz Decomposition Revisited: Vorticity Generation and Trailing Edge Condition - Part 1: Incompressible Flows". In: *Comput. Mech.* 1.1, pp. 65–90. ISSN: 0178-7675, 1432-0924. DOI: 10.1007/bf00298638.

— (1994). "Toward a Unification of Potential and Viscous Aerodynamics: Boundary Integral Formulation". In: *Applied Mathematics in Aerospace Science and Engineering*. Springer US, pp. 49–79. ISBN: 978-1-4757-9261-4. DOI: 10.1007/978-1-4757-9259-1_5.

Morino, L., F. Salvatore, and M. Gennaretti (1999). "A New Velocity Decomposition for Viscous Flows: Lighthill's Equivalent-Source Method Revisited". In: *Comput. Methods Appl. Mech. Eng.* 173.3-4, pp. 317–336. ISSN: 0045-7825. DOI: 10.1016/s0045-7825(98)00289-8.

Morison, J. R., J. W. Johnson, and S. A. Schaaf (1950). "The Force Exerted by Surface Waves on Piles". In: *Journal of Petroleum Technology* 2.05, pp. 149–154.

Moukalled, Fadl, L. Mangani, and Marwan Darwish (2016). *The Finite Volume Method in Computational Fluid Dynamics*. Vol. 6. Springer.

Musial, Walt (2007). "Offshore Wind Electricity: A Viable Energy Option for the Coastal United States". In: *Marine Technology Society Journal* 41.3, pp. 32–43.

Musiedlak, Pierre-Henri (2019). "Numerical Modelling of Responses of Offshore Wave Energy Converters in Extreme Waves". en. Thesis. University of Plymouth. URL: https://pearl.plymouth.ac.uk/handle/10026.1/15115 (visited on 08/03/2020).

O'Reilly, Christopher, Stéphan T. Grilli, Jason Dahl, Christian F. Janssen, Amir Banari, J. J. Schock, and Micha Uberrueck (2015). "A Hybrid Naval Hydrodynamic Scheme Based on an Efficient Lattice Boltzmann Method Coupled to a Potential Flow Solver". In: *13th International Conference on Fast Sea Transportation, Washington, DC, Sept*, pp. 1–4.

Oggiano, Luca, Fabio Pierella, Tor Anders Nygaard, Jacobus De Vaal, and Emile Arens (2017). "Reproduction of Steep Long Crested Irregular Waves with CFD Using the VOF Method". In: *Energy Procedia* 137, pp. 273–281. ISSN: 1876-6102. DOI: 10.1016/j.egypro.2017.10.351.

Ogilvie, T. Francis (1963). "First- and Second-Order Forces on a Cylinder Submerged under a Free Surface". In: *J. Fluid Mech.* 16.03, p. 451. ISSN: 0022-1120, 1469-7645. DOI: 10.1017/s0022112063000896.

Osher, Stanley and James A Sethian (1988). "Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations". In: *J. Comput. Phys.* 79.1, pp. 12–49. ISSN: 0021-9991. DOI: 10.1016/0021-9991(88)90002-2.

Patil, Akshay (2019). "Numerical Investigation of Nearshore Wave Transformation and Surf Zone Hydrodynamics". URL: https://repository.tudelft.nl/islandora/object/uuid:1b6037cf-6cb9-4ebf-b9c2-25e7fa6cf575.

Philippe, Maxime, Adrien Combourieu, Christophe Peyrard, Fabien Robaux, Gérard Delhommeau, and Aurélien Babarit (2015). "Introducing Second Order Low Frequency Loads in the Open-Source Boundary Element Method Code Nemoh". In: *EWTEC*. Proceedings of the 11th European Wave and Tidal Energy Conference. Nantes, France. URL: https://hal.archives-ouvertes.fr/hal-01198807.

Pinkster, Johannes Albert (1980). "Low Frequency Second Order Wave Exciting Forces on Floating Structures". PhD thesis. TU Delft.

Prandtl, Ludwig (1904). "Über Flussigkeitsbewegung Bei Sehr Kleiner Reibung". In: *Verhandl. III, Internat. Math.-Kong., Heidelberg, Teubner, Leipzig, 1904*, pp. 484–491.

Quéméré, P. and P. Sagaut (2002). "Zonal Multi-Domain RANS/LES Simulations of Turbulent Flows". In: *International journal for numerical methods in fluids* 40.7, pp. 903–925.

Quéméré, Patrick, Pierre Sagaut, and Vincent Couailler (2001). "A New Multi-Domain/Multi-Resolution Method for Large-Eddy Simulation". In: *International journal for numerical methods in fluids* 36.4, pp. 391–416.

Reliquet, Gabriel, Aurélien Drouet, Pierre-Emmanuel Guillerm, Erwan Jacquin, Lionel Gentaz, and Pierre Ferrant (2013). "Simulation of Wave-Body Interaction Using a Single-Phase Level Set Function in the Swense Method". In: *International Conference on Offshore Mechanics and Arctic Engineering*. Vol. 55416. American Society of Mechanical Engineers, V007T08A065.

Reliquet, Gabriel, Marie Robert, Lionel Gentaz, and Pierre Ferrant (2019). "Simulations de l'interaction Entre Le Catamaran Delft 372 et La Houle à l'aide Du Couplage SWENSE-Level Set". In: *La Houille Blanche* 5-6, pp. 59–66.

Riahi, H., Marcello Meldi, Julien Favier, Eric Serre, and E. Goncalves (2018). "A Pressure-Corrected Immersed Boundary Method for the Numerical Simulation of Compressible Flows". In: *Journal of Computational Physics* 374, pp. 361–383.

Rienecker, M. M. and J. D. Fenton (1981). "A Fourier Approximation Method for Steady Water Waves". In: *Journal of fluid mechanics* 104, pp. 119–137.

Robaux, Fabien and Michel Benoit (2018). "Modeling Nonlinear Wave-Body Interaction with the Harmonic Polynomial Cell Method Combined with the Immersed Boundary Method on a Fixed Grid". In: *Proc. 32th International Workshop on Water Waves and Floating Bodies, Guidel-Plages, France, April 4 to 7*.

— (2020). "Development and Validation of a Numerical Wave Tank Based on the Harmonic Polynomial Cell and Immersed Boundary Methods to Model Nonlinear Wave-Structure Interaction". In: *arXiv:2009.08937 [physics]*. arXiv: 2009.08937 [physics]. URL: http://arxiv.org/abs/2009.08937 (visited on 09/21/2020).

Rosemurgy, W., Deborah Osborn Edmund, Kevin J. Maki, and Robert F. Beck (2012). "A Velocity Decomposition Approach for Steady Free-Surface Flow". In: *29th Symposium on Naval Hydrodynamics, Gothenburg, Sweden*.

Saad, Yousef (2003). *Iterative Methods for Sparse Linear Systems*. Vol. 82. Society for Industrial and Applied Mathematics. ISBN: 978-0-89871-534-7. DOI: 10.1137/1.9780898718003.

Shao, Yan-Lin and Odd M. Faltinsen (2012a). "Solutions of Nonlinear Free Surface-Body Interaction with a Harmonic Polynomial Cell Method". In: *Proc. 27th International Workshop on Water Waves and Floating Bodies*.

