Creating a Numerical wave tank on OpenFOAM BlueCFD

By: Omar Sallam

August 17, 2022
Source : http://ocean-cfd.engr.tamu.edu/doc/NWT

Attachment : NWT.zip

Report No.: **OCEN CFD Technical Report #220003**

1 Introduction and problem statement

In this article a brief introduction is given on how to implement a Numerical Wave Tank (NWT) in OpenFOAM to simulate regular water waves. NWTs are used to simulate regular and irregular waves to study water free surface dynamics, sediment transport, fluid-structure interaction due to wave loads and more.

In this case, a regular linear airy wave is simulated in the created NWT. The NWT's dimension are (L = 10, W = 1.2 and H=0.6) m. The simulated wave height is 2 cm and the wave length is 3 m.

2 Software installation

For ease of use, this numerical simulation is implemented via the **blueCFD** software (a Windows version of OpenFOAM). This software can be downloaded and installed as an executable file from `http://bluecfd.github.io/Core/Downloads/`. After installation is completed, the user can use the **blueCFD** core terminal to execute OpenFOAM commands.

3 interfoam solver

OpenFOAM is a Finite Volume based open source code that includes several solvers dedicated to solve various fluid flow problems. The choice between these solvers depends on the user application of interest.

For this NWT application we will use the interFOAM solver (Damian (2012)) that is able to solve for multiphase flows using the Volume Of Fluid technique for fluid-fluid interface capturing. The solver is incompressible and isothermal. interFoam solves for the coupled PDE's, continuity equation 1, Navier-Stoke's equations 2 and the phase fraction equation 4). Where $\mathbf{u}$ is the flow velocity.p and ρ are the thermodynamic pressure and mixture density (5). $\mathbf{T}$ is the viscous stress tensor that is a function of the effective dynamic viscosity and the general rate of strain tensor $\mathbf{D}$. $\mathbf{f}_b$ represents body forces, including the gravitational force and the surface tension. The solver uses the PIMPLE algorithm for velocity-pressure coupling (Holzmann (2016)).

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

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = -\nabla p + \nabla \cdot \mathbf{T} + \rho \mathbf{f}_b \tag{2}$$

$$\mathbf{T} = 2\mu \mathbf{D} \tag{3}$$

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot \alpha \mathbf{u} = 0 \tag{4}$$

$$\rho = \alpha \rho_w + (1-\alpha)\rho_a \tag{5}$$

4 OpenFOAM case folder structure

OpenFOAM simulation cases should have a certain folder/file structure.

Figure 1 shows the folder/file structure for the NWT case. folder (**0**) contains the files that represents the boundary and initial conditions for the velocity field, pressure field and the phase fraction field. Folder (**system**) contains the files that control the numerical schemes such as **fvSchemes** and **fvSolution**. It also includes the **controlDict** file where the user can indicate the solver of interest such as (interFoam) and the simulation time in addition to other control parameters. The **system** folder includes the **blockMeshDict** file that is used to generate the numerical grid for the simulation case.

The **constant** folder includes additional information for the solver, not related to boundary condition or numerical schemes. The files in the **constant** folder can include a **g** file that determines the direction of the gravitational force, (**transportProperties**) includes the material properties such as viscosity and density of the fluid and file **waveProperties** contains the simulated wave properties such as wave amplitude and wave length. Figure 2 shows the the blueCFD terminal on the root folder location and the sub folders/files. (The **blueCFD** terminal can be opened on the root folder location by right-clicking on the root folder and choose "Open with blueCFD core terminal").Note: The folder structure for the example presented in this article is attached as NWT.zip.

