$(waveTheories)/$

Focused Wave generation based on Linear New Wave Theory, using OpenFOAM and waves2Foam toolbox

Eirini Katsidoniotaki

Department of Engineering Sciences/Electricity Division,
Uppsala University,
Uppsala, Sweden

2019-11-27

Contents

waves2Foam toolbox developed at the Technical University of Denmark by Niels G. Jacobsen

- Plug-in toolbox to the OpenFOAM package
- Generation and Absorption of free surface waves
- Modelling free-surface and bodies interaction

Intructions on how to download and install

waves2Foam toolbox can be downloaded

```
svn co http://svn.code.sf.net/p/openfoam-extend/svn \
/trunk/Breeder_1.6/other/waves2Foam
```

The toolbox's manual is available on the following link:

```
https://www.researchgate.net/publication/ \
319160515_waves2Foam_Manual
```

More info at:

```
http://openfoamwiki.net/index.php/Contrib/waves2Foam
```

Intructions for coupling with OpenFOAMv1906

1. In the file:

```
/waves2Foam/src/waves2FoamProcessing/Make/les
```

Comment out the line as shown below:

```
/* $(ppw)/$(spec)/$(specHelp)/complexExp.C */
```

2. In the file:

```
/postProcessing/postProcessingWaves/spectralAnalysis  \
/fftBasedMethods/reflectionAnalysis2DFFT/  \
/reflectionAnalysis2DFFT.C
```

In 4 instances in the code, change:

```
"complex::zero" to "complex(Zero)"
```

waves2Foam toolbox includes utilities for:

- Probes and Wave Gauges definition
- Wave Generation and Absorption through Relaxation Zone technique
- Wave Theory selection
- Initial conditions according to user defined wave theory
- Solvers for the wave interaction and propagation

Available Wave Theories in waves2Foam toolbox

1. Regular Wave Theories
   - First order Stokes
   - Second order Stokes
   - Fifth order Stokes
2. Bichromatic Wave Theories
   - First order bichromatic
   - Second order bichromatic
   - Irregular waves First order

res Potential Current

res Solitary First order

res Combined Waves

res External Wave Theories
   - Fast summation of irregular waves
   - OceanWave3D

waveProperties.input file is the key component for wave generation

- All the communication between the User, waves2Foam toolbox and OpenFOAM
- Located in `constant` folder of the case directory
- All the ocean wave related information

waves2Foam consists of the libraries:

1. waves2Foam
2. waves2FoamMooring
3. waves2FoamPorosity
4. waves2FoamProcessing
5. waves2FoamSampling

For the purpose of this tutorial, the following libraries are important:

- **waves2Foam** for the Wave Theories
- **waves2FoamProcessing** for setting the wave parameters

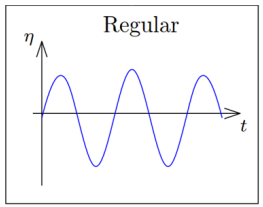
Extreme Wave Simulations

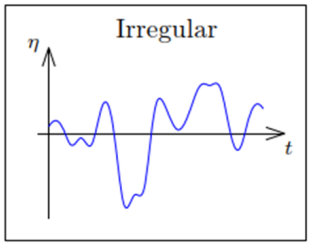
Extreme waves have the key role for the development of offshore Renewable Energy Systems

- Experiments in wave tanks are expensive and lack of flecxibility
- CFD has attracted a lot of attention
- Ability to predict impact forces
- But how to simulate Extreme Waves?

Extreme waves are highly transient events within a multi-frequency sea state

- High order Stokes waves are insufficient
- Random wave generation is computational expensive



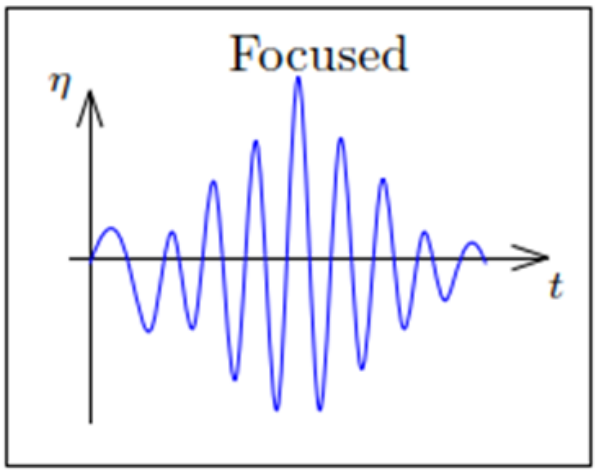


Need for a methodology for **Extreme Wave Simulation!**

Focused Wave: Concept for extreme wave representation

Motion of a single event with a **specific shape** and **crest height** over a single associated period.