Shao, Yan-Lin and Odd Magnus Faltinsen (2012b). "Towards Efficient Fully-Nonlinear Potential-Flow Solvers in Marine Hydrodynamics". In: *Volume 4: Offshore Geotechnics; Ronald w. Yeung Honoring Symposium on Offshore and Ship Hydrodynamics*. American Society of Mechanical Engineers, pp. 369–380. ISBN: 978-0-7918-4491-5. DOI: 10.1115/omae2012-83319.

— (2014a). "A Harmonic Polynomial Cell (HPC) Method for 3D Laplace Equation with Application in Marine Hydrodynamics". In: *J. Comp. Phys* 274, pp. 312–332.

— (2014b). "Fully-Nonlinear Wave-Current-Body Interaction Analysis by a Harmonic Polynomial Cell Method". In: *J. Offshore Mech. Arct. Eng.* 136.3, p. 031301. ISSN: 0892-7219, 1528-896X. DOI: 10.1115/1.4026960.

Siddiqui, Mohd Atif, Marilena Greco, Giuseppina Colicchio, and Odd Magnus Faltinsen (2018). "Validation of Damaged Ship Hydrodynamics by a Domain Decomposition

Approach Using the Harmonic Polynomial Cell Method and OpenFOAM". In: *Proceedings of 33rd International Workshop on Water Waves and Floating Bodies*. Scolan.

Siddiqui, Mohd Atif, Marilena Greco, Claudio Lugni, and Odd Magnus Faltinsen (2019). "Experimental Studies of a Damaged Ship Section in Forced Heave Motion". In: *Applied Ocean Research* 88, pp. 254–274.

— (2020). "Experimental Studies of a Damaged Ship Section in Beam Sea Waves". In: *Appl. Ocean Res.* 97, p. 102090.

Sobey, Rodney J. (1989). "Variations on Fourier Wave Theory". In: *Int. J. Numer. Meth. Fluids* 9.12, pp. 1453–1467. ISSN: 0271-2091, 1097-0363. DOI: 10.1002/fld.1650091203.

Spalart, Philippe and Steven Allmaras (1992). "A One-Equation Turbulence Model for Aerodynamic Flows". In: *30th Aerospace Sciences Meeting and Exhibit*, p. 439.

Sriram, V., Q. W. Ma, and T. Schlurmann (2014). "A Hybrid Method for Modelling Two Dimensional Non-Breaking and Breaking Waves". In: *Journal of computational physics* 272, pp. 429–454.

Strand, Ida M. and Odd Magnus Faltinsen (2019). "Linear Wave Response of a 2D Closed Flexible Fish Cage". In: *J. Fluids Struct.* 87, pp. 58–83. ISSN: 0889-9746. DOI: 10.1016/j.jfluidstructs.2019.03.005.

Sun, Hui (2007). "A Boundary Element Method Applied to Strongly Nonlinear Wave-Body Interaction Problems". In:

Sussman, Mark, Peter Smereka, and Stanley Osher (1994). "A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow". In:

Tamura, Tetsuro and Tetsuya Miyagi (1999). "The Effect of Turbulence on Aerodynamic Forces on a Square Cylinder with Various Corner Shapes". en. In: *Journal of Wind Engineering and Industrial Aerodynamics* 83.1, pp. 135–145. ISSN: 0167-6105. DOI: 10.1016/S0167-6105(99)00067-7.

Tanizawa, Katsuji (2000). "The State of the Art on Numerical Wave Tank". In: *Proceedings of 4th Osaka Colloquium on Seakeeping Performance of Ships 2000*, pp. 95–114.

Tavassoli, Armin and MH Kim (2001). "Interactions of Fully Nonlinear Waves with Submerged Bodies by a 2D Viscous NWT". In: *The Eleventh International Offshore and Polar Engineering Conference*. International Society of Offshore and Polar Engineers.

Tong, Chao, Yanlin Shao, Finn-Christian W. Hanssen, Ye Li, Bin Xie, and Zhiliang Lin (2019). "Numerical Analysis on the Generation, Propagation and Interaction of Solitary Waves by a Harmonic Polynomial Cell Method". en. In: *Wave Motion* 88, pp. 34–56. ISSN: 0165-2125. DOI: 10.1016/j.wavemoti.2019.01.007.

Troen, I. B. and Erik Lundtang Petersen (1989). "European Wind Atlas. Published for the Commission of the European Communities, Directorate General for Science". In: *Research and Development (Brussels, Belgium) by Risöo National Laboratory, Roskilde, Denmark*.

Tsai, Ching-Piao and Dong-Sheng Jeng (1994). "Numerical Fourier Solutions of Standing Waves in Finite Water Depth". In: *Appl. Ocean Res.* 16.3, pp. 185–193. ISSN: 0141-1187. DOI: 10.1016/0141-1187(94)90028-0.

Venugopal, Vengatesan (2002). "Hydrodynamic Force Coefficients for Rectangular Cylinders in Waves and Currents". PhD Thesis. University of Glasgow.

Venugopal, Vengatesan, Kamlesh S. Varyani, and Nigel D.P. Barltrop (2006). "Wave Force Coefficients for Horizontally Submerged Rectangular Cylinders". In: *Ocean Eng.* 33.11-12, pp. 1669–1704. ISSN: 0029-8018. DOI: 10.1016/j.oceaneng.2005.09.007.

Vukčević, Vuko (2016). "Numerical Modelling of Coupled Potential and Viscous Flow for Marine Applications-in Preparation". PhD Thesis. Ph. D. thesis. Faculty of Mechanical Engineering and Naval Architecture …

Vukčević, Vuko, Hrvoje Jasak, and Šime Malenica (2016a). "Decomposition Model for Naval Hydrodynamic Applications, Part I: Computational Method". In: *Ocean Engineering* 121, pp. 37–46.

— (2016b). "Decomposition Model for Naval Hydrodynamic Applications, Part II: Verification and Validation". In: *Ocean engineering* 121, pp. 76–88.

Vuorinen, V., J.-P. Keskinen, Christophe Duwig, and B. J. Boersma (2014). "On the Implementation of Low-Dissipative Runge–Kutta Projection Methods for Time Dependent Flows Using OpenFOAM®". In: *Computers & Fluids* 93, pp. 153–163.

Wang, Junxian, Shiqiang Yan, Qingwei Ma, Jinghua Wang, Zhihua Xie, and Sarah Marran (2020). "Numerical Simulation of Focused Wave Interaction with WEC Models Using qaleFOAM". In: *Proceedings of the Institution of Civil Engineers-Engineering and Computational Mechanics*, pp. 1–36.

Wei, Yanji, Thomas Abadie, and Frederic Dias (2017). "A Cost-Effective Method for Modelling Wave-OWSC Interaction". In: *International Journal of Offshore and Polar Engineering* 27.04, pp. 366–373.

Wilcox, David C. (1998). *Turbulence Modeling for CFD*. Vol. 2. DCW industries La Canada, CA.

Windt, Christian, Josh Davidson, Pál Schmitt, and John Ringwood (2019). "On the Assessment of Numerical Wave Makers in CFD Simulations". In: *JMSE* 7.2, p. 47. ISSN: 2077-1312. DOI: 10.3390/jmse7020047.

Wu, C. S., D. L. Young, and C. L. Chiu (2013). "Simulation of Wave–Structure Interaction by Hybrid Cartesian/Immersed Boundary and Arbitrary Lagrangian–Eulerian Finite-Element Method". en. In: *J. Comput. Phys.* 254, pp. 155–183. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2013.07.014.

