Description of the overset mesh approach in ESI version of OpenFOAM

Petra Tisovska

Technical University of Liberec,
Czech Republic

2019-11-27

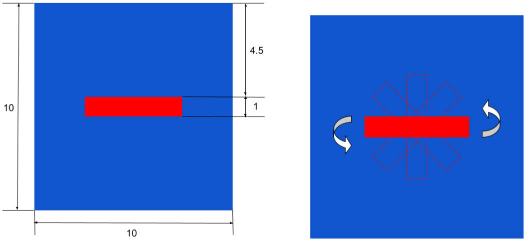
Table of Contents

Tutorial simpleRotor

In standard instalation the tutorial can be find in

`$FOAM_TUTORIALS/incompressible/overPimpleDyMFoam/simpleRotor`

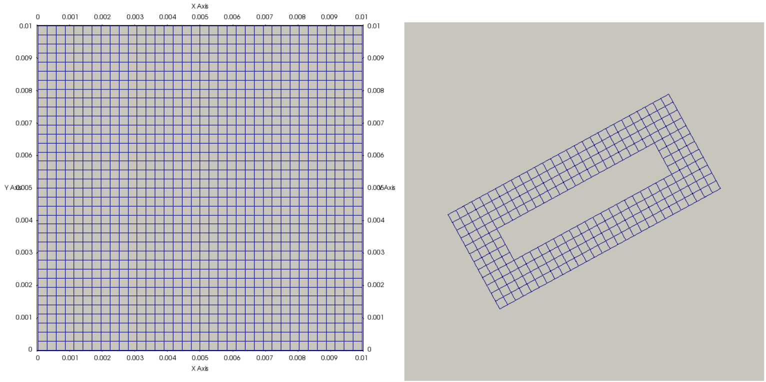
The rotor is rotating in domain and creates fluid motion.



(a) Domain dimensions (b) Rotor motion

Figure: Schematic view of simpleRotor tutorial case

The blockMesh creates two overlapping meshes



(a) Background mesh (b) Overset mesh

Figure: Meshes for simple rotor tutorial case

The utility topoSet is used for creating the hole and the zones for computation.

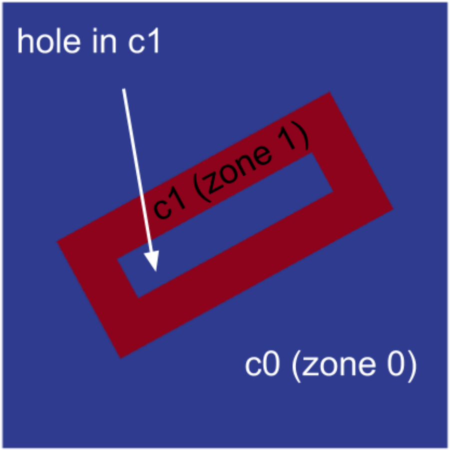


Figure: The zones of mesh distinguished by color

Theory

- disconnected meshes
- static and dynamic meshes
- mesh motion and interactions

Simplified mesh generation
: by spliting the domain into more parts more suitable elements can be used

Local refinement
: refined part of mesh around body – two block generation only

Moving parts
: simpleRotor tutorial, where mesh moves independently

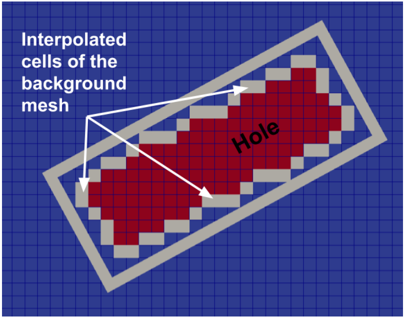
Optimization
: moving various parts of mesh without the need of remeshing the domain

Each cell of the background and overset mesh is marked as one of the following types

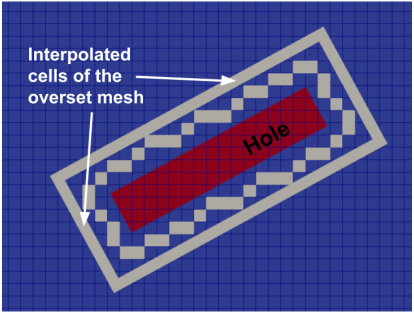
Calculated
: For this type of cells, the equations are solved.

Interpolated
: The values in these cells are computed by interpolation from the nearest elements of the second domain (background elements for the overset elements, and vice versa).

Holes
: No solution is computed here.