Examination of **peak surface elevation** and **loads**.



NewWave Theory

NewWave Theory is extensively referred in Offshore Engineering for modelling extreme wave interactions with offshore structures.

- Linear, Random, Gaussian sea
- Sea state discretization into a finite number of sinusoidal wave components
- In this tutorial, linear NewWave approximation is described

The focused wave group of localized waves is derived from a measured theoretical spectrum

The amplitude a_i, of each wave component N, for a specific frequency f_i is defined as:

$$a_i = A_o \frac{S(f_i)\Delta f}{\sum_{n=1}^{N} S(f_i)\Delta f} \tag{1}$$

where the frequency step Δf given by

$$\Delta f = \frac{f_u - f_l}{N-1} \tag{2}$$

Ao is the target theoretical linear crest amplitude of the focused wave given by

$$A_o = \sqrt{2m_0 ln(N)} \tag{3}$$

JONSWAP or Pierson-Moskowitz are frequently employed for the Surface spectral density $S(f_i)$.

The linear surface displacement η is given by:

$$\eta(x,t) = \sum_{n=1}^{N} a_i cos[k_i(x - x_0) - \omega_i(t - t_0)] \tag{4}$$

x_0,t_0 are the predefined focal location and focal time, respectively, $k_i = \omega_i^2/gtanh(k_i h)$ is the wave number and $\omega_i = 2\pi f_i$ is the frequency.

Implementation

waveTheory class part of waves2Foam library

Base class waveTheory and sub-classes located at:

`$WM_PROJECT_DIR/applications/utilities/waves2Foam/src \`
`/waves2Foam/waveTheories/waveTheory`

Each sub-class follows a different wave theory (Stokes first, irregular etc..)

waveTheory.H

```
//- Runtime type information
    TypeName("waveTheory");
    // Declare run-time constructor selection table
        declareRunTimeSelectionTable
        (
            autoPtr,
            waveTheory,
            dictionary,
            (
                const word& subDictName, const fvMesh& mesh_
            ),
            (subDictName, mesh_)
        );
```

waveTheory.H

Auto-pointer connects base class and sub-classes
The wave theory is passed as argument through the dictionary

```
 // Constructors
        //- Construct from components
        waveTheory
        (
            const word& type,
            const fvMesh& mesh_
        );
// Selectors

        //- Return a reference to the selected turbulence model
        static autoPtr<waveTheory> New
        (
            const word& subDictName,
            const fvMesh& mesh_
        );
```

waveTheory class

Virtual functions for defining the surface elevation η, pressure gradient p and velocity U

```
virtual scalar eta
        (
            const point&,
            const scalar&
        ) const = 0;

        virtual vector U
        (
            const point&,
            const scalar&
        ) const = 0;
```

irregular.C sub-class

Irregular Wave Theory is one of the sub-classes of the main class waveTheory.

```
class irregular
:
    public waveTheory
{
...
}
```

irregular.C sub-class

Member function for surface elevation η

```
scalar irregular::eta
(
    const point& x,
    const scalar& time
) const
{
    scalar eta(0);

    forAll (amp_, index)
    {
        scalar arg = omega_[index]*time - (k_[index] & x) + phi_[index];
        eta += amp_[index]*Foam::cos(arg);
    }
    eta *= factor(time);
    eta += seaLevel_;

    return eta;
}
```

wavePropertie.C part of waves2FoamProcessing library

Base class wavePropertie.C and sub-classes located at:

```
$WM_PROJECT_DIR/applications/utilities/waves2Foam/src \
/waves2FoamProcessing/preProcessing/setWavePropeties
```

Each sub-class follows a different wave theory (Stokes first, irregular etc..)