Yan, S. and Q. W. Ma (2007). "Numerical Simulation of Fully Nonlinear Interaction between Steep Waves and 2D Floating Bodies Using the QALE-FEM Method". en. In: *J. Comput. Phys.* 221.2, pp. 666–692. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2006.06.046.

Yan, Shiqiang, Jinghua Wang, Junxian Wang, Qingwei Ma, and Zhihua Xie (2019). "Numerical Simulation of Wave Structure Interaction Using QaleFOAM". In: *The 29th International Ocean and Polar Engineering Conference*. International Society of Offshore and Polar Engineers.

Zakharov, V.E. (1968). "Stability of Periodic Waves of Finite Amplitude on the Surface of a Deep Fluid". In: *J Appl Mech Tech Phys* 9.2, pp. 190–194. ISSN: 0021-8944, 1573-8620. DOI: 10.1007/bf00913182.

Zhang, X. (2018). "Multi-Model Method for Simulating 2D Surface-Piercing Wave-Structure Interactions". en. Doctoral. City, Universtiy of London. URL: https://openaccess.city.ac.uk/id/eprint/19772/ (visited on 08/26/2020).

Zhu, Wenbo, Marilena Greco, and Yanlin Shao (2017). "Improved HPC Method for Nonlinear Wave Tank". In: *Int. J. Nav. Archit. Ocean Eng.* 9.6, pp. 598–612. ISSN: 2092-6782. DOI: 10.1016/j.ijnaoe.2017.03.009.

List of Figures

1.1 (a) Growth in size and power of wind turbine (Igwemezie *et al.*, 2019). (b): European offshore wind resource (Troen and Petersen, 1989). 2

1.2 (a): bathymetry dependent solutions and associated ressource estimate by depth in the US (Musial, 2007). (b): existing solutions and proposed projects for floating offshore wind turbines (FOWT) (Igwemezie *et al.*, 2019). 3

1.3 Examples of experiment of floating wind turbine (Lacaze, 2015). 3

1.4 Examples of numerical simulation of floating wind turbines. 4

1.5 Validity domains of various wave theories. h is mean water depth, τ the wave period, H the wave height and g the gravitational acceleration. From wikimedia, after Le Méhauté (1976). 5

1.6 Dominant effect(s) of the wave-structure interaction depending on the wave height H, the wavelength λ and the characteristic dimension of the object D. From Bergdahl (2017). 7

1.7 Schematics of a body submitted to regular water waves. 11

2.1 Definition sketch of 9-node macro-cell used for the HPC method, with local numbering of the nodes 19

2.2 Schematic representation of the immersed free surface in a fixed grid . . . 23

2.3 Schematic representation of the immersed free surface and body below the free surface. A circle means an "interpolation" node (described in the text) while the colors are used to identify particular nodes on the two grids, as indicated in the legend. 28

2.4 Schematic representation of the background free surface and the fitted free surface. A matching is enforced between the two free surfaces. The difference between the two free surfaces is here exaggerated for clarity. . . 30

2.5 Schematic representation of a sharp coner (body in red) 31

3.1 Schematic representation of the nonlinear standing wave with steepness $H/\lambda = 10\%$ at $t = 0$. 37

3.2 L_2 error on η on the nonlinear standing wave case at four time instants ($t/T = 1$, 10, 50 and 100) as a function of the spatial and temporal discretizations. The color scale indicates the $L_2(\eta)$ error respective to the theoretical solution by Tsai and Jeng (1994). See text for explanations on the significance of the markers shapes. 38

3.3 Convergence of the L_2 error on η (crosses) with respect to the temporal discretization at two different physical times: $t/T = 1$ (left panel) and $t/T = 100$ (right panel). The spatial discretization is fixed for a given line. Solid lines represent power regression of the error in the "linear range", the computed power is reported in the legend of the fitted straight lines. 40
3.4 Convergence of the error on wave period and amplitude of the wave elevation at $x/\lambda = 0.5$ for $N_x = 30$. The $L_2(\eta)$ error -combination of both- is also added. 41
3.5 Convergence of the L_2 error on η in mesh refinement, with temporal discretization fixed. Solid lines correspond to the power regression of the error. 42
3.6 Schematic representation of the numerical set-up inspired from Chaplin (1984). Note that the mesh fitted to the cylinder is not represented in this figure. 44
3.7 Amplitudes of the various harmonic components of the vertical load on the horizontal circular cylinder. Current results (crosses) compared to numerical simulations from Guerber (2011) (lines with diamonds) and experiments from Chaplin (1984) (empty circles). The amplitude of the first order harmonic based on linear prediction is added as well as a third order model line, to compare with the evolution of the amplitude of the third order harmonic. 44
3.8 Photograph of the experimental setup of the rectangular barge in the wave flume close to the body. 46
3.9 Overview of the conducted experiments in the $(T, H/\lambda)$ plane. 46
3.10 Comparisons of the free surface elevations recorded at different positions: HPC (dashed lines), experimental wave gauges (solid lines) and extracted elevation from the experimental video (circle markers). Waves gauges are divided in three groups (corresponding to subfigures): Incident waves upstream (subfig 1), Front and rear run-up (subfig 2), transmitted waves (downstream waves, subfig 3). Case 30: $T = 1.1$ s, $H/\lambda = 4.8\%$. 47
3.11 Comparisons of the free surface elevations recorded at different positions: HPC (dashed lines), experimental wave gauges (solid lines) and extracted elevation from the experimental video (circle markers). Waves gauges are divided in three groups (corresponding to subfigures): Incident waves upstream (subfig 1), Front and rear run-up (subfig 2), transmitted waves (downstream waves, subfig 3). Case 21: $T = 1.1$ s, $H/\lambda = 1.2\%$. 48
3.12 Case 30 $T = 1.1$ s, $H/\lambda = 4.8\%$. Comparison of the computed free surface elevation in front of the rectangular barge with the experiment at six different time instants. Incident waves come from the left. 49
3.13 Case 21 and 30 $T = 1.1s$, $H/\lambda = 1.2\%$ and $H/\lambda = 4.8\%$. Time series of the hydrodynamic loads on the barge: experiments, linear results and HPC simulations. 50
3.14 $T = 1.1$ s, Amplitudes of harmonics of the vertical and horizontal loads on the barge. HPC models (dashed lines), experimental results (dots) and linear transfer functions (solid orange line). A power function of order 3 is added to compare the increase rate (solid purple line). 51

3.15 Geometric representation of the test case. Note that only the wavelength, wave elevation and the water depth are not represented at scale 52
3.16 Inertia and drag coefficient in the two directions of the Loads obtained with the HPC method with different wave periods. The L_2 norm of the error between the Morison model and the real loads is also shown. 55
3.17 Case $H/\lambda = 3.5\%$, $T = 2$ s. Horizontal and vertical loads predicted by HPC, and their reconstruction with a Morison model. Kinematics used for reconstruction, *i.e.* predicted by the wave model (stream function) at $z_c = -0.82$ m. Associated reconstruction errors $(L_2(F_x), L_2(F_z)) = (2.4\%, 3.9\%)$. 57
3.18 Case $H/\lambda = 5.5\%$, $T = 2.3$ s. Horizontal and vertical loads predicted by HPC, and their reconstruction with a Morison model. Kinematics used for reconstruction, *i.e.* predicted by the wave model (stream function) at $z_c = -0.82$ m. Associated reconstruction errors $(L_2(F_x), L_2(F_z)) = (5.0\%, 9.0\%)$. 58
3.19 Comparison of the obtained error $L_2(\eta)$ (see eq. (3.1)) with the fitted mesh method (right panels) and the IBM free surface (left panels, recalling fig. 3.2) on the standing wave. $H/\lambda = 0.1$. Top and bottom panels are the errors after $t/T = 10$ and $t/T = 100$ respectively. 60
3.20 Comparison of the convergence with CFL number of the $L_{2\eta}$ error obtained with the fitted mesh and immersed boundary methods, for two different discretizations ($N_x = 60$ and 80) 60
3.21 Screenshots of the coarsest fitted mesh tested as well as one of the finest in the vicinity of the body 61
3.22 Example of instability of the predicted load at a given time step 62
3.23 Example of obtained potential field at $t = 31.6366$ s 62
3.24 Resulting velocity potential ϕ, solution of the BVPs. The outer boundaries are imposed as Dirichlet condition respecting: eq. (3.6). The cylinder condition is either directly applyied as a the latter equation (Dirichlet, left) or as a Neumann condition which value is derived from the same equation (right). 64

