Coastal Engineering 91 (2014) 1–17

Contents lists available at ScienceDirect

Coastal Engineering

journal homepage: www.elsevier.com/locate/coastaleng

# Benchmarking of Navier–Stokes codes for free surface simulations by means of a solitary wave

Paweł A. Wroniszewski [a,*], Joris C.G. Verschaeve [b], Geir K. Pedersen [a]

[a] Department of Mathematics, University of Oslo, P.O. Box 1053 Blindern, 0316 Oslo, Norway
[b] Norwegian Geotechnical Institute (NGI), P.O. Box. 3930 Ullevål Stadion, N-0806 Oslo, Norway

ARTICLE INFO

*Article history:*
Received 3 June 2013
Received in revised form 27 April 2014
Accepted 28 April 2014
Available online xxxx

*Keywords:*
Navier–Stokes
Volume-of-fluid
Two-phase flow
Benchmark
Wave runup
Solitary wave

ABSTRACT

The paper presents a benchmark of four freely available solvers for Navier–Stokes equations: Gerris, OpenFOAM, Thétis and Truchas. These models are selected because they have been reported to deal successfully with oceanographic and coastal engineering applications. The benchmark includes two free surface problems: propagation of a solitary wave and runup of a solitary wave on a plane beach. The fluids are inviscid, which allows a detailed study of energy conservation and comparison of the results with a reference solution given by a boundary integral solver. The Navier–Stokes solvers use the finite volume discretization and free surface capturing techniques based on the volume-of-fluid (VOF) method. In the first benchmark test, we investigate the influence of numerical dissipation and other spurious effects on the energy balance of the wave. In the second problem, we focus on the runup heights and compare them with the reference solution. The beach in the runup problem is represented with a solid body immersed in the Cartesian mesh. The solid boundaries are described with a VOF type approach or a staircase representation, depending on the features of the solver. In addition to the immersed boundary description a couple of body fitted meshes are tested for the runup case.



## 1. Introduction

Flow simulations in oceanography and coastal engineering are dominated by depth averaged models. Such models offer high efficiency and simple treatment of the free surface, at the cost of loss of information concerning vertical distribution of flow properties. Even though they are well suited for wave propagation, they are often insufficient for problems involving complex flows on a variety of scales, like evolution of turbulence, green water loading or sub-aerial landslides. For this reason, primitive models, like Navier–Stokes solvers, are used in a growing number of applications. Examples of these include numerical prediction of green water on deck (Nielsen, 2003), modeling of fluid–structure interactions (Formaggia et al., 2008) or simulations of landslide-generated waves (Montagna et al., 2011) with deformation of solid bodies (Abadie et al., 2010). Some authors model the influence of turbulence by the use of the large-eddy-simulation approach, in applications like runup of waves generated by landslides (Liu et al., 2005), air entrainment under plunging breaking waves (Lubin et al., 2006) or propagation and breaking of solitary waves over submerged obstacles (Lubin, 2004). Likewise, simulations based on the Reynolds averaged Navier–Stokes equations have been performed for the computation of spilling breaking waves (Jacobsen et al., 2012) or for the computation of the propagation of a solitary wave, its interaction with a solid body and the breaking of a solitary wave on a sloping beach (Higuera et al., 2013a, 2013b). Somewhat more related to the present work is the work by Lin et al. (1999), where a non-breaking solitary wave runup has been computed by using a Navier–Stokes solver. However, even though Navier–Stokes solvers, due to their flexibility, are applied in a growing number of free surface problems related to oceanography and coastal engineering, a unified testing to assess their accuracy with respect to a nontrivial, highly accurate reference solution is not commonly formulated.

Free surface flows involve two fluids of different physical properties. Unless mixing is modeled, a jump in parameters is expected at the interface separating the fluids. There are several ways to model the interfaces in the Navier–Stokes solvers. Representing the interface with marker points, as in the interface tracking methods, automatically keeps the transition between fluids sharp, but brings difficulties with topological changes. An alternative approach is the interface capturing methods, where different fluids are marked by a scalar field and the position of the interface can be found by means of this scalar field. The advected scalar field may represent the distance from the interface, as in the level-set method, or the volume fractions, as in the volume-of-fluid (VOF) method. Advecting the volume fractions with the transport equation, requires special treatment for keeping the transition sharp. It may involve introduction of artificial compressibility terms or use of special numerical schemes, e.g. the total variation diminishing (TVD) method (Fletcher, 1991). Alternatively, the volume fractions can be advected

* Corresponding author. Tel.: +47 22 85 41 28.
*E-mail addresses:* pwronisz@gmail.com (P.A. Wroniszewski), joris@math.uio.no (J.C.G. Verschaeve), geirkp@math.uio.no (G.K. Pedersen).

http://dx.doi.org/10.1016/j.coastaleng.2014.04.012
0378-3839/

by a geometrical method, using the defined location of the interface, as in the VOF-PLIC (Piecewise Linear Interface Calculation) method (Tryggvason et al., 2011). Geometric advection of the interface has been proven to be fully mass conserving and very accurate in many applications, but can introduce small, spurious droplets, being ejected from the interface (Abadie et al., 2010). The use of the TVD method has been found more robust in certain applications (Abadie et al., 2010; Mory et al., 2011), as it smooths out the smaller features of the interface.

Similar to the interface capturing methods, which enable simulations of several fluids on a single mesh, a VOF type treatment can be used to model a solid body immersed in the fluid (Popinet, 2003). Alternatively, the presence of such a body can be modeled with source terms in the governing equations, as in the immersed boundary method. Such methods largely reduce the meshing effort in cases involving complex geometries and give good prospects for simulations of moving bodies, but usually at the cost of limited accuracy compared to body-fitted meshes.

This work has been undertaken in search for a robust and accurate method for advanced flow simulations in the oceanographic and coastal engineering context. The length and time scales in this context are extremely large and a detailed resolution of small scale features like droplets, boundary layers or small scale turbulence is not feasible. A solver should therefore ideally present a way to keep track of the influence on the large scales by these small scales in a sensible manner. However, the present work is not on the development nor on the testing of models keeping track of these influences, like large eddy simulations or Reynolds averaged Navier–Stokes based simulations. The present strategy is to formulate a nontrivial flow case where these small scale features are switched off. The flow solver designed for advanced flow simulations should then in the ideal case recover the reference solution with a certain level of accuracy. In the following, we shall see that this is, however, not always the case and that different issues can prevent the numerical solutions by this kind of solvers to converge smoothly to the reference solution.

There are several freely available Navier–Stokes solvers, able to perform multi-phase simulations. Among those, four finite volume solvers, all using the VOF method, were selected for this paper: Gerris, OpenFOAM, Thétis and Truchas. They were selected because they were reported successful in a variety of problems on different scales and validated against experimental data obtained for sample problems relevant in the coastal engineering context.

Nevertheless, lack of thorough, comparative benchmarking makes it difficult to choose the best tool for a given problem, e.g. wave propagation with an acceptable level of damping. A unified comparison of such solvers for well defined non-trivial problems would be helpful.

In this paper, two benchmark problems are defined and evaluated for the above-mentioned solvers. Both problems are two-dimensional and simulated on Cartesian grids. The fluids are modeled as inviscid, which, being a reasonable assumption for some kinds of flows, also enables comparison with the well controlled potential theory in order to estimate the influence of numerical viscosity and other spurious effects. In the first problem, a solitary wave is propagated over a certain distance. In this study we investigate to what degree the shape and energy of the wave remains constant, as it should according to the chosen physical model. A similar benchmark test was already used to validate Truchas (Wu, 2004) and another Navier–Stokes solver, Aquilon (Lubin, 2004), with allegedly good results. Our study is, however, more thorough, comparative and involves convergence tests. In the second benchmark problem, a solitary wave runs up on a plane beach. This test is significantly more difficult, because a thin swash tongue is generated along a solid boundary. This feature is demanding to capture properly by the methods under investigation. As all the simulations are performed on Cartesian meshes, immersed solid models are used when possible. The results of the runup are compared with a reference solution given by an accurate, fully non-linear boundary integral method solver. During the simulations, several difficulties and spurious effects were encountered and documented. To some extent the matter behind these issues is investigated. However, it is not feasible to study them all in detail, due to the variety of solvers used in this work.

The paper is organized as follows. Section 2 describes the Navier–Stokes solvers, with the emphasis on the differences between the implemented numerical techniques. Several remarks on initialization and post-processing are also given. Section 3 presents in detail the benchmark tests, which the above solvers are subject to, together with some information about the boundary integral method used for generation of the reference solutions. The results of the simulations are presented and discussed in Section 4, revealing a number of artifacts and giving an assessment on the usability of the models.

## 2. Numerics

### 2.1. Mathematical model

The governing equations for the free surface flow are the incompressible Navier–Stokes equations in the fluid domain $\Omega$:

$$\nabla \cdot \vec{u} = 0 \tag{1}$$

$$\partial_t \vec{u} + \left(\vec{u} \cdot \nabla\right)\vec{u} = -\frac{1}{\rho}\nabla p + \nu \Delta \vec{u} + \vec{g}. \tag{2}$$

However, we benchmark the Navier–Stokes solvers for cases where the viscosity is switched off. This way the numerical solution can be compared to an accurate reference solution for the inviscid flow. Therefore, we set $\nu = 0$ for the main part of the discussion. The resulting equations are called the Euler equations. These are used for both, the water and the air phase, since a one-fluid formulation is used. All simulations were performed for a high density ratio between air and water, 1:1000, with no deliberate smoothing at the interface. By setting $\nu$ to zero, boundary layers and turbulence will not be present in the problems constituting this benchmark. Therefore, we do not need to consider turbulence modeling nor resolution issues at the boundaries.

Subsequentially, we apply the slip boundary conditions at solid boundaries. We also neglect the surface tension, therefore the following boundary conditions hold at the free surface:

$$p|_{fs} = 0, \tag{3}$$

$$\vec{w} \cdot \vec{n} = \vec{u}|_{fs} \cdot \vec{n}, \tag{4}$$

where $\vec{u}|_{fs}$ is the velocity of the fluid at the boundary, $\vec{w}$ is the velocity of the boundary and $\vec{n}$ is the normal at the boundary. The outflow boundary conditions, used for the top of the computational domain, read

$$\frac{\partial \vec{u}}{\partial n} = 0, \tag{5}$$

$$\vec{u} \cdot \vec{t} = 0, \tag{6}$$

where $\vec{t}$ is a unit vector tangential to the boundary. The outflow boundary condition is used only in simulations in which the gravity vector is aligned with the mesh. In the rotated domain (see Section 3), the slip boundary condition is applied at all the boundaries of the domain.

To document the effect of viscosity, a few simulations in Section 4.2.5 are performed with $\nu = 10^{-6}$ and no-slip boundary conditions at the solid boundaries.

Navier–Stokes type models solve Eqs. (1) and (2) *directly* on a computational grid. Combined with an interface capturing method, the solvers may describe problems with complex configurations of the interface.

If there is initially no rotation present in an inviscid fluid, the velocity derives from a potential $\vec{u} = \nabla\phi$. Then, a boundary integral equation can be derived reducing the dimension of the problem by one. Methods of this kind are very accurate and are used in the present discussion to generate the reference solution for the runup of a solitary wave. However, they cannot deal with features like breaking waves and topology changes of the interface. Boundary integral equation methods are commonly used in coastal engineering related problems and more details can be obtained in Longuet-Higgins and Cokelet (1976), Vinje and Brevig (1981), Grilli and Svendsen (1989, 1990), Cooker et al. (1990), Cooker and Peregrine (1992), Grilli et al. (1994, 1997).

### 2.2. Computational methods

In the finite volume method, used by all of the solvers in the current investigation, the computational domain is subdivided into a number of cells, over which the governing equations are integrated. Pressure and material properties are located at cell centroids, while the velocity may either be stored at cell centroids (Fig. 1b) or on cell faces (staggered grid approach, see Fig. 1a). The cell-centered approach requires interpolation of the velocity on cell faces at each time-step. Several discretization schemes for the advection term can be adopted, from which the upwind scheme is preferred in this paper, due to its rugged stability. The main solution procedure is based either on the projection method (Tryggvason et al., 2011) or the Pressure Implicit with Splitting of Operators (PISO) method (Issa, 1985).

In the VOF method (Tryggvason et al., 2011), each fluid involved in a simulation is marked with the volume fraction $\gamma$ which it occupies in a cell. The volume fractions take values from 0 to 1, where we let $\gamma = 1$ stand for a cell full of liquid and $\gamma = 0$ for a cell full of gas. The fluid density in a cell is directly related to the volume fraction:

$$\rho = \gamma\rho_l + (1-\gamma)\rho_g \tag{7}$$

where $\rho_l$ and $\rho_g$ stand for the densities of liquid and gas, respectively. To describe the flow in a whole domain one can use a single velocity field, as in the one-fluid formulation (Tryggvason et al., 2011), or a separate velocity field for each of the fluids, as in the two-fluid formulation. The one-fluid approach is relatively simpler, as it does not require any additional treatments apart from the ones already described, and is therefore the only one used by the tested solvers.

The volume fractions are advected by the velocity field. The advection is governed by the transport equation, which, for an incompressible flow, reads:

$$\frac{\partial\gamma}{\partial t} + \nabla\cdot\left(\vec{u}\gamma\right) = 0. \tag{8}$$

(a) Staggered grid configuration, with staircase representation of the solid.

(b) Cell-centered grid configuration, with VOF treatment of the solid.

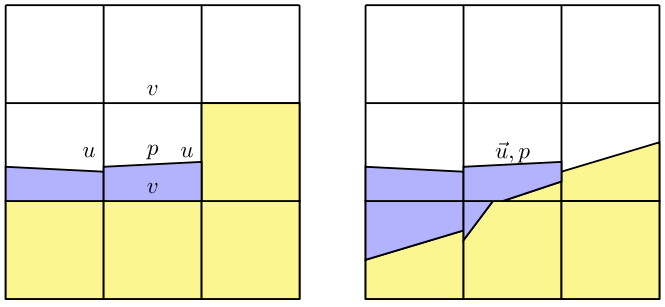


**Fig. 1.** Sketches of computational cell types used in the NS solvers. Solid fraction is sketched in yellow and the water fraction in blue. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Numerical solution of the above advection equation requires special treatment around the discontinuity in order to keep the interface sharp. Such a treatment may introduce an artificial compressibility term, either as a direct addition to the governing equation, or resulting from a numerical scheme. This procedure, however, always results in smoothing of the interface over one or more cells.

