Assignment 2: Mesh Interpolation and Mesh Deformation

- The assignment should be handed in at the latest Wedesday 20th March 2024 12:00 in order to receive full credit, otherwise a 2 point penalty is applied to the grade of the assignment.
- The assigment can be made in groups of **at most 3 students**.
- Please be sure to include the names and student numbers in your report.
- Please hand in an electronic version (also a scanned version of a (clearly) hand-written report is accepted) through Brightspace (only one submission per group is required).
- You do not have to make a full report (including summary, list of figures, list of symbols etc), but indicate clearly the question numbers when giving your answer / results / discussion of results.
- When an implementation is requested, please provide (only) the code snippets where you made your implementation.

Issue with Matlab versions

Starting from Matlab-R2013 there is an issue when using Matlab to compute the inverse of a matrix, which can result in erroneous interpolation results on the most refined meshes. Matlab version R2012b and before do not seem to have this issue. One way of circumventing the problem (tested for versions R2013a and R2013b) is to replace the direct computation of the inverse C and the subsequent right multiplication by the 'forward-slash' operator:

Instead of using

```
HsfRBF = [PHI P] * invC(:,1:Ns)
```

use

```
HsfRBF = [PHI P]/C
HsfRBF = HsfRBF(:,1:NS)
```

Introduction Interface Mesh Interpolation

In FSI computations it is required that pressure loads are transmitted from the fluid side of the fluid-structure interface to the structural nodes on that interface. Once the motion of the structure has been determined, the motion of the fluid mesh points on the interface has to be imposed. In FSI simulations it is usually not desirable to generate matching meshes at the fluid-structure interface, because different solvers may take care of the different physical domains. In addition, also the flow generally requires a much finer mesh than the structure. This means that the discrete interface between the domains may not only be non-conforming, but there can also be gaps and/or overlaps between the meshes. The exchange of data over the discrete interface becomes then far from trivial. In Figure 1 a 2D example of a non-matching discrete interface between a flow and structure domain is shown. When the meshes are non-matching, an interpolation/projection step has to be carried out to enable transfer of information between the two domains. As mentioned in the lectures different methods exist to transfer data between non-matching meshes, such as nearest neighbor interpolation, projection methods and methods based on interpolation by splines.

In this assignment we investigate the difference in accuracy between conservative and consistent coupling approaches. This is done for a simple analytic problem, using the Nearest Neighbor and Radial Basis Function interpolation schemes.

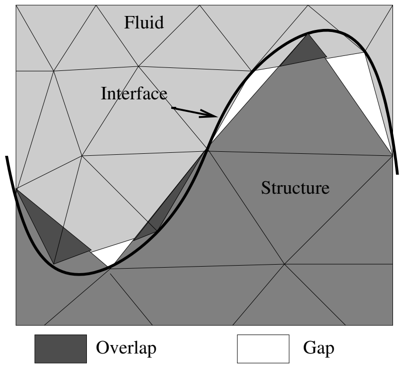

Figure 1: Non-matching meshes in 2D.

1 Conservative and consistent coupling approach

The interpolation of displacement from the structure mesh $\mathbf{U}_s$ to the fluid mesh $\mathbf{U}_f$ and the interpolation of pressures from the fluid mesh $\mathbf{P}_f$ to the structure mesh $\mathbf{P}_s$ can be formulated as

$$\mathbf{U}_f = H_{sf}\mathbf{U}_s \tag{1a}$$
$$\mathbf{P}_s = H_{fs}\mathbf{P}_f, \tag{1b}$$

wherein H_{fs} and H_{sf} the interpolation matrices. For a consistent coupling approach, the interpolation matrices can be generated separately, using any interpolation method that satisfies the criterion that the rowsums of H are one. The conservative coupling approach comes from the conservation of work over the interface. The matrices M_{ff} and M_{ss} the define the surface integral of the pressure to obtain the nodal forces on the interface. For the conservative approach it is shown in the lectures and in the lecture notes that in order for the work over the interface to be equal on both sides of the fluid-structure interface, we should satisfy

$$H_{fs} = \left[M_{ff}H_{sf}M_{ss}^{-1}\right]^T. \tag{2}$$

Therefore, the interpolation of fluid pressures to the structure mesh is defined by the interpolation chosen for interpolating the structure displacements to the fluid mesh. The interpolation methods that we will consider closely in this exercise are the nearest neighbor and radial basis function method.