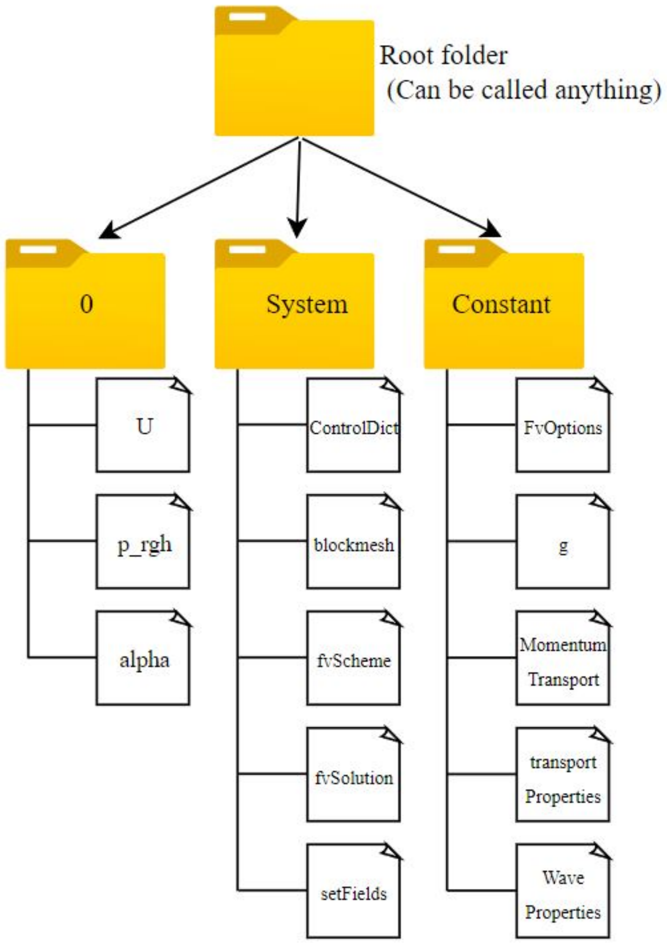
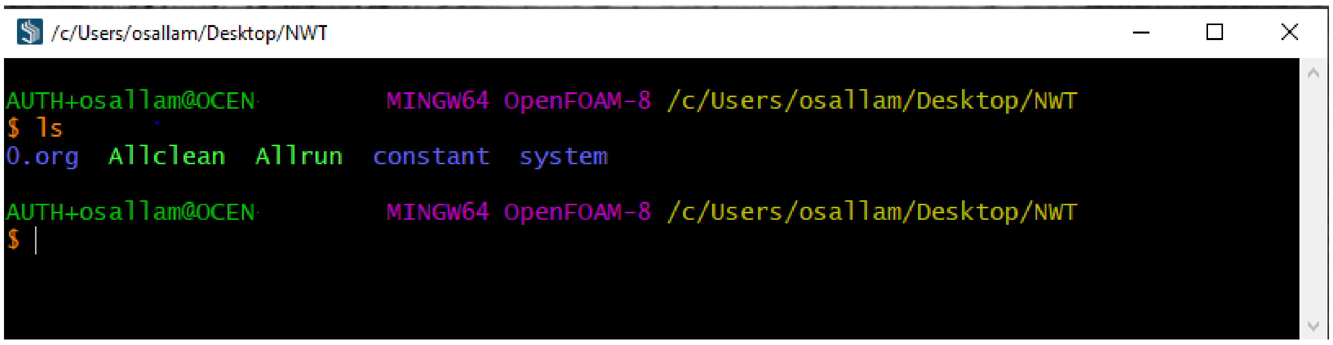


Figure 1: Folder structure for an interFoam solver case


/c/Users/osallam/Desktop/NWT

```
AUTH+osallam@OCEN          MINGW64 OpenFOAM-8 /c/Users/osallam/Desktop/NWT
$ ls
0.org  Allclean  Allrun  constant  system

AUTH+osallam@OCEN          MINGW64 OpenFOAM-8 /c/Users/osallam/Desktop/NWT
$ 
```

Figure 2: blueCFD terminal showing the main folders and files in the root folder

5 Mesh creation

To generate a simple computational domain for the NWT with cuboid shape, the **blockMesh** utility is used. With **blockMesh**, the user can assign Cartesian coordinates for all 8 vertices in the cuboid shape as well as defining certain boundary names and types for faces constructed from these vertices. The numbering system for the **blockMesh** utility is shown in figure 3.