Alternatively, the advection of the volume fractions can be performed by a geometric method. In the PLIC method, which is an example of such an approach, the interface is defined as a plane in each cell containing a mixture of fluids. Once the interface is reconstructed, it is used for the computation of the fluxes on the faces of the computational cells (Tryggvason et al., 2011).

For OpenFOAM a few simulations on body fitted meshes are performed for the runup case. However, in order to ease the comparison of solvers in the benchmark tests, the main simulations presented in this paper were performed on regular, Cartesian meshes. For such meshes, a solid body must be described as immersed. In a VOF type approach, the solid is described by its volume fractions and the solid boundary is approximated with a plane in each crossed cell, see Fig. 1b. When no higher order description method is available, a staircase representation is used, see Fig. 1a. In this approach, the cells intersected by a solid boundary are treated as either full or empty, depending on the location of the intersection. Such a description corresponds to a physical boundary made with blocks of cell size, therefore it brings additional roughness to the solid surface. A sort of artificial roughness is expected from the VOF type description as well, as the reconstruction schemes do not necessarily describe an arbitrary linear interface in the exact way (Tryggvason et al., 2011), especially when more then two phase fractions (e.g. two fluids and a solid) are involved in a single cell.

### 2.3. Navier–Stokes solvers

#### 2.3.1. Gerris

Gerris is an open-source solver, with the solution procedure based on the projection method (Popinet, 2003). The solver is using quad/oc-tree structured grids, which enable the use of an advanced adaptive mesh refinement technique. This feature is, however, not used in this work. The flow is described by using the one-fluid formulation with cell-centered variables. The solver has support for two gravity models:

- *Source term* approach – gravity is applied in a whole domain as a source term in the momentum equation.
- *Reduced gravity* approach – gravity is applied in a bond around the interface, similar to the surface-tension force.

The second model is more common among Gerris example files, although it was found unstable for stretched meshes, which are, however, not used in the present work. Both models are evaluated in this work. The time-step is adaptive, with the maximum Courant number equal to 0.5 for the VOF advection and 1 for the flow. The Gerris solver with 2D libgfs library version 1.3.2 (130112-095727) is used in this paper. In the oceanographic and coastal engineering context, the Gerris solver has been used for the air–water flow around a Series 60 cargo ship (Popinet, 2013) and the simulation of landslide induced waves (Popinet, 2013; Viroulet and Kimmoun, 2012).

#### 2.3.2. OpenFOAM

OpenFOAM is an open-source solver, using unstructured meshes with cell-centered values (OpenFOAM Foundation, 2013). The solution procedure is based on the PISO algorithm and implicit time discretization. Advection of the volume fractions is based on the solution of the transport equation. The solver uses the one-fluid formulation of the velocity field. The information about the relative velocities in the mixed cells is used to define an additional convective term in the transport equation (Berberović et al., 2009). This term has a compressing

effect on the interface and enables the use of standard numerical schemes for the whole domain.

Due to the lack of an official immersed boundary module, the staircase representation is adopted for the solid, for comparison to the other solvers. The excessive mesh is removed from the computation by using the snappyHexMesh pre-processing tool, available in the package. The time-step is adaptive, with the maximum Courant number chosen as 0.5 for both the flow and the VOF advection. InterFoam solver version 2.1.1, from the official OpenFOAM package, is used in this paper. The simulations of wave generation, propagation and absorption in Higuera et al. (2013a, 2013b) and Jacobsen et al. (2012) are examples on application of the OpenFOAM solver for coastal engineering related flows.

#### 2.3.3. Thétis

The Thétis solver is based on the one-fluid formulation and uses structured meshes with a staggered placing of variables (Abadie et al., 2010; Glockner, 2011). It uses the augmented Lagrangian method for the pressure velocity coupling. In this paper, the VOF-PLIC method is exclusively used for the advection of the volume fractions, even though the VOF-TVD and the level-set methods are also implemented in the solver. A staircase solid boundary description is adopted, but, opposed to the OpenFOAM solver, the excessive region is not removed from the computation. The boundary is introduced by a penalization method, making the immersed solid impermeable for the fluid. A higher order approach exists but the authors where unsuccessful in using it. The adaptive time-step is not supported, therefore the average time-step found in other solvers is prescribed. In this work, the 2.1 version of Thétis is used. Thétis has been used by Abadie et al. (2010) for the simulation of landslide-generated waves with deformation of solid bodies.

#### 2.3.4. Truchas

Truchas is an open-source solver, based on the one-fluid formulation and using unstructured grids with cell-centered variables (Francois et al., 2006). The solution procedure is based on the projection method with explicit time discretization. The PLIC method is used for the advection of the volume fractions and a VOF type description is adopted for the solid boundaries. Details of the method, in particular related to the movement of the contact line, can be found in the manual distributed with the software (The Telluride Team, 2008). The time-step is adaptive, with the maximum Courant number chosen as 0.5 for both the flow and the VOF advection. Truchas version 2.6 is used in this paper. Liu et al. (2005) used Truchas for the simulation of surface waves generated by a sliding wedge.

### 2.4. Initialization and post-processing

Simulations performed for this paper are initialized by using solitary wave data. The surface velocity and the surface elevation of a fully non-linear solitary wave of a given amplitude are generated by the method of Tanaka (Tanaka, 1986). The velocities in the bulk of the fluid are then found by application of Cauchy's formula (Pedersen et al., 2013). The kinetic and potential energies of the generated waves of given amplitudes are given in Table 1. The initial data for solitary waves can also be based on analytical solutions (e.g. Grimshaw's/Fenton's third order solution (Fenton, 1972)), but they diverge from the fully non-linear solution with increasing amplitude to depth ratio, leading to additional perturbations at the initial phase of the simulations. A comparison of different analytical and numerical approaches to the generation of solitary waves is given in Lubin and Lemonnier (2004).

**Table 1**
Kinetic and potential energies of the solitary wave, given by the reference solution.

| | Kinetic energy [J] | Potential energy [J] |
|---|---|---|
| $a/d = 0.1$ | 254.689 | 245.137 |
| $a/d = 0.3$ | 1452.05 | 1317.17 |

The data fields of the solitary waves are prescribed either by using the solver's internal integration schemes (Gerris, Thétis), or with the authors' own pre-processing scripts (OpenFOAM, Truchas). The velocity has been prescribed only to the cells with non-zero water fraction.

In post-processing, the kinetic energy of the water fraction is computed by using the following sum over all computational cells:

$$E_k = \rho_w \sum_{i=1}^{N} \frac{V_i u_i^2}{2} f_{wi}, \tag{9}$$

where $\rho_w$ is density of water, $V$ is volume, $u$ is velocity and $f_w$ is volume fraction of water. Index $i$ denotes the cell. This approach is slightly modified for Thétis, in which output (VTK files) is given in points, instead of cells. The cell values were obtained for Thétis by mapping of point data onto cell data by means of the Visualization Toolkit (VTK) routines (Kitware Inc., 2013).

The computation of the potential energy is not directly based on the volume fractions, as it would lead to wrong results in the case of OpenFOAM in which the volume fraction field is smoothed out over several cells around the interface. Instead, the numerical integration of the interface is performed, using the formula:

$$E_p = \frac{1}{2}\rho_w g \sum_{j=1}^{M-1} \left(\eta_j + \eta_{j+1}\right)^2 \left(x_{j+1} - x_j\right), \tag{10}$$

where the interface is given by $M$ points $(x_j, \eta_j)$. The exact location of the interface is estimated as a contour line marking a 0.5 level of the volume fraction field. Extracting a contour is straightforward if the point data is given at the output (OpenFOAM, Thétis). If only cell data is available (Gerris, Truchas), interpolation from cell data to point data is performed first. Both operations are done by using VTK.

Computation of energy, using the contour line approach presented above, is performed only for the propagation of a solitary wave. The procedure would require modifications for the runup case.

## 3. Benchmark tests

This section describes the two benchmark tests performed for this paper. In both problems, two heights of a solitary wave are studied: amplitude to depth ratio ($a/d$) equal to 0.1 and 0.3. The water depth $d$ is set to 1 m. To study the convergence of the results, three subsequently refined grids are used for each problem.

### 3.1. Propagation of a solitary wave

In this benchmark test, employed in Lubin and Lemonnier (2004), a solitary wave propagates over a certain distance. A solitary wave should ideally maintain a constant level of energy, when disregarding the dissipation by the boundary layers at the bottom and at the interface between water and air. However, finite volume solvers are expected to suffer from numerical viscosity, resulting in dissipation and reduction of the wave energy even for zero physical viscosity. The goal of this investigation is to estimate the influence of the numerical dissipation, as well as other spurious effects, e.g. spurious currents. Since the solvers simulate also the flow in the air phase, transfer of momentum and energy to the air phase might add to the numerical dissipation. For these reasons, the evolution of the total mechanical energy of the wave, being the sum of the kinetic and potential energies, is studied in detail.

Propagation of a solitary wave has been used as a validation for Truchas by Wu (2004) for a single amplitude of $a/d = 0.1$ and a single resolution. Without going much into detail, he reports that the results match the reference solution 'very well'. Lubin (2004) performed a more thorough analysis for the Aquilon solver, a predecessor to Thétis.

The present study is similar to Lubin's with the addition, that we compare different solvers with each other and analyze the convergence. Moreover, we have a closer look at the behavior of the total mechanical energy of the system.

In our test, solitary waves of two heights are being propagated for 12 s, with the initial position at $x = 15$ m. Sizes of the computational domains and parameters of the grids are collected in Table 2. The final position of the wave is around 15 m away from the end of the computational domain.

### 3.2. Runup of a solitary wave

In this benchmark test, the simulation is initiated with the same solitary waves as for the propagation, but this time the waves run up on a plane beach, see Fig. 2. As the wave reaches the maximum height, a thin film of water is generated on the beach. This setup, relatively simple for a boundary integral solver, brings difficulties to finite volume solvers.

The test is performed mainly in two configurations (Fig. 3). In the first, the domain is horizontal, and the beach is introduced as an immersed solid. In the other configuration the domain is rotated, so that one of the mesh boundaries becomes the beach, while a solid is used for the bottom of the channel. Comparison of these two configurations gives us a quantitative estimate on the accuracy of solid–fluid interactions, which includes interface advection in the vicinity of a solid. As was explained in Section 3, the Gerris and Truchas solvers possess higher order schemes for immersed solids, while OpenFOAM and Thétis rely on the staircase representation.

Solitary wave runup has also been used by other authors (Biausser et al., 2003; Lin et al., 1999; Lubin, 2004) in order to investigate mainly breaking waves. However, in the present discussion, non-breaking waves are used, since they allow comparison with the reference solution. Lin et al. (1999) did also computations for the case of a non-breaking wave on a 30-degree beach and observe good agreement between their inviscid Navier–Stokes solver and their boundary integral equation solver. A detailed comparison, not to mention an analysis of the numerical error for the runup in Lin et al. (1999) is, however, lacking. We shall see that, in addition, the 30-degree case is basically not enough to thoroughly test the solvers.

In the present work, the reference solution was obtained by means of a fully non-linear boundary integral solver based on potential theory. The boundary integral method used in this discussion is similar to the one in Dold (1992) and Cooker et al. (1990), with the exception that the high order polynomials are replaced by cubic splines, which allows us to extend the method to runup calculations. More details on the implementation can be found in Pedersen et al. (2013). This method is very accurate for the runup, provided that the numerical integrals of the swash tongue are computed carefully. However, during withdrawal, when the flow depth becomes extremely small and a shoreward facing bore is forming, the simulations can eventually become inaccurate and then unstable.

The runup of a solitary wave was performed for two angles of the inclination of the beach: 10 and 30°. The key parameter used for the evaluation of the solvers is the runup height, which is the maximum elevation of the free surface recorded during a simulation. The reference runup height to depth ratios, computed by means of the boundary integral solver, for the 10-degree inclination are equal to 0.369 for an $a/d = 0.1$ solitary wave and 1.275 for $a/d = 0.3$. For the inclination of 30° and $a/d = 0.3$, the reference runup height is 0.869. The accuracy of the reference data was verified up to the third decimal by grid refinement tests.

**Table 2**
Mesh parameters used for simulations of the solitary wave propagation.

| $a/d$ | 0.1 | 0.3 |
|---|---|---|
| $L \times H$ | 69 m × 1.5 m | 72.6 m × 2.2 m |
| Resolution 1 | 1472 × 32 | 1056 × 32 |
| Resolution 2 | 2944 × 64 | 2112 × 64 |
| Resolution 3 | 5888 × 128 | 4224 × 128 |

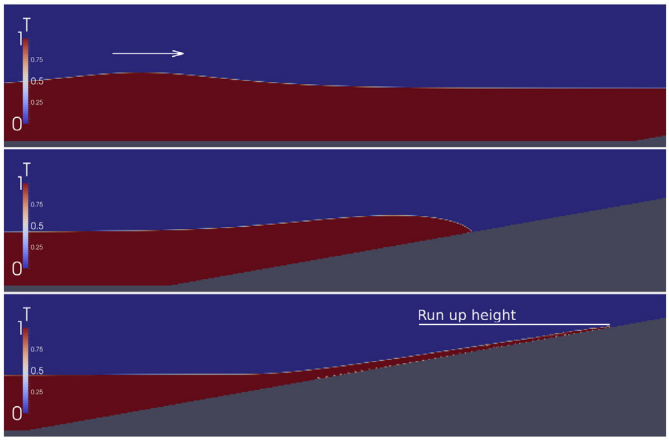


**Fig. 2.** Setup of the solitary wave runup problem. The picture presents results obtained with the Gerris solver, for the best resolution. The height of the wave is $a/d = 0.3$. Snapshots were taken at $t = 0$, 3.75 and 6 s, respectively.

