3. interFoam settings

In this section the default numerical settings for our simulations, as well as a general description of OpenFOAM®'s discretization practices, are presented. Our base numerical settings will be those found in the popular damBreak tutorial shipped with foam-extend-3.2. With this starting point we will change various settings to investigate their effect on the quality of the numerical solution. More specifically, we copy the controlDict, fvSchemes and fvSolution files directly from the damBreak tutorial. In the constant directory the mesh and the physical parameters of the case are specified: ρ_{water}=1000 kg/m^3, $\rho_{\text{air}} = 1.2$ kg/m^3, $\nu_{\text{water}} = 1 \cdot 10^{-6}$ m^2/s, $\nu_{\text{air}} = 1.45 \cdot 10^{-5}$ m^2/s, and $\sigma_T = 0.0$ N/m (i.e. no surface tension). We note that the analytic stream function solution does not take into account the presence of air, nor the effect of viscosity or surface tension. With the chosen wave parameters and boundary conditions (e.g. no slip at the bed) the physics are dominated by inertia and gravity. With a density rate of $\rho_{\text{water}}/\rho_{\text{air}} \sim 833$, the air will behave like a "slave fluid" moving passively out of the way for the water close to the surface. To confirm the insignificance of the physical viscosity in our setup, we have compared simulations with these set to their physical values and to $\nu = 1 \cdot 10^-16$ m^2/s, and confirmed that this had no effect on our results. We have also performed simulations with $\rho_{\text{air}} = 0.1$ kg/m^3 and $\rho_{\text{air}} = 10$ kg/m^3. This had almost no effect in the short term, but had some effect for long propagation distances. Increasing the density made the air behave less like a "slave fluid" and slowed the propagation of the wave. Decreasing the density created larger air velocities, but did not alter the wave kinematics significantly. We have confirmed that switching the surface tension between zero and its physical value ($\sigma_T = 0.07$ N/m) had next to no effect on our simulation results, as expected in the gravity wave regime. Finally, the simulations are performed without turbulence, as the results are intended to be compared with the idealized stream function (potential flow) solution.

The OpenFOAM® case setup is contained in a file called controlDict which, among others things, controls the time stepping method. The schemes used to discretize the different terms in the governing equations are specified in the fvSchemes file, and the file fvSolution contains various settings for the linear solvers and for the solution algorithm. In Table 1 the essential parameters for the base set up from these three files are indicated. The most important details of the scheme and solver choices presented in Table 1 will be described in the following. For descriptions of the remaining settings, the reader is referred to the OpenFOAM®user guide and programmers guides in Greenshields (2015, 2016).

3.1. *controlDict*

In this subsection the most important controlDict settings are presented. The time step can be specified either as fixed, such that the user defines the size of the time step, or as adjustable. In the latter case the time step is adjusted such that a maximum Courant number $Co = u_i \Delta t / \Delta x_i$ or a maximum AlphaCo (The Courant number in interface cells) is maintained at all times. Since these two for the remainder of this study are kept equal it is Co that controls the time step. In the damBreak tutorial an adjustable time step is used with $Co = 0.5$, hence this value will be utilized initially.

Table 1. Base setup from the damBreak tutorial

`controlDict`	Scheme/Value
`adjustTimeStep`	`true`
`maxCo`	0.5
`maxAlphaCo`	0.5
`fvSchemes`	
`ddt`	`Euler`
`grad`	`Gauss Linear`
`div(rho*phi,U)`	`Gauss LimitedLinearV 1`
`div(phi,alpha1)`	`Gauss VanLeer01`
`div(phirb,alpha1)`	`Gauss interfaceCompression`
`laplacian`	`Gauss linear corrected`
`interpolation`	`linear`
`snGrad`	`corrected`
`fvSolution`	
`pcorr(solver,prec,tol,relTol)`	`PCG, DIC`, 1e-10, 0
`pd(solver,prec,tol,relTol)`	`PCG, DIC`, 1e-07, 0.05
`pdFinal(solver,prec,tol,relTol)`	`PCG, DIC`, 1e-07, 0
`U(solver,prec,tol,relTol)`	`PBiCG, DILU`, 1e-06, 0
`cAlpha`	1
`momentumPredictor`	`yes`
`nOuterCorrectors`	1
`nCorrectors`	4
`nNonOrthogonalCorrectors`	0
`nAlphaCorr`	1
`nAlphaSubCycles`	2