waveProperties.C

```
autoPtr<setWaveProperties> setWaveProperties::New
(
    const Time& rT,
    dictionary& dict,
    bool write
)
{
    word waveTheoryTypeName;
    dict.lookup("waveType") >> waveTheoryTypeName;

    setWavePropertiesConstructorTable::iterator cstrIter =
        setWavePropertiesConstructorTablePtr_->find
        (
            waveTheoryTypeName+"Properties"
        );

    ....
    ....
    ....

    return autoPtr<setWaveProperties>(cstrIter()(rT, dict, write));
}
```

irregularProperties sub-class

`waveProperties` file is created according to the selected Wave Theory

```
void irregularProperties::set( Ostream& os )
{
    // Write the beginning of the sub-dictionary
    writeBeginning( os );

    // Write the already given parameters
    writeGiven( os, "waveType" );
    writeGiven( os, "spectrum");
    writeGiven( os, "N" );
    writeGiven( os, "Tsoft");

    if (dict_.found("writeSpectrum" ))
    {
        writeGiven( os, "writeSpectrum");
    }

    if (dict_.found("Tend"))
    {
        writeGiven(os, "Tend");
        writeGiven(os, "Tdecay");
    }
```

irregularProperties sub-class

Wave Spectra class is called by the irregularProperties

```
// Make a pointer to the spectral theory
    scalarField amp(0);
    scalarField frequency(0);
    scalarField phaselag(0);
    vectorField waveNumber(0);

    autoPtr<waveSpectra> spectra
    (
        waveSpectra::New(rT_, dict_, amp, frequency, phaselag, waveNumber)
    );
    // Computing the spectral quantities
    spectra->set( os );

    if (write_)
    {
        writeDerived( os, "amplitude", amp);
        writeDerived( os, "frequency", frequency);
        writeDerived( os, "phaselag", phaselag);
        writeDerived( os, "waveNumber", waveNumber);
    }
```

Modifications

Modify waves2Foam library

- Starting by creating a new wave type named `focusedWave` in the library `waves2Foam`
- Use the `irregular` wave type as base
- Follow the commands:

```
cd $WM_PROJECT_USER_DIR/applications/utilities  \
/waves2Foam/src/waves2Foam/waveTheories

mkdir --parents focusedWave/focusedWave

cp -r irregular/irregular/irregular.* \
focusedWave/focusedWave/
```

Rename file, folders and the source code

```
cd focusedWave/focusedWave
mv irregular.C focusedWave.C
mv irregular.H focusedWave.H
sed -i s/irregular/focusedWave/g focusedWave.H
sed -i s/irregular/focusedWave/g focusedWave.C
```

Create Make folder

- First, create a back-up of the original `Make` folder and rename it `Make_backup`.
- Then, a new `Make` folder is created and working on it.

```
cd $WM_PROJECT_USER_DIR/applications/utilities  \
/waves2Foam/src/waves2Foam

mv Make Make_backup

cp -r Make_backup Make

cd Make
```

Modify Make/files

Add the following piece of code at the end of the section `/*WAVE THEORIES*/`:

```
/* Focused wave theories */
focusedWave=focusedWave
$(waveTheories)/$(focusedWave)/focusedWave/focusedWave.C
```

focusedWave and focusedWaveProperties class are going to be part of the waves2Foam toolbox

Go to:

```
cd $WM_PROJECT_USER_DIR/applications/utilities/waves2Foam/src
```

Compile the libraries by executing:

```
./Allwmake
```

Tutorial

Description

Numerical Wave Tank (NWT) for NewWave Theory Focused Wave

- 2D NWT of COAST Lab Ocean Basin at the University of Plymouth, w/o floating body
- 3m depth, 8 wave gauges positions are used
- Pierson-Moskowitz spectrum together with NewWave Theory
- 65 wave components (simplified for faster simulation)
- Frequency range uniformly spaced between 0.101563Hz and 2Hz
- Theoretical focus location x_0=4.35m (wave gauge 5)
- Theoretical focus time t_0=20sec

NewWave Theory Characteristics

Ao (m)	fp (Hz)	Hs (m)	kA (m)	x_0 (m)	t_0 (sec)	f_l (Hz)	f_u (Hz)	N (-)
0.25	0.4	0.274	0.160972	4.35	20	0.101563	2	65

Create the case newWaveFocusedWave

- Copy the existing tutorial `waveFlume` included in the `waves2Foam toolbox`
- Modify the `blockMeshDict` to match the new NWT dimensions
- New boundary conditions `alpha.water`, `p_rgh`, `U`
- Modify the `fvSchemes` and `fvSolution` in the `system` folder
- Modify the `controlDict` in the `system` folder, mainly by adding the wave gauges
- `waveProperties.input` file is adjusted to the new wave type