The parameters of the grids used for computations are collected in Table 3. In all cases, the initial position of the solitary wave is 15 m away from the beach. The times of simulation are set to 8 s for the 10-degree slope and 6 s for the 30-degree slope. The simulations end soon after the maximum height was achieved. The Navier–Stokes solvers used for the benchmark tests are able to simulate the withdrawal of the wave, with possible breaking, but in this study we concentrate on the area well covered by the reference solution. It needs to be said that some simulations by some solvers did blow up during simulation of the withdrawal whereas the runup went fine with the chosen set of parameters.

The order of convergence is estimated by means of the following formula:

$$O = \frac{\ln \frac{\epsilon_N}{\epsilon_{2N}}}{\ln 2} \tag{11}$$

where $\epsilon_N$ is the runup height error, defined as:

$$\epsilon_N = \frac{h_{\max} - h_{\text{ref}}}{h_{\text{ref}}} * 100 \tag{12}$$

where $h_{\max}$ is the runup height, achieved for a particular mesh, and $h_{\text{ref}}$ is a reference value. In the *2N* case, as compared to the *N* case, the size of the mesh is doubled for every dimension.

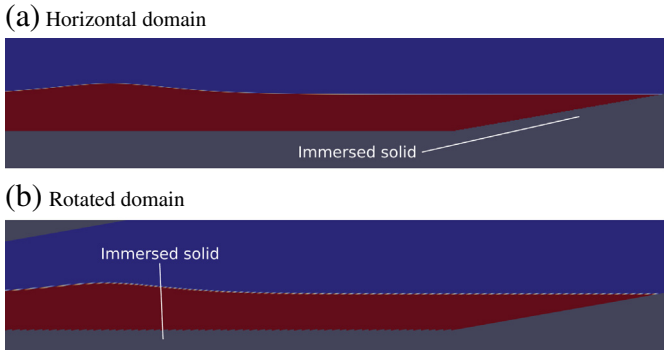


**Fig. 3.** Two configurations for the runup problem. The volume occupied by the immersed solid is not shown, whereas the colors indicate the volume fraction field. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

**Table 3**
Mesh parameters used for simulations of the solitary wave runup.

| | Horizontal | | Rotated | |
|---|---|---|---|---|
| $a/d$ | 0.1 | 0.3 | 0.1 | 0.3 |
| $L \times H$ | 34 m × 1.7 m | 39 m × 2.6 m | 35.1 m × 4.4 m | 40.5 m × 4.5 m |
| Resolution 1 | 640 × 32 | 480 × 32 | 256 × 32 | 288 × 32 |
| Resolution 2 | 1280 × 64 | 960 × 64 | 512 × 64 | 576 × 64 |
| Resolution 3 | 2560 × 128 | 1920 × 128 | 1024 × 128 | 1152 × 128 |

As mentioned in the beginning of this section, the benchmark test is performed mainly in two configurations, the horizontal and the rotated one. In order to give the reader an impression of how the results change when using a body fitted grid, we added a part, Section 4, on a simulation using OpenFOAM on simple body fitted meshes. A full discussion of the body fitted case is, however, beyond the scope of the present work, since it would need to address questions about the type of grid, e.g. structured, unstructured, about the type of cells, e.g. hexahedral and tetrahedral, or about the quality of the grid, e.g. optimization with respect to orthogonality and cell size.

The present benchmark is a highly idealized case, as flows with zero viscosity are very seldom. Nevertheless, it tells us already a lot about the functioning or nonfunctioning of the basic parts of the solvers. More advanced modeling, like turbulence modeling or modeling of the bottom friction effects will add a layer of difficulties which would need to be tested independently. Testing of these more advanced features is, however, not the aim of the present discussion. In order to give the reader an impression about how these additional model features add to the problem, we performed a couple of basic simulations using OpenFOAM on the problem of the viscous runup on a beach of 10-degree inclination, cf. subsection 5.

## 4. Results and discussion

### 4.1. Propagation of a solitary wave

All the solvers were able to propagate a solitary wave over a given distance in the form of a single crest. From Table 4, collecting the final heights of the solitary waves recorded at the end of the simulations, we see that in every case the results seem to converge with refinement to the nominal height of the wave. Nevertheless, for some solvers the final heights are lower than the nominal value, while for others they are higher. The latter behavior requires further study, as the growth of the solitary waves is alarming.

In addition to the wave heights, we recorded the final position of the crest and the final full-width at half maximum of the wave, cf. Table 5. From this table, we observe that for most of the solvers the solitary waves travel with the correct speed. However, for those solvers displaying an increased amplitude (Gerris with source term gravity and Thétis), we observe also a higher simulated speed of the solitary wave. For the full-width at half maximum, we observe for the same solvers that the wave shortens during propagation. These facts call for further discussion.

**Table 4**
Final heights of a solitary wave after the propagation, normalized by the nominal value of the amplitude. Normalized times of computation.

| | $h_{fin}/a$ | | | $t_{comp}/t_{ref}$ | | |
|---|---|---|---|---|---|---|
| Resolution | 1 | 2 | 3 | 1 | 2 | 3 |
| *(a) [$a/d = 0.1$, $t_{ref1} = 183$ s, $t_{ref2} = 1128$ s, $t_{ref3} = 9261$ s.]* | | | | | | |
| Gerris (source term gravity) | 0.999 | 1.044 | 1.006 | 1.85 | 2.86 | 2.58 |
| Gerris (reduced gravity) | 1.051 | 1.005 | 1.003 | 2.29 | 2.57 | 2.1 |
| OpenFOAM | 0.986 | 0.987 | 0.991 | 1 | 1 | 1 |
| Thétis | 1.029 | 1.020 | 1.007 | 4.68 | 4.36 | 3.12 |
| Truchas | 1.020 | 1.008 | 1.005 | 15 | 17.8 | 16.1 |
| *(b) [$a/d = 0.3$, $t_{ref1} = 153$ s, $t_{ref2} = 1355$ s, $t_{ref3} = 11{,}763$ s.]* | | | | | | |
| Gerris (source term gravity) | 1.114 | 1.030 | 1.006 | 4.41 | 2.58 | 2.04 |
| Gerris (reduced gravity) | 0.977 | 0.982 | 0.985 | 3.05 | 3.35 | 2.84 |
| OpenFOAM | 0.989 | 0.992 | 0.995 | 1 | 1 | 1 |
| Thétis | 1.144 | 1.056 | 0.991 | 3.63 | 2.19 | 1.48 |
| Truchas | 1.034 | 1.017 | 1.008 | 13.5 | 12.3 | 20.9 |

**Table 5**
Crest positions and full-width at half maximum of the solitary waves after propagation, normalized by the reference values for the position and the reference width, respectively.

| | $x_{fin}/x_{ref}$ | | | $\Delta x_{fin}/\Delta x_{ref}$ | | |
|---|---|---|---|---|---|---|
| Resolution | 1 | 2 | 3 | 1 | 2 | 3 |
| *(a) [$a/d = 0.1$, $x_{ref} = 39.41$ m, $\Delta x_{ref} = 6.62$ m.]* | | | | | | |
| Gerris (source term gravity) | 1.016 | 1.008 | 1.004 | 0.990 | 1.012 | 1.003 |
| Gerris (reduced gravity) | 0.999 | 0.998 | 1.000 | 0.976 | 0.968 | 0.990 |
| OpenFOAM | 0.999 | 0.999 | 0.999 | 1.011 | 0.997 | 1.001 |
| Thétis | 1.020 | 1.013 | 1.008 | 0.989 | 0.951 | 0.982 |
| Truchas | 1.001 | 0.999 | 1.000 | 0.990 | 0.998 | 0.999 |
| *(b) [$a/d = 0.3$, $x_{ref} = 42.75$ m, $\Delta x_{ref} = 3.95$ m.]* | | | | | | |
| Gerris (source term gravity) | 1.027 | 1.012 | 1.005 | 0.940 | 0.978 | 0.991 |
| Gerris (reduced gravity) | 1.000 | 0.999 | 0.998 | 0.988 | 0.998 | 1.000 |
| OpenFOAM | 0.999 | 0.999 | 0.999 | 0.987 | 0.998 | 0.998 |
| Thétis | 1.039 | 1.020 | 1.006 | 0.780 | 0.945 | 0.975 |
| Truchas | 0.999 | 1.000 | 1.000 | 1.004 | 1.002 | 1.002 |

Fig. 4 collects the plots of the evolution of the total energies of the waves in the $a/d = 0.3$ case. The energies were normalized with the reference values, given in Table 1, therefore the initial level of energy is expected to be equal to 1. This requirement is not fully satisfied in the case of Thétis (Fig. 4d), due to the mapping of point values onto cell values during post-processing, described in subsection 4. The situation improves with refinement, and the overall performance for all the solvers lets us believe that the initialization and post-processing were done in a proper way. The simulations for $a/d = 0.1$ reflected similar behavior.

The evolution of the energy of the system reveals several different patterns. Simulations performed with Gerris with the source term gravity model, and with Truchas (Fig. 4a and e) result in the evident growth of energy. In OpenFOAM a kind of balance is observed, with the energy staying at a constant level after initial perturbations (Fig. 4c). In Thétis the behavior is unpredictable with regards to refinement, as the dissipation rate is definitely smallest for resolution 2 (Fig. 4d). Gerris, with the reduced gravity model in use, produced results with a linearly decreasing level of energy (Fig. 4b) and with the numerical dissipation rate decreasing with refinement.

One of the reasons for the dissipation of energy may be the numerical bulk viscosity of the discretization scheme of the convective terms. This source of dissipation is supposed to be present in all of the tested solvers, even if a net growth of energy is observed. Extensive numerical dissipation is undesired, as it can limit the usability of the solver, but is safer than the uncontrolled growth of the energy. For this reason, the results given by the Gerris solver with the reduced gravity model (Fig. 4b), are considered the most stable and reliable, even though the final energies are the lowest observed among the solvers.

The estimation of the numerical dissipation rate level needs to be taken with care, since for all the solvers different discretization schemes for the convective terms can be chosen. For the above results only the upwind scheme has been tested.

To better understand the issues behind the growth of energy, we studied the velocity profiles along the vertical cross-section going through the crests of the solitary waves. The time is chosen as $t = 10$ s, so that the initial perturbations due to the initialization, visible in Fig. 4, have been smoothed out. Fig. 6 comprises the velocity profiles for the solitary waves with $a/d = 0.3$. The results obtained with all the solvers, except for Truchas and Gerris with the reduced gravity model, share a common feature, which is an increase of velocity in water fraction close to the interface, see Fig. 6a, c and d.

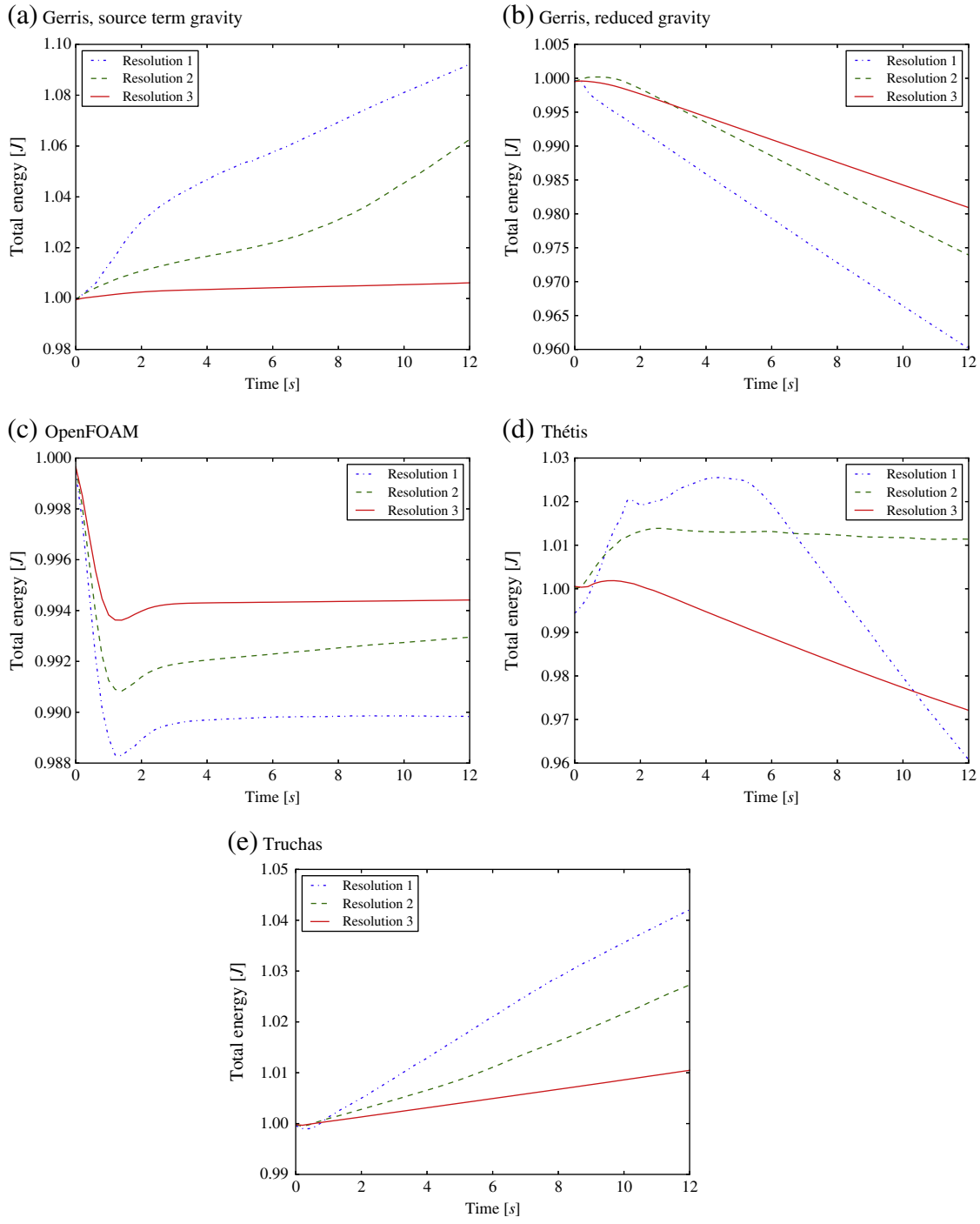


**Fig. 4.** Evolution of kinetic energy in water fraction during propagation of an $a/d = 0.3$ solitary wave.

The increase of velocity around the interface may seem surprising, as one would rather expect a development of a spurious boundary layer, smoothing out the jump of velocity at the free surface and acting as an additional source of numerical dissipation. In our case, the increase of velocity resembles the Gibbs phenomenon. The oscillation is most evident in the case of Thétis (Fig. 6d). The reason for such behavior may lie in the discontinuity of the parameters of the fluids, but the exact mechanisms are not pursued further.