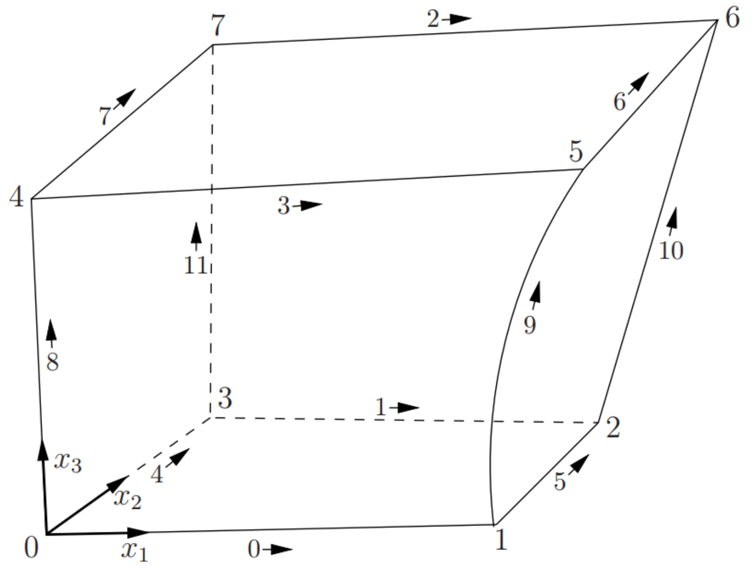


Figure 3: blockMesh cuboid vertex numbering, This image is adopted form (Greenshields (2018))

Code 1 shows the **blockMesh** utility dictionary file structure. First, the min/max dimension variables for the NWT are listed, then the vertices Cartesian coordinates (x_1, x_2, x_3) are defined based on the numbering system in figure 3. Then the cuboid shape is constructed by the **hex** object with the numbered vertices followed by the number of elements in (x_1, x_2, x_3) directions, here we used 45,15 and 45 elements in x_1,x_2 and x_3 directions respectively.

The **boundary** object defines the name, type and location of each boundary on the computational domain outer surface. In this case we define five boundaries for the NWT (Inlet, Outlet, sides, bottom and top). Each boundary is constructed by four vertices as shown in in figure 3.

Typing **blockMesh** in the **blueCFD** terminal makes OpenFOAM generate the computational domain (the higher number of elements the longer time it takes to construct the grid).

Now, the user is ready to type **paraFoam** in the terminal to show the grid in the ParaView post processing software as shown in figure 4

Code 1: blockMeshDict to generate a simple grid for a cuboid domain

```
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website:  https://openfoam.org
    \\  /    A nd           | Version:  8
     \\/     M anipulation  |
\*---------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      blockMeshDict;
}

// * * * * * * * * * * * * * * * * * * *
convertToMeters 1;
//  Tank dimensions
X_MIN   0  ;
X_MAX   10  ;
Y_MIN   -.6  ;
Y_MAX  .6  ;
Z_MIN   -0.3 ;
Z_MAX   .3  ;
// Vertices locations for the Tank
vertices
(
    ($X_MIN $Y_MIN $Z_MIN)
    ($X_MAX $Y_MIN $Z_MIN)
    ($X_MAX $Y_MAX $Z_MIN)
    ($X_MIN $Y_MAX $Z_MIN)
    ($X_MIN $Y_MIN $Z_MAX)
    ($X_MAX $Y_MIN $Z_MAX)
    ($X_MAX $Y_MAX $Z_MAX)
    ($X_MIN $Y_MAX $Z_MAX)
);
// Build the block Mesh based on the indices numbering
blocks
(hex (0 1 2 3 4 5 6 7) (45 15 45)  simpleGrading (1 1 1));
edges
(
);
defaultPatch
{
    name frontAndBack;
    type empty;}
// Give a name for each side
boundary
(
    inlet
    { type patch;
        faces
        ((0 4 7 3) );}
    outlet
    {
        type wall;
        faces
        ((2 6 5 1));}
    bottom
    {
        type wall;
        faces
        ((1 2 3 0)); }
    top
    {
        type patch;
        faces
        ( (7 4 5 6));}
      sides
    {
        type wall;
        faces
        ((4 5 1 0)
            (7 3 2 6));}    );
mergePatchPairs();
// **********************************
```