3.2. *fvSchemes*

In this subsection some of the discretisation schemes are presented to aid in the understanding of the forthcoming analysis. The `ddt` scheme specifies how the time derivative $\partial/\partial t$ is handled in the momentum equations. Available in `OpenFOAM` are: `steadyState`, `Euler`, `Backwards` and `CrankNicolson`. In this study, `steadyState` is naturally disregarded as the simulations are unsteady. The `Euler` scheme corresponds to the first-order backward implicit Euler scheme, whereas `Backward` corresponds to a second-order, `OpenFOAM` implemented time discretization scheme, which utilizes the current and two previous time steps. The `CrankNicolson` (CN) scheme includes a blending factor ψ, where $\psi = 1$ corresponds to pure (second-order accurate) CN and $\psi = 0$ corresponds to pure `Euler`. This blending factor is introduced to give increased stability and robustness to the CN scheme.

In the finite volume approach used in `OpenFOAM`, the convective terms in the mass (7) and momentum (2) equations are integrated over a control volume, and afterwards the Gauss theorem is applied to convert the integral into a surface integral:

$$\int_V \nabla \cdot (\phi u)\, dV = \oint_S \phi\, (n \cdot u)\, dS \approx \sum_f \phi_f F_f, \tag{9}$$

where $\phi(x,t)$ is the field variable, ϕ_f is an approximation of the face averaged field value. ϕ_f can be determined by interpolation, e.g. using central or upwind differencing. Central differencing schemes are second order accurate, but can cause oscillations in the solution. Upwind differencing schemes are first order accurate, cause no oscillations, but can be very diffusive.

`OpenFOAM` includes a variety of total variation diminishing (TVD) and normalized variable diagram (NVD) schemes aimed at achieving good accuracy while maintaining boundedness. TVD schemes calculate the face value ϕ_f by utilizing combined upwind and central differencing schemes according to

$$\phi_f = (1 - \Gamma)\phi_{f,UD} + \Gamma\phi_{f,CD0} \tag{10}$$

where $\phi_{f,UD}$ is the upwind estimate of ϕ_f, $\phi_{f,CD}$ is the central differencing estimate of ϕ_f. Γ is a blending factor, which is a function of the variable r representing the ratio of successive gradients,

$$r = 2\frac{d \cdot (\nabla\phi)_P}{\phi_N - \phi_P}. \tag{11}$$

Here d is the vector connecting the cell centre P and the neighbour cell centre N. In NVD-type schemes the limiter is formulated in a slightly different way. In the `damBreak` tutorial base setup the TVD scheme is utilized by specifying the keyword `limitedLinearV 1` for the momentum flux, `div(rho*phi,U)`, and `vanLeer01` for the mass flux, `div(phi,alpha1)`, where the keyword `phi` means face flux. With the `limitedLinear` scheme $\Gamma = max\left[min\left(2r/k, 1\right), 0\right]$, where k is an input given by the user, in this case $k = 1$. When using the scheme for vector fields a "V" can be added to the TVD schemes, which changes the calculation of r to take into account the direction of the steepest gradients. The `vanLeer` scheme calculates the blending factor as $\Gamma = (r + |r|)/(1 + |r|)$. The `01` added after the TVD scheme name means that Γ is set to zero if it goes out of the bounds 0 and 1, thus going to a pure upwind scheme to stabilize the solution. The other available TVD/NVD schemes differ in their definition of Γ and resulting degree of diffusivity. Since r depends on the numerically calculated gradient of ϕ, the choice of gradient scheme can also play an important role. In general the gradients are calculated utilizing a Gauss linear scheme, but this might lead to unbounded face values, and therefore gradient limiting can be applied. As an example the gradient scheme can be specified as `Gauss faceMDLimited`. The keyword `face` or `cell` specifies whether the gradient should be limited base on cell values or face values and the keyword `MD` specifies that it should be the gradient normal to the faces. In addition to the linear choice of gradient schemes there also exists a least square scheme as well as a fourth order scheme.