The presence of spurious currents around an interface was reported by several authors, especially for cases with static equilibrium in surface tension driven flows (Francois et al., 2006; Popinet, 2009; Raeini et al., 2012). Recently, Wemmenhove et al. (Wemmenhove et al., 2012)

**Table 6**
Maximum deviation of the profiles in Fig. 6 with respect to the reference solution. The maximum has been sought among the cells completely submerged in the water phase, i.e. having a volume fraction of 1. The values are for resolution 3.

| Solver | $\max_y \lvert u - u^{ref} \rvert [m/s]$ |
|---|---|
| Gerris (source term gravity) | 0.358 |
| Gerris (reduced gravity) | 0.023 |
| OpenFOAM | 0.115 |
| Thétis | 0.396 |
| Truchas | 0.005 |


**Fig. 5.** Velocity field in the case of an $a/d = 0.1$ solitary wave during propagation, with a high velocity cell visible at the front face of the wave. Results obtained with Truchas.

suggested that similar problems are present in the case of gravity driven flows with interfaces at equilibrium, not aligned with the mesh. The source of the problem is believed to lie in the application of gravity in the mixed cells, and a new averaging of density at faces is suggested as a possible solution. These findings seem to relate to our problem.

The increase of velocity is visible in the vicinity of the interface, while in the bulk of the fluid the velocities follow the reference solution reasonably well (Fig. 6). Comparison of the two gravity models used in Gerris suggests, that the way the body forces are applied can have a dramatic influence on the results. The source term gravity model induces strong spurious currents around the interface and growth of energy of the system (cf. Figs. 6a and 4a), while the reduced gravity model is almost free of spurious currents and reflects no signs of energy production (cf. Figs. 6b and 4b). In Table 6, we list the maximum deviation of the profiles in Fig. 6 with respect to the reference solution for the finest resolution. The values are taken only for cells entirely submerged in the water phase. As for most solvers, except Truchas, the overshoot is much larger than the standard discretization error of the solver, the values in Table 6 represent therefore a measure of the overshoot of the profile. Since the characteristic particle velocity is about $1 m/s$, we

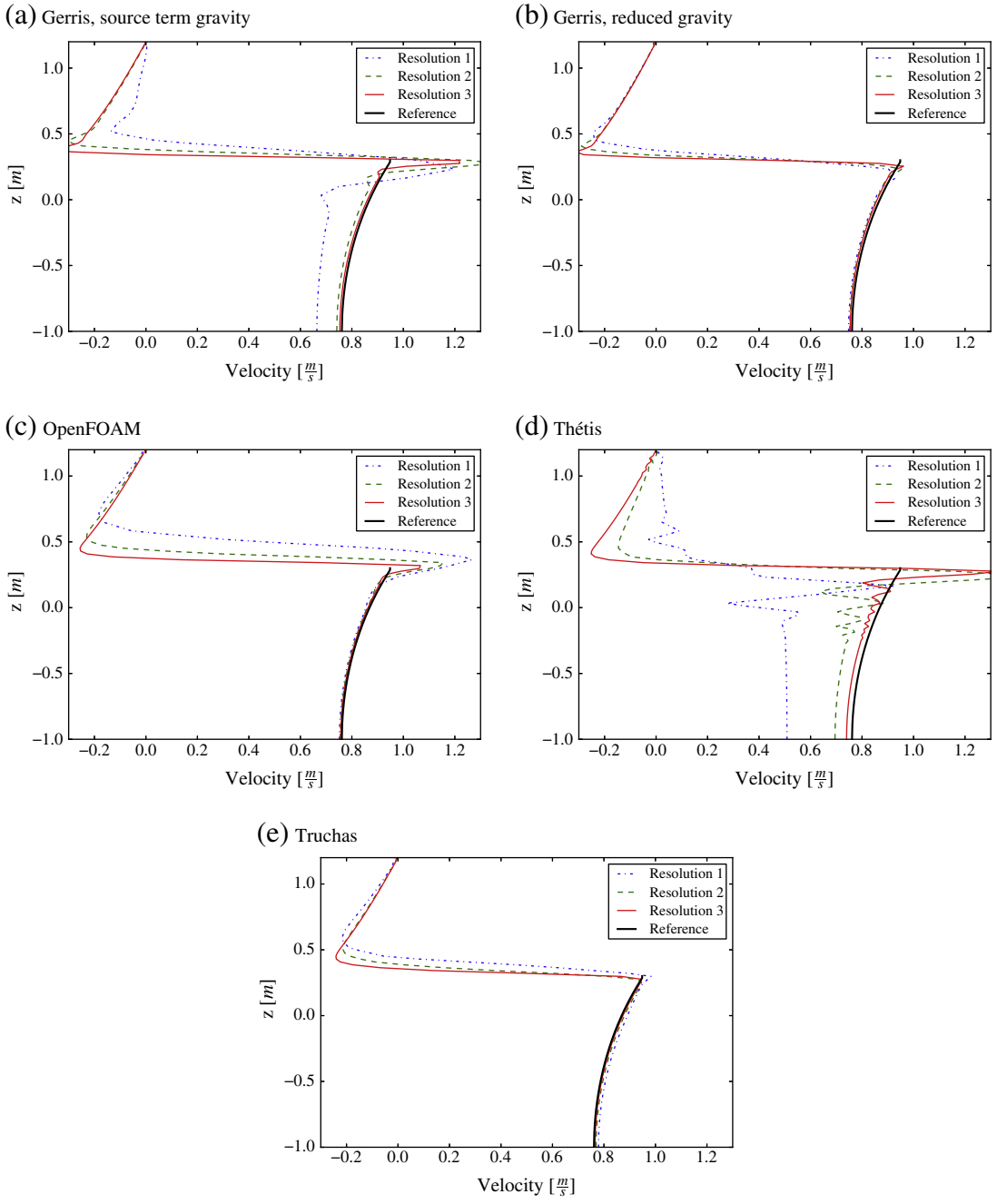


**Fig. 6.** The horizontal velocity profiles along a vertical cross-section going through the crest of a propagating $a/d = 0.3$ solitary wave, after 10 s of propagation.

see that the relative error due to the overshoot can be from less than 1% to about 40% depending on the solver.

The velocity profiles obtained with Truchas seem to follow the reference solution very well. Nevertheless, a closer look at the results (Fig. 5) reveals, that single cells with artificially high velocities are always present at the front of a solitary wave, which may be the source of the observed increase of energy (Fig. 4e). The origin of these spurious velocities has not been identified.

A strong increase of velocity is observed in OpenFOAM (Fig. 6c), compared to the reference. It has to be stressed here, that the reference solution is valid for a problem with a sharp discontinuity of density at the interface. In OpenFOAM, the transition between the fluids is distributed over several cells. In this case, the surface dynamics may be reminiscent of a physically continuous stratification, altering the transfer of momentum between the water and air fractions, as compared to a jump. The detailed study of these mechanisms is beyond the scope of this work, but it has to be taken into account that the increased velocities observed in OpenFOAM may be due to a different nature of the problem that is actually simulated (Deshpande et al., 2012). The observed stable level of energy seems to support this assumption (cf. Fig. 4c). For each solver, the magnitude of spurious velocities becomes smaller with refinement and the velocity profile converges to a reference solution. Large discrepancies visible for the lower resolutions in the case of Gerris with the source term gravity model (Fig. 6a) and Thétis (Fig. 6d), find reflection in a high production of energy (cf. Fig. 4a) and a large increase of height of the wave during propagation (cf. Table 4).

A feature of the VOF method in OpenFOAM is the compression of the interface in normal direction to limit diffusion of the volume fraction field. This compression is controlled by the parameter $c_\alpha$, which in our case has been chosen to be $c_\alpha = 1$, corresponding to conservative compression (OpenFOAM Foundation, 2013). In Fig. 7, profiles of the final horizontal velocity are compared for $c_\alpha = 0.5$ and $c_\alpha = 2$ for the case $a/d = 0.3$. Comparing to Fig. 6c, the profiles for the different values of $c_\alpha$ are relatively similar. For $c_\alpha = 1$, we observe that below the interface the profiles follow the reference profile somewhat better than for the other values of $c_\alpha$. The reason might be that for lower values of $c_\alpha$ the compression is weaker and the volume fraction tends to diffuse more into the air region making the transition between the phases less steep. Somewhat surprising is the fact that for $c_\alpha = 0.5$, the overshoot at the interface is larger than for enhanced compression. Compared to enhanced compression, $c_\alpha = 2$, the profiles do not follow the reference profile as well as for $c_\alpha = 1$. In this respect the conservative compression $c_\alpha = 1$ seems to be a good value for two phase flow simulations.

The usability of the solver can be limited by the computational time it requires for a particular problem. Comparing computational times collected in Table 4, we see that OpenFOAM is definitely the fastest solver tested, while Truchas is up to an order of magnitude slower than the rest of the solvers.

Gerris, with the reduced gravity model used, seems to produce the most reliable results among all the solvers, both in sense of the energy dissipation and the velocity profile. Nevertheless, decrease of the influence of the spurious effects with grid refinement, observed among all the other solvers, entitles them for further use, even though special care is advised. It is feared, that spurious effects can have a dominant role in long time simulations, balancing the numerical viscosity in the best case, and blowing up the simulation in the worst. The source-term gravity model in Gerris brings much worse results then the reduced gravity model, therefore only the latter is used in the further sections.

### 4.2. Runup of a solitary wave

#### 4.2.1. Immersed solid beach, 10-degree slope

After the study of energy dissipation during propagation of a solitary wave, we proceed to investigate the abilities of the solvers to simulate the runup of a solitary wave on a plane beach. The beach is modeled as an immersed solid and is inclined by an angle of 10°. The recorded runup heights are collected in Table 7. The growth of the runup height with increasing amplitude, apparent for the reference solution, is reproduced by the Navier–Stokes solvers. Still, all the computed runup heights are lower than the reference values and the average error grows with the height of the wave. The runup height errors, computed for the best resolutions using formula (12), are around 10% for the $a/d = 0.1$ case, whereas they are as much as around 30% for $a/d = 0.3$, see Fig. 9 and Table 7. The possible factors responsible for the underestimation of the runup heights are studied in the next paragraphs.

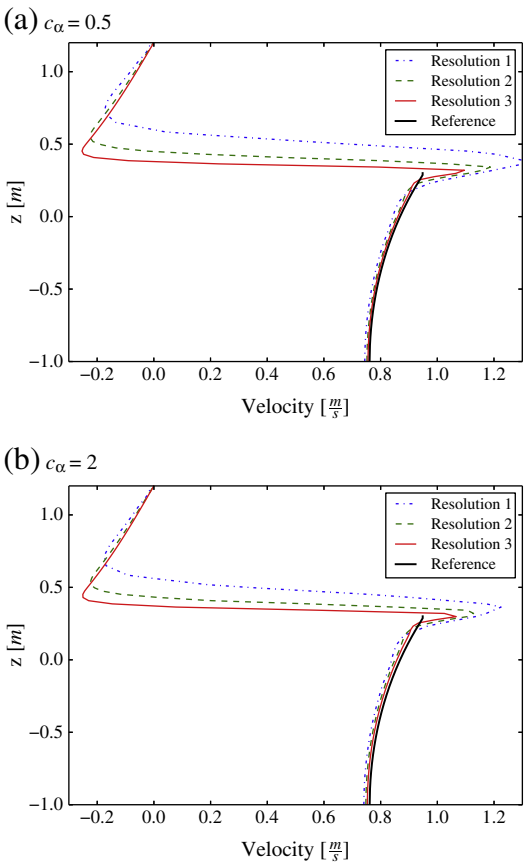


**Fig. 7.** The horizontal velocity profiles along a vertical cross-section going through the crest of a propagating $a/d = 0.3$ solitary wave, after 10 s of propagation using the OpenFOAM solver for different values of $c_\alpha$.