Nearest Neighbor

Nearest neighbor interpolation (NN) is a very simple method to transfer data from mesh A to mesh B. A search algorithm determines the point x_A in mesh A that is closest to a given point x_B in mesh B. The variable in x_B is then assigned the same value as in x_A. In this way the transformation matrix H_{AB} becomes a Boolean matrix, with a single one in each row which implies that the transformation is indeed consistent when the consistent approach is used.

Radial basis function interpolation (RBFI)

The quantity to be transferred from mesh A to mesh B is approximated by a global interpolation function which is a sum of basis functions

$$\mathbf{w}(\mathbf{x}) = \sum_{j=1}^{n_A} \gamma_j \phi(||\mathbf{x} - \mathbf{x}_{A_j}||) + q(\mathbf{x}), \qquad \mathbf{w} = \{\mathbf{u}, p\mathbf{n}\}, \tag{3}$$

where $\mathbf{x}_{A_j}$ are the centres in which the values are known, in this case the nodes at the interface of mesh A, q a polynomial, and ϕ a given radial basis function with respect to the Euclidean distance $||\mathbf{x}||$. The coefficients γ_j and the polynomial q are determined by the interpolation conditions

$$\mathbf{w}(\mathbf{x}_{A_j}) = \mathbf{W}_{A_j}, \tag{4}$$

with $\mathbf{W}_A$ containing the discrete values of $\mathbf{w}$ at the interface of mesh A, and the additional requirements

$$\sum_{j=1}^{n_A} \gamma_j q(\mathbf{x}_{A_j}) = 0. \tag{5}$$

The radial basis function that we will use is the C^2 radial basis function defined as

$$\phi(||\bar{\mathbf{x}}||) = \begin{cases} (1 - ||\bar{\mathbf{x}}||/r)^4 \, (4||\bar{\mathbf{x}}||/r + 1) & ||\bar{\mathbf{x}}|| \leq r, \\ 0 & ||\bar{\mathbf{x}}|| > r, \end{cases} \tag{6}$$

where the support radius $r = 1$ is chosen and the polynomial is defined by

$$q(\mathbf{x}) = c_0 + c_x x + c_y y. \tag{7}$$

The requirements in (4) and (5) result in a linear system of equations that defines the RBF interpolation function coefficients γ and c, e.g.

$$\begin{bmatrix} \mathbf{W}_A \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{\Phi} & \mathbf{L} \\ \mathbf{L}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \gamma \\ \mathbf{c} \end{bmatrix}, \tag{8}$$

wherein the matrix $\mathbf{\Phi}$ contains the evaluation of the radial basis functions $\Phi_{ij} = \phi(||\mathbf{x}_{A_i} - \mathbf{x}_{A_j}||)$, $\mathbf{L}$ is the matrix for the linear polynomial $\mathbf{L}_i = (1 \; x_i \; y_i)$. Once the system is solved and the coefficients γ and $\mathbf{c}$ are obtained, the values in mesh B $\mathbf{W}_B$ can be computed by evaluating (3) for mesh point locations $\mathbf{x}_B$.

2 Analytic problem

We consider an interface defined in two dimensions of a cosine shape:

$$\Gamma : y(x) = \cos(2\pi x), \quad 0 \leq x \leq 1. \tag{9}$$

The interface is discretized by linear approximation between discrete points on a 'fluid' and 'structure' mesh, each having N_f and N_s points distributed evenly in x. Therefore, for each discrete point i, the mesh has the discrete coordinate (x_i, y_i), where $y_i = cos(2\pi x_i)$. We define two exact functions to represent a displacement and pressure on the interface Γ

$$d(x) = \sin(2\pi x), \text{ on } \Gamma, \tag{10}$$
$$p(x) = \sin(2\pi x), \text{ on } \Gamma. \tag{11}$$

