Acoustic streaming modeling

Milad Setareh

Applied Mechanics/Fluid Dynamics,
Amirkabir University of Technology, Tehran Polytechnic
Tehran, Iran

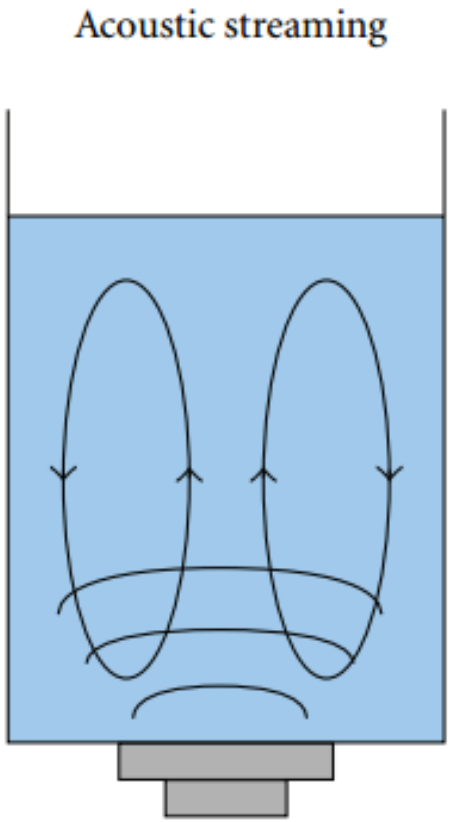
December, 2016

Outline

- Introduction
- Mathematical modeling of acoustic streaming
- Modifications
- Running the case

Introduction

Acoustic streaming is a vortex flow which shows the intraction between acoustic waves and flow.



Expansion of variables

$$\begin{aligned} \rho &= \rho^0 + \rho^1 + \rho^2 \\ u &= u^0 + u^1 + u^2 \\ p &= p^0 + p^1 + p^2 \end{aligned} \tag{1}$$

First order equations

$$\frac{\partial \rho^1}{\partial t} + \rho^0 \nabla \cdot \left(u^1\right) = 0 \tag{2}$$

$$\rho^0 \frac{\partial u^1}{\partial t} = -\nabla \cdot p^1 + \mu \nabla^2 u^1 \tag{3}$$

$$p^1 = c^2 \rho^1 \tag{4}$$

Second order equations

$$\frac{\partial \rho^2}{\partial t} + \rho^0 \nabla \cdot \left(u^2\right) = \left\langle -\nabla \cdot \left(\rho^1 u^1\right) \right\rangle \tag{5}$$

$$\rho^0 \frac{\partial u^2}{\partial t} = -\nabla \cdot p^2 + \mu \nabla^2 u^2 + \left\langle -\rho^1 \frac{\partial u^1}{\partial t} - \rho^0 \left(u^1 \cdot \nabla\right) u^1 \right\rangle \tag{6}$$

$$p^2 = c^2 \rho^2 \tag{7}$$

Copy the existing solvers

Copy the sonicLiquidFoam and buoyantBoussinesqSimpleFoam solvers in the local solvers directory and change its name.

```
cd $WM_PROJECT_USER_DIR
mkdir -p applications/solvers/compressible/sonicLiquidFoam
cd applications/solvers/compressible
cp -r $FOAM_SOLVERS/compressible/sonicLiquidFoam .
cd sonicLiquidFoam
mv sonicLiquidFoam.C mySonicLiquidFoam.C
sed -i "s/sonicLiquidFoam/mySonicLiquidFoam/g" Make/files
sed -i "s/FOAM_APPBIN/FOAM_USER_APPBIN/g" Make/files
wclean
wmake
```

Copy the existing solvers

```
cd $WM_PROJECT_USER_DIR
mkdir -p applications/solvers/heatTransfer/buoyantBoussinesq\
SimpleFoam
cd applications/solvers/heatTransfer
cp -r $FOAM_SOLVERS/heatTransfer/buoyantBoussinesq\
SimpleFoam .
cd buoyantBoussinesqSimpleFoam
mv buoyantBoussinesqSimpleFoam.C myBuoyantBoussinesq\
SimpleFoam.C
sed -i "s/buoyantBoussinesqSimpleFoam/myBuoyantBoussinesq\
SimpleFoam/g" Make/files
sed -i "s/FOAM_APPBIN/FOAM_USER_APPBIN/g" Make/files
wclean
wmake
```

creatFileds.H

Add the following lines into `creatFileds.H` for mySonicLiquidFoam solver. Also, replace `createphi.H` instead of `compressibleCreatePhi.H`

```
volVectorField sumAcousticForce
(
    IOobject
    (
        "sumAcousticForce",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::NO_WRITE
    ),
    mesh,
    dimensionedVector("meanAcousticForce",  dimensionSet(0,1,-2,0,0,0,0),
    vector(0,0,0))
);
```