**Table 7**
The solitary wave runup heights and times of computation. The immersed solid beach is inclined by an angle of 10°.

| | $h_{max}/h_{ref}$ | | | $t_{comp}/t_{ref}$ | | |
|---|---|---|---|---|---|---|
| Resolution | 1 | 2 | 3 | 1 | 2 | 3 |
| *(a) [$a/d = 0.1$, reference value equals $h_{ref} = 0.369$ m, $t_{ref1} = 48$ s, $t_{ref2} = 315$ s, $t_{ref3} = 2888$ s.]* | | | | | | |
| Gerris | 0.824 | 0.917 | 0.941 | 5.83 | 7.49 | 5.48 |
| OpenFOAM | 0.713 | 0.795 | 0.852 | 1 | 1 | 1 |
| Thétis | 0.866 | 0.925 | 0.914 | 207 | 107 | 52.6 |
| Truchas | 0.740 | 0.804 | 0.839 | 212 | 212 | 129 |
| *(b) [$a/d = 0.3$, reference value equals $h_{ref} = 1.275$ m, $t_{ref1} = 41$ s, $t_{ref2} = 308$ s, $t_{ref3} = 2949$ s.]* | | | | | | |
| Gerris | 0.672 | 0.754 | 0.785 | 6.61 | 7.22 | 5.29 |
| OpenFOAM | 0.474 | 0.545 | 0.615 | 1 | 1 | 1 |
| Thétis | 0.482 | 0.565 | 0.627 | 91.9 | 61.6 | 35.4 |
| Truchas | 0.528 | 0.540 | 0.630 | 141 | 139 | 158 |

Fig. 8 shows the evolution of the maximum elevation of the free surface for the more difficult, $a/d = 0.3$ case. The results improve with refinement, but slowly. The low convergence rate is also demonstrated in Fig. 9, presenting the errors of the results for both amplitudes. Nevertheless, the error decreases, with an exception of Thétis, which is not displaying a monotonic behavior in the $a/d = 0.1$ case. The reason for the low convergence rate in the studied problem may be the basic difficulty in the advection of the interface close to the solid. The runup leads to the formation of a thin, elongated swash tongue, due to the small inclination of the beach and the high amplitude of the solitary wave. Regardless of the refinement, always some elongated, single-cell thick features appear, challenging the performance of the VOF method. For all the solvers, the swash tongue splits into several parts in the early draw down, and often already during the runup. In Truchas, blowing up of the simulation happened several times, due to extensive currents, ranging across the entire air fraction, initiated at the location of the splitting of the water film. In Gerris, small bubbles stuck to the beach during runup, and moved downstream during draw down, influencing the flow. Thétis revealed some local spurious currents and movement of the free surface close to the contact line, which were responsible for the deviation from the reference solution, visible in Fig. 8c just before the runup actually starts. All the observed problems are believed to be due to strong local effects present in the close neighborhood of the solid boundaries. A main issue may be the changing stencil of the VOF-PLIC reconstruction scheme, close to a solid or a boundary, as no essential spurious behavior was observed in the bulk of the fluid in any case. In this light, OpenFOAM, which solves the transport equation for the volume fractions, is found to be the most stable, robust and reliable solver. The VOF advection method used by the solver leads to an increased smoothing of the volume fractions during the runup (see Fig. 10b). In more complex situations this smoothing may lead to ambiguity in the definition of the runup. However, in the present case the splitting of the runup tongue, most pronounced for the other solvers, caused more severe problems. In order to quantify the spurious effects due to the presence of an immersed boundary, we run the above test case for a static interface at equilibrium for a similar duration as for the solitary wave runup. We measured total kinetic energy and maximum height of the interface after 8 s. The values are listed in Table 8. As such the magnitude of the spurious effects due to the presence of the immersed boundary seems to be rather limited. However, as we will see in Section 2, once the interface is not anymore aligned with the grid axes, spurious currents can become important in magnitude.

Separation of spurious droplets from the tip of the swash tongue can lead to over-shots, as, after the release, they can move in a spurious way. This problem is mostly common in Gerris and Thétis (Fig. 10a), but is partially present in OpenFOAM (Fig. 10b) and Truchas as well. Problems with small droplets, considered as numerical noise, are often reported by Gerris users and are often solved by removal of droplets of certain size during the simulation. While other solvers do not have such features, the procedures available in Gerris are not found satisfactory, as the spurious droplets present in our problem become very elongated along the beach, resulting in the need of removal of droplets as big as 10 cells. To address the spurious droplets issue, and to enable a fair comparison between the solvers at the same time, only the largest connected water region is taken into account in the computation of the free surface (cf. Section 4), while no droplets are being removed during

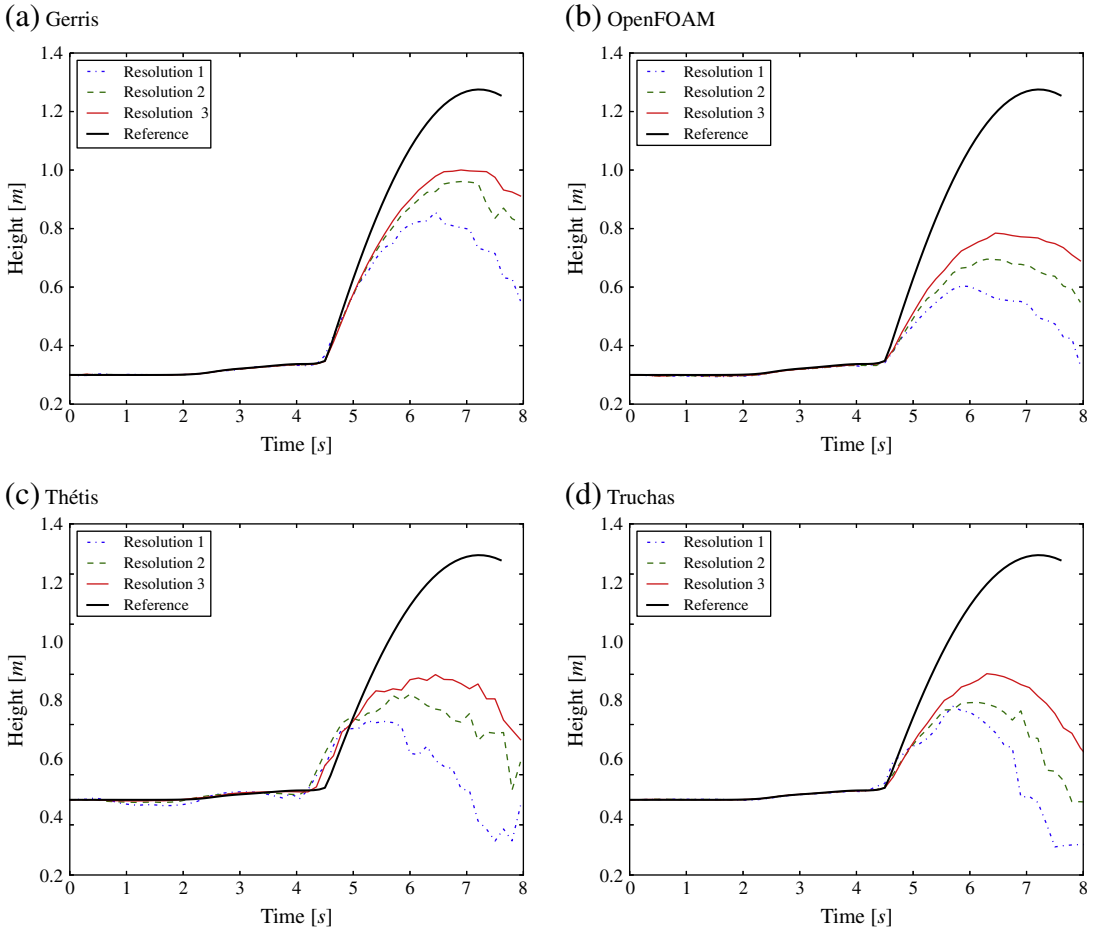


**Fig. 8.** Evolution of the maximum elevation of free surface during runup of an $a/d = 0.3$ solitary wave on an immersed solid beach, inclined by an angle of 10°.

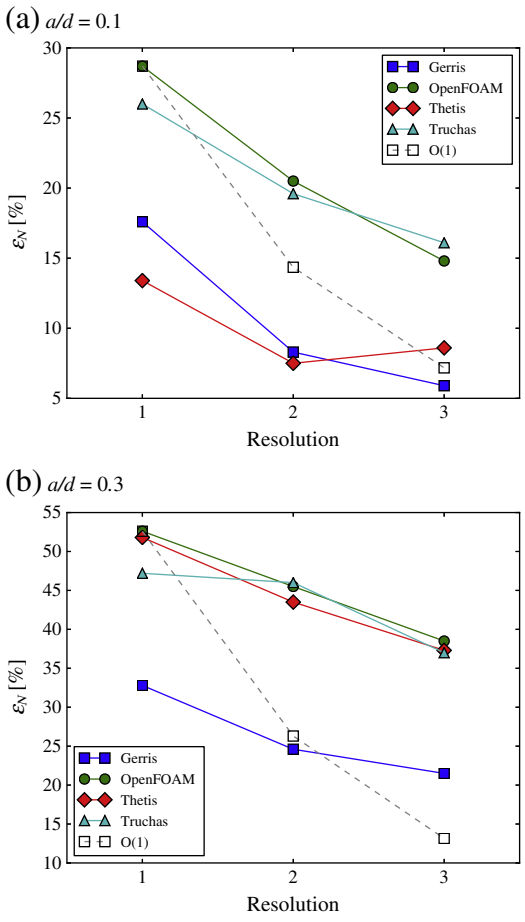


**Fig. 9.** The runup height errors, computed for the immersed solid beach inclined by an angle of 10°. The dashed line stands for the first order of convergence.

the simulations. This method is justified for the computation of the runup height, as the boundary integral method shows the runup tongue to be smooth and not fragmented for all times during runup.

The accuracy of the results may be influenced by the representation of the boundary. Definitely the best results were obtained with Gerris, in which a VOF type treatment of solids is implemented. However, no increased accuracy was observed in Truchas, which is surprising, as the immersed solids are treated in a similar way (see Section 2). It must be noted, that the consistency between the immersed solid implementation and the volume fraction advection scheme may be an important factor in this context, being responsible for the movement of the contact line. The artifacts, reported in the previous paragraph, may imply that the coupling between these two modules of the code may not be perfect in any of the solvers, resulting in the formation of bubbles and droplets in Gerris, as well as increased probability of blowing up as a result of the splitting of the water film in Truchas. As we will see in Section 4, OpenFOAM gives much better results in the case of a body fitted mesh. However, good body fitted meshes can be problematic to generate for very complex geometries.

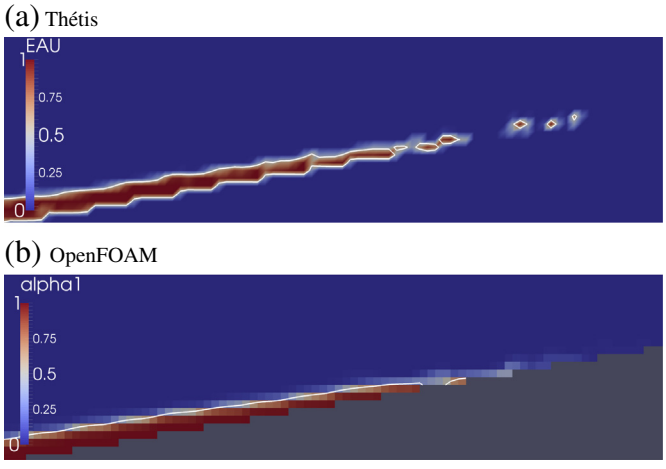


**Fig. 10.** The tip of the swash tongue, obtained during runup of an $a/d = 0.3$ solitary wave on the 10-degree immersed solid beach. Best resolution. White line marks the 0.5 level of the volume fraction field. The droplets were not taken into account during computation of the run up height. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

**Table 8**
Total kinetic energy and water height $h_{fin}$ after 8 s of simulation for a static interface at equilibrium in contact with the 10-degree immersed solid beach; $h_{ref} = 1m$. The kinetic energy of a solitary wave with amplitudes $a/d = 0.1, 0.3$ can be found in Table 1. The zero values for Thétis for resolutions 1 and 2 do not mean that the kinetic energy was actually zero. Its value must just have been below the number of digits recorded.

| | Total kinetic energy [J] | | | $h_{fin}/h_{ref}$ | | |
|---|---|---|---|---|---|---|
| Resolutions | 1 | 2 | 3 | 1 | 2 | 3 |
| Gerris | $5.4\cdot10^{-1}$ | $1.6\cdot10^{-1}$ | $9.3\cdot10^{-1}$ | $1.4\cdot10^{-2}$ | $5.6\cdot10^{-3}$ | $2.8\cdot10^{-3}$ |
| OpenFOAM | $3.3\cdot10^{-5}$ | $2.9\cdot10^{-5}$ | $5.5\cdot10^{-4}$ | $2.2\cdot10^{-5}$ | $4.4\cdot10^{-5}$ | $8.4\cdot10^{-5}$ |
| Thétis | 0.0 | 0.0 | $4.9\cdot10^{-2}$ | $5.4\cdot10^{-3}$ | $-1.7\cdot10^{-3}$ | $3.4\cdot10^{-3}$ |
| Truchas | $1.8\cdot10^{-3}$ | $1.2\cdot10^{-5}$ | $4.1\cdot10^{-5}$ | $1.5\cdot10^{-4}$ | $3.2\cdot10^{-4}$ | $5.6\cdot10^{-5}$ |

All the spurious effects appearing in the simulations make it difficult to compare the efficiency of the solvers, as they always increase the computational effort by decreasing the time-step due to the CFL condition. Moreover, an increase of the number of iterations in the pressure solver was observed in many cases, connected with the appearance of spurious droplets. Besides the artifacts already mentioned, Truchas and Thétis are particularly prone to droplets sticking at the beach during withdrawal. The water fraction of a partially filled cell, instead of being advected downwards the beach, sticks and induces spurious currents in the air fraction. Nevertheless, computational times collected in Table 7 show, that OpenFOAM is again the fastest solver, as in the case of propagation. Gerris is almost as fast as OpenFOAM, Thétis is on the average more than an order of magnitude slower, while Truchas can be almost two orders of magnitude slower. The difference in computational times decreases for finer meshes.

The runup of a solitary wave on an immersed solid beach inclined at an angle of 10° was found to be a relatively difficult problem for the tested finite volume solvers. In order to gain a broader perspective on the possible sources of difficulties, as well as the usability of the solvers, two variations of the problem are studied: runup on the boundary of a domain rotated by 10°, and runup on an immersed solid beach inclined by 30°.

**Table 9**
The runup heights and times of computation, obtained for the runup of a solitary wave on a 10-degree slope in the rotated configuration.

| | $h_{max}/h_{ref}$ | | | $t_{comp}/t_{ref}$ | | |
|---|---|---|---|---|---|---|
| Resolution | 1 | 2 | 3 | 1 | 2 | 3 |
| *(a) [$a/d = 0.1$, the reference values equal to: $h_{ref} = 0.369$ m, $t_{ref1} = 181$ s, $t_{ref2} = 112$ s, $t_{ref3} = 1133$ s.]* | | | | | | |
| Gerris | 0.560 | 0.766 | 0.868 | 2.78 | 3.34 | 3.17 |
| OpenFOAM | 0.549 | 0.688 | 0.841 | 1 | 1 | 1 |
| Thétis | 0.898 | 1.323 | 1.196 | 211 | 159 | 41.1 |
| Truchas | 0.590 | 0.723 | 0.900 | 33.5 | 53.1 | 59.1 |
| *(b) [$a/d = 0.3$, the reference values equal to: $h_{ref} = 1.175$ m, $t_{ref1} = 26$ s, $t_{ref2} = 193$ s, $t_{ref3} = 1912$ s.]* | | | | | | |
| Gerris | 0.502 | 0.632 | 0.793 | 5.04 | 5.16 | 3.56 |
| OpenFOAM | 0.466 | 0.624 | 0.776 | 1 | 1 | 1 |
| Thétis | 0.726 | 0.745 | 0.827 | 128 | 83.7 | 31.7 |
| Truchas | 0.494 | 0.622 | 0.745 | 37 | 55.2 | 63.5 |

*4.2.2. Domain rotated by 10°*

The study in the present subsection is undertaken in order to give an estimate of how the immersed solid, used previously for the beach, decreases the accuracy of the overall results for the runup. In the rotated configuration the immersed solid is used only to describe the bottom of the channel, and therefore has a relatively small influence. The approach presented hereby is not suggested for real applications in complex geometries, but is rather used for its resemblance to a simple body fitted mesh. A similar setup has, however, been used for the simulation of landslide induced waves (Popinet, 2013). A short depiction of a true body fitted mesh case for the OpenFOAM solver is presented in subsection 4 as an illustration on the additional issues for body fitted meshes.

