**05/03/2023**

**Installation of OpenFOAM2012 and Waves2Foam on TU Delft AE Cluster**

1. **When you install**

It’s better to start an interactive job on the one of the worker nodes and compile software in order to avoid causing a high load on cluster.

In other way, if you prefer to compile on master node because of the large amount of file IO, you can limit the compilation to 10 cores by typing the following command.

export WM\_NCOMPPROCS=10  
Alternatively / additionally it is also possible to set a different "nice" value for processes, which means setting it to a lower priority. This means that if many other users are on the master node, their processes get priority. Once logged in, you can open a new Bash shell with a higher nice value (the higher, the less priority) and execute the compilation script from that Bash shell:  
Once logged in type:  
nice -n 10 bash  
to open a new bash session with a nice value of 10 (default a users has value zero; the maximum (lowest priority) you can select is 19).

1. **Installation of OpenFOAM2012 on CentOS**

Referred document:

<https://openfoamwiki.net/index.php/Installation/Linux/OpenFOAM%2B-v1606%2B/CentOS_SL_RHEL>

Source file location of OpenFOAM2012:

<https://www.openfoam.com/download/release-history.php>

1. **Pre-require software (AE cluster already has them, and so you can skip this step.)**

The following software is required for OpenFOAM installation.

git, gcc-c++, bison, flex, m4, glibc-devel, glibc-devel.i686, zlib-devel

If you do not have, you need to install them as a root.

yum groupinstall 'Development Tools'

yum install openmpi openmpi-devel zlib-devel texinfo gstreamer-plugins-base-devel \

libXext-devel libGLU-devel libXt-devel libXrender-devel libXinerama-devel libpng-devel \

libXrandr-devel libXi-devel libXft-devel libjpeg-turbo-devel libXcursor-devel \

readline-devel ncurses-devel python python-devel

*#This one is useful, but not crucial*

yum upgrade

1. **Download and unpack**

*#OpenFOAM+ downloading and installation*

cd ~

mkdir OpenFOAM

cd OpenFOAM

wget "https://sourceforge.net/projects/openfoamplus/files/v2112/OpenFOAM-v2012.tgz"

wget "https://sourceforge.net/projects/openfoamplus/files/v2112/ThirdParty-v2012.tgz"

tar -xzf OpenFOAM-v2012.tgz

tar -xzf ThirdParty-v2012.tgz

Note: Basically, the folder name should be OpenFOAM. If you would like to change the folder name, you need to change the file of /etc/bashrc.0. If the installation through wget doesn’t work, then OpenFOAM-v2012.tgz and ThirdParty-v2012.tgz files need to be installed manually and copied to the home/<netID> folder in the hpc.

1. **A few details need to be fixed, before proceeding:**

The version name changed. From MPFR\_ARCH\_PATH to GMP\_ARCH\_PATH

sed -i -e 's=gmp-system=gmp-5.1.2=' -e 's=mpfr-system=mpfr-3.1.2=' -e 's=mpc-system=mpc-1.0.1=' OpenFOAM-v2012/etc/config.sh/compiler

sed -i -e 's=**\(**MPFR\_ARCH\_PATH)/lib**\)**=**\1**$(WM\_COMPILER\_LIB\_ARCH)=' -e 's=**\(**GMP\_ARCH\_PATH)/lib**\)**=**\1**$(WM\_COMPILER\_LIB\_ARCH)=' OpenFOAM-v2012/wmake/rules/General/CGAL

The symbol \ when copied in the CentOS terminal of the hpc should become a backslash. It is fine this way.

1. **Load Open-MPI into the environment and set the shell setting**

module load mpi/openmpi-1.8.8-gnu || export PATH=$PATH:/usr/lib64/openmpi/bin

Here, you can confirm which Open-MPI you can use by typing module avail

You will get error message if you load invalid Open-MPI in your cluster.