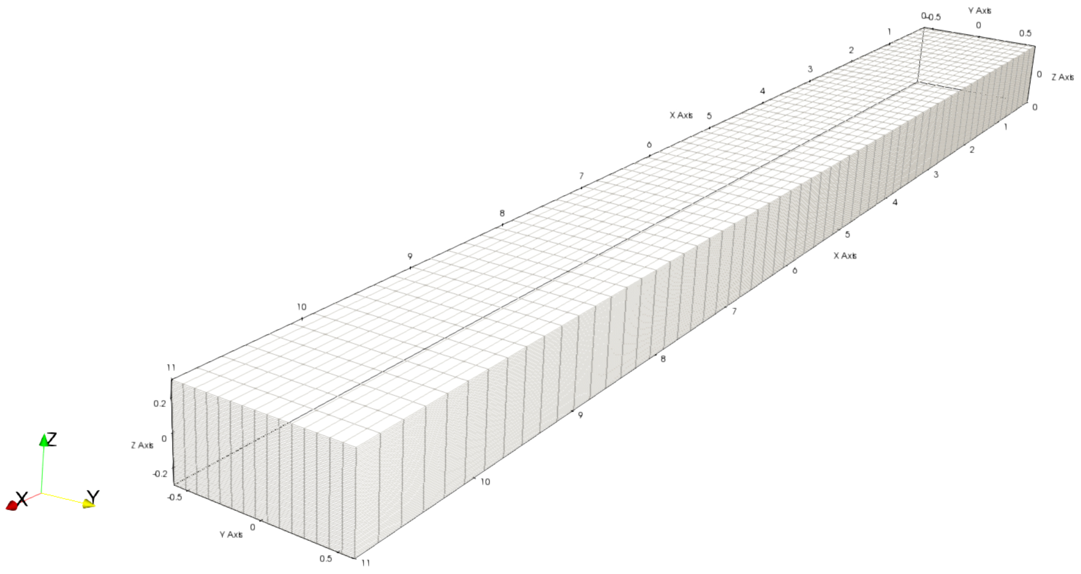


Figure 4: Grid generated by the blockMesh utility

6 Boundary and initial conditions

In this section we define the initial and boundary conditions IC/BC for the interFoam solver when solving the set of equations (1, 2 and 4). The IC/BC are defined for the velocity field **u** in the **U** file, pressure field p in the **p_rgh** file and the phase fraction field α in the **alpha.water** file.

In addition, we need to define two more files , **waveProperties** where the incident wave parameters are defined and **setFields** where the IC for the water/air phase fraction α are defined. This will essentially fill the tank with water and define the mean water level.

The velocity BCs are shown in code 2. For the Inlet boundary, the type is wave velocity which will be defined in code 5. **noSlip** BC is applied to the tank side, bottom and outlet walls.

A **pressureInletOutletVelocity** BC is applied to the top wall, this BC is usually used for boundaries that are open to the atmosphere and where inflow or outflow may take place. When inflow occurs the velocity is calculated by interpolating the neighbouring cells, for outflow, the pressure gradient is set to zero at that boundary.

The attached test case folder contains the BC called **0.org**, it is customary to keep the original folder untouched and to make a copy of it calling it **0** where the phase fraction BC is set.

More information about the different OpenFOAM boundary conditions can be found in (Greenshields (2018))

Code 2: U. Velocity boundary condition

```
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website:  https://openfoam.org
    \\  /    A nd           | Version:  8
     \\/     M anipulation  |
\*-----------------------------------------
FoamFile
{
    version     2.0;
    format      ascii;
    class       volVectorField;
    location    "0";
    object      U;
}
// * * * * * * * * * * * * *  * * * * * * * * * * *
dimensions      [0 1 -1 0 0 0 0];
internalField   uniform (0 0 0);
boundaryField
{
    #includeEtc "caseDicts/setConstraintTypes"
    inlet
    {type            waveVelocity;}
    outlet
    {type            noSlip;}
    top
    {type            pressureInletOutletVelocity;
        value           uniform (0 0 0); }
    bottom
    {type            noSlip;}
     sides
    {type            noSlip; }}
// *************************************************** //
```

Code 3 shows the pressure field BC, for all boundaries zero gradient is enforced except for the top boundary where a total pressure BC is applied that represents atmospheric datum pressure.