Create the new solvers (mySonicLiquidFoam)

```
volVectorField meanAcousticForce
    (
        IOobject
        (
            "meanAcousticForce",
            runTime.timeName(),
            mesh,
            IOobject::NO_READ,
            IOobject::AUTO_WRITE
        ),
        mesh,
         dimensionedVector("meanAcousticForce", dimensionSet(0,1,-2,0,0,0,0),
         vector(0,0,0))
    );
```

Modify the solver

Remove compressibleCourantNo.H and replace it with CourantNo.H

```
sed -i "s/compressibleCourantNo.H/CourantNo.H/g" mySonicLiquidFoam.C
```

Also, we change the initial value of ρ to 1+psi*p in the createFields.H. For considering kinematic viscosity, it is better to change mu to nu in readTransportProperties.H file.

```
cd $WM_PROJECT_USER_DIR/applications/solvers/compressible/sonicLiquidFoam
sed -i "s/rho0/1/g" createFields.H
sed -i "s/mu/nu/g" readTransportProperties.H
```

mySonicLiquidFoam.C

Delete the line `#include "rhoEqn.H"`

Add `float iter=0.0` before run.time loop.

Add `#include "readPISOControls.H"` after `#include "createMesh.H` Remove `//#include "pimpleControl.H"` and `pimpleControl pimple(mesh);` Remove all lines started form pimple loop and ended with correct density.Then, paste the below code instead.

```
fvVectorMatrix UEqn
(
    fvm::ddt(U)
  - fvm::laplacian(nu, U)
);
solve(UEqn == -fvc::grad(p));
// --- PISO loop
  for (int corr=0; corr<nCorr; corr++)
{
    volScalarField rAU(1.0/UEqn.A());
    U = rAU*UEqn.H();
    surfaceScalarField phid
    (
        "phid",
        psi
       *(
            (fvc::interpolate(U) & mesh.Sf())
         // + fvc::ddtPhiCorr(rAU, rho, U, phi)
        )
    );
```

Create the new solvers (mySonicLiquidFoam)

```
            fvScalarMatrix pEqn
            (
                fvm::ddt(psi,p)
              + fvc::div(phi)
              //+ fvm::div(phid, p)
              - fvm::laplacian(rAU, p)
            );

            pEqn.solve();

            phi += pEqn.flux();

#include "continuityErrs.H"

            U -= rAU*fvc::grad(p);
            U.correctBoundaryConditions();
        }
        rho = 1 + psi*p;

    iter=iter+1.0;

     sumAcousticForce=sumAcousticForce+fvc::div(phi, U)+(rho-1)*fvc::ddt(U);

     meanAcousticForce=sumAcousticForce/iter;
```

creatFileds.H

Add the following lines into `creatFileds.H` for myBuoyantBoussinesqSimpleFoam solver.

```
volVectorField meanAcousticForce
(
    IOobject
    (
        "meanAcousticForce",
        runTime.timeName(),
        mesh,
       IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);
```

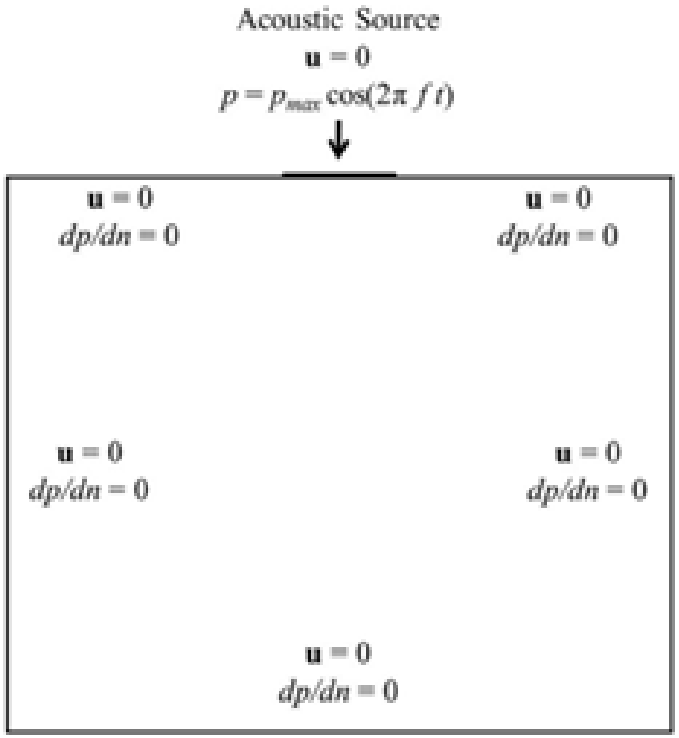
To import the driving momentum source into U equation, we should add `meanAcousticForce` at the last line of UEqn.