* For building with the normal 32-bit integer support (maximum 2.147×109 cells, faces or points):

source $HOME/OpenFOAM/OpenFOAM-v2012/etc/bashrc WM\_COMPILER\_TYPE=ThirdParty

Also, save an *alias* in the personal .bashrc file, simply by running the following command:

echo "alias of2012='module load mpi/openmpi-1.8.8-gnu; source **\$**HOME/OpenFOAM/OpenFOAM-v2012/etc/bashrc $FOAM\_SETTINGS'" >> $HOME/.bashrc

**Note**: This last line means that whenever you start a new terminal window or tab, you should run the *alias* command associated to the OpenFOAM+ 2012 shell environment. In other words, whenever you start a new terminal, you should run:

of2012

Note: You probably will see a message similar to this one and it's meant to do so until we've built the custom Gcc version:

Warning in /home/user/OpenFOAM/OpenFOAM-v2012/etc/config/settings.sh:

Cannot find /home/user/OpenFOAM/ThirdParty-v2012/platforms/linux64/gcc-4.8.5 installation.

Please install this compiler version or if you wish to use the system compiler,

change the 'WM\_COMPILER\_TYPE' setting to 'system'

1. **Download GCC and binutils source file**

Now we'll have to get all of the scripts we'll need to build GCC and binutils (because OpenFOAM-v2012 requires at least GCC 4.5 and CentOS only provides GCC 4.4):

cd $WM\_THIRD\_PARTY\_DIR

wget "https://raw.github.com/wyldckat/scripts4OpenFOAM3rdParty/master/getGcc"

wget "https://raw.github.com/wyldckat/ThirdParty-2.0.x/binutils/makeBinutils"

wget "https://raw.github.com/wyldckat/ThirdParty-2.0.x/binutils/getBinutils"

chmod +x get\* make\*

1. **Build CMake 3.0 or newer version**

CentOS 6.7 comes with CMake 2.6.4, but ParaView 5.0.1 needs a one of the versions from the more recent CMake 3.0 or newer and CGAL also needs CMake. Therefore, we'll need to do a custom build and we have to do it before we build the custom GCC+binutils (reason: [[1]](http://www.cfd-online.com/Forums/openfoam-installation/130205-paraview-4-redhat-possible.html#post478025)):

cd $WM\_THIRD\_PARTY\_DIR

wget "https://raw.github.com/wyldckat/scripts4OpenFOAM3rdParty/master/getCmake"

chmod +x getCmake

sed -i -e 's=2**\.**8=3.2=' -e 's=version}**\.**3=version}.1=' getCmake

./getCmake

./makeCmake > log.makeCM 2>&1

wmREFRESH

wmREFRESH command not found ... no worries

1. **Build Gcc and company (in ThirdParty directory)**

./getGcc gcc-4.8.5 gmp-5.1.2 mpfr-3.1.2 mpc-1.0.1

./makeGcc > log.mkgcc 2>&1 ( takes a long time, very long)

wmREFRESH

wmREFRESH command not found ... no worries

1. **Build Binutils**

./getBinutils

./makeBinutils gcc-4.8.5 > log.mkbinutils 2>&1

1. **Build Qt version more than 4.7**

CentOS 6.7 comes with Qt 4.6.2, but ParaView 5.0.1 needs at least 4.7. Therefore, we'll need to do a custom build of Qt 4.8.6:

cd $WM\_THIRD\_PARTY\_DIR

*#Get the scripts we need*

wget https://github.com/wyldckat/scripts4OpenFOAM3rdParty/raw/master/getQt

*#make it executable*

chmod +x getQt

*#define correct download version and download it*

sed -i -e 's=4**\.**6=4.8=' -e 's=4**\.**8**\.**4=4.8.6=' -e 's=/**\$**major/**\$**tarFile=/$major/$version/$tarFile=' getQt

./getQt

./makeQt qt-4.8.6 > log.mkqt 2>&1

1. **Build ParaView 5.0.1**

Now, in order to build ParaView 5.0.1 that comes with OpenFOAM:

For building ParaView 5.0.1 with Python and MPI, run:

cd $WM\_THIRD\_PARTY\_DIR

*#this will take a while... somewhere between 30 minutes to 2 hours or more*

./makeParaView -qmake $WM\_THIRD\_PARTY\_DIR/platforms/$WM\_ARCH$WM\_COMPILER/qt-4.8.6/bin/qmake -mpi -python > log.makePV 2>&1

Finally, update the shell environment:

wmREFRESH

wmREFRESH command not found ... no worries

1. **Build ThirdParty**

Now let's build the ThirdParty folder, because we need the shell environment to be updated afterwards, for CGAL/Boost/FFTW to be properly picked up for building OpenFOAM+:

cd $WM\_THIRD\_PARTY\_DIR

*#we need a little fix for FFTW*

sed -i -e 's=unset \_foamAddPath=unset -f \_foamAddPath=' makeFFTW

*#need to fix CGAL 4.12.2...*

sed -i -e 's=**\(**#include <CGAL/IO/Color.h>**\)**=**\1\n**#include <CGAL/assertions.h>=' CGAL-4.12.2/include/CGAL/IO/io.h