4.1 Schematic representation of the selected case. Note that the sea bottom is not represented, and everything is at scale, considering a wavelength $\lambda = 6.15$ m (half of it represented) and a wave steepness $H/\lambda = 7.0\%$. Profiles in the following of this work will be sampled over the blue lines l_{v1}, l_{h1} and l_{h2}. 84
4.2 Example of mesh grid. (bg045x045dx0056) 86
4.3 Loads on the cylinder with $H/\lambda = 3.5\%$, laminar vs turbulent ($k-\omega$ SST buoyant). Computations with OpenFOAM®. 89
4.4 Turbulent kinetic energy field at two different physical times. Native $k-\omega$ SST model. Black dots mark the free surface position ($\alpha = 0.5$) 90
4.5 Comparison of modified turbulent $k-\omega$ SST model results on k and ν_t . . 91
4.6 Investigations on the stabilized $k-\omega$ SST model 93
4.7 ν_t profile along a horizontal line at time $t/T = 15$ just above the body $z_l = -0.63$ m for the different computed cases. 94

4.8 ν_t along a horizontal line at time $t/T = 15$ just above the body $z_l = -0.63$m for the different computed cases. 95
4.9 ν_t along a horizontal line at time $t/T = 15$ just above the body $z_l = -0.63$m for the different computed cases. 95
4.10 Turbulent viscosity just above the body ($z_l = -0.63\,\mathrm{m}$) at differrent physical times, for the different versions of the $k-\omega$ SST model (stabilized, buoyant and smoothed), $\lambda_2 = 0.05$, $f_s = 0.5$. 97
4.11 Horizontal velocity profiles at the top-center of the cylinder. Two different periods are represented, namely $t//T = 15$ and $t//T = 16$. $\bar{t}$ is the adimentionalized remainder of $t//T$, thick lines represent the extrema of the horizontal velocities. Note that only those extrema are represented in the inset of each graph. 99
4.12 Temporal loads comparisons of the computation on the different meshes presented table 4.4."nc" stands for "no corners" which means that the corners were not additionnaly refined (see section 4.4.2 for details). 101
4.13 Amplitude of the loads on the cylinder with respect to the average discretization in the vicinity of the cylinder for different meshes. "nc" stands for "no corners" which means that the corners where not additionnaly refined. 101
4.14 Profile of the computed fields on a horizontal line at $z_l = -0.72\,\mathrm{m}$ (top of the cylinder) at time $t/T = 15$ 102
4.15 Temporal series of the loads applied on the cylinder for different time step sizes. A computation with the CFL number controlling the time step is also added, mesh:"bg045x045dx0056" 103
4.16 Amplitude of the loads as a function of the time step size. A CFL controled case is added ($\max(C_o) = 0.5$ *i.e.* $dt \approx 7.5 \times 10^{-4}\,\mathrm{s}$ to $10 \times 10^{-4}\,\mathrm{s}$) with an arbitrary abscissa of $dt = 7.5 \times 10^{-4}\,\mathrm{s}$. Two different meshes are used: bg045x045dx0028 ($dx = 0.0028\,\mathrm{m}$) and bg045x045dx0056 ($dx = 0.0056\,\mathrm{m}$). 104
4.17 Fields sampled over a vertical line at the top most altitude of the cylinder ($z_l = -0.72\,\mathrm{m}$), at $t/T = 21$ for the coarser mesh "bg045x045dx0056" . . . 105
4.18 Comparisons of the obtained envelopes of the horizontal velocity (computed over 40 time steps in a period) along a vertical line ($x_l = 0$) obtained with different wall models and two meshes ("baseMesh" and "noBdL") differing by their boundary layer discretizations (see text for details). (b) is a zoom on the boundary layer region. 107
4.19 Free surface elevation over time at $x = 0$ predicted by OpenFOAM® without body for the steepest wave case ($H/h = 8.9\%$) - Comparison between OpenFOAM® and a Stokes 5^{th} order theory. 111
4.20 Inertia and drag coefficients in the two directions obtained with OpenFOAM® with different wave periods. The L_2 norm of the error between the Morison model and the real loads is also shown, *i.e.* a large error means the Morison model is not well suited to represent the wave loads obtained with our RANS model. 113

5.1 Generic schematic of a domain decomposition (DD) solved by different models. 118

5.2 Schematic representation of the domain coupling method applied here. Note that only one "external" and one "internal" mesh are represented here. In our application, two external meshes will be used, corresponding to the different HPC grids, namely background and immersed. 122
5.3 Coupled mesh used vs mesh used for the computation with waveFoam alone 128
5.4 Horizontal and vertical loads obtained from the coupled algorithm compared with the waveFoam results. 131
5.5 Comparaison of the vorticity fields predicted by the domain coupled model and waveFoam at $t/T = 15$ 132
5.6 Horizontal velocity profiles, domainCoupling, waveFoam and HPC 132
5.7 Vertical velocity profiles, domainCoupling, waveFoam and HPC 134
5.8 Dynamic pressure profiles, domainCoupling, waveFoam and HPC 135
5.9 Comparaison of the turbulent viscosity field between the domain coupled case and the OpenFOAM® only case at $t/T = 15$. The horizontal black line on fig. 5.9a is the upper boundary of the CFD region, denoted Γ_t on fig. 5.2. 136
5.10 Turbulent viscosity profiles, domainCoupling and waveFoam 137
5.11 turbulent specific dissipation rate profiles, domainCoupling and waveFoam 138
5.12 Temporal series of the loads when performing a "start" and "hotstart" a time $t_0/T = 12$. See text for details. 140
5.13 Horizontal velocity (figs. 5.13a and 5.13b) and turbulent kinematic viscosity (figs. 5.13c and 5.13d) envelopes, computed from different 20 time steps values per half wave period. For waveFoam and the base domainCoupling $t/T \in [15, 16]$. For the "hotstart" case - begining at $t_0/T = 12$, the used values are extracted from times $t/T \in [12 + nT, 13 + nT]$. 141
5.14 Extracted maxima of the loads obtained with different CFD mesh breadths 142
5.15 Loads series obtained with the different BC sets, see table 5.2 143
5.16 Fields profile over a vertical line located at $x_l = 0.$ at time $t/T = 18$ (6 simulated periods) obtained with different boundary condition sets, see table 5.2. 144
5.17 Temporal loads series with two different sets of finite volumes schemes and parameters (see text for details). 145
5.18 Envelope over one period with two different sets of finite volume schemes and parameters (see text for details). 146
5.19 ν_t profiles obtained with domainCoupling, several turbulence model variations 147
5.20 Loads series for different wave heights, $T = 2\,\mathrm{s}$ 149
5.21 Inertia and drag coefficients in the two directions of the loads obtained with the DD method at different wave periods. The L_2 norm of the error between the Morison model and the real loads is also shown in the bottom panel. 150