Cases

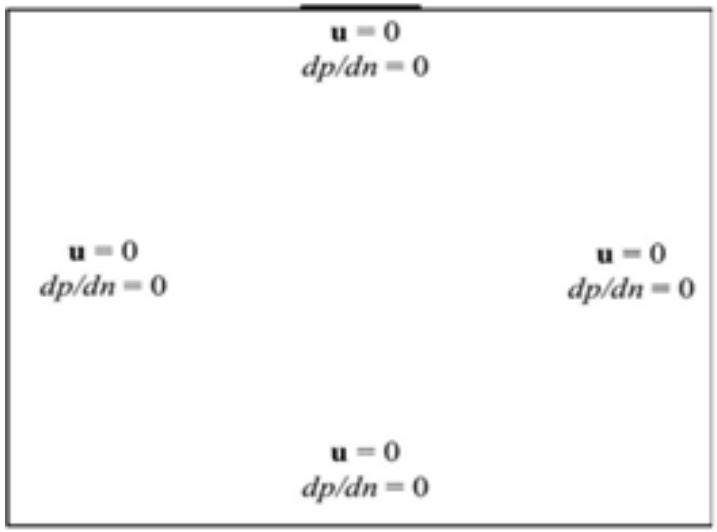
Download all files from course homepage and copy them into `FOAM_RUN` directory.
you can see 2 directories into cavity.
cavityFirstOrder : the case for testing mySonicLiquidFoam
cavitySecondOrder : the case for testing myBuoyantBoussinesqSimpleFoam

Boundary conditions

This figure shows boundary and initial conditions for first and second order equations.



First order equations

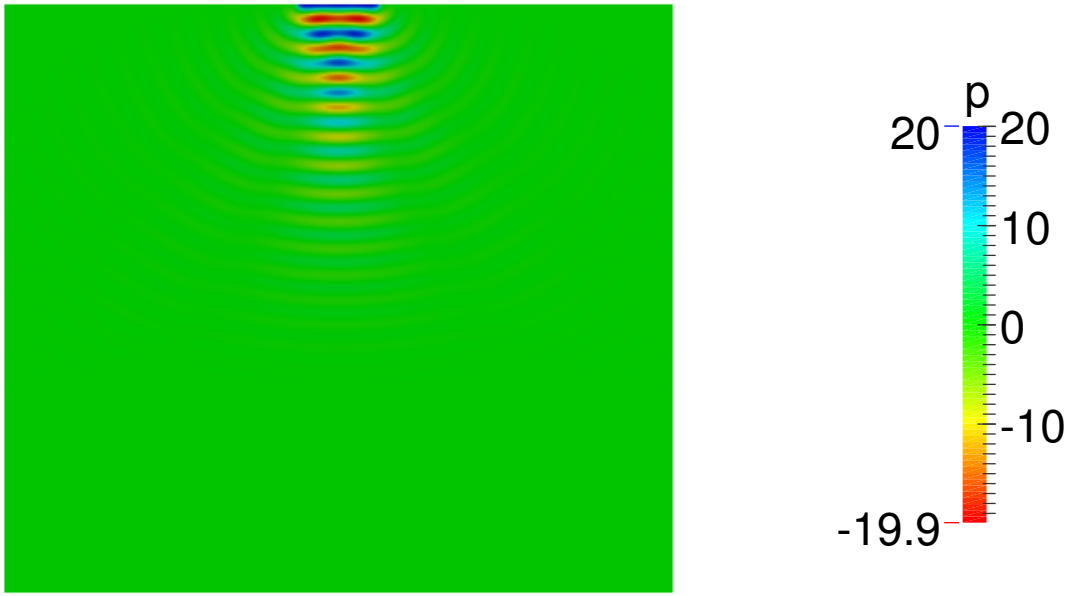


Second order equations

Property of fluid and simulation parameter

Parameter	Value
Sound Speed	1440 m/s
Frequency	10 MHz
Pressure Amplitude	20 kPa
Density	998 kg/m^3
Kinematic viscosity	89 e-08 m^2/s
Time Step–$1^{st} order$	$10^{-9} s$
Simulation time–$1^{st} order$	O$\left(10^{-5}\right)$ s
psi	4.44 e-07 s^2/m^2

For running cavityFirstOrder, execute the file named AllrunFirstOrderCavity. this file is a script file. Below figure shows pressure field into domain. As you see, acoustic waves propagate from the source and move along vertical direction.



For running cavitySecondOrder, execute the file named AllrunSecondOrderCavity. this file is a script file. below figure shows velocity field into domain. as shown, there are two large steady vortex which is created due to propagation of acoustic waves.