The runup heights computed in the rotated domain are collected in Table 9. Once again, the $a/d = 0.3$ wave is more difficult then the $a/d = 0.1$ one, but the difference is not as severe as in the case for the immersed solid beach. The evolution of the maximum height of the free surface for the $a/d = 0.3$ wave is presented in Fig. 11. It has to be mentioned, that the current results are not directly comparable to the results obtained for the immersed solid beach (Table 7), as the resolutions used for the two configurations are different. In the rotated domain the mesh is around two times coarser. Nevertheless, we see, that a substantial gain of accuracy is observed only for the higher resolutions, and rather for the $a/d = 0.3$ wave (compare Tables 7 and 9). This is especially true in the case of OpenFOAM and Truchas. Gerris also produced good results in the $a/d = 0.3$ case for the highest resolution, but it revealed weak performance for the lower resolutions and for the $a/d = 0.1$ wave. Moreover, increased splitting of the swash tongue is observed, manifesting itself in the plots of the evolution of the maximum height, which are now less smooth in the runup part (compare Figs. 8a and 11a). A very good accuracy is observed for Thétis for the highest resolution case of the $a/d = 0.3$ wave, but the over-shots present in the $a/d = 0.1$ case suggest, that the simulations are highly influenced by spurious effects. This is probably also the reason for the steep decrease of the run up height in Fig. 11c. The swash tongue separates into multiple smaller fragments and some of them behave in a spurious way.

Despite the observed mixed performance of the solvers, the convergence rate is much better compared to the immersed solid beach case (compare Figs. 9 and 12) and is close to first order for all the solvers apart from Thétis.

Taking all the above into consideration, we see that the representation of the beach is important for the runup height. The thin swash tongue is easier to resolve in the rotated configuration, comparing to the immersed solid beach, leading to better results for the best resolution case. Nevertheless, the spurious effects observed during the propagation, like possible difficulties with the static equilibrium in the gravity field, might have amplified in the rotated configuration. As a result, rotation of the domain seems to decrease the overall performance, and is therefore not suggested for practical applications. This can also be observed when performing a simulation with an interface at equilibrium as initial condition. This way we measure the spurious currents due to the interface not aligned with the grid axes in the whole domain, not only at the boundary as was done in Section 1. After 8 s of simulation time, we measured total kinetic energy and maximum height of the interface, cf. Table 10. As can be seen from the values the levels of spurious kinetic energy and interface height are substantially higher when compared to the non rotated case (Table 8), in particular for Thétis. Concerning the runup for the rotated and the non rotated case, it is,

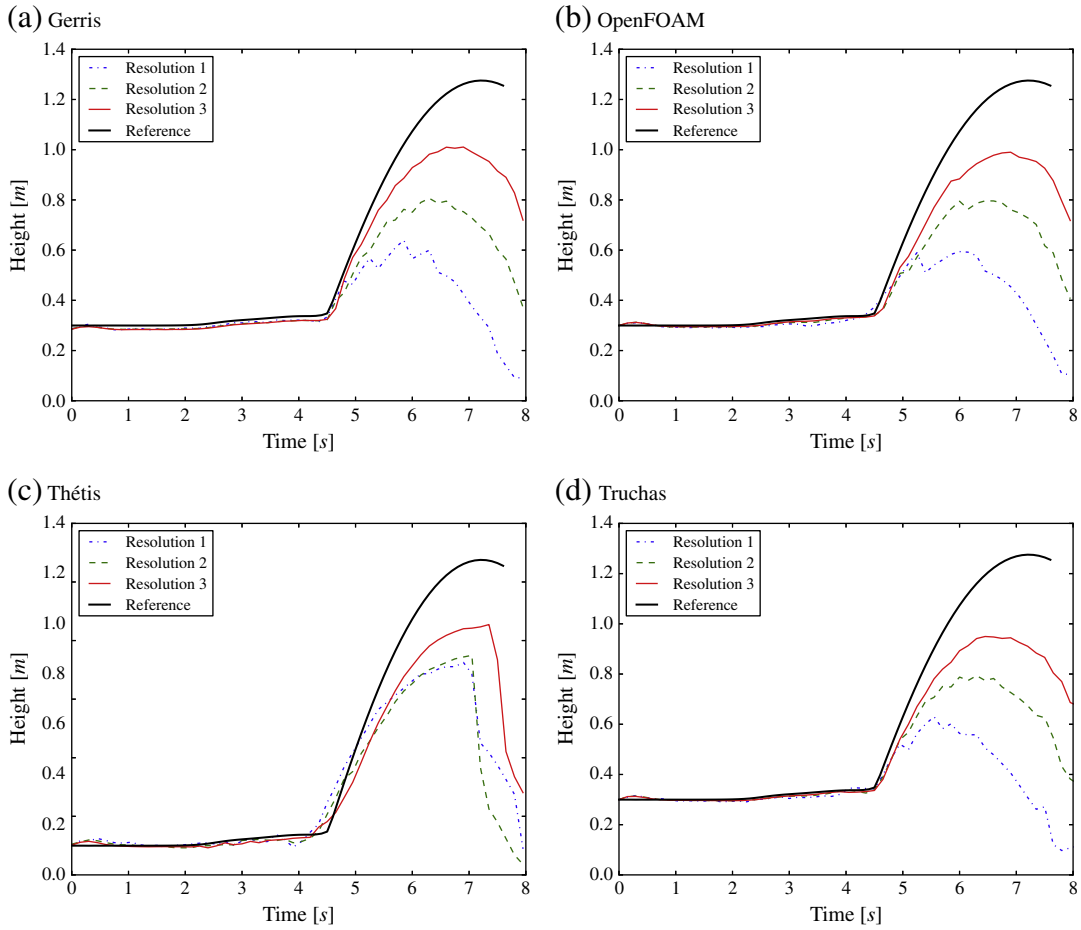


**Fig. 11.** Evolution of the maximum height of free surface during runup of an $a/d = 0.3$ solitary wave on a 10-degree slope, in the rotated configuration.

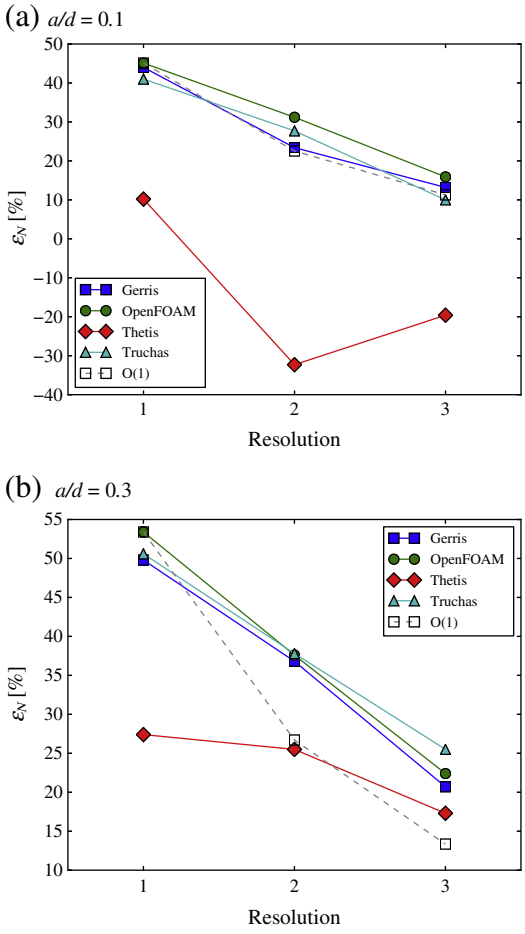


**Fig. 12.** The runup height errors, computed for the rotated configuration. The dashed line stands for the first order of convergence.

however, difficult to single out this spurious effect, since the interface will be inclined with respect to the grid axes.

#### 4.2.3. Immersed solid beach, 30-degree slope

The study of the 10-degree slope showed, that the tested solvers have problems with thin films of water moving over a solid surface. While the 10-degree beach acts as a demanding benchmark test, some practical applications may involve steeper slopes. To check the performance of the solvers in less rigid situations, the inclination angle of 30° is tested hereby, with the more difficult, $a/d = 0.3$ solitary wave. Lin et al. (1999) performed numerical simulations and experiments for a solitary wave runup on a beach inclined by 30°. However, their simulation was less demanding, since the amplitude of their solitary wave was $a/d = 0.17$.

**Table 10**
Total kinetic energy and water height $h_{fin}$ after 8 s of simulation for a static interface at equilibrium for the rotated domain configuration; $h_{ref} = 1m$. The kinetic energy of a solitary wave with amplitudes $a/d = 0.1, 0.3$ can be found in Table 1.

| | Total kinetic energy [J] | | | $h_{fin}/h_{ref}$ | | |
|---|---|---|---|---|---|---|
| Resolutions | 1 | 2 | 3 | 1 | 2 | 3 |
| Gerris | 7.9 | $3.2\cdot10^{1}$ | $8.2\cdot10^{1}$ | $6.6\cdot10^{-2}$ | $2.8\cdot10^{-2}$ | $2.7\cdot10^{-2}$ |
| OpenFOAM | $8.4\cdot10^{2}$ | $2.7\cdot10^{2}$ | $8.3\cdot10^{1}$ | $1.0\cdot10^{-1}$ | $5.2\cdot10^{-2}$ | $2.7\cdot10^{-2}$ |
| Thétis | $4.8\cdot10^{3}$ | $3.1\cdot10^{3}$ | $1.4\cdot10^{3}$ | 1.1 | $9.8\cdot10^{-1}$ | $3.3\cdot10^{-1}$ |
| Truchas | $3.9\cdot10^{2}$ | $7.9\cdot10^{1}$ | $2.2\cdot10^{1}$ | $9.8\cdot10^{-2}$ | $4.9\cdot10^{-2}$ | $2.5\cdot10^{-2}$ |

**Table 11**
The runup heights and times of computation, obtained for the runup of an $a/d = 0.3$ solitary wave on an immersed solid beach inclined at an angle of 30°. The reference values equal to: $h_{ref} = 1.275$ m, $t_{ref1} = 36$ s, $t_{ref2} = 307$ s, $t_{ref3} = 2185$ s.

| | $h_{max}/h_{ref}$ | | | $t_{comp}/t_{ref}$ | | |
|---|---|---|---|---|---|---|
| Resolution | 1 | 2 | 3 | 1 | 2 | 3 |
| Gerris | 0.933 | 0.952 | 0.972 | 1.25 | 2.22 | 2.84 |
| OpenFOAM | 0.868 | 0.919 | 0.947 | 1 | 1 | 1 |
| Thétis | 0.984 | 0.994 | 0.972 | 69 | 40 | 32.9 |
| Truchas | 0.839 | 0.871 | 0.921 | 85.4 | 77.8 | 85.1 |

The simulations reveal, that the computed runup heights, collected in Table 11 and Fig. 13, are much closer to the reference solution, than in the case of the 10-degree beach. The average error for the $a/d = 0.3$ wave decreased to a few percent for the best resolution, from more then 22 (see Table 7). The results slightly underestimate the reference height, and usually demonstrate a stable convergence. Gerris is the most accurate solver, following the reference solution during the runup, as well as during the draw down (Fig. 13a), and giving only a 3% underestimation of the runup height in the best resolution case (Table 11). OpenFOAM is the second most accurate, as well as the fastest solver. However, the temporal evolution of the runup height lags behind the reference solution (Fig. 13b). Truchas produced slightly poorer results than OpenFOAM, and took the most time for the computations (cf. Table 7). An unexpected behavior is observed in the case of Thétis, which gives an over-shot of the results in the initial phase of the runup (Fig. 13c). It may be due to the spurious effects appearing at the contact line, already observed for the 10-degree slope and having a stronger influence here. Moreover, even though all the recorded runup heights do not exceed the reference value, a decrease of accuracy is observed between the second and the third resolution (see Table 7). Such behavior would be impractical in use, as the convergence is ambiguous.

The overall good performance of the solvers proves, that the presented methods may produce reliable results in many practical applications involving steep slopes. Once again, the best accuracy is observed in Gerris, which may be due to the implemented immersed solid model.

The good performance for runup on steep slopes has also been observed by Lin et al. (1999). They mentioned that the results of their inviscid Navier–Stokes solver and their boundary integral method were comparing well.

#### 4.2.4. Body fitted mesh, 10-degree slope

In practical applications solid boundaries might either be described by an immersed method as discussed in Section 1 or by a mesh fitted to the solid body. The subject of meshing is extremely vast (Thompson et al., 1985) and has many facets, the discussion of which would go beyond the scope of the present treatise. In the present section, only OpenFOAM is used as a solver, since we are interested in only highlighting a few of the issues associated with body fitted meshes. As such we limit to three examples of meshes. The first mesh has only the lower edge aligned with the beach but lets the upper edge horizontal, cf. Fig. 14. The cells at the tip of the beach become thus very small and elongated. The second mesh has the upper edge parallel to the beach, cf. Fig. 15. The cells above the beach in this mesh are thus equal in size and shape. For both meshes the cells above the beach are not square anymore and display nonorthogonal angles. In the present discussion, no optimization of the mesh has been performed with respect to orthogonality or to cell size. The third mesh, cf. Fig. 16, was produced by using the snappyHexMesh pre-processing tool, as in Section 1, but this time triangular cells were added in order to fit the mesh to the surface.