6.1 Schematic representation of the coupling method 162

6.2 Temporal series of the loads obtained with the velocity decomposition method with different exit conditions for the PIMPLE loop, either a fixed number of iterations (nOuterCorr) or controlled by the residual target. Boundary set 2. (b) is a focus on $t/T \in [15, 16]$. 174
6.3 Temporal series of the loads obtained with the velocity decomposition method with different exit conditions for the PIMPLE loop, either a fixed number of iterations (nOuterCorr) or controlled by the residual target. See text for detail between "separated" and "aggregated". Boundary set 2.(b) is a focus on $t/T \in [15, 16]$. 175
6.4 Temporal series of the loads obtained with the velocity decomposition method with different tolerance targets. γ parametrizes the restriction of the target residual values compared to the base case. See text for details. (b) is a focus on $t/T \in [15, 16]$. 176
6.5 Fields at $t/T = 18$ (6 simulated periods) over a vertical line at $x_l = 0$ for different γ parameters. A marker is added every 5 faces encountered. ω variable is shown on a log-linear scale: it ranges from $1\ \mathrm{s}^{-1}$ to $1 \times 10^4\ \mathrm{s}^{-1}$. 177
6.6 Obtained loads for different complementary velocity boundary conditions presented in table 6.1. For further details the reader is referred to section 6.3.5. 178
6.7 velocity Coupling local results 180
6.8 waveFoam, domain coupling and velocity coupling total fields sampled over a vertical line ($x_l = 0\,\mathrm{m}$, $z = -0.72\,\mathrm{m}$ to $-0.3\,\mathrm{m}$) at 40 time instants split in two half-periods (left and right subfigures). 182
6.9 Temporal series of the loads obtained with the waveFoam solver, the domain decomposition method, the velocity decomposition method and HPC 183
6.10 Inertia and drag coefficients in the horizontal (x) and vertical (z) directions obtained with the domain decomposition method, the velocity decomposition method, the VoF-FVM method (waveFoam) and the HPC method for different KC numbers. The L_2 error of the Morison fitting to the temporal loads series is also shown in the lower panels. Important error reveals that the Morison model is not well adapted to describe the temporal series. . . 185

A.1 Sketch of the period, amplitude and total error of a perturbed cosine (grey) at two different whole periods. The assumption is made that e_A and e_T do not depend on time. 220
A.2 Relative error of models in period and amplitude (e_T and e_A respectively). Total relative error e after $\dfrac{t}{T_{th}}$ periods between two cosine (equation A.5) as a function of d which parameterize the order 4 convergence between their amplitudes and period. $e^{(n)}$ is the Taylor expansion of e at order n . 222

List of Tables

2.1 Analytical expressions of the integral over a straight line 33

3.1 Waves parameters for the simulated cases. KC and Re are based on the horizontal width of the object and the maximum horizontal velocity at the cylinder submergence depth predicted by the wave theory, see section 1.4. $\beta = Re/KC$. 56

4.1 Values of the different parameters of the $k-\omega$ SST model in OpenFOAM®. Table from Devolder *et al.* (2017). 75
4.2 Table describing the base mesh densities (in m) 86
4.3 Qualitative performance in fulfilling requirement "1)" (left table) and requirement "2)" (right table). See section 4.5.4 for definition of requirements 96
4.4 Table describing the tested mesh principal densities (in m), sorted by the number of cells 100
4.5 sets of wall boundary conditions. zG stands for `zeroGradient` while fV for `fixedValue` and their associated values are precised. 106
4.6 Execution times on 4 processor (*) or 1 processor, for some of the presented meshes and variations. Note that the parallel decomposition seems to not have been correctly parametrized, especially for the coarsest meshes. . . . 115

5.1 Table of the base set of boundary conditions. IO stands for inletOutlet, fV for fixedValue. A prefix "c" is added when their coupled counterparts are used. 129
5.2 Table of the tested sets of boundary conditions. IO stands for `inletOutlet` , fV for `fixedValue`. A prefix "c" is added when their coupled counterparts are used. For details about these boundaries, see section 5.2.6. 142

6.1 Table of the base set of boundary conditions. IO stands for `inletOutlet`, fV for `fixedValue`. Values are precised when necessary (and not dummies). For IO conditions, we also indicate the face flux used to determine the direction of the flow. 178

7.1 CPU costs with the main employed mesh ($dx = 0.056$ m in the body vicinity) using one computational core. Note that those CPU time values might not be perfectly accurate (see text for details). 193

Appendix A

Convergence of a perturbed cosine in both period and amplitude at long time

In order to analyze the rate of convergence of our implementation of the HPC method on the case of the nonlinear standing wave presented in section 3.2, we consider that the theoretical (reference) solution for the free surface elevation at the center of the numerical wave tank, denoted $f_{th}(t)$, is a cosine function of period T_{th} and amplitude a_{th}:

$$f_{th}(t) = a_{th} \cos\left(2\pi \frac{t}{T_{th}}\right) \tag{A.1}$$

Assume then that the HPC solution for the free surface elevation at the same location can also be expressed as a cosine function:

$$f(t) = a \cos\left(2\pi \frac{t}{T}\right) \tag{A.2}$$

with amplitude amplitude a and period T, being close to a_{th} and T_{th} respectively.

Our numerical convergence tests have shown (see figure 3.4) that the errors on amplitude and period of the surface elevation at the center of the tank decrease as C_o^4. Lets generalize this with a more convenient parameter d and thus assume that the error both in amplitude and period decreases as d^4. In this case $d \equiv C_o$ but the exact same reasoning can be applied with $d \equiv \delta x$. With this notation we have:

$$a = a_{th}(1 - e_A) = a_{th}(1 - f_A d^4) \tag{A.3}$$
$$T = T_{th}(1 - e_T) = T_{th}(1 - f_T d^4) \tag{A.4}$$

Note that for every representation in this article, we set $f_A = -5.5 \cdot 10^{-3}$ and $f_T = -5.4 \cdot 10^{-5}$. Those values are extracted from the convergence depicted figure (3.4) with the aim of fitting the convergence of our HPC model on the standing wave for $d \equiv C_o$ and $t/T = 100$.

Lets define e the total relative error at a given time t:

$$e(t) = \frac{f_{th}(t) - f(t)}{a_{th}} \tag{A.5}$$

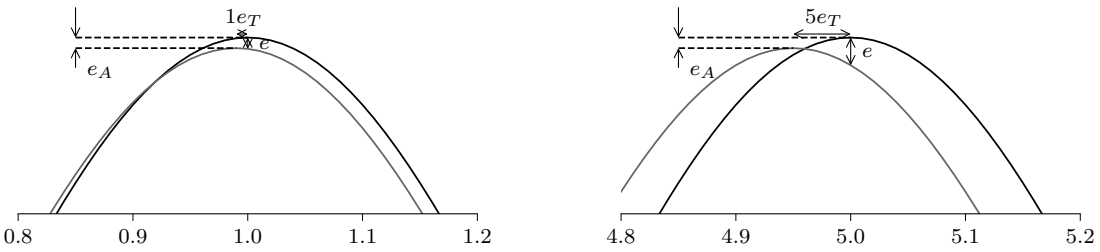


Figure A.1: Sketch of the period, amplitude and total error of a perturbed cosine (grey) at two different whole periods. The assumption is made that e_A and e_T do not depend on time.

