MOX–Report No. 40/2012

Radial basis functions for inter-grid interpolation and mesh motion in FSI problems

Lombardi, M.; Parolini, N.; Quarteroni, A.

MOX, Dipartimento di Matematica "F. Brioschi"
Politecnico di Milano, Via Bonardi 9 - 20133 Milano (Italy)

mox@mate.polimi.it http://mox.polimi.it

Radial basis functions for inter-grid interpolation and mesh motion in FSI problems

M. Lombardi$^{(1)}$, N. Parolini$^{(2)}$, A. Quarteroni$^{(1,2)}$

September 20, 2012

$^{(1)}$ CMCS - Institut d'Analyse et Calcul Scientifique
Ecole Polytechnique Fédérale de Lausanne
Station 8, CH-1015, Lausanne, Switzerland

$^{(22}$ MOX– Modellistica e Calcolo Scientifico
Dipartimento di Matematica "F. Brioschi"
Politecnico di Milano
via Bonardi 9, 20133 Milano, Italy

Abstract

When addressing multi-domain/multi-physics problems, the correct exchange of mathematical information at subdomain interfaces is crucial. In this paper, such transfer is analyzed in the particular framework of a fluid-structure interaction (FSI) problem. A genuine FEM-FEM formulation is considered firstly, followed by a mixed FVM-FEM formulation. In both cases, we focus on two critical issues: how to interpolate numerical quantities at the interface, and how to achieve the property of conservation of energy transfer. In the second part of this work, we analyze the use of Radial Basis Functions (RBF) to handle both mesh motion and interpolation of numerical variables over non-matching interface grids. Different kinds of radial basis functions are considered and numerical tests comparing their performances in terms of accuracy and stability are presented and discussed.

1 Introduction

The simulation of multi-physics/multi-domains problems has become common practice in many different fields. These problems feature a strong interaction between one domain/physics and the others, thus it becomes fundamental to exchange information consistently and accurately. Such coupling is achieved by ensuring that suitable physical variables are continuous across the interface separating the sub-domains. In this paper we will focus on fluid-structure interaction (FSI) problems even if our approach can be extended to the more general framework of multi-physics/multi-domains problems.

In the FSI framework, a classical way to couple the fluid and structure domains consists in imposing the continuity of stress and displacement (or, equivalently, velocity) at the interface. Furthermore, it has been shown in literature that in order to ensure stability of the coupling algorithm, it is important to ensure the correct energy transfer at the interface. We first consider the case where both fluid and structure problems are solved via the same discretization technique, the finite element method (FEM). The analysis is performed for both the simple case where the spatial discretization is conforming, i.e. the interface nodes are coincident, and in the more general case when non-conforming meshes are used. Afterwards, the same analysis is extended to a different framework where a mixed finite volume-finite element (FVM-FEM) spatial discretization is considered.

The second part of this paper focuses more specifically on the use of the Radial Basis Functions (RBF) technique as interpolatory procedure. Different basis functions are considered and the trade-off between accuracy and stability is investigated. A battery of numerical tests on different geometries and data are carried out to analyze the convergence properties of the different kind of radial bases proposed in the literature and the influence of the radial basis shape factor.

In FSI problems based on the Arbitrary Lagrangian-Eulerian approach [11, 14, 24], a critical step is represented by the successful mesh motion due to the FSI interface deformation. In this paper, the same RBF tool adopted for the interface data interpolation is used also to deform the volume mesh, as done for example in [9]. In this work, this approach is considered in order to analyze, on realistic deformation fields, the behavior of the different radial basis functions in terms of mesh quality. We also discuss some specific issues related to the implementation of the RBF method on parallel architectures.

The numerical models presented in this paper have been recently used for wind-sail FSI simulations [20]; however, the same approach can be applied in the context of other FSI problems as well.

The paper is organized as follows: after recalling the formulation for a general fluid-structure interaction problem and the conditions imposed at the interface (section 2), in section 3 we analyze the case of FEM-FEM discretization framework, on both conforming and non-conforming grids. In section 4 the particular case of FVM-FEM discretization is discussed and the considerations presented in the previous section are modified according to this new framework. The attention is then focused on the RBF interpolation technique, presented in section 5. Numerical results assessing the performance of the method in the context of FSI problems in terms of accuracy and conservation properties are given in section 6, as well as its application when used also for moving the computational domain. Finally, in section 7 some implementation issues on parallel architectures are discussed, followed by a section on conclusions.

2 Fluid-Structure interaction framework

A fluid-structure interaction (FSI) problem is defined by the coupled dynamics of a moving or deformable structure interacting with a surrounding fluid. In this work, we will assume that the flow field is governed by the time-dependent incompressible Navier-Stokes equations, whose governing variables are the velocity and pressure pair $(\mathbf{u}_F, p_F)$, and the structure is modelled by any elasto-dynamic equation whose unknown is the structural displacement $\mathbf{d}_S$.

In the classical formulation of fluid-structure interaction problems, the fluid and structure equations are coupled by a kinematic condition

$$\mathbf{u}_F = \mathbf{u}_S \quad \text{on } \Gamma, \tag{1}$$

which states that the fluid velocity $\mathbf{u}_F$ and the structure velocity $\mathbf{u}_S = \dot{\mathbf{d}}_S$ are continuous at the interface Γ, plus a dynamic condition

$$\boldsymbol{\sigma}_F \mathbf{n}_F = \boldsymbol{\sigma}_S \mathbf{n}_S \quad \text{on } \Gamma, \tag{2}$$

which ensures the balance of fluid and structure normal stresses at the interface, with $\boldsymbol{\sigma}_{F,S}$ and $\mathbf{n}_{F,S}$ denoting the stress tensors and the outward normal unit vector, respectively.

After space and time discretizations, although monolithic FSI approaches [17, 13, 8, 7] can be adopted, with the fluid and structure equations solved simultaneously (together with the coupling conditions), in many engineering applications partitioned FSI schemes are most often preferred [22, 12, 21, 6]. The latter are based on suitable domain decomposition paradigms according to which the solution of fluid and structure problems are decoupled from one another thanks to suitable splitting of the interface conditions (1) and (2). In this way, existing codes previously developed (and optimized) for the solution of each separate field only need to be coupled through suitable interface conditions: informations between codes are exchanged only at the interface. In the classical Dirichlet-Neumann domain decomposition coupling, which is considered in this work, the displacement field at the interface is transferred from the structure to the fluid while the stress field from the fluid to the structure through condition (2), see [23].

Furthermore, it has been shown [22] that the stability of the FSI coupling depends on whether or not the global conservation of energy is fulfilled across the interface, more precisely, the work W_F done by the fluid forces on the interface equals the work W_S done by the structural forces.

We consider the general case where the fluid and structural grids are non-matching at the interface, so that suitable inter-grid interpolation (or projection) is required when transferring data between the two domains. Not only the interface can be non-matching (Figure 1(left)) but also the fluid and structure solvers may be derived from different spatial discretization techniques with different arrangements of the degrees-of-freedom over the grids (Figure 1(right)).

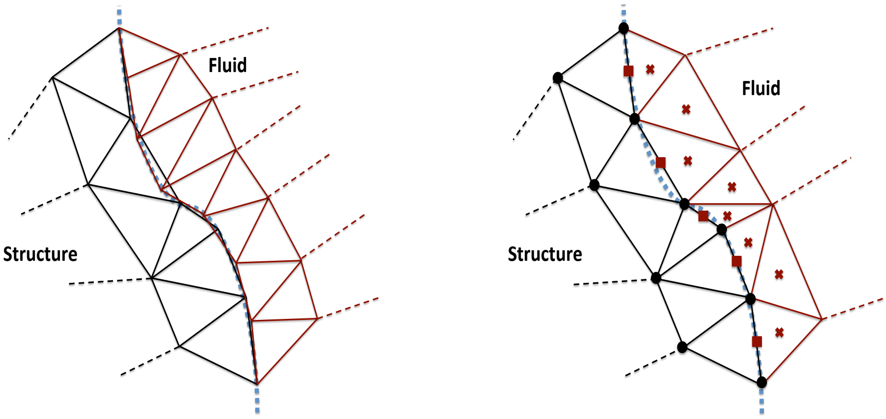


Figure 1: (Left) Non-matching grids. (Right) Different spatial discretizations: P1 finite elements on the left, cell centered finite volumes on the right. Symbol explanation is as follows: • FE nodes, × FV cell centers and ■ FV face centers on the interface.

Finally, the flow problem is defined on a domain which deforms according to the structure motion. The problem is here formulated in an Arbitrary Lagrangian-Eulerian framework, which implies that a correction term related to the rate of mesh deformation should be added to the convective term of the momentum equation. The deformation of the fluid mesh is obtained imposing the structural displacement at the interface. The grid displacement inside the fluid domain can be obtained in many different ways, such as, for example, the harmonic extension or the spring-analogy method. In this work, the application of the RBF technique for the deformation of the volume mesh will be considered.

3 General formulation for Dirichlet-Neumann interface coupling

The FSI problem just introduced can be recast in a more general abstract framework which can be shared by different classes of multi-physics (or multi-domain) problems. We will follow the general abstract approach of [23].

Given two disjoints domains Ω_1 and Ω_2 with a common interface $\Gamma = \overline{\Omega}_1 \cap \overline{\Omega}_2$, we consider two differential problems defined, in strong form, by

$$L_1 u_1 = f_1, \quad \text{in } \Omega_1, \tag{3}$$

$$L_2 u_2 = f_2, \quad \text{in } \Omega_2. \tag{4}$$

For simplicity of exposition we are assuming that L_1 and L_2 are two linear second order operators and that (3) and (4) are two scalar equations. At the interface Γ, the two problems (3) and (4) are coupled by the two following conditions:

$$u_1 = u_2, \quad \text{on } \Gamma, \tag{5}$$

$$\psi_1(u_1, \mathbf{n}) = \psi_2(u_2, \mathbf{n}), \quad \text{on } \Gamma, \tag{6}$$

where the former is a Dirichlet-type interface condition and the latter a Neumann-type interface condition, with $\mathbf{n}$ denoting the unit normal on Γ pointing from Ω_1 to Ω_2. In the FSI problem, the Dirichlet condition corresponds to the kinematic condition (1) and the Neumann condition to the continuity of the normal stress (2). For the sake of simplicity, on the other boundaries of domains Ω_1 and Ω_2, ($\partial\Omega_1 \backslash \Gamma$ and $\partial\Omega_2 \backslash \Gamma$, respectively), we consider homogeneous Dirichlet boundary conditions.