We recorded the runup heights as a function of time in Fig. 17 for the three different meshes. As for the rotated domain case, we observe

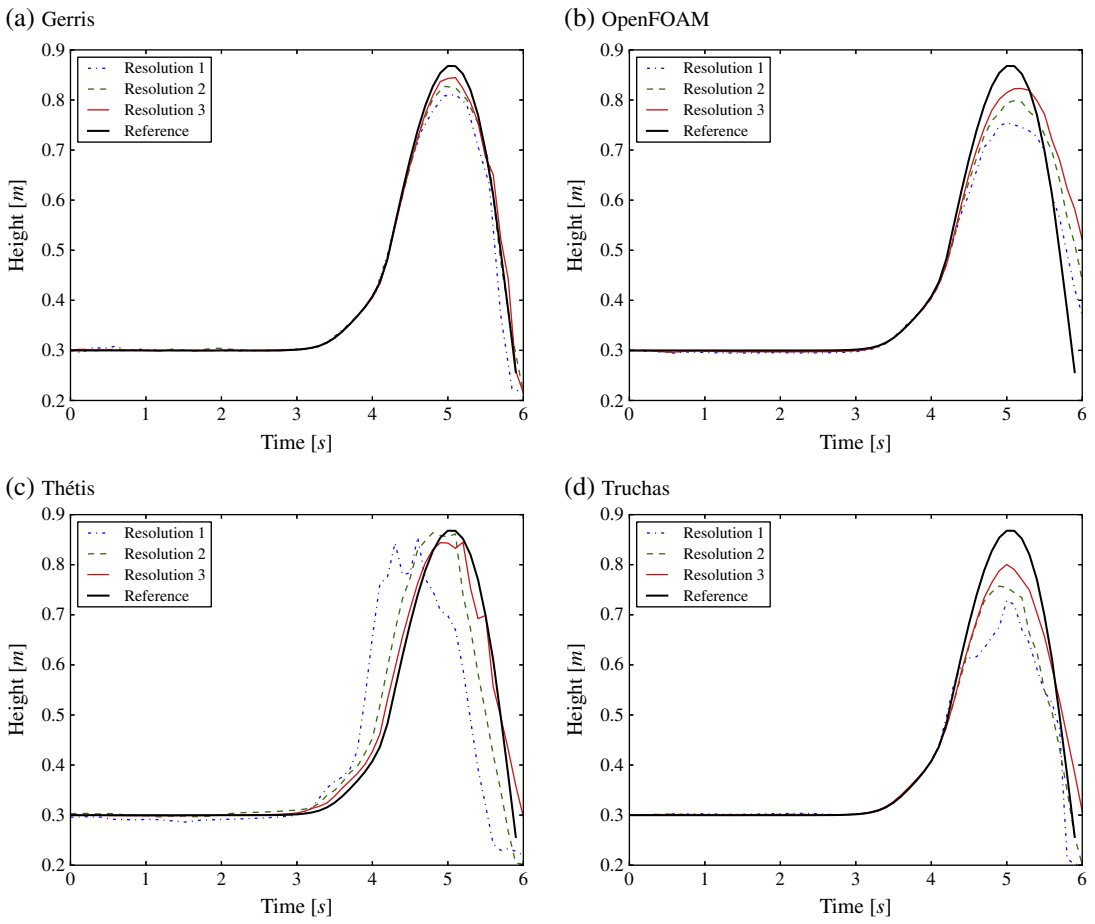


**Fig. 13.** Evolution of the maximum height of free surface, during the runup of an $a/d = 0.3$ solitary wave on an immersed solid beach inclined at an angle of 30°.

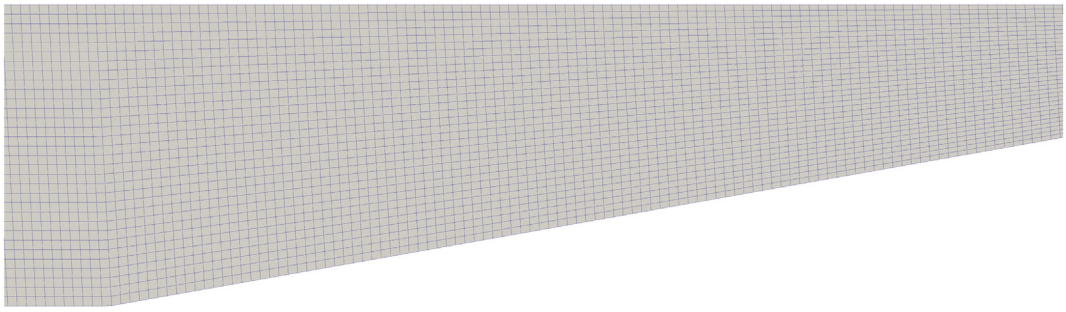

**Fig. 14.** The first mesh configuration. Zoom onto the start of the beach. The lower edge follows the beach, whereas the upper edge stays horizontal.

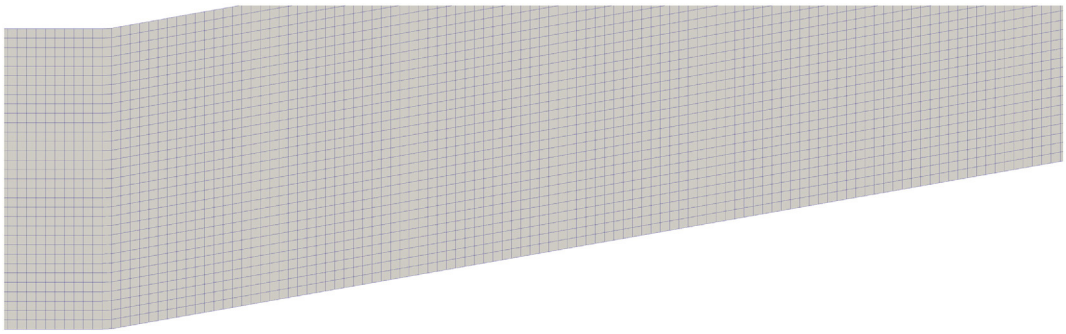

**Fig. 15.** The second mesh configuration. Zoom onto the start of the beach. The lower edge follows the beach and the upper edge is parallel to the lower one.

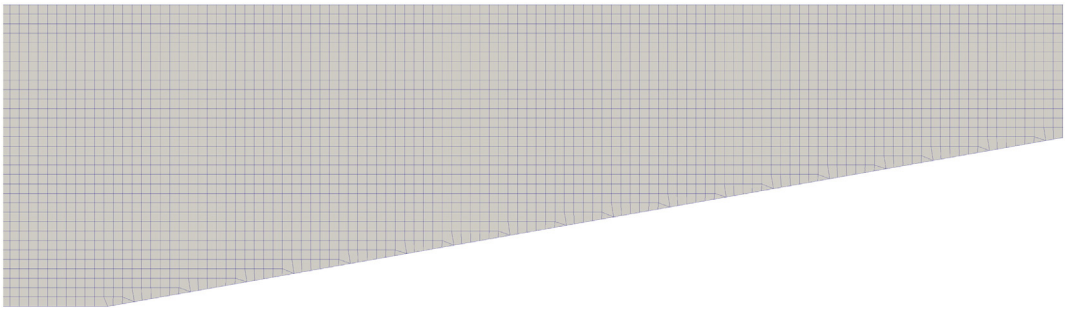

**Fig. 16.** The third mesh configuration. Zoom onto the start of the beach. The cells are mostly square for this configuration with the exception of cells close to the beach, where they can be triangular.

spurious currents due to the interface not being aligned anymore with the grid axes (figure not shown). Another probable source of spurious currents adding to these are those due to the nonorthogonality of the cell edges. The effect of the spurious currents can be seen in Fig. 17b for the runup produced by using the second mesh. We observe that the interface already starts to move up the beach before the wave arrives. These spurious currents can grow in time to significant values and might in a real case corrupt parts of the simulation. An in-depth study of this spurious effect is, however, beyond the scope of the present discussion. These currents are present for all the three meshes, but the total time of simulation is sufficiently short, so that they do not develop to such levels that the runup height might be significantly altered. These currents are present in the vicinity of the interface on the entire mesh for the first two meshes, whereas they are mostly located where the interface touches the beach for the third mesh. This is probably due to the fact that the cells for the third mesh are square in the core of the domain and only at the beach assume different shapes. For the first mesh, the spurious currents are less than for the second one. An explanation might be that the cells in the first mesh are closer to a square than for the second mesh.

Concerning the maximum runup height, the first mesh produces much better results than the other two meshes, cf. Fig. 17b. The reason is that the spatial resolution gets a lot finer in the critical region where the swash tongue becomes thinner. As such the partitioning of the cells is much more favorable to the problem than for the other two meshes, indicating that there are many more parameters to be discussed concerning meshing than is done in the present discussion. The second mesh produces results, which are comparable to the ones by OpenFOAM for the rotated domain case, cf. Fig. 11b. There are small differences but it is difficult to assess whether this is due to the mesh or the spurious currents already lifting parts of the water surface up the beach. The third mesh produces slightly better results than the second mesh or the rotated domain. This is probably due to a reduction of spurious currents when the interface is aligned with the grid lines.

To conclude this section, similar to what was found for the rotated domain case the use of a mesh aligned with the boundaries improves considerably the accuracy of the simulation. However, depending on the complexity a body fitted mesh might be prohibitive to use. In such cases an immersed description of the solid body is necessary. A particularly interesting variant of an immersed description is the snappyHexMesh pre-processing tool, where the advantages of both approaches, the immersed and the body fitted description, are combined. As such the partitioning of cells is an important criterion for meshes. A full discussion is, however, beyond the scope of the present discussion.

#### *4.2.5. Viscous effects, 10-degree slope*

The previous cases have been computed by setting viscosity to zero and using slip boundary conditions on the solid boundaries, which allowed us to relate the results to a very accurate reference solution computed by means of the velocity potential. In the real case viscosity will be present and the boundary condition at the solid boundary will be a no-slip boundary condition. In this section, we use OpenFOAM on the first mesh of the previous section in order to highlight some issues concerning the viscous case. This specific mesh has been chosen as spurious effects (artificial wall roughness or spurious currents) are less significant than for the other cases. The effects due to viscosity should therefore become more visible. As before we used only OpenFOAM since we are only interested in highlighting some issues related to introducing viscosity. We produced two different sets of simulations using no-slip boundary conditions on the solid boundaries. For the first one turbulence modeling was switched off, such that the full Navier–Stokes equations were solved, whereas for the second one turbulence modeling was switched on, such that the unsteady Reynolds averaged Navier–Stokes equations were solved. The runup heights recorded for these cases can be seen in Fig. 18. Surprisingly the viscosity and the no-slip boundary condition hardly alter the runup height for the full Navier–Stokes solution, cf. Fig. 18a, when compared to the inviscid solution, cf. Fig. 17a. This is somewhat surprising, since one would expect some influence by the no-slip boundary condition on the result. As it seems, OpenFOAM does not suffer from spurious effects due to an underresolved boundary layer. The boundary layer is rather ignored on this scale. Other solvers might show a different behavior, possibly break down due to underresolution or artificially thick boundary layers. The situation changes, however, once turbulence modeling is switched on. The solution is smoother, cf. Fig. 18b, and this could also be seen when looking at a snapshot of the solution (figure not shown). The fields are much smoother and spurious currents have virtually disappeared. The reason might be in an enhanced dissipation by the eddy viscosity of the turbulence model. This enhanced dissipation is probably also at the root of the reduced maximum runup heights for this model. For this flow depth, the flow in the runup is mostly laminar. Transition can be expected in the boundary layer (Pedersen et al., 2013) but turbulence is unlikely to grow significantly into the bulk flow. The turbulence modeling introduces thus additional dissipation which on one hand suppresses spurious currents but on the other hand changes the physics of the problem. This can also be seen in a retarded run up of the wave, cf. Fig. 18b. A similar effect has been observed by Higuera et al. (2013b) for their simulations.

## 5. Concluding remarks

We designed two benchmark tests for free-surface Navier–Stokes codes, with focus on applications in oceanography and coastal engineering. The first problem, propagation of a solitary wave, showed that numerical dissipation, due to the discretization of the convective terms, is not the only mechanism influencing the energy balance in the simulations. In several cases, growth of the total energy of the wave was observed. The investigation of the velocity field revealed

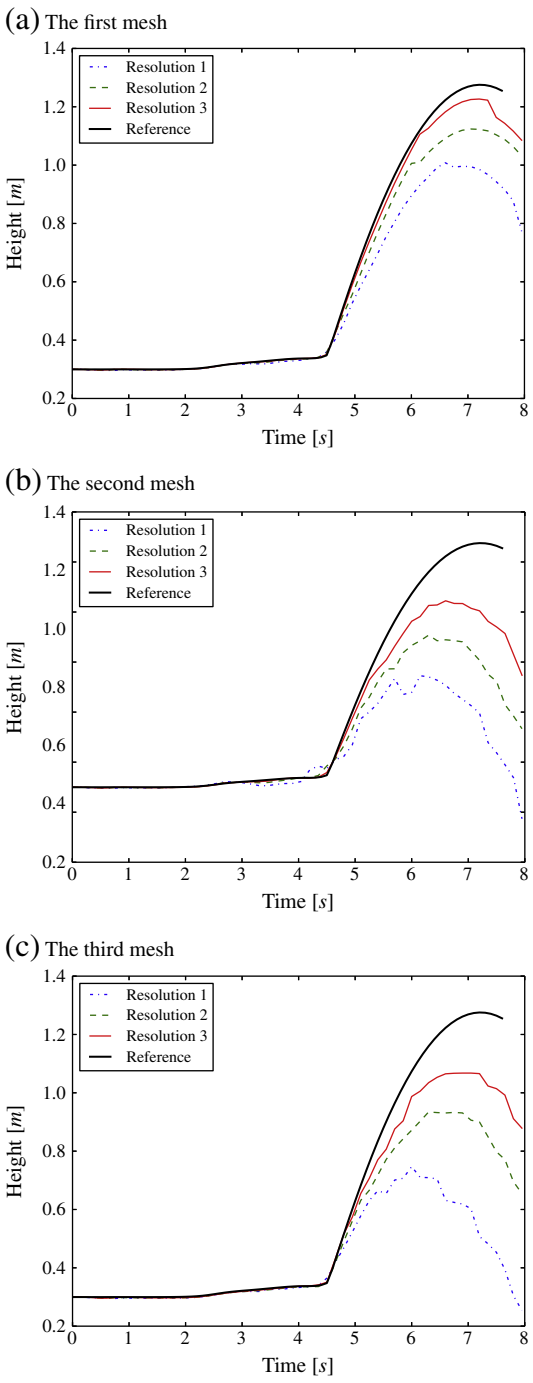


**Fig. 17.** Evolution of the maximum elevation of free surface during runup of an $a/d = 0.3$ solitary wave on a solid beach inclined by an angle of 10° using OpenFOAM on three different meshes.