This error is the combined effect of both the error on amplitude and the error on period. A representation of the impact of e_A and e_T and their combined impact - with the assumption that they are constant in time - are shown in sketch A.1.

Developing this expression with our cosine models (A.1 and A.2), and injecting the modeled expression of the error (A.3 and A.4) yield an expression of the total error as a function of t and d.

$$e(t) = \cos(2\pi\frac{t}{T_{th}}) - \frac{a}{a_{th}}\cos(2\pi\frac{t}{T}) \tag{A.6}$$

$$e(t) = (\cos(2\pi\frac{t}{T_{th}}) - \cos(2\pi\frac{t}{T})) + f_A d^4 \cos(2\pi\frac{t}{T}) \tag{A.7}$$

This error is computed for different times $\frac{t}{T_{th}}$ and shown on figure (A.2) as a function of d. In the range $d \in [1.5, 2.5]$ a power interpolation is computed and added.

A.1 Taylor expansion

In this part, a Taylor expansion of the latter expression of the total error is done in the vicinity of a whole number of period t/T_{th}:

$$e(t/T_{th} \in \mathbb{N}) = (1. - \cos(2\pi\frac{t}{T})) + f_A d^4 \cos(2\pi\frac{t}{T}) \tag{A.8}$$

Let's first expand t/T using the modeled behavior of T when $d \to 0$ given in the expression (A.4):

$$2\pi\frac{t}{T} = 2\pi\frac{t}{T_{th}}\frac{1}{1 - f_T d^4} \tag{A.9}$$

$$2\pi\frac{t}{T} = 2\pi\frac{t}{T_{th}}(1 + f_T d^4 + f_T^2 d^8 + O(d^{16})) \tag{A.10}$$

$$2\pi\frac{t}{T} = 2\pi\frac{t}{T_{th}} + X \tag{A.11}$$

where $X = 2\pi\frac{t}{T_{th}}(f_T d^4 + f_T^2 d^8 + O(d^{16}))$. X also tends to 0, so it is possible to expand the cosine in the vicinity of its maximum (whole number of periods):

$$e(t/T_{th} \in \mathbb{N}) = (1 - [1 - X^2/2 + O(X^4)]) + f_A d^4[1 - X^2/2 + O(X^4)] \tag{A.12}$$

$$e(t/T_{th} \in \mathbb{N}) = (2\pi^2\frac{t^2}{T_{th}^2} f_T^2 d^8 + 2\pi\frac{t}{T_{th}} f_T^3 d^{12} + O(d^{16})]) + f_A d^4[1 - 2\pi^2\frac{t^2}{T_{th}^2} f_T^2 d^8 + O(d^{12})] \tag{A.13}$$

At the end, we obtain the Taylor expansion of e/a_{th} at order 12:

$$e(t/T_{th} \in \mathbb{N}) = f_A d^4 + 2\pi^2\frac{t^2}{T_{th}^2} f_T^2 d^8 + (2\pi\frac{t}{T_{th}} f_T^3 - 2 f_A \pi^2\frac{t^2}{T_{th}^2} f_T^2) d^{12} + O(d^{16}) \tag{A.14}$$

A.2 Representation and interpretation

Equation (3.2) is computed at $o(8)$ and $o(12)$ and represented as a function of d on figure (A.2) (respectively labelled $e^{(8)}$ and $e^{(12)}$) at $\frac{t}{T_{th}} = 100$. The parameters are kept fixed to $f_A = 5.5 \cdot 10^{-3}$ and $f_T = 5.4 \cdot 10^{-5}$ so as to fit the obtained results depicted on figure (3.4). Thus, we model the case where $N_x/\lambda = 30$ and the error is computed as the relative difference of the free surface elevation at the center point ($x/\lambda = 0.5$) with $d \equiv C_o$. Moreover, expression (A.5) of e is also shown at different times and for different convergence parameter in the range $[1, 3.5]$.

For most lines a power regression is computed and added in the figure (dashed lines). For the sake of simplicity, the values of f_A and f_T are supposed to be independent of t/T_{th} and set from the observed convergence at t/T_{th}. In reality, this assumption is not verified.

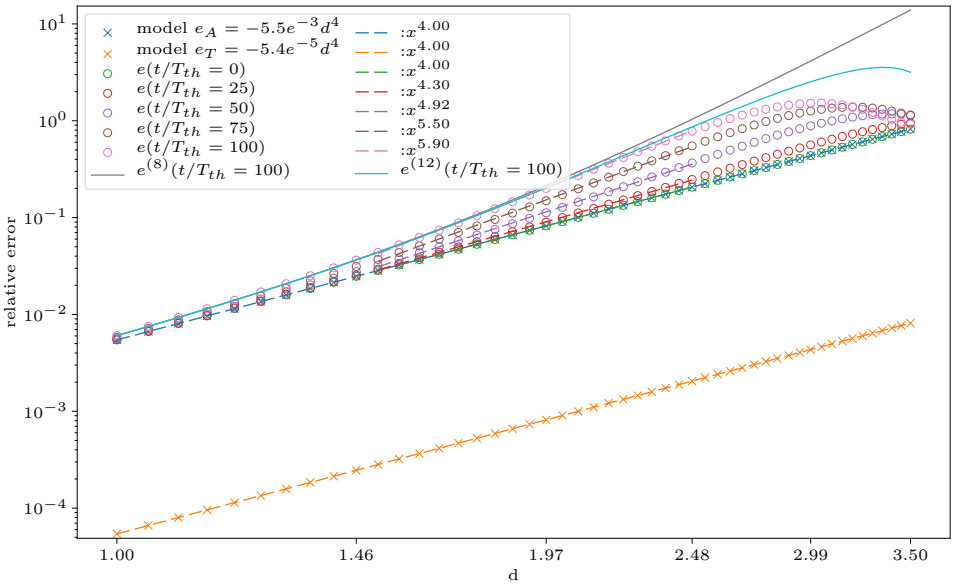


Figure A.2: Relative error of models in period and amplitude (e_T and e_A respectively). Total relative error e after $\frac{t}{T_{th}}$ periods between two cosine (equation A.5) as a function of d which parameterize the order 4 convergence between their amplitudes and period. $e^{(n)}$ is the Taylor expansion of e at order n

Note that the order of convergence is 4 when $\frac{t}{T_{th}} = 0$ (*i.e.* $t = 0$). For $t \neq 0$ the asymptotic error when d converges to 0 is equal to the amplitude error. Mathematically, this is due to the square elevation when expanding the cosine. Thus, on figure (A.2), the d^4 order model is not shown as it corresponds to the amplitude error.