Code 3: p_rgh. Pressure boundary condition

```
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website:  https://openfoam.org
    \\  /    A nd           | Version:  8
     \\/     M anipulation  |
\*------------------------------------------
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    location    "0";
    object      p_rgh;}
// * * * * * * * * * * * * * * * * * *
dimensions      [1 -1 -2 0 0 0 0];
internalField   uniform 0;
boundaryField
{
    #includeEtc "caseDicts/setConstraintTypes"
    "(inlet|outlet|bottom|sides)"
    {
        type            fixedFluxPressure;
        value           uniform 0;}
    top
    {
        type            totalPressure;
        p0              uniform 0;}}
// ***************//
```

Code 4 shows the **alpha.water** BC file for the phase fraction α. For the outlet, sides and bottom boundaries a **zeroGradient** BC is used. For the inlet boundary, **waveAlpha** BC is used that is linked to the **waveProperties** file discussed later in code 5. **inletOutlet** BC is applied for the top boundary, this is a mixed boundary condition which applies **zeroGradient** for outflow condition and user specified value for inflow condition.

Code 4: alpha.water. Phase fraction boundary condition

```
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website:  https://openfoam.org
    \\  /    A nd           | Version:  8
     \\/     M anipulation  |
\*--------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    location    "0";
    object      alpha.water;
}
// * * * * *  * * * * * * * * * * * * * * * * //
dimensions      [0 0 0 0 0 0 0];
internalField   uniform 0;
boundaryField
{
    #includeEtc "caseDicts/setConstraintTypes"
    "(outlet|bottom|sides)"
    {
        type            zeroGradient;}
    inlet
    {
        type            waveAlpha;
        U               U;
        inletOutlet     true;}
    top
    {
        type            inletOutlet;
        inletValue      uniform 0;
        value           uniform 0; }}
// ***************************************** //
```

In code 5 the wave properties are defined by the wave amplitude, length and the water current speed **UMean**. In addition, the user can define the wave direction (**angle**). The wave origin and direction are specified with respect to the computational domain coordinate system generated by the **blockMesh** utility. Here we are simulating an (**Airy**) wave that obeys linear wave theory. Other higher order nonlinear wave models are also included in the OpenFOAM wave library such as Stoke's waves.

At the end of the **waveProperties** file, **scale** object is defined to linearly damp the generated waves in the range 7-9 m of the tank length in the wave propagation direction.

Code 5: waveProperties. Wave properties definition

```
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website:  https://openfoam.org
    \\  /    A nd           | Version:  8
     \\/     M anipulation  |
\*------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "constant";
    object      waveProperties;
}
// * * * * * *  * * * * * * * * * * * * * * //
origin          (0 0 0);
direction       (1 0 0);
waves
(
    Airy
    //Stokes2
    {
        length      3;
        amplitude   0.01;
        phase       0;
        angle       0; });
UMean           (0 0 0);   // current speed
scale           table ((7 1) (9 0));
crossScale      constant 1;
// ********************************** //
```

Code 6 shows the IC for the phase fraction α in the computational domain. First, the whole domain α has value of zero that represents air. Then using the **boxToCell** object, all cells in the in the box dimension of interest can be assigned new value of $\alpha = 1$ that represent water. The z_{max} value in the **box** object is set to zero which puts the undisturbed mean water free surface level at zero on the z axis.

The user can run the **setFields** command in the **blueCFD** terminal by typing **setFields**. To view the α field run **paraFoam** in **blueCFD** terminal to open ParaView as shown in figure 5.

Code 6: setFileds. Defining the initial mean water level in the NWT

```
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website:  https://openfoam.org
    \\  /    A nd           | Version:  8
     \\/     M anipulation  |
\*------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      setFieldsDict;
}
// * * * * * *  * * * * * * * * * * * //
defaultFieldValues
```

```
( volScalarFieldValue alpha.water 0
);
regions
(
    // Set cell values
    // (does zerogradient on boundaries)
    boxToCell
    {  box (-999 -999 -999) (999 999 0);
        fieldValues
        (
            volScalarFieldValue alpha.water 1
        );});
// ****************************** //
```