*# This next command will take a while... somewhere between 5 minutes to 30 minutes.*

./Allwmake > log.make 2>&1

*#update the shell environment*

wmREFRESH

wmREFRESH command not found ... no worries

1. **Build OpenFOAM**

Go into OpenFOAM's main source folder:

cd $WM\_PROJECT\_DIR

This next command will take a while... somewhere between 30 minutes to 3-6 hours:

./Allwmake -j 4 > log.make 2>&1

**Note:** The "4" refers to the number of cores to be used for building in parallel. In addition, the amount of RAM needed for building scales with the number of cores used, something like 1GB of RAM per core; a minimum of 1.5GB is needed for linking the libraries, which is not done in parallel.

Run it a second time for getting a summary of the installation:

./Allwmake -j 4 > log.make 2>&1

**Note:** Technically, the build will stop as soon as the first error is found. Running a second time only makes it easier to sort through between what's already been built and where the crash occurred.

To check if everything went well: Check if icoFoam is working, by running this command:

1. **Check the success of installation**

icoFoam -help

which should tell you something like this:

Usage: icoFoam [OPTIONS]

options:

-case <dir> specify alternate case directory, default is the cwd

-noFunctionObjects

do not execute functionObjects

-parallel run in parallel

-roots <(dir1 .. dirN)>

slave root directories for distributed running

-srcDoc display source code in browser

-doc display application documentation in browser

-help print the usage

It’s fine also if it is not exactly the same message, but a similar one

1. **Installation of waves2Foam**

Waves2Foam install manual :

<https://www.researchgate.net/profile/Niels_Jacobsen3/publication/319160515_waves2Foam_Manual/links/5995c1e7aca27283b11b21a2/waves2Foam-Manual.pdf>

1. **Pre-require software (AE cluster already has them, and so you can skip this step.)**

The following software is required for Waves2Foam installation manual.

gfortrun, git, subversion(SVN), GNU Scientific Library (GSL)

If you do not have, you need to install them as a root.

1. **Download and installation of waves2Foam ( Installation Manual pp. 10-11 )**

cd $WM\_THIRD\_PARTY\_DIR

## First, source your OpenFoam version

mkdir -p $FOAM\_RUN/../applications/utilities

cd $FOAM\_RUN/../applications/utilities

svn co http://svn.code.sf.net/p/openfoam-extend/svn/trunk/Breeder\_1.6/other/waves2Foam

cd waves2Foam

./Allwmake

There is a chance that waves2Foam is not correctly installed. While running Allwmake in applications/utilities/waves2Foam this error could show up

/usr/bin/ld: cannot find -lwaves2Foam

/usr/bin/ld: cannot find -lwaves2FoamGABC

/usr/bin/ld: cannot find -lwaves2FoamSampling

collect2: error: ld returned 1 exit status

The error comes from waves2Foam not finding the gfortran library (libgfortran.so). Turns out waves2Foam is not looking for it in the right directory.

To overcome this problem, you need to create a link for the libgfortran.so file into the OpenFoam library folder, which waves2Foam automatically checks:

ln -s /usr/lib/gcc/x86\_64-linux-gnu/9/libgfortran.so /home/lscarlatti/OpenFOAM/OpenFOAM-v2012/platforms/linux64GccDPInt32Opt/lib

Check the version of your laptop, it may not be the same as linux64Gcc63DPInt32Opt.

When installation has succeeded, if you type ‘wave’ and enter the tab, ‘waveFoam’, ‘wavesetParameters’, ‘waveField’ appears.

After install, in Waves2Foam folder, the shell setting need to be reflected.

source ./bin/bashrc

1. **Installation of waveDyMFoam ( Installation Manual pp. 45-46 )**

The interDyMFoam needs to be merged to waves2Foam as waveDyMFoam.