Getting Started

Go to:

```
cd $WM_PROJECT_USER_DIR/applications/utilities/waves2Foam/tutorials/waveFoam
cp -r waveFlume newWave_focusedWave
cd newWave_focusedWave
```

waveProperties.input file

```
inletCoeffs
{
         waveType   focusedWave;
         N 65;

        // Ramp time of 2 s
            Tsoft       10 ;

        //Define the phases
           phaseMethod focusingPhase;
           focusTime 20.0;
           focusPoint (4.25 0 0);
```

waveProperties.input file

```
//Define the spectrum
            spectrum  PiersonMoskowitz_FW;
            Hs 0.274;
            Tp 2.5;
            depth         3.0;
            direction (1 0 0);
             Ao 0.25;
//Define the frequency
  frequencyAxis
           {
              discretisation equidistantFrequencyAxis;
              lowerFrequencyCutoff 0.0869595;
              upperFrequencyCutoff 2;
              writeSpectrum false;
           }
```

wavesProperties file is created by applying setWaveProperties utility

In the new file the wave components characteristics are calculated:

```
amplitude  nonuniform List<scalar>
    65
    (
        4.12942e-75
        1.31611e-30
        5.28844e-15
        ....
        ....
        ....
        3.55175e-05
        3.29281e-05
        3.05617e-05
    );
```

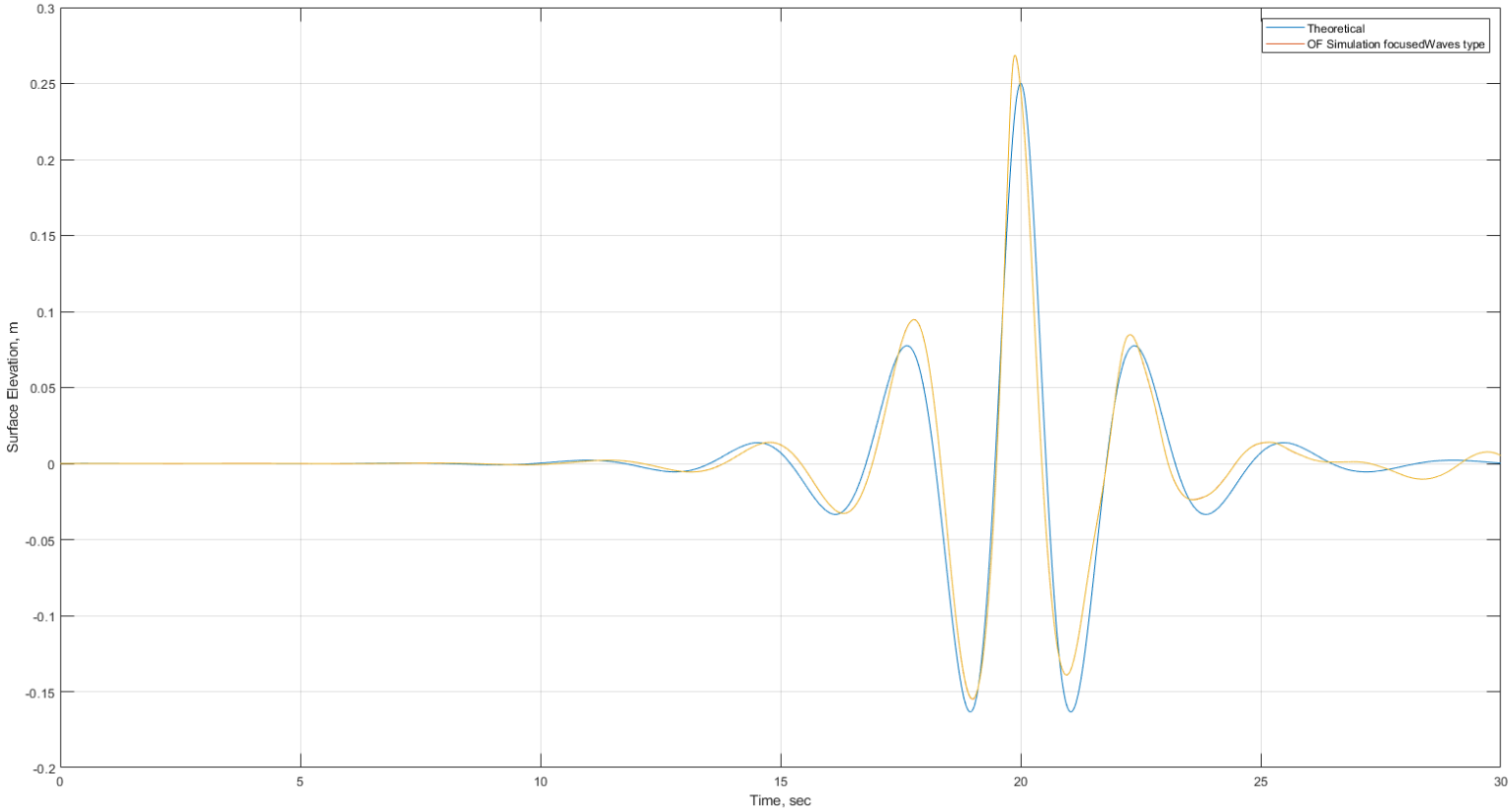
```
frequency nonuniform List<scalar>
 65
 (
     0.101675
     0.131107
     0.160538
     ....
     ....
     ....
     1.92642
     1.95585
     1.98528
 );
```