The displacement is sampled in the structure points

$$\mathbf{d}_s = \begin{pmatrix} d(x_{s,1}) \\ d(x_{s,2}) \\ \vdots \\ d(x_{s,N_s}) \end{pmatrix}, \tag{12}$$

and the pressure is sampled in the fluid points

$$\mathbf{d}_f = \begin{pmatrix} p(x_{f,1}) \\ p(x_{f,2}) \\ \vdots \\ p(x_{f,N_f}) \end{pmatrix}. \tag{13}$$

Matlab code

For each of the interpolation methods (NN and RBF), two interpolation matrices need to be constructed, which will be implemented into the Matlab script `Analyt.m` which can be found in the folder `matlab\Analytic\`:

- Interpolation from the structure mesh to the fluid mesh H_{sf} so that

$$\mathbf{d}_f = H_{sf}\mathbf{d}_s, \tag{14}$$

  where the matrices in `Analyt.m` are named `HsfNN` and `HsfRBF`.

- Interpolation from the fluid mesh to the structure mesh using the *consistent* approach so that

$$\mathbf{p}_s = H_{fs}\mathbf{p}_f, \tag{15}$$

  where the matrices in `Analyt.m` are named `HfsNN` and `HfsRBF`.

Furthermore, when the pressures are interpolated in a conservative way, the resulting interpolation follows the formulation in Eq. (2), and therefore this matrix does not need to be constructed explicitly.

Exercise 1.1 In `Analyt.m` implement the NN and RBF interpolation matrices and the conservative interpolation. Check the rowsum of the matrices `HsfNN`, `HsfRBF`, `HfsNN` and `HfsRBF` - are the matrices consistent?

For the analysis, the error with respect to the exact solution is determined. This error is approximated by taking a large number of samples (M) at the locations $\boldsymbol{\chi}$, e.g. for the exact displacement

$$\tilde{\mathbf{d}}_{ex} = \begin{pmatrix} d(\chi_1) \\ d(\chi_2) \\ \vdots \\ d(\chi_M) \end{pmatrix}, \tag{16}$$

and using a linear interpolation between the fluid point locations to simulate the use of linear basis functions to approximate the displacements at the same locations $\boldsymbol{\chi}$ to obtain a vector $\tilde{\mathbf{d}}_f$ with the same M number of displacement data. The root-mean-square (RMS) error between the exact displacement and the interpolated data is

$$\epsilon_{RMS} = \sqrt{\frac{\sum_{i=1}^{M}(\tilde{d}_{ex,i} - \tilde{d}_{f,i})^2}{M}}. \tag{17}$$

Finally the error in the conservation of work over the interface is determined

$$\epsilon_{\partial W} = |\partial W_f - \partial W_s|. \tag{18}$$

Exercise 1.2 Implement the error in computational work in `Analyt.m` in terms of the given and interpolated displacement and pressure fields for the conservative and consistent interpolations with NN and RBF, e.g.
`err_dW_NN = abs( (Mf*Pf)' * Df_NN - (Ms*Ps_NN)' * Ds )`
How large should the error be for the conservative methods?

For the analysis a grid refinement study is performed by using the script `loop.m`, which can be found in `matlab\Analytic\`. The number of points for the fluid mesh and structure mesh are determined by $N_s = 5 \cdot 2^{i-1} + 1$ and $N_f = 7 \cdot 2^{i-1} + 1$, for $i = 1 \ldots 8$, so that the ratio between the number of intervals in which the structure and fluid domains are divided is always 5/7. For each of those settings `Analyt.m` is run to compute the errors of the interpolated displacement and pressure with respect to the exact solution and the error in conservation of work over the interface.

Exercise 1.3 Run the `loop.m` script and insert the figures generated in the report. Discuss the results in the figures.
When comparing e.g. the displacement received by the fluid to the exact solution, the error contains two different contributions - which two? (Hint: if you select matching meshes, e.g. $N_f = N_s$, there is no interpolation required - is there still an error with respect to the exact solution?)
Is the error in work over the interface as expected (Exercise 1.2)?