(a) Hole (red cells) and interpolated cells (white cells) defined for background mesh



(b) Hole (red cells) and interpolated cells (white cells) defined for overset mesh

Figure: Display of interpolation cells (white) for simpleRotor tutorial

Interpolation schemes

Interpolation schemes implemented in the overset library

- cellVolumeWeight
- inverseDistance
- leastSquares
- trackingInverseDistance

Inverse distance

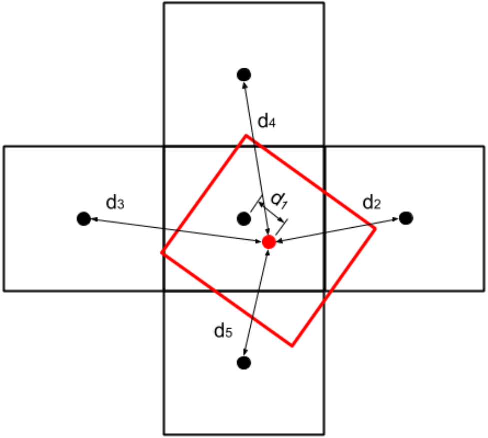


Figure: Distance of cell centers for determination of the weights. The red cell is acceptor and the black ones are donors.

To determine the weights the sum S of of their inverse distances is needed

$$S = \sum_{i}^{n} \frac{1}{|d_i|}, \tag{1}$$

where n is number of donors and d_i is the distance between their centers and center of aceptor. Then the weights w_i are

$$w_i = \frac{\frac{1}{|d_i|}}{S}. \tag{2}$$

Finally, the interpolated value is obtained for one cell by sum over all neighbours

$$\varphi = \sum_{i}^{n} w_i \varphi_i, \tag{3}$$

where φ is the field that is interpolated, for example the pressure.

Neighbors inverse distance

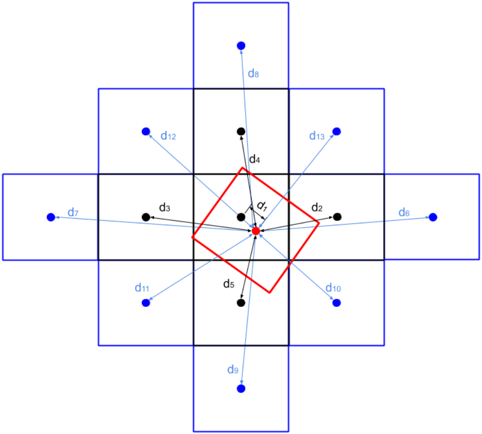


Figure: Distance of cell centers for determination of the weights. The red cell is acceptor, the black ones are original donors and the blue cells are newly added donors.

```
OFv1906
ufoam
mkdir applications  %DONT DO IF THE FOLDER ALREADY EXISTS
cd applications
cp -r $FOAM_SRC/overset/cellCellStencil/leastSquares nbInverseDistance
cd nbInverseDistance
mkdir Make
```

options

In dictionary Make create options file

```
gedit Make/options
```

with contents

```
EXE_INC = \
    -I$(LIB_SRC)/finiteVolume/lnInclude \
    -I$(LIB_SRC)/overset/lnInclude

LIB_LIBS = \
    -lfiniteVolume \
    -loverset
```

files

In dictionary Make create files

```
gedit Make/files
```

with contents

```
nbInverseDistance.C

LIB = $(FOAM_USER_LIBBIN)/libnbInverseDistance
```

Rename files

The files leastSquares.C and leastSquares.H should be renamed.

```
mv leastSquaresCellCellStencil.C nbInverseDistance.C
mv leastSquaresCellCellStencil.H nbInverseDistance.H
```

Structure

The structure of dictionary logarithmInverseDistance is obtained by command `tree`

```
nbInverseDistance
   |--- nbInverseDistance.C
   |--- nbInverseDistance.H
   |--- Make
         |--- files
         |--- options
```

Renaming inside the files

The word leastSquare has to be changed inside the files by executing

```
sed -i 's/leastSquaresCellCellStencil.H/nbInverseDistance.H/g' nbInverseDistance.C
sed -i 's/leastSquares/nbInverseDistance/g' nbInverseDistance.C
sed -i 's/leastSquares/nbInverseDistance/g' nbInverseDistance.H
```

Compile the code with `wmake`

The output should be checked for errors.

Adding the extension into code

Open

```
$FOAM_SRC/overset/cellCellStencil/inverseDistance/inverseDistanceCellCellStencil.C
```

and coppy function createStencil (line 1467 – 1650) and put it insted of the stencilweights function in nbInverseDistance.C.
In function createStencil localize

```
// Get neighbours (global cell and centre) of donorCells.
labelListList donorCellCells(mesh_.nCells());
pointListList donorCellCentres(mesh_.nCells());
globalCellCells
(
    globalCells,
    mesh_,
    isValidDonor,
    donorCells,
    donorCellCells,
    donorCellCentres
);
```

and add following code.

```
forAll(donorCells,cellI)
{
   //index of donors
   label someCell = donorCells[cellI];
   //neighbors for acceptor, finding their neighbors
   labelList neighborCells = donorCellCells[someCell];

   //new field for new neighbours
   labelListList nbDonorCellCells(mesh_.nCells());
   //new field for cell centers of new nb
   pointListList nbDonorCellCentres(mesh_.nCells());
   //finds neibours of "neighborCells" and fills following arrays with cells
   // and cell centers
   globalCellCells
   (
      globalCells,
      mesh_,
      isValidDonor,
      neighborCells,
      nbDonorCellCells,
      nbDonorCellCentres
   );
```

```
    for(label index = 1; index < neighborCells.size();index++)
    {
        label someNbCell = donorCellCells[someCell][index];
        forAll(nbDonorCellCells[someNbCell], k)
        {
            bool addNb = true;
            forAll(donorCellCells[someCell], j)
            {
                if(donorCellCells[someCell][j] ==  nbDonorCellCells[someNbCell][k])
                {
                    addNb = false;
                }
            }
            if (addNb)
            {
                donorCellCentres[someCell].append(nbDonorCellCentres[someNbCell][k]);
                donorCellCells[someCell].append(nbDonorCellCells[someNbCell][k]);
            }
        }
    }
}
```