The weak formulation of problems (3) and (4) reads

Problem 1: *Find $u_1 \in V_1(\Omega_1)$ such that:*

$$a_1(u_1, v_1)_{\Omega_1} = (f_1, v_1)_{\Omega_1}, \quad \forall v_1 \in V_1(\Omega_1). \tag{7}$$

Problem 2: *Find $u_2 \in V_2(\Omega_2)$ such that:*

$$a_2(u_2, v_2)_{\Omega_2} = (f_2, v_2)_{\Omega_2}, \quad \forall v_2 \in V_2(\Omega_2), \tag{8}$$

where V_1 and V_2 are suitable functional spaces defined on the two domains Ω_1 and Ω_2, respectively, and $(\cdot, \cdot)_{\Omega_i}$ denotes the standard $L^2(\Omega_i)$ ($i = 1, 2$) scalar product.

We assume that the Neumann condition (6) is the natural condition associated to the weak form of the two problems, namely

$$(L_1 u_1, v_1)_{\Omega_1} = a_1(u_1, v_1) + \int_{\partial\Omega_1} \psi_1(u_1, \mathbf{n}) v_1, \;\; \forall v_1 \in V_1(\Omega_1), \tag{9}$$

$$(L_2 u_2, v_2)_{\Omega_2} = a_2(u_2, v_2) - \int_{\partial\Omega_2} \psi_2(u_2, \mathbf{n}) v_2, \;\; \forall v_2 \in V_2(\Omega_2). \tag{10}$$

Within this framework, the Neumann interface condition (6) is naturally imposed in a weak sense. Indeed, if we take as test function in equations (7) and (8), an interface function λ defined over a functional space $\Lambda(\Gamma)$ with a suitable regularity, we have, $\forall \lambda \in \Lambda(\Gamma)$,:

$$a_1(u_1, R_1\lambda) + a_2(u_2, R_2\lambda) = (f_1, R_1\lambda)_{\Omega_1} + (f_2, R_2\lambda)_{\Omega_2}, \tag{11}$$

with $R_i\lambda$ denoting a prolongation (or lifting) of λ into $V_i(\Omega_i)$, $i = 1, 2$. This implies, using (9) and (10), that

$$\begin{aligned}(L_1 u_1, R_1\lambda)_{\Omega_1} &- \int_{\partial\Omega_1} \psi_1(u_1, \mathbf{n})\lambda + \\ &+ (L_2 u_2, R_2\lambda)_{\Omega_2} + \int_{\partial\Omega_2} \psi_2(u_2, \mathbf{n})\lambda = \\ &= (f_1, R_1\lambda)_{\Omega_1} + (f_2, R_2\lambda)_{\Omega_2}, \quad \forall \lambda \in \Lambda(\Gamma).\end{aligned}$$

Using the strong form (3) and (4), we get

$$\int_{\Gamma} [\psi_1(u_1, \mathbf{n}) - \psi_2(u_2, \mathbf{n})]\, \lambda = 0, \quad \forall \lambda \in \Lambda(\Gamma), \tag{12}$$

which corresponds to the weak form of (6).

3.1 FEM-FEM formulation

If we consider a finite element discretization of both problems defined in Ω_1 and Ω_2, the formulation of the interface condition is easily derived by the continuous weak formulation. Indeed, given two triangulations of Ω_1 and Ω_2, we can consider two finite dimensional subspaces $V_{h,1}$ and $V_{h,2}$ of $V_1(\Omega_1)$ and $V_2(\Omega_2)$, respectively. The discrete counterparts of problems (7) and (8) read

Problem 1 (FEM): *Find* $u_{h,1} \in V_{h,1}$ *such that:*

$$a_1(u_{h,1}, v_{h,1}) = (f_1, v_{h,1})_{\Omega_1}, \quad \forall v_{h,1} \in V_{h,1}. \tag{13}$$

Problem 2 (FEM): *Find* $u_{h,2} \in V_{h,2}$ *such that:*

$$a_2(u_{h,2}, v_{h,2}) = (f_2, v_{h,2})_{\Omega_2}, \quad \forall v_{h,2} \in V_{h,2}. \tag{14}$$

For interface conforming finite element grids, we denote with Γ_h the common restriction of the two triangulations at the interface Γ. In this case, the Dirichlet interface condition (5) simply implies that

$$u_{h,1} = u_{h,2}, \quad \text{on } \Gamma_h,$$

while the Neumann interface condition (6) can be written, as in the continuous problem, in weak form:

$$\int_{\Gamma_h} \psi_1(u_{h,1}, \mathbf{n}_{h,1})\lambda_h = -\int_{\Gamma_h} \psi_2(u_{h,2}, \mathbf{n}_{h,2})\lambda_h, \quad \forall \lambda_h \in \Lambda_h(\Gamma_h),$$

where $\Lambda_h(\Gamma_h)$ is the finite dimensional subspace of $\Lambda(\Gamma)$, while $\mathbf{n}_{h,1} = -\mathbf{n}_{h,2}$ are the two unit outward normal directions across Γ_h.

In the more general case where the triangulations are non-conforming at the interface, let us denote with $\Gamma_{h,1}$ and $\Gamma_{h,2}$ the restrictions of the triangulations over Ω_1 and Ω_2, respectively, and with $\Lambda_{h,1}$ and $\Lambda_{h,2}$ the corresponding finite dimensional subspaces of $\Lambda(\Gamma)$. In this case, to impose the Dirichlet interface condition (5) a suitable interpolating operator $\mathcal{I}_h$ from $\Lambda_{h,2}$ to $\Lambda_{h,1}$ is required, resulting in

$$u_{h,1} = \mathcal{I}_h(u_{h,2}).$$

Introducing the bases $\{\xi_{1,i}\}$, $i = 1, \ldots, \dim \Lambda_{h,1}$ and $\{\xi_{2,i}\}$, $i = 1, \ldots, \dim \Lambda_{h,2}$ of the discrete spaces $\Lambda_{h,1}$ and $\Lambda_{h,2}$, respectively, $u_{h,1}$ and $u_{h,2}$ can be developed in terms of the respective basis functions:

$$\begin{aligned} u_{h,1}(\mathbf{x}) &= \sum_j U_{1,j}\, \xi_{1,j}(\mathbf{x}), \quad \mathbf{x} \in \Gamma_{h,1}, \\ u_{h,2}(\mathbf{x}) &= \sum_j U_{2,j}\, \xi_{2,j}(\mathbf{x}), \quad \mathbf{x} \in \Gamma_{h,2} \end{aligned}$$

where $\mathbf{U}_1 = \{U_{1,j}\}$ and $\mathbf{U}_2 = \{U_{2,j}\}$ are the nodal value vectors of $u_{h,1}$ on $\Gamma_{h,1}$ and $u_{h,2}$ on $\Gamma_{h,2}$, respectively.

Denoting with H the matrix representing the linear interpolating operator $\mathcal{I}_h$, we get

$$\mathbf{U}_1 = H\mathbf{U}_2.$$

In the non-conforming case, the weak form of condition (6) is more involved. The discrete equivalent of (11) can be written, for $\lambda_{h,i} \in \Lambda_{h,i}(\Gamma_{h,i}), i = 1, 2$, as

$$\begin{aligned} a_1(u_{h,1}, &R_{h,1}\lambda_{h,1}) + a_2(u_{h,2}, R_{h,2}\lambda_{h,2}) \\ &= (f_1, R_{h,1}\lambda_{h,1})_{\Omega_1} + (f_2, R_{h,2}\lambda_{h,2})_{\Omega_2}, \end{aligned} \tag{15}$$

with $R_{h,i}\lambda_{h,i}$ denoting a prolongation of $\lambda_{h,i}$ into $V_{h,i}(\Omega_i)$, $i = 1, 2$. Proceeding similarly to what done at continuous level, we can retrieve the discrete equivalent of condition (12), for $i = 1, \ldots, \dim \Lambda_{h,2}$, as:

$$\int_{\Gamma_{h,1}} \psi_1(u_{h,1}, \mathbf{n}_{h,1})\, \tilde{\xi}_{2,i} = \int_{\Gamma_{h,2}} \psi_2(u_{h,2}, \mathbf{n}_{h,2})\, \xi_{2,i} \tag{16}$$

where $\tilde{\xi}_{2.i}$ is the interpolation of $\xi_{2,i}$ on $\Gamma_{h,1}$ and can be rewritten in terms of the basis of $\Lambda_{h,1}$, as follows

$$\tilde{\xi}_{2,i} = \sum_k H_{k,i}\xi_{1,k}.$$

Note that to obtain the $i-th$ basis function $\tilde{\xi}_{2,i}$, the coefficients of the linear combination are the $i-th$ column of H.

Furthermore, if ψ_1 and ψ_2 are developed in terms of the corresponding basis functions

$$\begin{aligned} \psi_1(u_{h,1}, \mathbf{n}_{h,1}) &= \sum_j \Psi_{1,j}\, \xi_{1,j}(\mathbf{x}), \\ \psi_2(u_{h,2}, \mathbf{n}_{h,2}) &= \sum_j \Psi_{2,j}\, \xi_{2,j}(\mathbf{x}), \end{aligned}$$

the algebraic counterpart of the right-hand-side of (16) then reads, for $i = 1, \ldots, \dim \Lambda_{h,2}$:

$$\begin{aligned} \int_{\Gamma_{h,2}} \psi_2(u_{h,2}, \mathbf{n}_{h,2})\, \xi_{2,i} &= \int_{\Gamma_{h,2}} \sum_j \Psi_{2,j}\, \xi_{2,j}\, \xi_{2,i} = \\ &= \sum_j \Psi_{2,j} \int_{\Gamma_{h,2}} \xi_{2,j}\, \xi_{2,i} = (M_2\mathbf{\Psi}_2)_i, \end{aligned} \tag{17}$$

where M_2 is the mass matrix relative to $\Gamma_{h,2}$.

Similarly the left-hand-side of (16) can be rewritten in algebraic form, as

$$\int_{\Gamma_{h,1}} \psi_1(u_{h_1}, \mathbf{n}_{h,1})\, \tilde{\xi}_{2,i} \approx \int_{\Gamma_{h,1}} \sum_j \Psi_{1,j}\, \xi_{1,j}\, \tilde{\xi}_{2,i}$$
$$\approx \int_{\Gamma_{h,1}} \sum_j \Psi_{1,j}\, \xi_{1,j} \sum_k H_{k,i}\, \xi_{1,k} = (H^T M_1 \mathbf{\Psi}_1)_i, \qquad (18)$$

where we recognize the mass matrix M_1 relative to $\Gamma_{h,1}$.

By combining equations (17) and (18), for every value of i, we end up with the following vector condition

$$M_2 \mathbf{\Psi}_2 = H^T M_1 \mathbf{\Psi}_1, \qquad (19)$$

relating the energy balance at the interface. The same condition will be found in the following section for a mixed-discretization case via a more involved algebraic reasoning.