The `laplacian` scheme specifies how the Laplacian in the pressure correction equation within the `PISO` algorithm, as well the third term on the right hand side of equation (2), should be discretized. It requires both an interpolation scheme for the dynamic viscosity, μ, and a surface normal gradient scheme `snGrad` for ∇u. Often a `linear` scheme is used for the interpolation of μ and the proper choice of surface normal gradient scheme depends on the orthogonality of the mesh. Besides being used in the Laplacian, the `snGrad` is also used to evaluate the second and fourth term on the right hand side of equation (2). Often a linear scheme will be used, with or without orthogonality correction. Another option is to use a fourth order surface normal gradient approximation. Finally, the interpolation scheme determines how values are interpolated from cell centres to face centres.

3.3. *fvSolution*

In the fvSolutions file the iterative solvers, solution tolerances and algorithm settings are specified. The available iterative solvers are preconditioned (bi-)conjugate gradient solvers denoted PCG/PBiCG, a smoothSolver, generalised geometric-algebraic multi-grid, denoted GAMG, and a diagonal solver. Each solver can be applied with different preconditioners and the smooth solver also has several smoothing options. The GAMG solver works by generating a quick solution on a coarse mesh consisting of agglomerated cells, and then mapping this solution as the initial guess on finer meshes to finally obtain an accurate solution on the simulation mesh. The different preconditioners and smoothers will not be discussed here, but Greenshields (2015, 2016) can be consulted for additional details.

In addition to the solver choices the PISO, PIMPLE and SIMPLE controls are also given in the fvSolution file. The cAlpha keyword controls the magnitude of the numerical interface compression term in equation (7). cAlpha is usually set to 1 corresponding to a "compression velocity" of the same size as the flow velocity at the interface. The momentumPredictor is a switch specifying enabling activation/deactivation of the predictor step in the PISO algorithm. The parameter, nOuterCorrectors is the number of outer correctors used by the PIMPLE algorithm and specifies how many times the entire system of equations should be solved within one time step. To run the solver in "PISO mode" we set nOuterCorrectors to 1. The parameter nCorrectors is the number of pressure corrector iterations in the PISO loop. The parameter nAlphaSubCycles enables splitting of the time step into nAlphaSubCycles in the solution of the α equation (7). Finally, the parameter nAlphaCorr, specifies how many times the alpha field should be solved within a time step, meaning that first the alpha field is solved for, and this new solution is then used in solving for the alpha field again.

4. Results and discussion

In this section the simulated results involving the propagation of the regular stream function wave will be presented and discussed for various settings.

4.1. *Perfomance of interFoam utilizing the damBreak settings*

First, the "default" performance of interFoam in the progression of the stream function wave is presented, utilizing the settings from the damBreak tutorial. The setup utilized here will be considered as the base setup, and the remainder of the simulations in this study will utilize this base setup with minor adjustments.

Starting from the analytical stream function solution imposed as an initial condition (utilizing the waves2Foam toolbox of Jacobsen et al. (2012)), the simulation is performed for 200 s (corresponding to 100 periods). This is sufficiently long to highlight certain strengths and problems of interFoam. Results are sampled at the cyclic boundary 20 times per period. In Figure 1 the surface elevation time series is shown. Quite noticeably, even though the depth is constant, the wave height immediately starts to increase, and this continues until the wave at some point (approximately at $t = 20T$) breaks. This rather surprising result demonstrates the potentially poor performance of interFoam, as the wave does not come close to maintaining a constant form. A similar result has been shown in Afshar (2010). A feature that seems to contribute, though is not solely responsible for, the un-physical steepening of the