3 Mesh motion for two-dimensional mesh

In this exercise, the Radial Basis Function interpolation is used for mesh deformation. In the first Matlab exercise for the DGCL, a two-dimensional mesh with 9 cells was used (Fig. 2). This

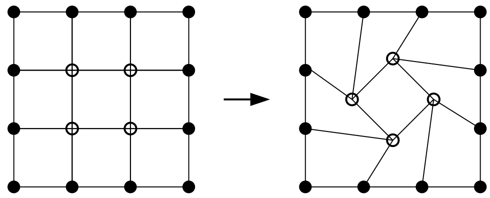

Figure 2: Two dimensional mesh.

meant that when the displacement on the outer boundary was set equal to zero and the four internal points were rotated, all mesh points were on a boundary so that no internal mesh nodes were present. We will use the same program, but this time with $N = M = 11$ cells, so that also points in the iterior of the mesh exist (that are not on the boundary).

Matlab code

The matlab code that is available for this exercise can be found in `matlab\MeshDef\`. The files that are important in this exercise are:

`MeshDef2D.m` This is the main program; it takes the following steps:

- determine node locations,
- create cells by assigning 4 nodes for each cell,
- for each time step:
  1. compute displacements of nodes,
  2. move the mesh (update node locations),
  3. compute the mesh quality,

`moveMesh.m` Updates the mesh nodes to the new location.

`plotMesh.m` Displays the mesh and creates colors depending on skewness of the cells.

Mesh interpolation

When we run the program with `NdtP=20, Np=0.25, N=M=11, maxAngle=45`, you will find that the center cell rotates, but that all internal nodes do not move. We are going to implement a mesh motion algorithm, based on a radial basis funtion interpolation. First of all, the function determines the number of internal and boundary points and splits the vector of all the nodes into a vector containing only the boundary node coordinates `Xb` and boundary displacements `Db` and a vector containing the internal node coordinates `Xi` and the yet unknown vector of internal displacements `Di`.

Radial basis function interpolation

The displacements of the domain boundary `Db`, located at `Xb` have to be interpolated to dislacements `Di` at the internal node locations `Xi`. For the interpolation, a displacement field is computed using radial basis functions. The displacement at any location $\mathbf{x}$ is approximated by a sum of basis functions

$$s(\mathbf{x}) = \sum_{j=1}^{N_b} \gamma_j \phi(||\mathbf{x} - \mathbf{x}_{b_j}||) + q(\mathbf{x}), \tag{19}$$

where N_b the number of boundary points, $\mathbf{x}_{b_j}$ are the centres on the domain boundary in which the values for the displacement $\mathbf{d} = (d_x d_y)^T$ are known, q a polynomial, and ϕ a given radial basis function with respect to the Euclidean distance $||\mathbf{x}||$. The coefficients γ_j and the polynomial q are determined by the interpolation conditions

$$\mathbf{s}(\mathbf{X}_{b_j}) = \mathbf{D}_{b_j}, \tag{20}$$

with $\mathbf{D}_b$ a vector containing the discrete values of the displacement at the mesh boundary, $\mathbf{X}_b$ a vector containing the coordinates of the boundary points and the additional requirements

$$\sum_{j=1}^{N_b} \gamma_j q(\mathbf{X}_{b_j}) = 0. \tag{21}$$

The radial basis function that we will use is the Thin-Plate-Spline (TPS) radial basis function defined as

$$\phi(||\mathbf{x}||) = ||\mathbf{x}||^2 \ln ||\mathbf{x}||, \tag{22}$$

and the polynomial is defined by

$$q(\mathbf{x}) = c_0 + c_x x + c_y y. \tag{23}$$

A separate interpolation function is made for the displacements in x and y direction. The requirements in (20) and (21) result in a linear system of equations that defines the RBF interpolation function coefficients γ and c, e.g. for the displacements in x:

$$\begin{bmatrix} \mathbf{D}_x \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{\Phi} & \mathbf{L} \\ \mathbf{L}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \gamma_x \\ \mathbf{c}_x \end{bmatrix}, \tag{24}$$

wherein the matrix $\mathbf{\Phi}$ contains the evaluation of the radial basis functions $\Phi_{ij} = \phi(||\mathbf{x}_i - \mathbf{x}_j||)$, $\mathbf{L}$ is the matrix for the linear polynomial $\mathbf{L}_i = (1 \; x_i \; y_i)$. Once the system is solved and the coefficients γ_x and $\mathbf{c}_x$ are obtained, the x displacements of the internal nodes can be computed by evaluating (19) for all internal mesh point locations $\mathbf{X}_i$. Note that for the dislacements in y, essentially the same system of equations has to be solved as $\mathbf{\Phi}$ and $\mathbf{L}$ are the same.
The implementation is performed in several steps in `moveMesh.m`:

1. Setting up the matrix `rbfMat` for the constraints, such that `Db = rbfMat*P`, with `P` a vector containing the coefficients γ and c_0, c_x, c_y of the linear polynomial.

2. Compute the coefficients `P`

3. Evaluate the interpolation function for each internal node.

Note that the interpolation is broken down into two steps: 1) determine the RBF coefficients and 2) evaluate the interpolation function at each internal grid point.

Exercise 2.1 Implement the mesh deformation steps in `moveMesh.m` for the Thin-Plate-Spline radial basis function $\phi = ||x||^2 \ln ||x||$. This time when you run the code with `NdtP=20, Np=0.25, N=M=11, maxAngle=45`, you will find that the mesh deforms smoothly. At the end of the simulation the mesh orthogonality is plotted as is a plot of the mesh orthogonality in time.

Absolute or relative displacements The displacement of the mesh nodes can be computed either from the intitial mesh (absolute displacements) or from the mesh at t_n (relative displacements). Which method is used can be selected by setting `absdisp = 1` for absolute displacements or `absdisp = 0` for relative displacements. Close all figures and use the following settings: `NdtP=20, N=M=11, maxAngle=90`. Run computations using the settings from Table 1.

Table 1: Simulation time, displacement mode and format settings.

case	`Np`	`absdisp`	`fmt`
1	0.25	1	'-b'
2	0.25	0	'-r'
3	2.0	1	'-g'
4	2.0	0	'-c'

Exercise 2.2 Discuss which method performs best when comparing cases 1 and 2 are compared. Explain the differences in results for computations 3 and 4; what would happen when the simulation time is increased?

Number of steps per period When using the relative interpolation, the number of steps performed per period is of importance. To investigate, we use the settings: `Np=1, N=M=11, maxAngle=90`. Close all figures and run computations using the settings from Table 2.

Table 2: Number of steps per period, displacement mode and format settings.

case	`NdtP`	`absdisp`	`fmt`
1	40	1	'-b'
2	20	1	'–b'
3	10	1	'ob'
4	5	1	'xb'
5	40	0	'-r'
6	20	0	'-g'
7	10	0	'-c'
8	5	0	'-m'

Exercise 2.3 Discuss the results for the mesh quality for the absolute displacement method and for the relative displacement method with respect to the number of time steps taken per period.

Exercise 2.4 What are your overall conclusions comparing the absolute and relative displacement methods (i.e. which method would you prefer what kind of application / test case characteristic)?

BONUS Exercise 2.5 (optional) *This exercise is optional: you can receive 1 point bonus.* So far we only used the TPS RBF for the mesh deformation, you could also use a compact supported function (like the C^2 function of Wendland). Implement a radial basis function mesh deformation based on the compact supported radial basis function

$$\phi(||\mathbf{x}||) = \begin{cases} (1 - ||\mathbf{x}||/r)^4(4||\mathbf{x}||/r + 1) & : \quad ||\mathbf{x}|| \leq r \\ 0 & : \quad ||\mathbf{x}|| > r \end{cases}, \tag{25}$$

wherein r the support-radius. Perform an investigation into the influence of the support radius on the mesh quality and discuss the results.