It is important to notice that, in the case of FSI problems, the weak imposition of the continuity of the normal stress corresponds to imposing the principle of virtual works at the interface, for any given displacement, and thus the balance of energy transfer at the interface.

4 FVM-FEM formulation

Most available commercial fluid flow solvers, as well as many open source codes, are based on finite-volume discretization, mainly because of the high efficiency of such methods for the numerical solution of turbulent flows on large computational grids. On the other hand, the finite elements method has become the standard tool for the numerical simulations of structural problems. It is thus common and quite efficient in fluid-structure interaction problems to couple a finite-volume fluid solver to a finite-element structural solver. In this section we will generalize and adapt the results obtained in the previous section to this particular class of problems, still in the case of non-conforming grids at the interface.

In particular, we will consider the case in which the structural solver is based on a Lagrangian piecewise-linear finite element discretization, where the degrees-of-freedom are the displacement values at the grid nodes. The fluid solver considered is based instead on a finite volume discretization, with a colocated arrangement for the velocity and pressure variables at the center of the grid cells and the conditions at the interface are imposed on the face centers. Both pressure and velocity are considered constant over each control volume. Many different interpolation schemes to transfer data from cell center to face center (and *vice versa*) exist [15], however the specific algorithm used is not relevant in this context.

In this framework, the kinematic condition (1) requires the transfer of the displacement of the interface structural nodes to the fluid nodes. This can be achieved via an interpolatory operator that usually acts from the nodal value of the structural displacement to the fluid interface grid nodes.

The interpolation of the structure displacement $\mathbf{D}_{S_n}$ to the fluid interface nodes can thus be expressed as a vector relation

$$\mathbf{D}_{F_n} = H_{F_n S_n} \mathbf{D}_{S_n} \, , \tag{20}$$

where the index S_n and F_n denotes the structure and fluid nodes respectively and $H_{F_n S_n}$ is a general interpolatory operator (a rectangular matrix) from the value on the structural nodes to the fluid nodes.

It should be noted that such operation already implies a first approximation: condition (1) states that fluid and structure velocities (and displacements) should coincide at the interface; however, in case of non matching interfaces, this is usually not feasible. In fact, let us assume that the displacement is transferred from the structure to the fluid nodes in order to deform the grid through any interpolation technique. The same interpolation could in principle be used to transfer the velocity from the structure to the fluid face centers (where it will be used as velocity boundary condition). Unfortunately, in so doing, the *geometric conservation law* (GCL) will not be respected. The GCL states that the mesh velocity must be equal to volume swept by the cell faces divided by the time interval [18]. If such condition is not respected, incorrect artificial mass sources are introduced in the Navier-Stokes equations. The opposite approach could be based on transfering, at first, the velocity from the structure to the face centers and then recovering the actual grid points motion via geometric considerations. Unfortunately, this problem may not be easy to solve, and in some cases its solution may not even exist, since we are considering the general case where the mesh is composed by polyhedral cells and the number of points is arbitrarily large, as well as the number of the faces connected to a given face.

Furthermore, it is interesting to notice that although the cell faces are usually planar at the beginning of a simulation, when the mesh is deformed, and if the mesh is composed of arbitrary polyhedra, such motion usually will not preserve this property. In this sense, the mesh velocity, but even more basic geometrical quantities such as the face normal or area, are usually retrieved by ad-hoc averaging/interpolation techniques yielding a first approximation.

To give an example, in the code used in this work, the face normal and area magnitude of each face are reconstructed by evaluating the face barycentre and then subdividing each face in the triangles obtained joining the face vertices and the barycentre (see Figure 2), and then evaluating for each of those sub-triangles the quantity needed. The total face value is finally obtained as the area-weighted average of the values ϕ_i on each sub-triangle, i.e. $\phi_f = \sum_i \frac{\phi_i A_i}{\sum_i A_i}$.

Let us now focus on condition (2) and the transfer of informations in the opposite direction, from FV to FE discretization. When using a finite volume

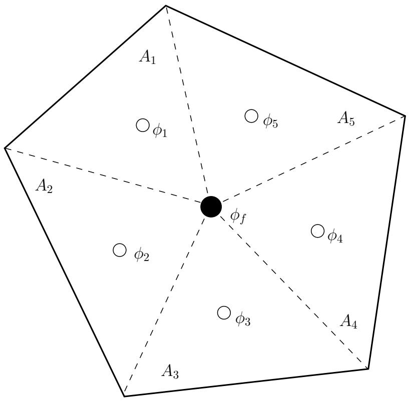


Figure 2: Subdivision of a generic polyhedron face in triangles. Face values are obtained as area-weighted average of the values ϕ_i on each sub-triangle. A_i is the area of each sub-triangle.

discretization, the normal stress at the interface is naturally defined on the face centers.

On the other hand, in the finite element structural solver, the Neumann condition on the normal stress usually appears in the weak formulation as a right-hand-side term of the form

$$\int_\Gamma (\boldsymbol{\sigma}_S \mathbf{n}_S) \cdot \boldsymbol{\xi}, \tag{21}$$

(with $\boldsymbol{\xi}$ denoting a test function) and thus the proper choice of interpolation points would be the location of the quadrature nodes of each element. Other less accurate choices would be to interpolate the required field on the nodal points location and then use the finite-element basis to obtain the value on the quadrature nodes or, even more bluntly, to assume the normal stress to be constant over the whole structural face (in a similar way to what is imposed on the fluid side) and compute the interpolation only at the center of the structural faces.

To keep the notation simple, hereafter we will consider the case where the interpolation is required on the nodes of the structural grid. The interpolation of the normal stress from the fluid face centers to the structure is thus given by

$$\boldsymbol{\Sigma}_{S_n} = H_{S_n F_f} \, \boldsymbol{\Sigma}_{F_f}, \tag{22}$$

where $\boldsymbol{\Sigma}_{S_n}$ and $\boldsymbol{\Sigma}_{F_f}$ represents the structural normal stress at the grid nodes and the fluid normal stress at the face centers respectively, and $H_{S_n F_f}$ is a general interpolatory operator (again, a rectangular matrix) acting from the fluid face centers to the structural nodes.

To better clarify our procedure, in Figure 3 a schematic description of the non-matching finite element/finite volume discretization is reported. The displacement is transferred from the structure interface nodes (•) to the nodes of

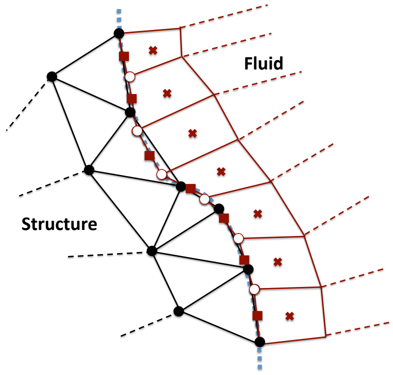


Figure 3: On the left, P1 finite elements. On the right, cell/face centered finite volume elements.Symbol explanation is as follows: • FE grid nodes, ∘ FV grid nodes, × FV cell centers and ■ FV face centers on the interface.

the interface of the fluid grid (∘). The normal stress instead is transferred from the fluid interface face centers (■), properly obtained by the neighbor cell center (×) values, to the structure interface nodes.

4.1 Energy conservation at the interface

In this section, we will analyze the problem of conserving the energy transfer at the interface in the framework of a FVM-FEM discretization on non-conforming interfaces.

In the case of a common interface Γ, the energy is conserved if the work W_S done by the structural forces on the interface equals the work W_F done by the fluid forces, namely

$$W_S(\mathbf{d}) = \int_\Gamma (\boldsymbol{\sigma}_S \mathbf{n}_S) \cdot \mathbf{d} = \int_\Gamma (\boldsymbol{\sigma}_F \mathbf{n}_F) \cdot \mathbf{d} = W_F(\mathbf{d}), \quad \forall\, \mathbf{d},$$

with $\mathbf{d} = \mathbf{d}_S = \mathbf{d}_F$ being the displacement of the interface.

At discrete level, since the fluid solver is based on finite volume approximation and values are constant over a whole face, the work $\tilde{W}_F$ done by the discrete fluid forces at the fluid interface can be computed as follows

$$\tilde{W}_F = \int_{\Gamma_F} \sum_{i=1}^{N_{F_f}} \Sigma_{F_f,i}\, \chi_i \sum_{j=1}^{N_{F_f}} D_{F_f,j}\, \chi_j = \sum_{i=1}^{N_{F_f}} \Sigma_{F_f,i} A_{F_f,i} D_{F_f,i} = (M_F \mathbf{D}_{F_f})^T \mathbf{\Sigma}_{F_f},$$

where χ_j is the characteristic function on the i-th fluid interface face, $\Sigma_{F_f,i}$ and $D_{F_f,i}$ are the face center values of normal stress and displacement, respectively,

$\mathbf{\Sigma}_{F_f}$ and $\mathbf{D}_{F_f}$ denote the vectors containing all $\Sigma_{F_f,i}$ and $D_{F_f,i}$ values, respectively. Note that, in finite-volume discretization, the fluid interface mass matrix M_F is simply a diagonal matrix whose entries are the areas of the corresponding faces $A_{F_f,i}$.

Analogously, the approximation $\tilde{W}_S$ of the work acting on the structure at the discrete level can be obtained as follows

$$\tilde{W}_S = \int_{\Gamma_S} \sum_{i=1}^{N_{S_n}} \Sigma_{S_n,i}\, \xi_i \sum_{j=1}^{N_{S_n}} D_{S_n,j}\, \xi_j = (M_S \mathbf{D}_{S_n})^T \mathbf{\Sigma}_{S_n} \,,$$

where $\mathbf{\Sigma}_{S_n,i}$ is the discrete value of the normal stress at the i-th structure interface node , M_F is the finite-element interface mass matrix for the structure and $\mathbf{D}_{S_n}$ is the displacement evaluated at the nodes of the structural mesh.

We can thus conclude that, at discrete level, energy is globally conserved if the following algebraic identity is satisfied:

$$(M_S \mathbf{D}_{S_n})^T \mathbf{\Sigma}_{S_n} = (M_F \mathbf{D}_{F_f})^T \mathbf{\Sigma}_{F_f},$$

therefore

$$\mathbf{D}_{S_n}^T M_S \mathbf{\Sigma}_{S_n} = \mathbf{D}_{F_f}^T M_F \mathbf{\Sigma}_{F_f}, \tag{23}$$

where we have used the fact that the mass matrices are obviously symmetric.