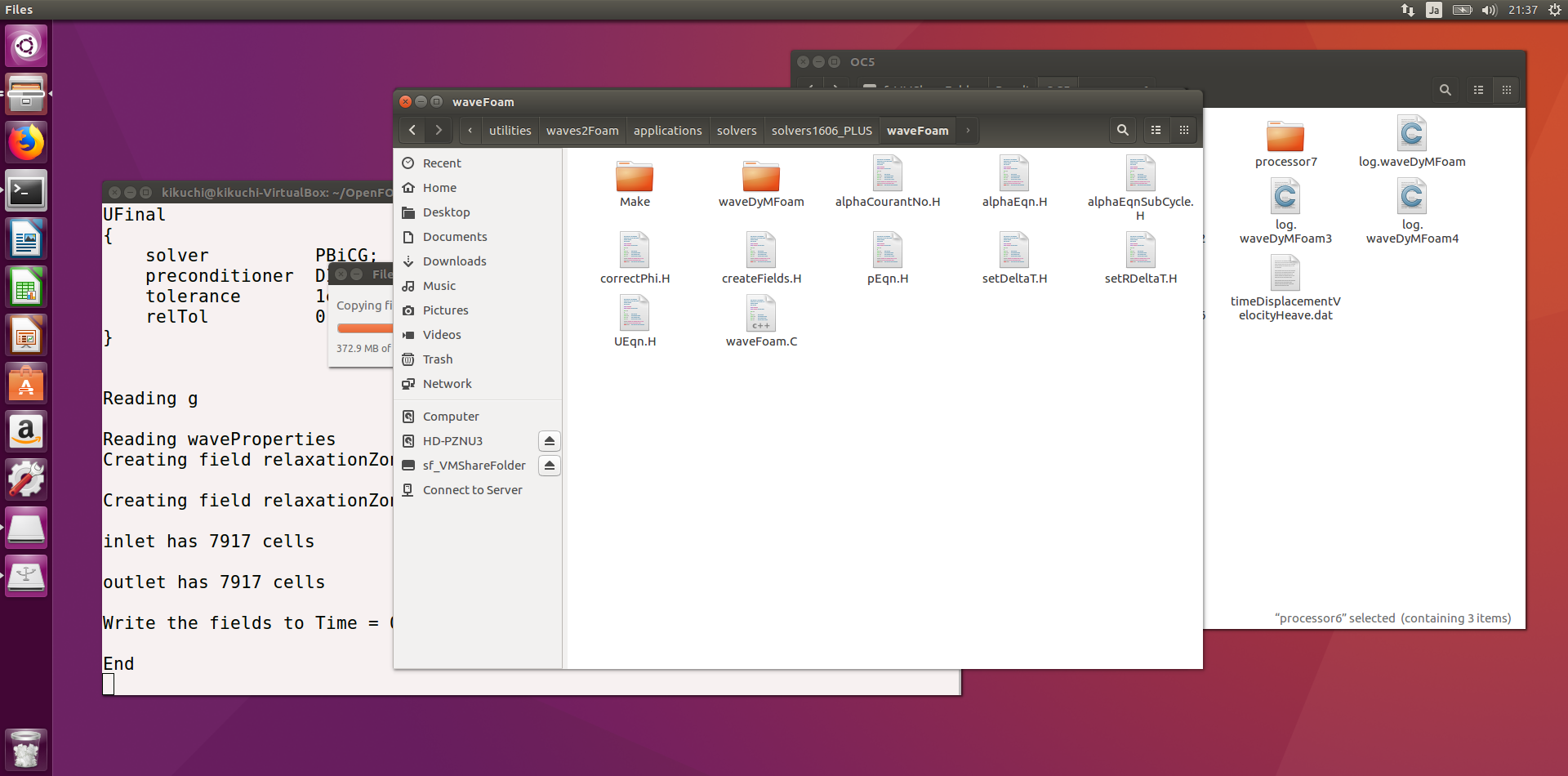
* 1. Copy interDyMFoam in OpenFOAM and locate in the following folder, renaming waveDyMFoam.

FOAM\_RUN/../applications/utilities/waves2Foam/applications/solvers/solvers1606\_PLUS/waveFoam

interDyMFoam in OpenFOAM is located in the following folder.

OpenFOAM/OpenFOAM-v2012/applications/solvers/multiphase/interFoam

After copying the interDyMFoam folder in the waveFoam folder, rename the interDyMFoam folder to ‘waveDyMFoam’. Now the content of the waveFoam folder should look like the image below:



(Before reading ②, you can skip to ⑥)

*Note: at the end of this document and extra description has been attached on what shall be changed after the step above. Try that if installation doesn’t succeed.*

* 1. In interDyMFoam (waveDyMFoam), the following files and folder are located.