In order to compare the different regimes of convergence, the ratio between the 8$^{\text{th}}$ order and 4$^{\text{th}}$ order is computed

$$r_{8,4} = \frac{2\pi^2 \frac{t^2}{T_{th}^2} f_T^2}{f_A} d^4 \tag{A.15}$$

If f_A increases with time slower than $t^2 f_T^2$, then the order 8 will have a growing importance over time ($r_{8,4}$ increase). As f_A, and f_T where set to model the magnitude of the error after $t/T_{th} = 100$ and $d \equiv C_o$, the ratio presented above is computed at this time step for different parameter d: 8^{th} order and 4^{th} (*i.e.* $r_{8,4} = 1$) are of the same importance when $\dfrac{t}{T_{th}} = 100$ as soon as $d = 1.75$. For d as small as 1.00, the 8^{th} order already represents $r_{8,4} = 10\%$ of the error. Thus, it is both mathematically and graphically predictable that the convergence in time in our standing wave case would be of order higher than 4 after $t/T_{th} = 100$ in our range of the C_o parameter given the convergence rate of the amplitude and period at this time step.

It is then also consistent that the orders of convergence seem to increase with $\dfrac{t}{T_{th}}$ for all other computations.

Appendix B

Finite Volume Method (FVM) applied in OpenFOAM for solving the RANS equations

This appendix briefly describes the implementation of the finite volume method applied to a multiphase case - treated by the volume of fluid method - in the OpenFOAM® framework. The solver of interest is `interFoam`, and some of the presents corrections are also detailed, mostly concerning the velocity-pressure resolution.

The call to a `PIMPLE` loop is done within the main solver directly (alg. B.1).

Algorithm B.1: PISO loop

```
/*************** interFoam.C *************************/
    #include "UEqn.H"

    // --- Pressure corrector loop
    while (pimple.correct())
    {
        #include "pEqn.H"
    }
```

Amongst these lines, `UEqn.H` (shown in alg. B.2) generate the equation for **u** at a given time step eqs. (4.9a) and (4.9b) while `pEqn.H` (shown in alg. B.3) solve the Poisson equation for the pressure.

Algorithm B.2: UEqn.H

```

/*************** UEqn.H ************************/
    MRF.correctBoundaryVelocity(U);

    fvVectorMatrix UEqn
    (
        fvm::ddt(rho, U) + fvm::div(rhoPhi, U)
      + MRF.DDt(rho, U)
```

```
9       + turbulence->divDevRhoReff(rho, U)
10     ==
11        fvOptions(rho, U)
12     );
13
14     UEqn.relax();
15
16     fvOptions.constrain(UEqn);
17
18     if (pimple.momentumPredictor())
19     {
20        solve
21        (
22           UEqn
23        ==
24           fvc::reconstruct
25           (
26              (
27                 mixture.surfaceTensionForce()
28               - ghf*fvc::snGrad(rho)
29               - fvc::snGrad(p_rgh)
30              ) * mesh.magSf()
31           )
32        );
33
34        fvOptions.correct(U);
35     }
```

The main objective here, is to compute the matrices $\mathbb{H}$ and $\mathbb{A}$ yielded by eq. (4.35) and needed on eq. (4.36). Thus, the pressure ∇p is omitted in the `UEqn` equation (lines 5-12) as it is not needed to compute $\mathbb{H}$ and $\mathbb{A}$. Note that this equation resolution is not needed, and done only when the `momentumPredictor` is activated. In that case only, we solve for the previously defined equation to be equal to the pressure gradient.

The pressure equation file is more complex, as it includes the loop correcting for non-orthogonal meshes, the source file is shown in alg. B.3. The eq. (4.36) is solved trough the definition of HbyA ($\mathbb{H}/\mathbb{A}$). Note that the divergence operator is defined as the sum of a flux over the faces of a cell. The function `laplacian(q_1, q_2)` is defined as the divergence of the product of q_1 and the gradient of q_2. Thus, q_1 needs to be a surface field too. The q_2 variable is a volume field, and thus the function perform an implicit computation of the surface normal gradient of q_2.

main algorithm

- HbyA=$\mathbb{H}/\mathbb{A}$ (line 3)
- forceFlux=-interpolate(gh) * $\nabla_n \rho$* interpolate($1/\mathbb{A}$) * magSf (lines 17-23). Note that ∇_n operator is the surface normal gradient operator. It computes the scalar value of the gradient along the normal of a face. Its return value is then of type surfaceScalarField. The interpolate function used here is the explicit interpolation

of a field defined on cell (e.g. volScalarField) to a field defined on the faces (e.g. surfaceScalarField). magSf is the magitude of the surface normal field, thus it contains the information on the area of the faces.

- phiHbyA=flux(HbyA)+forceFlux (line 8-13 and 25)
- p_rghEqn: `laplacian(interpolate(`$1/\mathbb{A}$`),` $p - \rho g h$`)` $==$ $\nabla\cdot$(phiHbyA)

Corrections A lot of corrections are done on top of the steps presented above:

- The flux used in eq. (4.36) to compute the r.h.s. (line 34) is not directly set to the flux of HbyA. A second flux ϕ_2 is added to HbyA (line 12). This corrective flux is computed as

$$\phi_2 = \left[\frac{\rho}{\mathbb{A}}\right]_f \frac{\gamma}{\Delta t}[\phi^{n-1} - S_f \mathbf{u}_f^n] \tag{B.1}$$

  where the exponent $n-1$ represent the fields at the previous time step. Members of this equation are indiced with f to represent them as field defined on the faces. S_f is the surface magnitude of the faces. The coefficient γ is computed from the difference of the flux and the velocity interpolated at the faces with

$$\gamma = 1 - \min\left(\frac{\phi^{n-1} - S_f \mathbf{u}_f^{n-1}}{\phi^{n-1}}, 1\right) \tag{B.2}$$

  Allegedly, this term introduce a numerical stabilization, even if it also introduce numerical dissipation. More details are given in Vuorinen *et al.* (2014, section 2.6 and 4.4) where it is shown on the inviscid Taylor-Green vortex case that this extra terms dissipate energy, particularly for coarse grids.
- The multi reference frame correction (MRF) is disabled in our case, as its purpose is to allow to solve the equation a non inertial reference frame.
- On line 15, an adjustment of the face flux is done on the boundary conditions. Wherever the outward flux is adjustable (inlet-outlet conditions, zeroGradient conditions, *etc.*), we modify it such that the mass is conserved. In this function **u** is needed only to know if a given boundary has an adjustable mass.

Remark. *This function that adjusts the flux ϕ is also used on the input flux which, after interpolation onto the CFD mesh, is not conservative enough.*

Algorithm B.3: pEqn.H

```
1  /*************** pEqn.H ************************/
2  {
3      volScalarField rAU("rAU", 1.0/UEqn.A());
4      surfaceScalarField rAUf("rAUf", fvc::interpolate(rAU));
5
6      volVectorField HbyA(constrainHbyA(rAU*UEqn.H(), U, p_rgh));
7
8      surfaceScalarField phiHbyA
9      (
10         "phiHbyA",
11         fvc::flux(HbyA)
```

```
12        + fvc::interpolate(rho*rAU)*fvc::ddtCorr(U, phi)
13    );
14    MRF.makeRelative(phiHbyA);
15    adjustPhi(phiHbyA, U, p_rgh);
16
17    surfaceScalarField phig
18    (
19        (
20            mixture.surfaceTensionForce()
21          - ghf*fvc::snGrad(rho)
22        )*rAUf*mesh.magSf()
23    );
24
25    phiHbyA += phig;
26
27    // Update the pressure BCs to ensure flux consistency
28    constrainPressure(p_rgh, U, phiHbyA, rAUf, MRF);
29
30    while (pimple.correctNonOrthogonal())
31    {
32        fvScalarMatrix p_rghEqn
33        (
34            fvm::laplacian(rAUf, p_rgh) == fvc::div(phiHbyA)
35        );
36
37        p_rghEqn.setReference(pRefCell, getRefCellValue(p_rgh,
              pRefCell));
38
39        p_rghEqn.solve(mesh.solver(p_rgh.select(pimple.finalInnerIter())));
40
41        if (pimple.finalNonOrthogonalIter())
42        {
43            phi = phiHbyA - p_rghEqn.flux();
44
45            p_rgh.relax();
46
47            U = HbyA + rAU*fvc::reconstruct((phig -
                  p_rghEqn.flux())/rAUf);
48            U.correctBoundaryConditions();
49            fvOptions.correct(U);
50        }
51    }
52
53    #include "continuityErrs.H"
54
55    p == p_rgh + rho*gh;
56
57    if (p_rgh.needReference())
58    {
```

```
59          p += dimensionedScalar
60          (
61              "p",
62              p.dimensions(),
63              pRefValue - getRefCellValue(p, pRefCell)
64          );
65          p_rgh = p - rho*gh;
66      }
67  }
```