Substituting (20) into (23), and neglecting the mismatch between the fluid node and fluid face location of the displacement (which will be further discussed in section 5.2), we obtain

$$\mathbf{D}_{S_n}^T M_S \mathbf{\Sigma}_{S_n} = \mathbf{D}_{S_n}^T H_{F_f S_n}^T M_F \mathbf{\Sigma}_{F_f}. \tag{24}$$

Therefore, if we require that

$$M_S \mathbf{\Sigma}_{S_n} = H_{F_f S_n}^T M_F \mathbf{\Sigma}_{F_f}, \tag{25}$$

then equation (24) is automatically satisfied for any displacement field $\mathbf{D}_{S_n}$. This condition is, not surprisingly, equivalent to equation (19) found in the previous section.

Using equation (25) we finally find the following relation between fluid and structural stresses:

$$\mathbf{\Sigma}_{S_n} = M_S^{-1} H_{F_f S_n}^T M_F \mathbf{\Sigma}_{F_f}. \tag{26}$$

In conclusion, in order to conserve energy at the interface it is sufficient to use as interpolatory operator to transfer the normal stress from fluid to structure the transposed of the operator $H_{F_f S_n}$ used to transfer the information from the structure to the fluid weighted by the inverse of the structural interface mass matrix and the fluid interface mass matrix, i.e.

$$H_{S_n F_f} = M_S^{-1} H_{F_f S_n}^T M_F. \tag{27}$$

Finally, since in the finite element structural solver the normal stress appears in the weak formulation as an integral term (equation (21)) which, in algebraic form corresponds to the left side of equation (25), there is no need to actually evaluate the inverse of the structural interface mass matrix. Indeed, the fluid normal stress can be directly transferred to the structural solver as an already integrated quantity, namely $H^T_{F_f S_n} M_F \mathbf{\Sigma}_{F_f}$.

It is interesting to notice that actually no information about the finite element discretization is used to transfer the stress from fluid to solid since the right hand side of equation (25) is independent of the structural discretization schemes.

Remark 1 *The previous analysis has concerned a steady fluid-structure coupling. To extend it to time-dependent problems, the possibly different time-step and time discretization schemes used in the fluid and structural solvers should be taken into account. This, however, in general is not a critical issue.*

5 RBF interpolation in FSI problems

In this section, we will analyze the role of the RBF method as interpolatory tool in the framework of FSI problems.

5.1 Introduction to Radial Basis Functions interpolation

In recent years, the Radial Basis Function method has become a well-established tool for the multivariate interpolation of scattered data, thanks to its peculiar approximation properties [3, 4]. This technique has been successfully adopted in different areas such as, *e.g.*, computer graphics [25], three-dimensional surface reconstruction [5], fluid-structure interaction problems [1, 10] and mesh motion [9, 2].

Here we consider the adoption of the RBF technique for the interpolation of data on non-matching grids, in order to transfer those data from the structure to the fluid and *vice versa*. Given the known values $\mathbf{f}_C$ of a function in a set of N_C control points $\{\mathbf{x}_{C_j}\}_{j=1,\ldots,N_C}$, the Radial Basis Function interpolation defines a global interpolation function $f(\mathbf{x})$ in the whole domain. No restriction on the admissible location of the control points applies.

The value of f at a generic location $\mathbf{x}$ is obtained as a weighted sum of radial basis functions $\phi(|\mathbf{x}-\mathbf{x}_{C_j}|)$ based on the Euclidean distance between the control points position $\mathbf{x}_{C_j}$ and $\mathbf{x}$:

$$f(\mathbf{x}) = \sum_{j=1}^{N_C} \gamma_j \phi(|\mathbf{x} - \mathbf{x}_{C_j}|) + q(\mathbf{x}),$$

where γ_j is the weight associated to the j-th basis function and $q(\mathbf{x})$ is a polynomial term. In order to obtain the weights γ_j and the coefficients of the polynomial

q, we impose the interpolation conditions at the control points

$$f(\mathbf{x}_{C_j}) = f_{C_j} \quad \text{for} \quad j = 1, ..., N_C. \tag{28}$$

where f_{C_j} denotes the j-th element of vector $\mathbf{f}_C$. Moreover, it has been demonstrated (see [4]) that in order to guarantee the positive definiteness of the RBF problem, the following condition

$$\sum_{j=1}^{N_C} \gamma_j p(\mathbf{x}_{C_j}) = 0 \tag{29}$$

should hold for any polynomial p with degree less or equal than the degree of q; the choice of the degree of the polynomial q depends on the basis functions used [10]. We assume in the following that q is linear: in such case, constant fields are exactly interpolated.

Denoting with $\boldsymbol{\beta}$ the vector containing the coefficients of q, i.e. $q(\mathbf{x}) = \beta_0 + \beta_1 x + \beta_2 y + \beta_3 z$, equations (28) and (29) result in the linear system

$$\begin{bmatrix} B_{CC} & P_C \\ P_C^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\gamma} \\ \boldsymbol{\beta} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_C \\ \mathbf{0} \end{bmatrix} \tag{30}$$

where B_{CC} is a $N_C \times N_C$ matrix whose entries are $B_{CC_{i,j}} = \phi(|\mathbf{x}_{C_i} - \mathbf{x}_{C_j}|)$ and P_C is a $N_C \times 4$ matrix with the j-th row given by $[1 \; x_{C_j} \; y_{C_j} \; z_{C_j}]$.

The interpolation on a set of N_I interpolation points $\{\mathbf{x}_{I_k}\}_{k=1,...,N_I}$ can be computed by

$$\mathbf{f}_I = \underbrace{[B_{IC} \; P_I]}_{R_{IC}} \begin{bmatrix} \boldsymbol{\gamma} \\ \boldsymbol{\beta} \end{bmatrix} = [B_{IC} \; P_I] \underbrace{\begin{bmatrix} B_{CC} & P_C \\ P_C^T & 0 \end{bmatrix}^{-1}}_{R_{CC}^{-1}} \begin{bmatrix} \mathbf{f}_C \\ \mathbf{0} \end{bmatrix}$$

and, denoting with $\tilde{R}_{CC}^{-1}$ the first N_C columns of R_{CC}^{-1}, we obtain

$$\mathbf{f}_I = \underbrace{R_{IC} \, \tilde{R}_{CC}^{-1}}_{H} \mathbf{f}_C = H_{IC} \, \mathbf{f}_C \, , \tag{31}$$

where H_{IC} represents the interpolation matrix from the control points to the interpolation points.

Remark 2 *In practice, unless N_C and N_I are very small, the matrix H_{IC} is never assembled because of its high computational cost. In fact, the cost of computing eq. (31) assembling matrix H_{IC} is of the order $(N_I \times N_C^2)$ operations, while if we first multiply $\tilde{R}_{CC}^{-1}$ by f_C and then multiply the resulting vector by R_{IC} the total cost is only of the order of $(2N_I \times N_C)$ operations. In principle, the matrix H_{IC} could be evaluated once for all at the beginning and stored, so that at run-time the first method would cost only $(N_I \times N_C)$ operations, thus about twice as fast as the second approach. However, in practice, if the number of control points is large (order of thousands), assembling the matrix H_{IC} is usually too expensive.*

Many different radial basis functions have been proposed and analyzed in the literature [4]; among the most common options, we mention:

- the Inverted Multi-Quadric biharmonic splines (IMQ):

$$\phi(|\mathbf{x}|) = \frac{1}{\sqrt{|\mathbf{x}|^2 + r^2}} \, ;$$

- the Gaussian splines:

$$\phi(|\mathbf{x}|) = e^{-|\mathbf{x}|^2/r^2} \, ;$$

- the Thin Plate splines (TP):

$$\phi(|\mathbf{x}|) = |\mathbf{x}/r|^2 \ln |\mathbf{x}/r| \, ;$$

- the Beckert and Wendland [1] C^2 splines:

$$\phi(|\mathbf{x}|) = \left((1 - |\mathbf{x}|/r)_+\right)^4 (4|\mathbf{x}|/r + 1) \ ,$$

where r is a scaling factor that controls the shape of the basis and the subscript $_+$ indicates that only positive values are taken into consideration.

The first three bases have global support, while the last one has compact support. The advantage of using basis with compact support is that the matrices R are sparse and, therefore, considerable speed-up may be obtained in both the matrix-vector multiplication and the solution of the linear system (30), when adopting an iterative method. The drawback is that, in theory, the interpolation is less accurate [4]. The scaling factor r has a significant impact on the accuracy and stability of the interpolation. The larger it is, the wider the radius of influence of each control point. Optimal value of this parameter can be computed analytically only in very simple cases [4]. However, it is common practice to consider a value of r large enough such that the support of each radial basis function includes at least all nearest neighbor control and interpolation points. The choice of the scaling factor r is limited from above by the fact that the condition number of the R_{CC} matrix increases exponentially with r [4]: if the basis are very wide and flat, it will be hard to distinguish the contribution of every single control point on a nearby interpolation point and the matrix will be almost singular. The best solution would be to use a different radius r for each radial basis function; however, the well-posedness of problem (30) is guaranteed only for constant r [1].

In section 6, the influence of the shape factor r on the accuracy and condition number of the RBF methods for the different bases will be analyzed on different numerical test cases.

Remark 3 *Usually, in the literature the TPS basis does not include a shape factor r, since it has no effect when the polynomial term is used (which is required* *to guarantee that the RBF matrix is positive definite). Indeed, if we include the factor r in the TPS basis, we can show that*

$$\phi(|\mathbf{x}|) = |\mathbf{x}/r|^2 \ln |\mathbf{x}/r| = \frac{1}{r^2} \left(|\mathbf{x}|^2 \ln |\mathbf{x}| - |\mathbf{x}|^2 \ln r \right) ,$$

therefore, the scaling factor is equivalent to an additional second order polynomial term scaled by a factor $\ln(r)$*: this term is negligible compared to the linear polynomial, unless we consider very large values of r and $|\mathbf{x}|$. In this work, we have decided to include the factor r in the TPS basis and analyze whether such independence is indeed verified in practice (see section 6).*

5.2 RBF in FSI problems

Let us now focus on the use of the RBF interpolation in the case of the FSI problem introduced in section 2. As mentioned before, the displacement values should be transferred from the grid nodes on the structural interface to the grid nodes on the fluid interface.

Using the RBF interpolation notation introduced in the previous section (see equation (31)), the interpolation of the structure displacement vector $\mathbf{D}_{S_n}$ to the fluid one $\mathbf{D}_{F_n}$ via equation (20) can thus be expressed as

$$\mathbf{D}_{F_n} = H_{F_n S_n} \mathbf{D}_{S_n} = R_{F_n S_n} \tilde{R}^{-1}_{S_n S_n} \mathbf{D}_{S_n} , \tag{32}$$

where the structural nodes S_n are used as control points and the fluid interface nodes F_n are the interpolation points.