Here, three files of interDyMFoam.C, files and options need to be modified following manual.

|-- correctPhi.H

|-- createControls.H

|--interDyMFoam.C

|--Make

| |-- files

| |-- options

|-- pEqn.H

|-- readControls.H

* 1. Then, executing wmake to compile waveDyMFoam.
  2. That’s all process in manual, but the following error occurs about gravitation definition. This is maybe what the Manual suggested in Warning 5.2 in pp.46.

waveDyMFoam.C:139:43: error: ‘ghRef’ was not declared in this scope  
                     gh = (g & mesh.C()) - ghRef;

1. And so, three files of createFields.H ,  waveFoam.C and waveDyMFoam.C file were modified.

In createFields.H file

All sentences relating with gravity were commented out.

//#include "readGravitationalAcceleration.H"  
//#include "readhRef.H"  
//#include "gh.H"  
//volScalarField gh("gh", g & (mesh.C() - referencePoint));  
//surfaceScalarField ghf("ghf", g & (mesh.Cf() - referencePoint));

 In waveFoam.C and waveDyMFoam.C file

# include "readhRef.H" and # include "gh.H" are added after #include "readGravitatioalAcceleration.H"

  #include "readGravitationalAcceleration.H"  
 #include "readhRef.H"  
 #include "gh.H"     /\* ghRef is defined in this file/  
 #include "readWaveProperties.H"  
 #include "createExternalWaveForcing.H"

("readhRef.H" and "gh.H"  are located in the following path.

/home/kikuchi/OpenFOAM/OpenFOAM-v2112/src/finiteVolume/lnInclude)

Like in waveFoam.C make sure that the waveDyMFoam.C file contains the line relaxing.correct():

#include "alphaControls.H"

#include "alphaEqnSubCycle.H"

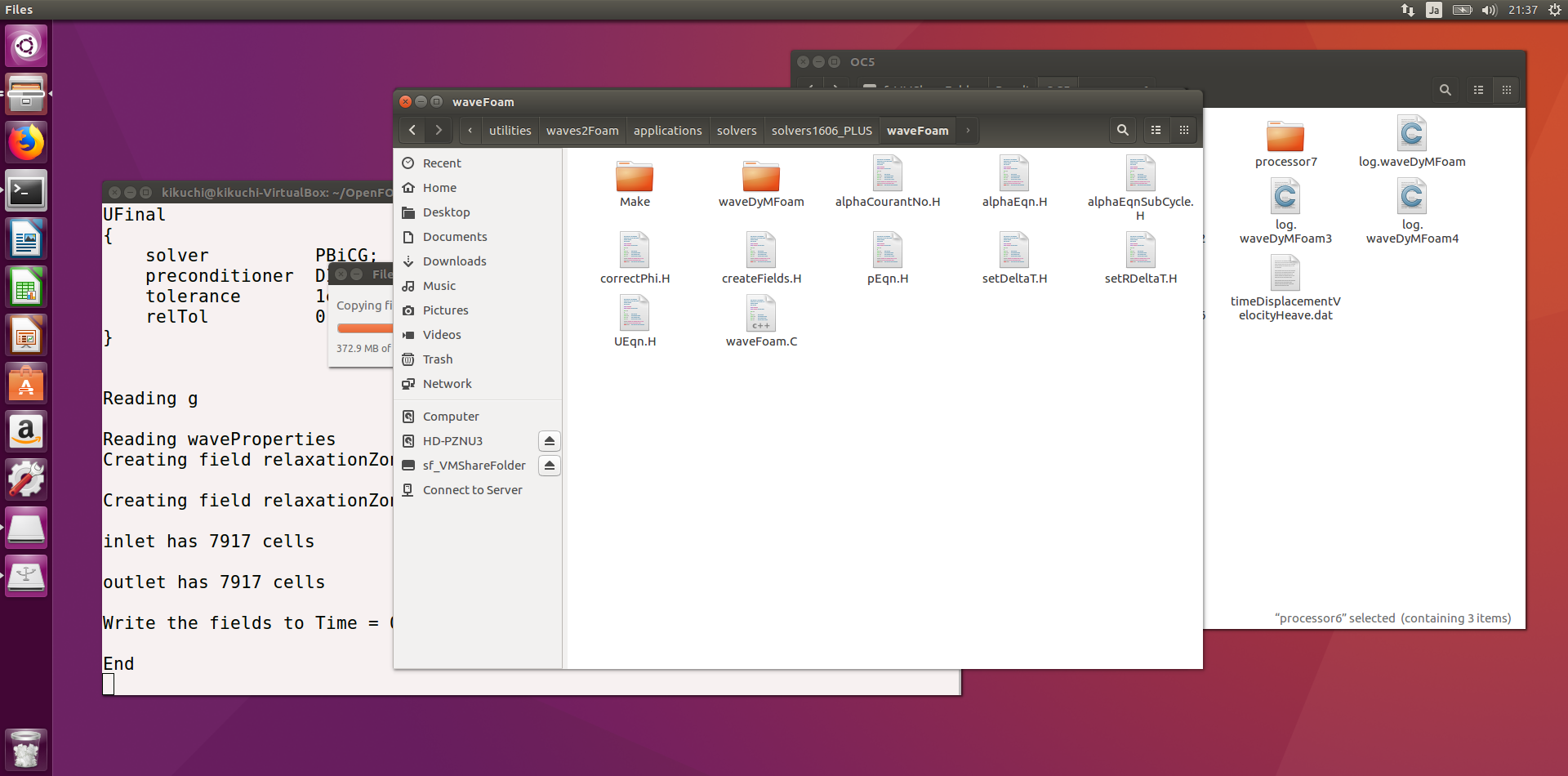
// include here relaxing.correct() like in waveFoam.C