In OpenFOAM®, the effective mean rate or stain tensor is computed trough the function divDevRhoReff:

Algorithm B.4: Computing the mean rate of stain tensor

```
1  /***************** UEqn.H ************************/
2  fvVectorMatrix UEqn
3      (
4          fvm::ddt(rho, U) + fvm::div(rhoPhi, U)
5       + MRF.DDt(rho, U)
6       + turbulence->divDevRhoReff(rho, U)
7       ==
8          fvOptions(rho, U)
9      );
10
11 /************ linearViscousStress.C ***************/
12 Foam::linearViscousStress<BasicTurbulenceModel>::divDevRhoReff
13 (
14     const volScalarField& rho,
15     volVectorField& U
16 ) const
17 {
18     return
19     (
20       -
          fvc::div((this->alpha_*rho*this->nuEff())*dev2(T(fvc::grad(U))))
21       - fvm::laplacian(this->alpha_*rho*this->nuEff(), U)
22     );
23 }
24 /***************** tensorI.H ********************/
25 //- Return the deviatoric part of a tensor
26 template<class Cmpt>
27 inline Tensor<Cmpt> dev2(const Tensor<Cmpt>& t)
28 {
29     return t - SphericalTensor<Cmpt>::twoThirdsI*tr(t);
30 }
31
32
33 }
```

In alg. B.4, the operator T() is the transpose operator, alpha_ is a member which was previously set as uniformly equal to one. The deviatoric operator of $\mathbb{A}$ is given as $dev(\mathbb{A}) = \mathbb{A} - 1/3tr\mathbb{A}$. Thus dev2 is the same with a 2/3 factor in place of the 1/3. Note also that the equality $\nabla \cdot B = tr(\nabla B^T)$ was used.

Abstract

This thesis is devoted to the mathematical modeling and numerical simulation of water waves and their interaction with fixed bodies. A numerical wave tank is first developed within the fully nonlinear potential theory. The mathematical formalism and the Harmonic Polynomial Cell (HPC) method, used to solve the boundary value problems on the wave potential and its time derivative, are described. Numerical strategies employed to include a body of arbitrary shape and to simulate the free surface are developed. Extensive evaluations of the model stability and its independence to numerical parameters (mesh, time-step, *etc.*) are conducted on a highly nonlinear standing wave case. Afterwards, validations on wave-body interactions problems are realized by comparing with both literature results and a dedicated experiment performed during this study. Then a Reynolds Averaged Navier-Stokes (RANS) approach associated with the Volume of Fluid method (VoF) is extensively tested on a case that was shown to emphasize some limitations of the potential approach. The solver, available in the OpenFOAM® toolbox, successfully captures additional effects, at the cost of an increase in the computational resources.

Thus, two one-way coupling approaches are developed within the OpenFOAM® library. First, a domain decomposition strategy is employed to solve the RANS equations only in the vicinity of the body, enforcing values yielded by the potential HPC model at the outer boundaries: a large reduction of the domain size is achieved. Moreover, the incident wave propagation is simulated by the potential method, more efficient, accurate and easier to set up than the RANS-VoF method for this task. In the second coupling method, on the local grid close to the body, the pressure and velocity variables are decomposed into their potential components and complementary parts, these latter ones being expected to vanish away from the body. Modified version of the RANS equations for the complementary variables are derived and implemented. The two coupling approaches are extensively tested on the case of a horizontal cylinder of rectangular cross-section. Good agreements with the original OpenFOAM® results are found, for only a contained part of the computational cost, both in the time-series of loads on the body and the local flow field variables. Predictions of the associated hydrodynamic coefficients are accurate when compared to existing experimental and numerical results.

Résumé

Cette thèse est consacrée à la modélisation mathématique et à la simulation numérique des vagues et de leurs interactions avec des corps fixes. Un canal à houle numérique est tout d'abord développé dans le cadre de la théorie potentielle non-linéaire. En premier lieu sont décrits le formalisme mathématique ainsi que la méthode des cellules aux polynômes harmoniques (HPC), utilisée pour résoudre les problèmes aux valeurs limites sur le potentiel et sur sa dérivée temporelle. Des stratégies numériques utilisées pour inclure un corps de forme arbitraire et pour simuler la surface libre sont développées. Des évaluations approfondies de la stabilité du modèle et de son indépendance par rapport aux paramètres numériques (maillage, pas de temps, *etc.*) sont menées sur le cas d'une onde stationnaire fortement non-linéaire. Ensuite, la méthode est validée par comparaison avec différents résultats d'interactions vagues-structure, pour partie issus de la littérature, et complétés par des expériences spécifiques réalisées au cours de ce travail. Par la suite, une méthode RANS (Reynolds Averaged Navier-Stokes) combinée avec la méthode de Volume of Fluid (VoF) est mise en œuvre et testée de manière approfondie sur un cas de validation dont certains effets physiques sont ignorés dans le cadre de la théorie potentielle. Le solveur, disponible dans la boîte à outils OpenFOAM®, permet de capturer ces effets supplémentaires, au prix toutefois d'une augmentation significative du coût de calcul.

Ainsi, pour réduire ces coûts additionnels, deux approches de couplage unidirectionnel sont développées, utilisant la librairie OpenFOAM®. En premier lieu, une stratégie de décomposition de domaine est proposée pour résoudre les équations RANS uniquement à proximité du corps, en appliquant les valeurs fournies par le modèle potentiel HPC sur les frontières extérieures : une réduction importante de la taille du domaine de calcul est ainsi obtenue. De plus, la propagation de l'onde incidente est simulée par la méthode potentielle, plus efficace, plus précise et plus facile à mettre en place que la méthode RANS-VoF pour cette tâche. Dans la seconde méthode de couplage, sur la grille locale proche du corps, les variables de pression et de vitesse sont décomposées en leurs composantes potentielles et complémentaires, ces dernières étant supposées tendre vers zéro lorsque l'on s'éloigne du corps. Une version modifiée des équations RANS pour les variables complémentaires est dérivée et mise en œuvre. Les deux approches de couplage sont ensuite validées sur le cas d'un cylindre horizontal de section rectangulaire. De bons accords avec les résultats originaux de OpenFOAM® sont obtenus, à la fois pour les séries temporelles d'efforts sur le corps et les variables décrivant l'écoulement local, avec une réduction importante du coût de calcul. Les coefficients hydrodynamiques associés sont également correctement prédits, en comparaison aux résultats expérimentaux et numériques existants.