Similarly, equation (22) becomes

$$\mathbf{\Sigma}_{S_n} = H_{S_n F_f} \mathbf{\Sigma}_{F_f} = R_{S_n F_f} \tilde{R}^{-1}_{F_f F_f} \mathbf{\Sigma}_{F_f} , \tag{33}$$

where, in this case, the fluid interface face centers F_f are used as control points and the structural nodes S_n are now the interpolation points.

We have seen in section 4.1 that, in order to conserve the energy at the interface, equation (27) should be used instead of (22). Moreover, to obtain equation (24) the mismatch between face and nodal fluid displacements has been neglected: formally only the operator $H_{F_n S_n}$ relating solid node displacements to fluid node displacements has been defined and not the operator $H_{F_f S_n}$ relating solid node displacements to fluid face displacements.

One possible solution would be to obtain the fluid displacement $\mathbf{D}_{F_f}$ at the face centers from the node center values through a further node-to-face interpolation, namely

$$\mathbf{D}_{F_f} = K_{F_f F_n} \mathbf{D}_{F_n} ,$$

with $K_{F_f F_n}$ being the matrix that performs such node-to-face interpolation. In this case, using (32) and (31), the nodal displacement $\mathbf{D}_{F_n}$ would be given by

$$\mathbf{D}_{F_f} = K_{F_f F_n} \mathbf{D}_{F_n} = K_{F_f F_n} H_{F_n S_n} \mathbf{D}_{S_n} = K_{F_f F_n} R_{F_n S_n} \tilde{R}^{-1}_{S_n S_n} \mathbf{D}_{S_n} . \tag{34}$$

We prefer, instead, computing the fluid displacement $\mathbf{D}_{F_f}$ at the face centers directly from the structural nodal displacement $\mathbf{D}_{S_n}$, where equation (32) is now written using the face centers as interpolation points, namely:

$$\mathbf{D}_{F_f} = H_{F_f S_n} \mathbf{D}_{S_n} = R_{F_f S_n} \tilde{R}^{-1}_{S_n S_n} \mathbf{D}_{S_n}. \tag{35}$$

Comparing (34) and (35), we note that the *core* factor of the RBF interpolation, i.e. $\tilde{R}^{-1}_{S_n S_n}$, is the same and only the *projections* (i.e. $K_{F_f F_n} R_{F_n S_n}$ and $R_{F_f S_n}$) are different. Moreover, using (35) the potentially critical node-to-face interpolation $K_{F_f F_n}$ is avoided. Notice that the matrix $H_{F_f S_n}$ requires the calculation of the new matrix $R_{F_f S_n}$.

Remark 4 *Using equation (26), we guarantee the conservation of the total energy at the expenses, however, of the consistency of the interpolation. Nothing in fact ensures that the row-sum of $M_S^{-1} \tilde{H}^T_{F_f S_n} M_F$ is still equal to one (and indeed usually this would not be the case). This means that with this approach if a constant normal stress is acting on the fluid interface, the stress imposed on the structure side of the interface is not exactly constant. If the standard approach is used, i.e. equation (22), consistency would have been ensured, but at the cost of a worse energy balance.*

5.3 RBF for mesh motion

One advantage of using the RBF technique for the interpolation of data between non-matching interfaces in FSI problems is that it allows using the same map to extend the interface deformation inside the three-dimensional fluid domain. Indeed, it is sufficient to enlarge the set of interpolation points so to include all the interior fluid mesh nodes. In this way, the correct mesh motion on the fluid-structure interface is naturally achieved, while to control the deformation far from the interface special adjustment should be made depending on the case at hand.

In particular, we first consider the case where the moving interface is located at the center of the domain and the other fluid domain boundaries are far away from it, see Figure 4. To move the volume mesh is thus sufficient to use as control points only the structural points located on the interface and then multiply the predicted deformation by a smooth cut-off function [2], given for example by:

$$b(\mathbf{x}) = \begin{cases} 1 & \text{if } t < 0, \\ 1 - t^2(3 - 2t) & \text{if } 0 < t < 1, \\ 0 & \text{if } t > 1\,, \end{cases}$$

with $t = |\,||(\mathbf{x} - \mathbf{x}_c)|| - R_{min}|/(R_{max} - R_{min})$.

As shown in Figure 5, this correction imposes a cubic decay for all points whose distance from a predefined "central" point $\mathbf{x}_c$ lies between distance R_{min} and the maximum distance R_{max}. For all points farther than R_{max} the deformation is set to zero. It is thus sufficient to ensure that the external boundaries

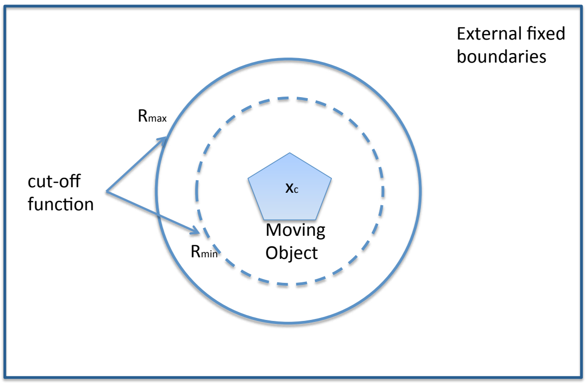


Figure 4: Example case where the moving object is located at the middle of the domain and far away from every other boundary. The local support of the RBF map due to the introduction of the cut-off function has also been highlighted.

are farther away than R_{max} from the interface to be sure that the they will not be deformed. If such cut-off function is used, it is worth noting that the points outside R_{max} are not moved and thus there is no need to insert them in the matrix R_{IC}: considerable speed-up can be achieved.

However, this approach cannot be used when the moving boundary is close to other fluid domain boundaries since the cut-off radius to be chosen would be too small. In this case, a possible solution is to use as control points also all the nodes of the boundaries close to the moving one and impose on those a displacement equal to zero.

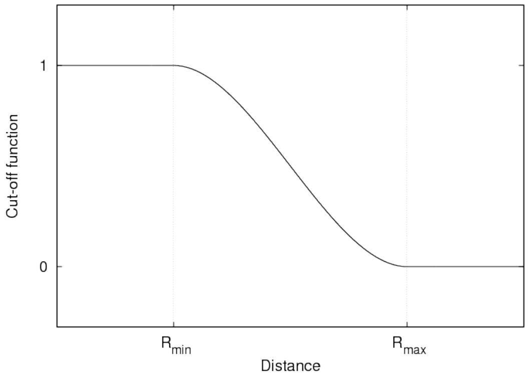


Figure 5: Smooth cut-off function applied to the RBF map.

6 Numerical results

In this section, we present the numerical simulations carried out to assess the behavior of the RBF technique described in the previous sections. In particular, we will first consider the interpolations based on different families of RBF bases and we will compare their behavior on different geometries and with different values of the shape factor r. The interpolation will be tested on simple geometries as well as on a realistic configuration, the latter being referred to a wind/sail fluid-structure interaction problem.

6.1 Numerical assessment of RBF interpolation

As a first test case, the properties of the RBF interpolation are analyzed by computing the interpolation of a constant function defined over a planar unit square domain. Let us assume that the solution (on the structure) is given over a 25×25 quadrilateral uniform grid and it is interpolated over a 40×40 uniform and structured quadrilateral (fluid) grid, thus non-conforming w.r.t the structural one. Since the fluid mesh is finer, we can consider that the upper limit of the distance between one control point and its neighbor points (with respect to both the other control points and the points where the interpolation is requested) is of the order $\sqrt{2}/25 \approx 0.0565$

In Figure 6, the L^2-norm and the maximum-norm of the interpolation error for a constant field ($f(x, y) = 1$) is reported for different RBF bases, without the polynomial term. For the IMQ and C^2 bases, results are in agreement with the theory: the error decreases as the support radius increases. For the Gaussian

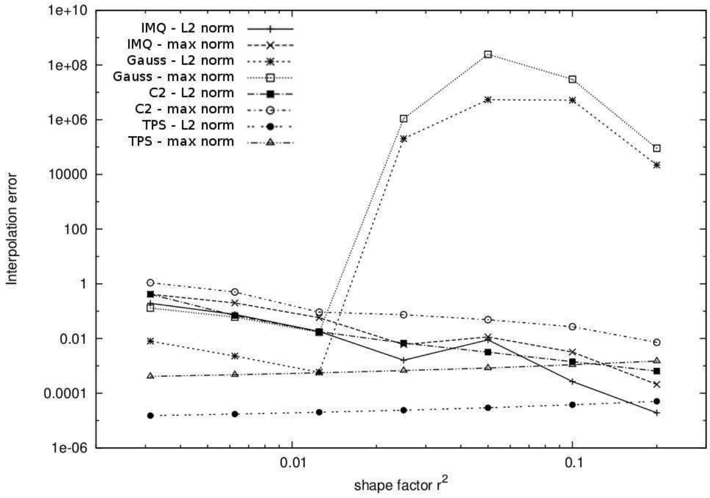


Figure 6: Interpolation error of a constant field on unit square domain for different RBF bases, without using the polynomial term.

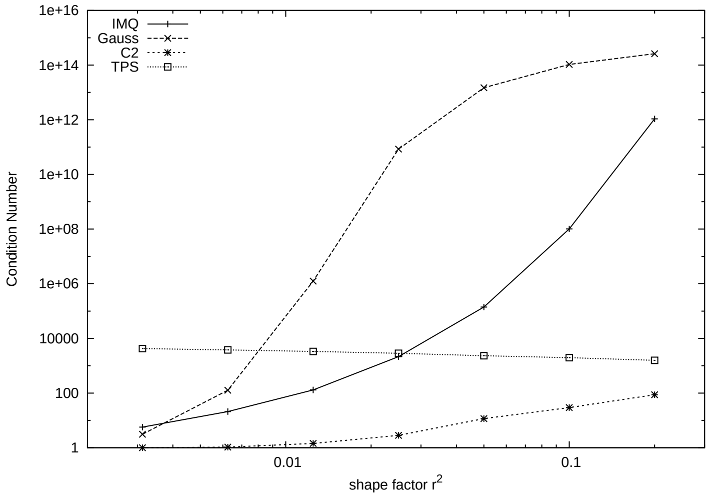


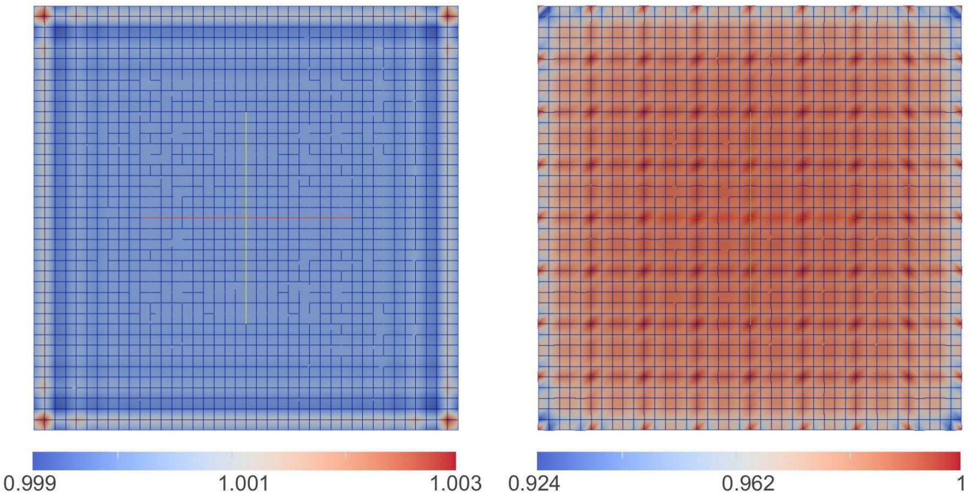
Figure 7: Condition number of the interpolation matrix R_{CC} on a unit square domain for different RBF bases and shape factors r.

basis functions, the error decreases only for small values of r, while for larger values, the problem becomes unstable. This is due to the fact that the condition number of the RBF matrix grows exponentially when r increases, see Figure 7. In the case of the Gaussian basis, when r is large, the condition number is so high that the round-off errors become dominant.