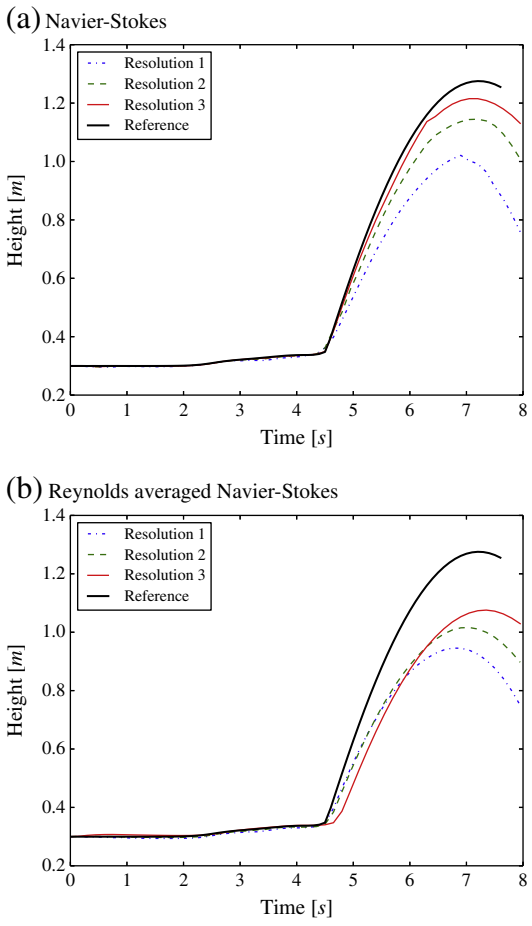


**Fig. 18.** Evolution of the maximum elevation of free surface during runup of an $a/d = 0.3$ solitary wave on a solid beach inclined by an angle of 10° using OpenFOAM and no-slip boundary conditions at the solid boundaries in connection with a Navier–Stokes solver and a Reynolds averaged Navier–Stokes solver.

spurious currents appearing in the vicinity of the interface, resulting in a local, but substantial, increase of the velocity, which might have influenced the energy balance of the system. The observed artifacts, most vivid for the Thétis solver, may be connected with the jump in fluid parameters, as no intentional smoothing of the density field was performed. Test of two gravity models in the Gerris solver suggested, that change of the way the body forces are applied can be a possible remedy. Luckily, for the propagation case, the spurious effects diminished with grid refinement, therefore all the solvers were recommended for further use and validation, even though special care concerning these issues is advised.

In the second test, Navier–Stokes solvers were used to simulate the runup of a solitary wave on a plane beach. The simulations were performed for non-breaking wave parameters and the results were compared to a reference solution based on potential theory. During runup a thin and elongated swash tongue is produced, amplified by small inclination of the beach and high wave amplitude. The study showed, that the thin swash tongue on the beach brings increased difficulties to the proper advection of the water fraction. As a result, the swash tongue often splits, which can result in small droplets, subject to spurious currents. The local effects, that can be responsible for such problems, include the changing stencil of the VOF reconstruction scheme, the interaction with the immersed solid and the artificial roughness introduced by the staircase representation of the beach. In this light, solution of the transport equation for the volume fractions was found much more robust than the geometric advection, even though it is responsible for increased smoothing of the volume fraction field. Moreover, the runup heights obtained with OpenFOAM, which solves the advection equation for the volume fraction, were similar to the ones computed with Thétis and Truchas, which use the geometric method. For this reason, the high accuracy observed in the Gerris solver, seems to be due to the implemented VOF treatment of solids, even though a similar method used in Truchas did not result in increased runup heights. For practical simulations another strategy for representing the solid boundary is the use of grids aligned with the boundary. This has been tested by means of a rotated domain and by means of body fitted meshes for the OpenFOAM solver. This description is not a lot more accurate than the immersed one but depends on a large extent on the properties of

the mesh, such as the clustering of cells. A problem identified for all cases, also mentioned by Wemmenhove et al. (2012), was the development of spurious currents for interfaces not aligned with the grid axes. We feel that this is a serious issue and that it requires further study. As such the use of a turbulence model might pose a remedy by presenting a higher level of dissipation due to the eddy viscosity. However, the influence of this dissipation on the physical solution must be carefully checked. Further research must also point out, how well small scale models, like turbulence modeling, provide solutions for viscous and other effects in the oceanographic and coastal engineering context.

The encountered difficulties imply, that when using Navier–Stokes solvers for flows found in oceanography and coastal engineering, such as the runup of a tsunami, not only the discretization has to be closely monitored but also the numerical artifacts. The artifacts can have surprising effects, like growth of the total energy or spurious currents. The study showed the necessity of convergence tests, as the balance between the influence of the artifacts and other effects, like numerical dissipation, can change with refinement. As needs to be said, for verification and validation purposes it is important to compare the results by primitive models to the results by more accurate methods. For our case this required to refrain from viscous effects, since the boundary integral method is based on potential theory. This, however, does not lower the validity and significance of the present benchmark tests, since thanks to the more accurate reference solution, weaknesses and problems of the primitive models could be found and analyzed in detail. Once these issues are addressed, more physics can be added to the model, e.g. boundary layers, which then needs to be validated separately by another reference method. A comparison to experimental data may not be reliable, since not all physical phenomena present in the laboratory are modeled or modeled correctly in the solver and since laboratory measurements often provide only a few measured quantities and not for example the entire flow field. For the present investigation, physical phenomena, such as boundary layers or surface tension, important on a laboratory scale, would have reduced the maximum runup heights in experiments (Pedersen et al., 2013).

## Acknowledgment

The authors are grateful to Jose M. Gonzalez-Ondina and Stéphane Glockner for providing help with the Truchas and Thétis code, respectively. The work has been funded by the Research Council of Norway project “Laboratory experiments and numerical modeling of tsunamis generated by rock slides into fjords” (NFR 205184/F20).

## References

Abadie, S., Morichon, D., Grilli, S., Glockner, S., 2010. Numerical simulation of waves generated by landslides using a multiple-fluid Navier–Stokes model. Coast. Eng. 57, 779–794.
Berberović, E., van Hinsberg, N.P., Jakirlić, S., Roisman, I.V., Tropea, C., 2009. Drop impact onto a liquid layer of finite thickness: dynamics of the cavity evolution. Phys. Rev. E. 79, 036306.
Biausser, B., Grilli, S.T., Fraunié, P., 2003. Numerical simulations of three-dimensional wave breaking by coupling of a VOF method and a boundary element method. Proceedings of the Thirteenth International Offshore and Polar Engineering Conference, Honolulu Hawaii.
Cooker, M.J., Peregrine, D.H., 1992. Wave impact pressure and its effect upon bodies lying on the sea bed. Coast. Eng. 18, 205–229.
Cooker, M.J., Peregrine, D.H., Vidal, C., Dold, J.W., 1990. The interacting between a solitary wave and a submerged semicircular cylinder. J. Fluid Mech. 215, 1–22.
Deshpande, S.S., Anumolu, L., Trujillo, M.F., 2012. Evaluating the performance of the two-phase solver interfoam. Comput. Sci. Disc. 5, 36.
Dold, J.W., 1992. An efficient surface-integral algorithm applied to unsteady gravity waves. J. Comput. Phys. 103, 90–115.
Fenton, J., 1972. A ninth-order solution for the solitary wave. J. Fluid Mech. 53, 257–271.
Fletcher, C.A.J., 1991. Computational Techniques for Fluid Dynamics 2. Springer-Verlag.
Formaggia, L., Miglio, E., Mola, A., Parolini, N., 2008. Fluid–structure interaction problems in free surface flows: application to boat dynamics. Int. J. Numer. Methods Fluids 56, 965–978.
Francois, M.M., Cummins, S., Dendy, E., Kothe, D., Sicilian, J., Wiliams, M., 2006. A balanced-force algorithm for continuous and sharp interfacial surface tension models within a volume tracking algorithm. J. Comput. Phys. 213, 143–171.
Glockner, S., 2011. Thétis Web Page. Thetis.enscbp.fr.
Grilli, S., Svendsen, I.A., 1989. Runup and reflection of a solitary wave on steep slopes in a numerical wave tank. The Fourth International Workshop on Water Waves and Floating Bodies. International Workshop on Water Waves and Floating Bodies, pp. 77–80.
Grilli, S., Svendsen, I.A., 1990. Computation of nonlinear wave kinematics during propagation and run-up on a slope. Kluwer Academic Publishers. NATO ASI Series E 178, 378–412.
Grilli, S., Subramanya, R., Svendsen, I., Veeramony, J., 1994. Shoaling of solitary waves on plane beaches. J. Waterw. Port Coast. Ocean Eng. 120, 609–628.
Grilli, S., Svendsen, I., Subramanya, R., 1997. Breaking criterion and characteristics for solitary waves on slopes. J. Waterw. Port Coast. Ocean Eng. 123, 102–112.
Higuera, P., Lara, J.L., Losada, I.J., 2013a. Realistic wave generation and active wave absorption for Navier–Stokes models. Application to OpenFOAM®. Coast. Eng. 71, 102–118.
Higuera, P., Lara, J.L., Losada, I.J., 2013b. Simulating coastal engineering processes with OpenFOAM®. Coast. Eng. 71, 119–134.
Issa, R.I., 1985. Solution of the implicitly discretised fluid flow equations by operator-splitting. J. Comput. Phys. 62, 40–65.
Jacobsen, N.G., Fuhrman, D.R., Fredsøe, J., 2012. A wave generation toolbox for the open-source CFD library: OpenFOAM®. Int. J. Numer. Methods Fluids 70, 1073–1088.
Kitware Inc., 2013. The Visualization Toolkit (VTK) web page. http://www.vtk.org.
Lin, P., Chang, K.A., Liu, P.L.F., 1999. Runup and rundown of solitary waves on sloping beaches. J. Waterw. Port Coast. Ocean Eng. 125, 247–255.
Liu, P.L.F., Wu, T.R., Raichlen, F., Synolakis, C., Borrero, 2005. Runup and rundown generated by three-dimensional sliding masses. J. Fluid Mech. 536, 107–144.
Longuet-Higgins, M.S., Cokelet, E.D., 1976. The deformation of steep surface waves on water—I. A numerical method of computation. Proc. R. Soc. Lond. A 350, 1–26.
Lubin, P., 2004. Simulation des grandes échelles du déferlement plongeant des vagues. (Ph.D. Thesis) UMR Trefle.
Lubin, P., Lemonnier, H., 2004. Test-case no 33: propagation of solitary waves in constant depths over horizontal beds. Multiph. Sci. Technol. 16, 239–250.
Lubin, P., Vincent, S., Abadie, S., Caltagirone, J.P., 2006. Three-dimensional large eddy simulation of air entrainment under plunging breaking waves. Coast. Eng. 53, 631–655.
Montagna, F., Bellotti, G., Di Risio, M., 2011. 3D numerical modeling of landslide-generated tsunamis around a conical island. Nat. Hazards 58, 591–608.
Mory, M., Abadie, S., Mauriet, S., Lubin, P., 2011. Run-up flow of a collapsing bore over a beach. Eur. J. Mech. B/Fluids 30, 565–576. http://dx.doi.org/10.1016/j.euromechflu.2010.11.005 (URL: http://www.sciencedirect.com/science/article/pii/S099775461000124X, special Issue: Nearshore Hydrodynamics).
Nielsen, K.B., 2003. Numerical prediction of green water loads on ships. (Ph.D. Thesis) Technical University of Denmark.
OpenFOAM Foundation, 2013. The OpenFOAM web page. http://www.openfoam.org.
Pedersen, G.K., Lindstrøm, E., Bertelsen, A.F., Jensen, A., Laskovski, D., Sælevik, G., 2013. Runup and boundary layers on sloping beaches. Phys. Fluids 25, 23. http://dx.doi.org/10.1063/1.4773327.
Popinet, S., 2003. Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries. J. Comput. Phys. 190, 572–600.
Popinet, S., 2009. An accurate adaptive solver for surface-tension-driven interfacial flows. J. Comput. Phys. 228, 5838–5866.
Popinet, S., 2013. Gerris Flow Solver web page. http://gfs.sourceforge.net.
Raeini, A.Q., Blunt, M.J., Bijeljic, B., 2012. Modelling two-phase flow in porous media at the pore scale using the volume of fluid method. J. Comput. Phys. 231, 5653–5668.
Tanaka, M., 1986. The stability of solitary waves. Phys. Fluids 29, 650–655.
The Telluride Team, 2008. Truchas Physics and Algorithms. http://telluride.lanl.gov.
Thompson, J.F., Warsi, Z.U.A., Mastin, C.W., 1985. Numerical Grid Generation, North-Holland.
Tryggvason, G., Scardovelli, R., Zaleski, S., 2011. Direct Numerical Simulations of Gas–Liquid Multiphase Flows. Cambridge University Press.
Vinje, T., Brevig, P., 1981. Numerical simulation of breaking waves. Adv. Water Resour. 4, 77.
Viroulet, S., Kimmoun, O., 2012. Impulse wave generated by the collapse of a dry granular media. Proceedings of the 2nd International Conference on Violent Flows.
Wemmenhove, R., Luppes, R., Veldman, A.E., Bunnik, T., 2012. Simulation of hydrodynamic wave loading by a compressible two-phase flow method. Proceedings of the 2nd International Conference on Violent Flows.
Wu, T.R., 2004. A numerical study of three-dimensional breaking waves and turbulence effects. (Ph.D. Thesis) Cornell University.