relaxing.correct();

mixture.correct();

#include "UEqn.H"

1. In summary, easy installation way is copying already modified waveDyMFoam folder, waveFoam.C and createField.H from the folder *waveFoamInstallation*, and compile.



Fin.

If the installation of waveDyMFoam doesn’t succeed using the above mentioned instructions, try the following approach.

**Continue here after copying interDyMFoam and renaming it to waveDyMFoam.**

The following indicates the content of *‘/home/<USERNAME>/OpenFOAM/<USERNAME>-v2112/applications/utilities/waves2Foam/applications/solvers/solvers1606\_PLUS/waveFoam’* (replace <USERNAME> with your own username). The files that are marked red need to be changed, the green text indicates the changes. Instead of changing the files manually, they can simply be copied from the attached folder ‘waveFoam’.

* Make <dir>
  + Linux64GccDPInt32Opt <dir>  
    remove this folder
  + files
  + options
* waveDyMFoam <dir>
  + Make <dir>
    - Linux64GccDPInt32Opt  
      this folder shouldn’t be here yet, if it is present remove it. It comes back after compilation.
    - files  
      the name of the application shall be changed:   
      *waveDyMFoam.C*

*EXE = $(FOAM\_APPBIN)/waveDyMFoam*

* + - options  
      The wave libraries are missing in the options file, therefore replace the options file with the file in the attached folder.
  + correctPhi.H
  + createControls.H
  + pEqn.H
  + readControls.H
  + waveDyMFoam.C
* alphaCourantNo.H
* alphaEqn.H
* alphaEqnSubCycle.H
* correctPhi.H
* createFields.H
  + The following is commented out using ‘//’:

//#include "readGravitationalAcceleration.H"

//#include "readhRef.H"

//#include "gh.H"

//volScalarField gh("gh", g & (mesh.C() - referencePoint));

//surfaceScalarField ghf("ghf", g & (mesh.Cf() - referencePoint));

* + At the very end the following line is added:  
    relaxationZone relaxing(mesh, U, alpha1);
* pEqn.H
* setDeltaT.H
* setRDeltaT.H
* UEqn.H
* waveFoam.C
  + This file is almost identical to interFoam.C from ‘OpenFOAM/OpenFOAM-v2112/applications/solvers/multiphase/interFoam’ except for two lines in the beginning of the file:   
    #include "relaxationZone.H"

#include "externalWaveForcing.H"

Once you’ve made sure that the files the in waveFoam folder and its subdirectories are changed according to the instructions described above, go to the waveFoam folder (*‘/home/.../ solvers/solvers1606\_PLUS/waveFoam’)* and type the command *wmake* in the terminal. Now, compilation shall start. After this is done, go to the waveFoam/waveDyMFoam folder and also execute the *wmake* command in this folder. Now compilation should be done.