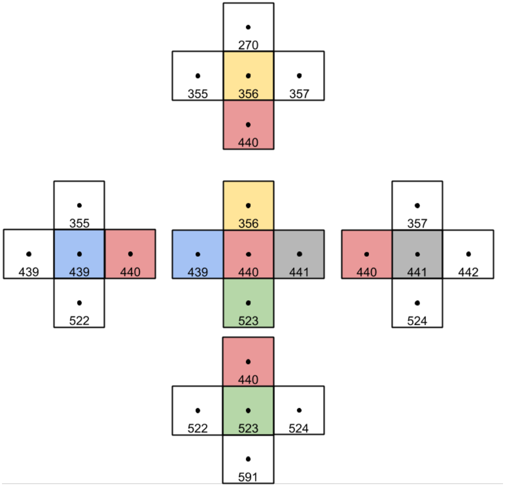


Figure: The graphical scheme of neighbors of original donors. Some of the overlapping ones are colored.

After the added code

```
// Determine the weights.
scalarListList donorWeights(mesh_.nCells());
forAll(donorCells, i)
```

should follow.
The function header should change from

```
void Foam::cellCellStencils::inverseDistance::createStencil
```

to

```
void Foam::cellCellStencils::nbInverseDistance::createStencil
```

The last modification has to be added into nbInverseDistance.H. Since the name of function had changed, the declaration has to be changed as well. Instead of

```
//- Calculate lsq weights for single acceptor
virtual void stencilWeights
(
     const point& sample,
     const pointList& donorCcs,
     scalarList& weights
) const;
```

The declaration from file inverseDistanceCellCellStencil.H must be added.

```
virtual void createStencil(const globalIndex&);
```

The library now have to be compiled by `wmake`.

Run the case

The tutorial simpleRotor is copied into run folder.

```
cp -r $FOAM_TUTORIALS/incompressible/overPimpleDyMFoam/simpleRotor/ $FOAM_RUN/nbSimpleRotor
cd $FOAM_RUN/nbSimpleRotor
```

The case has to be slightly modified.

```
sed -i 's/inverseDistance/nbInverseDistance/g' $FOAM_RUN/simpleRotor/system/fvSchemes
sed -i 's/"libfvMotionSolvers.so"/"libfvMotionSolvers.so" "libnbInverseDistance.so"/g' \
$FOAM_RUN/simpleRotor/system/controlDict
```

Run the case with `./Allrun`

Comparison

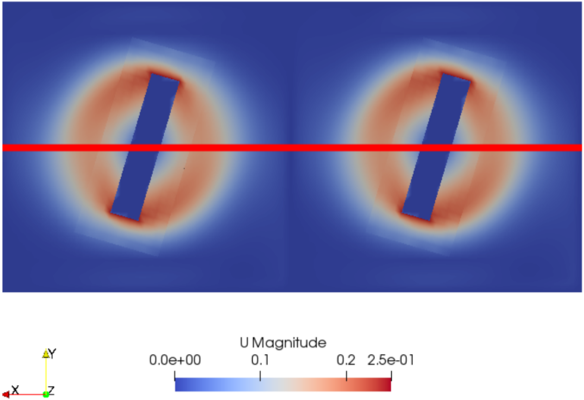


Figure: The contours for simpleRotor tutorial, left one is for usage of original the inverse distance interpolation scheme and the right one is for newly implemented nbInverseDistance. The red line is displayed for plotting purposes.

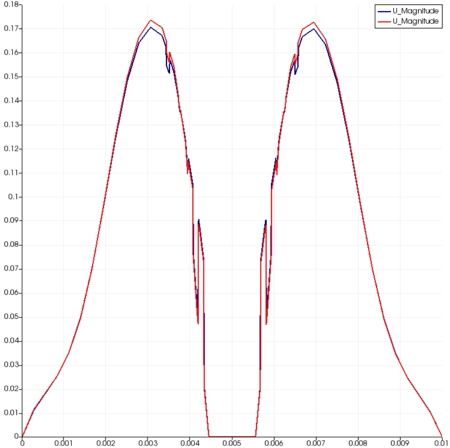


Figure: The magnitudes of velocity for original inverseDistance interpolation scheme (red) and for nbInverseDistance (blue) plotted over red line (the cases overlap for the plotting).

Thank you for your attention and cooperation!