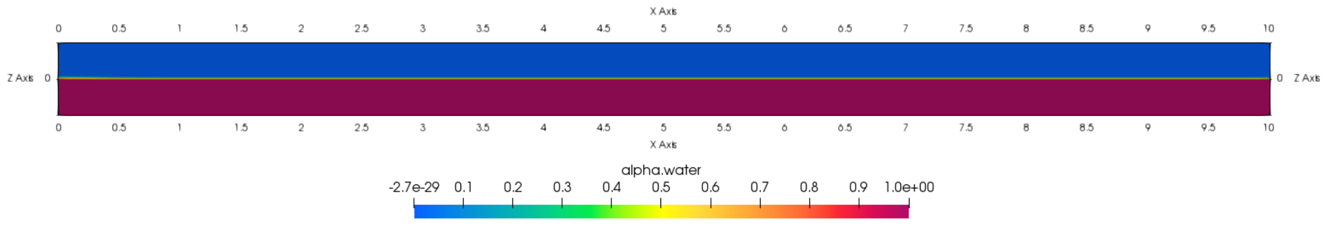


Figure 5: Undisturbed phase fraction α. α=1 is water, $\alpha = 0$ is air

7 OpenFOAM simulation control

The user can control the major parameters of the case using the **controlDict**. These parameters include solver type (**interFoam**), simulation **endtime**, data saving and writing intervals and the maximum allowable Courant number for simulation stability control. Also in **controlDict**, the user can call the libraries used in the simulation such as the wave generation library **libwaves.so**. In addition, other post post processing and, output controls and data logging at certain locations or surfaces can be achieved. In this example we used the **interfaceHeight** function to monitor the water free surface elevation at certain location in the NWT.

Other simulation control files are placed in the **system** folder such as **fvSchemes** and **fvSoluiton**, these two files can control the numerical schemes such as the spatio-temporal numerical differentiation in the **interFoam** solver or the parameters of the velocity-pressure coupling algorithm.

To start the **interFoam** solver, run **interFoam** command in the **blueCFD** terminal. The user will notice new folders are being created in the root folder while the simulation is running. These have names corresponding to the time stamps being written. A **postProcessing** folder is also generated which includes the **interfaceHeight** output containing the wave elevation at the wave probe point.

Code 7: ControlDict.

```
  =========                 |
  \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
   \\    /   O peration     | Website:  https://openfoam.org
    \\  /    A nd           | Version:  8
     \\/     M anipulation  |
\*------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      controlDict;
}
// * * * * * * * * * ** * * * * * * * * * * * * * * //
application     interFoam;
startFrom       latestTime;
startTime       0;
stopAt          endTime;
endTime         80;
deltaT          0.01;
writeControl    adjustableRunTime;
writeInterval   0.1;
purgeWrite      0;
writeFormat     ascii;
writePrecision  6;
writeCompression off;
timeFormat      general;
timePrecision   6;
runTimeModifiable yes;
adjustTimeStep  yes;
maxCo           1.5;
maxAlphaCo      1.5;
maxDeltaT       0.1;
libs
( "libwaves.so");

functions
{
    interfaceHeight1
    { type            interfaceHeight;
        libs            ("libfieldFunctionObjects.so");
        locations       (( 5 0 -1) );
        alpha           alpha.water;
    }}
// ******************************************** //
```

8 Results and post processing

After the solver finishes, the user can view the results in **ParView** by typing **paraFoam** in the **blueCFD** terminal. Figure 6 shows the pressure contour for the water free surface at isosurface $\alpha = 0.5$.

Figure 7 shows the free surface elevation at the virtual wave probe point (The output of the **interfaceHeight** function defined in the **controlDict**).