The TPS basis is the one that performs the best and has a very small dependence on the choice of the radius r. It is worth noting that the optimal choice of the shape factor is usually much bigger than the distance between one control point and its neighbor point ($r \approx 0.056$), as usually suggested in literature. In Figure 6 such value is the smallest radius evaluated and it is clear that the optimum is achieved for larger values of r.

In Figure 8 the interpolation of the unit constant field with the IMQ basis for two different radii is shown. For large shape factor, the relative L^2 error is equal to 0.0265% , and indeed the field is well interpolated all over the domain unless near the edges and corners where the deviation gets a bit larger but always acceptable. For small radius instead, the error rises up to 7.8% and we can clearly see many small peaks all over the domain due to the fact that the support radius of the basis is too small.

A second test has been performed using the same grids, but interpolating the non-constant function $f(x, y) = \sin(2\pi x)\cos(3\pi y) + e^{xy}$, see Figure 9. The trend is very similar to the one obtained when interpolating a constant field. As shown in Figure 10, when the support radius is not large enough, the method may produce spurious oscillations in the interpolated field which may be detrimental for the FSI problem to converge.



(a) IMQ, $r^2 = 0.1$

(b) IMQ, $r^2 = 0.00625$

Figure 8: Interpolation of a constant field equal to 1 with the RBF technique using the IMQ basis for two different shape factors r.

The results reported above have been computed without the use of the polynomial term in the RBF interpolation due to the fact that all control points are coplanar, and thus the corresponding matrix R_{CC} would have become singular (if a three-dimensional RBF library is used). The theory of RBF interpolation guarantees that, with the adoption of the polynomial term, the interpolation is

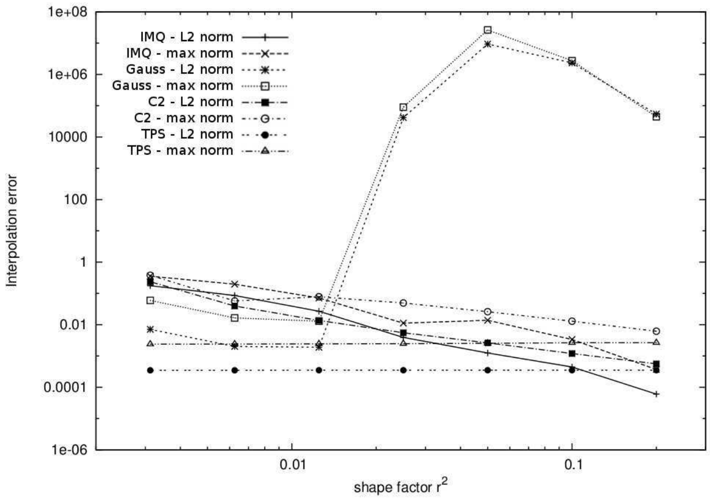
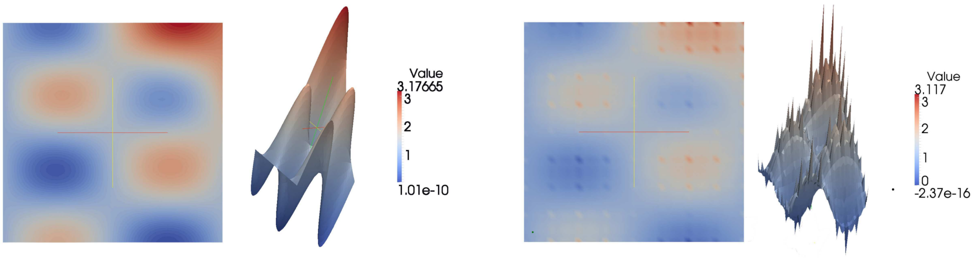


Figure 9: Interpolation error for the function $f(x, y) = \sin(2\pi x)\cos(3\pi y) + e^{xy}$ on a unit square domain for different RBF bases, without the polynomial term.



(a) IMQ, $r^2 = 0.1$

(b) IMQ, $r^2 = 0.00625$

Figure 10: Interpolation of the field $f(x, y) = \sin(2\pi x)\cos(3\pi y) + e^{xy}$ on uniform cartesian grids with the RBF technique using IMQ basis for two different shape factors r.

consistent (i.e. constant functions are interpolated exactly). This property has been assessed on another test case where a cylindrical domain has been used. The control points grid is structured and uniform all over the domain, while the interpolation points are clustered near the inlet and outlet of the cylinder. On the non-matching interface, quadrilateral elements have been used for both fluid and structural grid.

In Figure 11 the condition number of the RBF matrix for the IMQ and TPS bases with and without polynomial term is plotted: in both cases, the introduction of the polynomial term produces a rise in the condition number. For the IMQ basis, once again the condition number gets larger when the support radius increases, while for the TPS basis, it is almost constant when no polynomials are used, otherwise it gets smaller for larger radius.

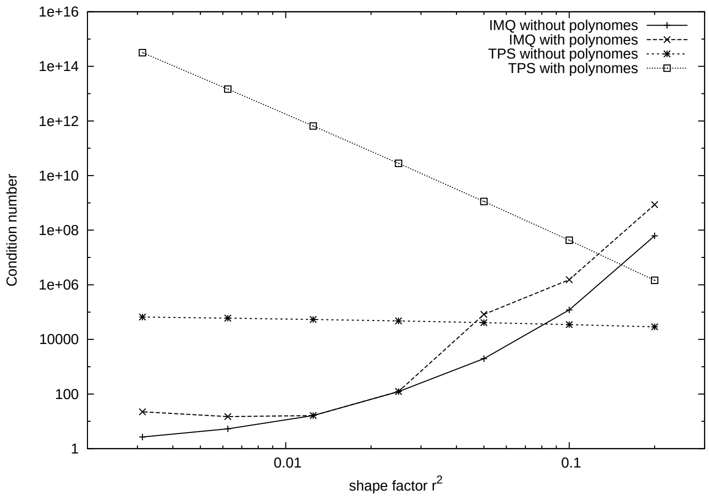


Figure 11: Condition number of the interpolation matrix $R_{cp,cp}$ for the cylindrical test case for different RBF bases, shape factors r and with or without the polynomial term.

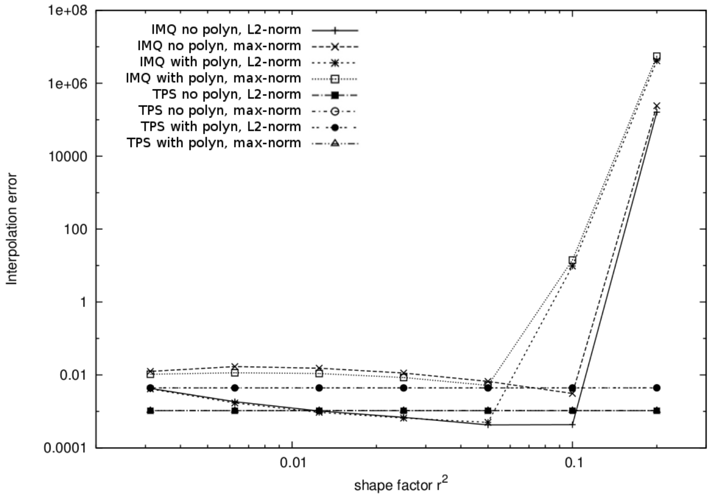


Figure 12: Interpolation error for the function $f(x, y) = \sin(2\pi x)\cos(2\pi y) + e^{xy}$ on the cylindrical test case for different RBF bases, without the polynomial term.

If the polynomial term is used, constant functions are interpolated exactly.

Interpolating the non-constant function $f(x, y) = \sin(2\pi x)\cos(3\pi y) + e^{xy}$ on the same meshes, the polynomial term greatly improves the accuracy of the interpolation for the IMQ basis while it has no effect on the TPS basis, see Figure 12. The IMQ basis function interpolation becomes more accurate as the radius gets larger until the condition number gets too large and the interpolation blows-up. The TPS basis function interpolation has no dependence on the shape factor r.

To better assess the interpolation properties of the different radial bases, we have performed a spatial convergence numerical test, computing the interpolation error with three different grid resolutions. The coarse grid is composed by 12×12 elements on the structure side and 20×20 elements on the fluid side, the medium grid by 24×24 elements on the structure side and 40×40 elements on the fluid side and, finally, the fine grid by 48×48 elements on the structure side and 80×80 elements on the fluid side. The values of the shape factor for the different radial bases (which are kept constant for all meshes so to test also the "flexibility" of each basis to perform on meshes with different sizes) are: $r^2 = 0.005$ for IMQ, $r^2 = 0.01$ for Gaussian and $r^2 = 0.2$ for C2. For the TPS basis, the shape factor has no major influence and has thus been set equal to 1.

Furthermore, we have decided to compare the RBF interpolation accuracy with a geometric-projection based interpolation. In this simple interpolation, which is often still adopted in practice, each control point is projected to the closest face on the surface where the interpolation is required. Then, for every face of the target surface, a weighted average (based on the distance from the face center) of the value of all points projected to the surface is computed.

In Figure 13, the interpolation errors for the different RBF bases and for the geometrical interpolations are shown. As can be observed, all the RBF interpolations converge faster than the geometrical interpolation that is only linear both in L^2 and maximum norms. In particular, we can remark that the IMQ basis displays a fourth order convergence in L^2-norm and third order in maximum-norm, while the TPS basis is third order in L^2-norm and second order in maximum-norm.