waveProperties file

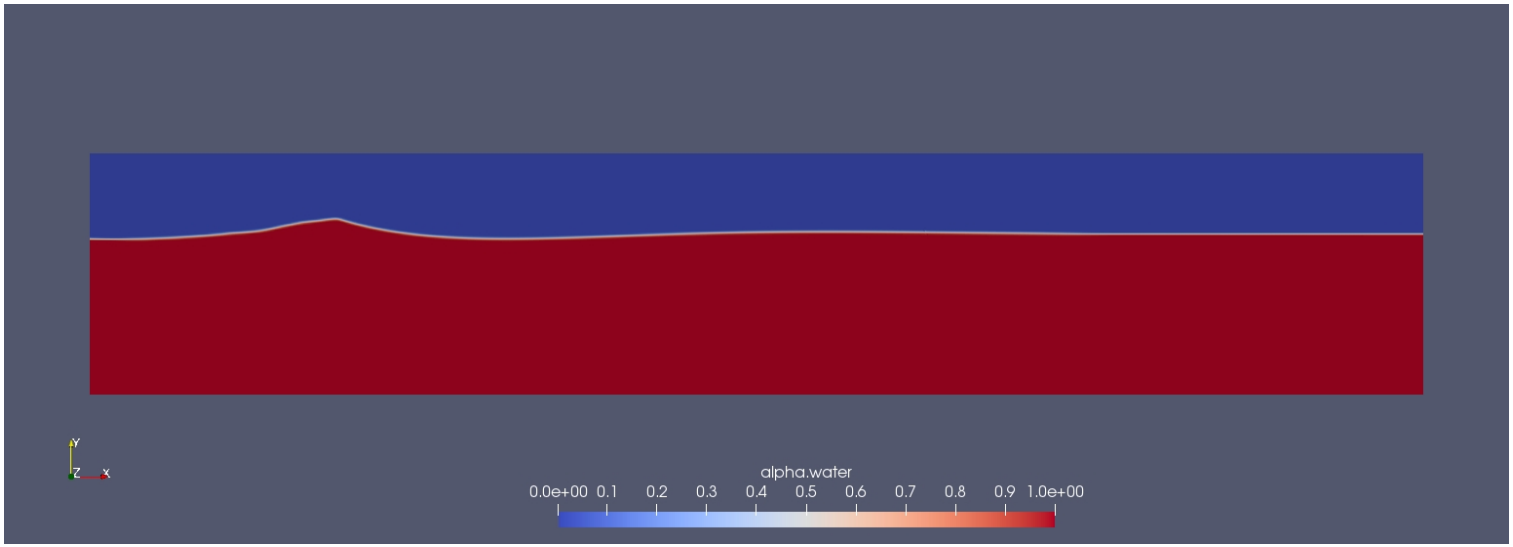
```
phaselag   nonuniform List<scalar>
    65
    (
        -12.2657
        -15.8068
        -19.3402
        ....
        ....
        ....
        -178.609
        -180.353
        -182.068
    );
```

```
waveNumber nonuniform List<vector>
 65
 (
     (0.120267 0 0)
     (0.157304 0 0)
     (0.196146 0 0)
     ....
     ....
     ....
     (14.9346 0 0)
     (15.3944 0 0)
     (15.8612 0 0)
 );
```

Theoretical vs Numerical Simulation



Animation



Thank you for your attention!