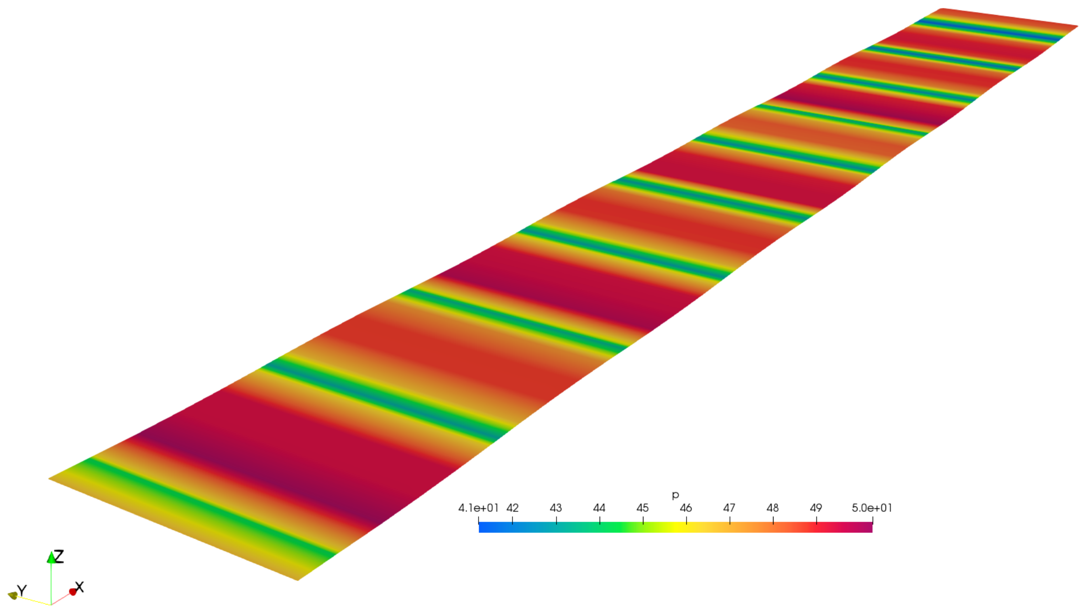


Figure 6: Pressure contour at the water free surface

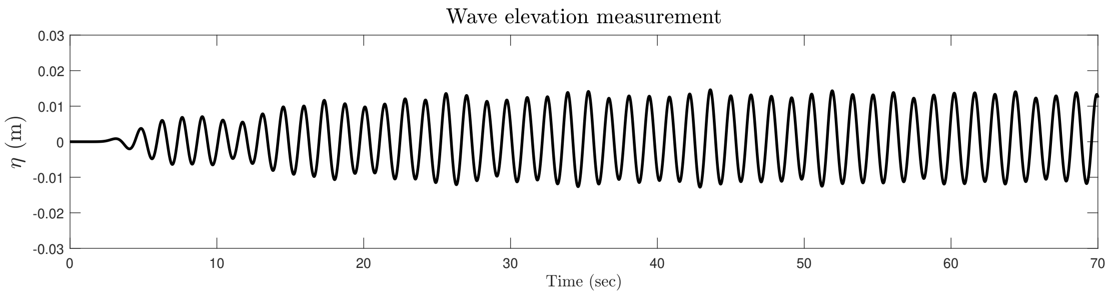


Figure 7: Free surface water elevation at the virtual wave probe point

9 Allrun/Allclean scripts

The user can automate the case running process by running the **Allrun** script in the **blueCFD** terminal as shown in code 8. The **Allrun** script first makes a copied directory for the initial condition **0** folder, then runs **(blockMesh, setFields,interFoam)** in order. To run the **Allrun** script, simply type **./Allrun** in the **blueCFD** terminal. To delete all results and data logging of the simulation in an efficient and easy way, the user can run the **Allclean** script shown in code 9.

Code 8: Allrun script

```
. $WM_PROJECT_DIR/bin/tools/RunFunctions
cp -r 0.org 0    # To copy the 0.org folder to 0 folder
runApplication blockMesh
runApplication setFields
runApplication interfoam
```

Code 9: Allclean script

```
# Source tutorial clean functions
. $WM_PROJECT_DIR/bin/tools/CleanFunctions
# Remove surface
rm -f constant/triSurface/DTC-scaled.stl.gz > /dev/null 2>&1
rm -rf constant/extendedFeatureEdgeMesh > /dev/null 2>&1
rm -f constant/triSurface/DTC-scaled.eMesh > /dev/null 2>&1
cleanCase
rm -r 0
```

References

Damian, S. M. (2012). Description and utilization of interfoam multiphase solver. *International Center for Computational Methods in Engineering*, pages 1–64.

Greenshields, C. (2018). Openfoam–the openfoam foundation–user guide.

Holzmann, T. (2016). Mathematics, numerics, derivations and openfoam®. *Loeben, Germany: Holzmann CFD*.