A similar analysis on the interpolation properties of different RBF bases has been performed on a more realistic case related to a wind/sail fluid-structure interaction problem. We consider the RBF interpolation over a sail geometry. The structural mesh is made of 2694 unstructured quadrilateral elements, with finer elements near the sail edges, while the volume fluid mesh is composed of 383879 tetrahedra refined near the sails, and 13029 triangles on the sail patch (refined near the sail edges as well).

To test the RBF interpolation, we have considered a pseudo-realistic analytic deformation distribution of a flapping sail, defined by the function $\mathbf{d} = (0, 0, 0.02 \sin(\pi x/5) \cos(\pi y/10) xy)$ (see Figure 14), which has been interpolated from the structure to the fluid surface grids.

Table 1 summarizes the results on the RBF interpolation on the sail geometry with the IMQ and TPS bases which were the ones performing better in the previous test cases. Once again we observe a strong dependence of the quality of the IMQ interpolation with respect to the shape factor, with very good results for $r = 1$, while lower accuracy for smaller r and instability for larger r are observed. When using the polynomial, the accuracy is improved. For the TPS basis instead, the level of accuracy is always very high for a large range of shape factor r: when using polynomial, there seems to be no effect when varying the shape factor r, while some small dependence is observed when no polynomials are used. The best and safest option, in terms of accuracy of the interpolation, is thus the use of the TPS with the polynomial term since this choice is always

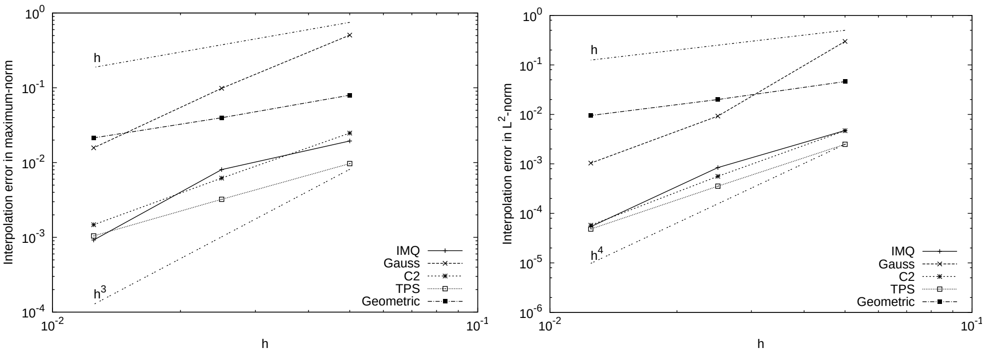
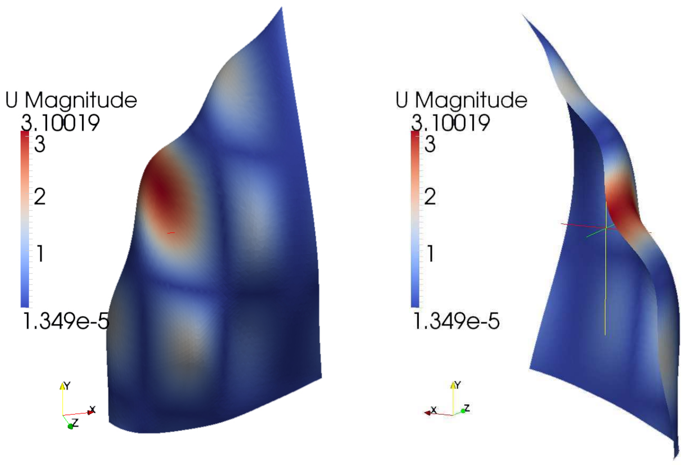


Figure 13: Interpolation error in L^2-norm (left) and maximum-norm (right) for different grid sizes.



(a) Front view.

(b) Side view.

Figure 14: Deformation of sail geometry: deformation field $\mathbf{d} = (0, 0, 0.02 \sin(\pi x/5) \cos(\pi y/10) xy)$.

very accurate without the need of tuning the shape factor parameter.

6.2 RBF-based mesh motion in FSI problems

As mentioned before, the RBF technique in FSI problems can be used to accomplish both the surface interpolation over the interface and the mesh motion. For

Type	r^2	Poly	cond(R_{CC})	L^2 error	L^∞ error
IMQ	0.05	no	5.010	0.0186	0.0824
IMQ	0.5	no	609.1	0.0038	0.0064
IMQ	1	no	2665	0.0034	0.0054
IMQ	10	no	6.0e10	1e12	1e13
IMQ	0.05	yes	539.53	0.0182	0.0821
IMQ	0.5	yes	2.69e4	0.00384	0.00633
IMQ	1	yes	1.36e5	0.00335	0.00547
IMQ	10	yes	7.8e12	NaN	NaN
TPS	0.01	no	2.53e5	0.003624	0.00565
TPS	1	no	1.23e5	0.00366	0.00568
TPS	100	no	3.23e4	0.003545	0.00556
TPS	0.01	yes	8.7e14	0.00355	0.00556
TPS	1	yes	6.11e5	0.00355	0.00556
TPS	100	yes	9.19e4	0.00355	0.00556

Table 1: RBF interpolation on a sail geometry. NaN means that no data are available because the error was so large that lead to a floating exception error during the execution of the code.

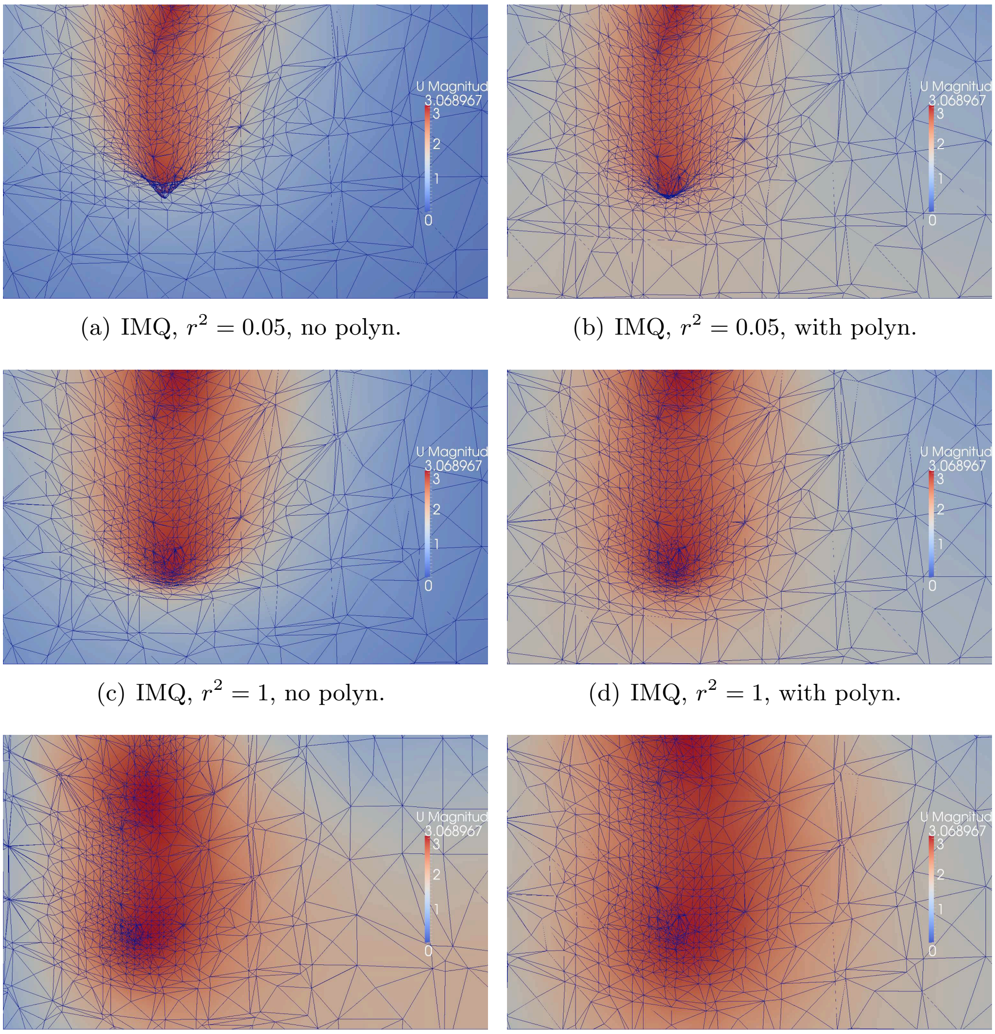
the sail test case, we have deformed the grid accordingly to the imposed displacement $\mathbf{d}$ using the different RBF bases. In Table 2, we report some statistics on the mesh quality of the fluid volume mesh after such deformation. In particular, for each case we report the number of cells that have become invalid (because of negative volume, wrong faces orientation, flipped cells) after the mesh motion, as well as the number of low quality cells. The mesh quality is measured in terms of orthogonality factor, that is the maximum angle between the vector connecting the centers of two contiguous cells and the normal of the face shared by the two cells. The low quality cells are those with an orthogonality factor larger than 75.

The original mesh has no degenerated cells and the highest orthogonality factor is 62.8.

Deforming the mesh with RBF for this kind of problems is quite challenging since all control points are located on a two-dimensional surface at the middle of the domain and such motion should be absorbed by a portion of the domain as large as possible. When using the IMQ basis, the grid quality is improved when the radius is increased. Indeed, if the basis functions have a too small support, the deformations is localized in a very small volume around the structure so that grid inversion may happen (Figure 15(a)). When the radius is increased, the area of influence of every RBF basis function becomes larger and the mesh motion becomes smoother and more robust (Figure 15(b)). The polynomial term has a beneficial effect since it accounts for the averaged rigid body translation motion which therefore does not need to be resolved by the radial basis term of the transformation (Figure 15(c-d)).

Type	r^2	Poly	# Inverted	# Low quality
IMQ	0.05	no	9904	4921
IMQ	0.5	no	262	764
IMQ	1	no	0	168
IMQ	10	no	79235	727435
IMQ	0.05	yes	2024	1680
IMQ	0.5	yes	0	26
IMQ	1	yes	0	0
IMQ	10	yes	NaN	NaN
TPS	0.01	no	0	0
TPS	1	no	0	0
TPS	100	no	0	0
TPS	0.01	yes	0	0
TPS	1	yes	0	0
TPS	100	yes	0	0

Table 2: RBF-based mesh motion on a sail geometry. NaN means that no data are available because the error was so large that lead to a floating exception error during the execution of the code.



(a) IMQ, $r^2 = 0.05$, no polyn.