Check that you can run the case ‘test\_case’. In principle the commands are (remember to remove the files in the 0 folder and copy them from the org one, and to remove all the files inside polyMesh (in constant) but the blockMeshDict file, of course):

- blockMesh

- relaxationZoneLayout

- waveGaugesNProbes

- snappyHexMesh -overwrite

- extrudeMesh

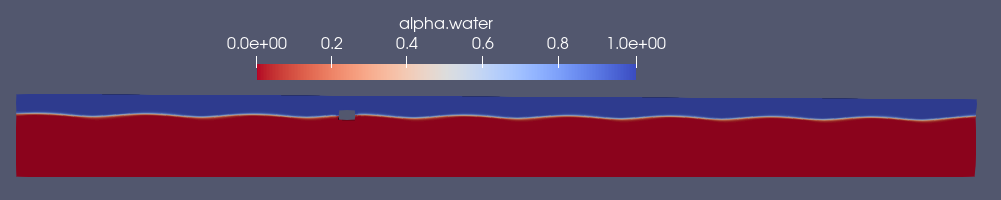
- setWaveParameters

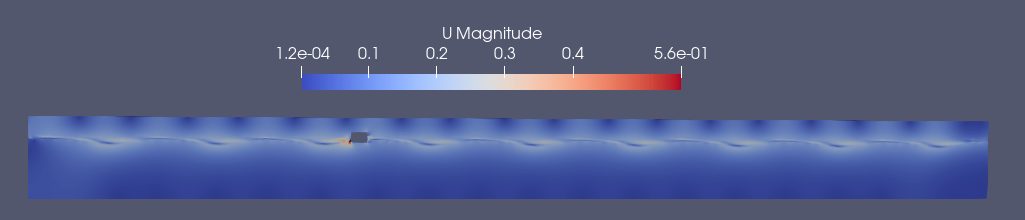
- setWaveField

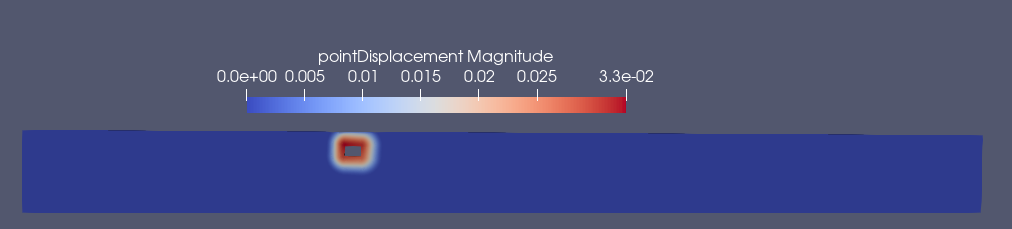
- relaxationZoneLayout

- waveDyMFoam

Note: the execution stops after the solver solved for up to time = 0.5 s. Now you can already see if the structure moves and if waves are generated. The following screenshots are from Paraview (load results in Paraview by making a <random\_name>.foam file and clicking on it) at t = 0.35 s:







If installation didn’t succeed remove the folder waves2foam (*/home/<USERNAME>/OpenFOAM/<USERNAME>-v2112/applications/utilities/waves2Foam*)

When you are in the folder */home/<USERNAME>/OpenFOAM/<USERNAME>-v2112/applications/utilities/* use the following command: *rm -rf waves2foam* to remove the folder and its subdirectories. Then, install and source waves2foam again:

cd $FOAM\_RUN/../applications/utilities svn co <http://svn.code.sf.net/p/openfoam-extend/svn/trunk/Breeder_1.6/other/waves2Foam>

cd waves2Foam

./Allwmake

*>Check that it works by typing wave… (and then waveFoam and something else should appear).*

source ./bin/bashrc

Now, try again to follow the steps listed under ‘Continue here after copying interDyMFoam and renaming it to waveDyMFoam’

On the OpenFoam forum cfd-online.com there is a topic on the installation of waves2foam, which might be helpful if you run into problems (note that a different version of OpenFoam is used here): <https://www.cfd-online.com/Forums/openfoam-community-contributions/205425-question-about-integration-moving-mesh-cases-waves2foam.html>