(b) IMQ, $r^2 = 0.05$, with polyn.

(c) IMQ, $r^2 = 1$, no polyn.

(d) IMQ, $r^2 = 1$, with polyn.

(e) TP, $r = 1$, no polyn.

(f) TP, $r = 1$, with polyn.

Figure 15: A detail of an horizontal section (height=20 $[m]$) of the deformed volume mesh near the sail for different RBF bases.

The volume mesh deformation obtained with the TPS basis produce very good quality meshes for almost any value of r (Figure 15(e-f)). When polynomials are used, the shape factor r seems to have no effect. On the other hand, without the linear polynomial term, the effect of r is observed mainly on the far field (where x^2 is large) and for small values of r and results become slightly better with higher values of r.

It is worth to notice that, for the volume mesh deformation far away from the control points, the actual type of RBF basis used has an important impact on the spatial distribution of the deformation. The IMQ basis has a bell-like shapes and decays to zero far from the control points (Figure 16(a-c)). The

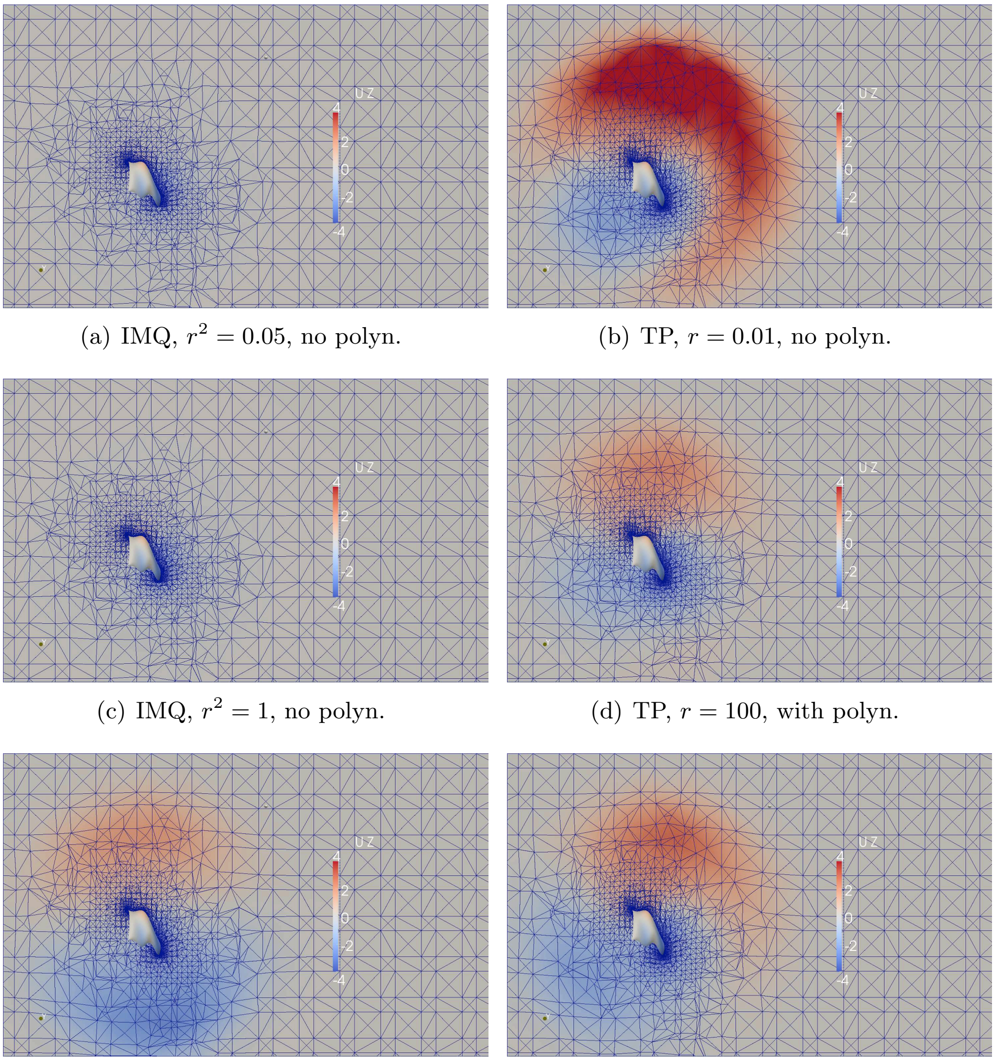
TPS basis instead goes to infinity when the distance increases. In theory, the IMQ basis is thus preferable because a mesh motion that increases the farther a point is from the moving surface is clearly not optimal. In practice, as discussed in section 5.3 it is often enough to multiply the computed deformation by a suitable cut-off function. In Figure 16(b-d-e-f) it is clearly visible how the mesh deformation becomes actually important when moving away from the sail, before getting smoothed out by the cut-off function. In Figure 16(d-f), we can once again observe that the variation of the parameter r has no noticeable effect when using the TPS basis with polynomial term. Finally, when adding the polynomials to the IMQ basis, Figure 16(e), the deformation increases far away from the sail as well. This is due to the fact that the average motion recognized is proportional to the longitudinal direction, and thus when going farther away from the sail in such direction, the linear term becomes large. Once again this is not a problem if the cut-off function is chosen appropriately.

Remark 5 *In this paper, we have considered an absolute approach for the RBF interpolation/mesh motion: the deformation is always referred to the initial configuration. This choice allows us to evaluate/factorize the RBF matrices only once at the beginning of the simulations and thus save much computational effort. Furthermore, this approach ensures that the mesh motion is reversible, in the sense that if a cyclic motion is imposed, the mesh always goes through the same configurations. The alternative strategy would have been to use an incremental approach and thus after every motion consider as control points position the new position of the structural points. This method has better stability properties and produce valid mesh motion even for global large mesh deformations (since they would be subdivided in many small steps). The drawback is that the computational cost is much high since, at every time step, the RBF matrices have to be re-evaluated and a linear problem has to be solved.*

6.3 Energy conservation

As shown in section 5.2, the interpolation scheme has been designed to conserve energy at the interface. In case of finite-element discretization for both fluid and structure, or whatever other discretization approach where the variables are located at the mesh nodes, the proposed scheme guarantees exactly the energy balance at the interface. In case of finite-volume/finite-element coupling, the energy is not exactly conserved due to the possible face/point mismatch in equation (35). However, we can show that the proposed method features better conservation properties when compared to standard interpolation approaches.

A simple test on the unit square has been designed to quantify the energy balance at the interface obtained with the proposed method and compare it with a standard interpolation, based on equation (22), where the interpolation matrix used to transfer the fluid stress from the fluid face centers to the structural faces, has been computed using the geometrical interpolation. The displacement



(a) IMQ, $r^2 = 0.05$, no polyn.

(b) TP, $r = 0.01$, no polyn.

(c) IMQ, $r^2 = 1$, no polyn.

(d) TP, $r = 100$, with polyn.

(e) IMQ, $r^2 = 1$, with polyn.

(f) TP, $r = 1$, with polyn.

Figure 16: A global vision of an horizontal section (height=20 $[m]$) of the deformed volume mesh for different RBF bases.

$\mathbf{d}(x, y) = (0, 0, 1e - 4\sin(2\pi x)\cos(3\pi y) + e^{xy})$ is interpolated from the structure mesh to the fluid mesh using the TPS basis and the pressure field $p(x, y) = xy^2 + y\cos(5\pi x)$ is transferred back from the fluid to the structure using relation (25). The difference between the work done by the structure and the one done by the fluid for the three mesh resolutions considered in the previous section is displayed in Figure 17. We can see that, by adopting the projection (25) for the stress transfer, an order of magnitude in the energy conservation is gained. Moreover, a second order convergence rate is obtained for both projections.

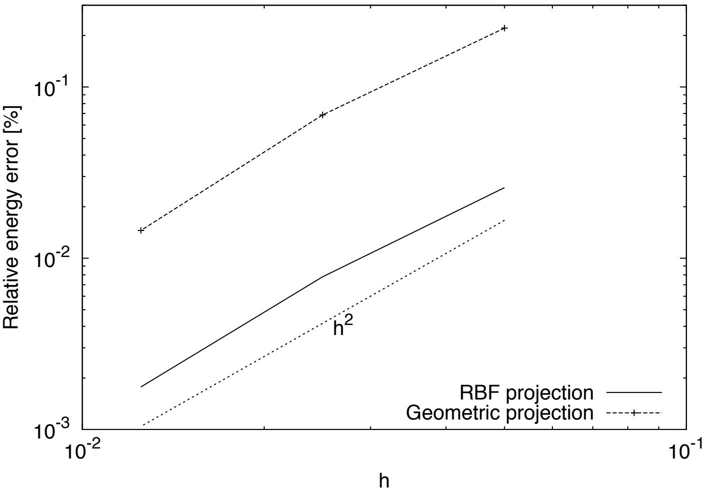


Figure 17: Relative error on energy balance at the interface.

7 Implementation issues and parallel scalability

The RBF interpolation technique, as it has been conceived in this work, can be easily adapted to parallel simulations. In particular, it has been implemented in the open-source library OpenFOAM: here, the structure grid is read by the master node which builds and factorizes the RBF matrix R_{CC} (via LU decomposition) once for all at the beginning of the simulation. This procedure could be further improved and made parallel itself, however it is not too much penalizing in terms of computational efficiency as long as the number of control points is not too large. In Table 3 we report the CPU time required to factorize and invert the RBF matrix for two different meshes, with a number of control points of 1925 and 2850, respectively. The computations were performed on a Intel Xeon E5506 (2.13 GHz) processor. We compared the performance obtained with the LU factorization supplied in the OpenFOAM release, with two new implementations based on the Boost library [16] and the optimized ATLAS library [26]. The Boost solution outperforms the standard OpenFOAM algorithm by a factor close to 2 for the smaller matrix and 1.22 for the bigger matrix. The ATLAS implementation is even faster, with a speed-up factor of about 11 in the first case and about 8 in the second one.

At run time, only two matrix-vector multiplications are required to interpolate the data from the structure to the fluid. To transfer the displacement of the FSI interface predicted by the structural solver to the fluid interface points every partition k should create its own $R_{I_k,C}$ matrix. This process is easily parallelizable since every partition will move just its own points. At the partition interfaces, the nodes are duplicated and, whatever mesh motion technique is used, special care has to be adopted to ensure that the corresponding points in two different partitions are moved exactly in the same way. In our case, however, this condition is automatically satisfied, since the RBF map and control points displacements are the same on every processor and so will be the resulting mapping. If the domain has been partitioned in a way such that the fluid FSI interface has been split evenly over the whole sub-domains, the scalability factor is optimal (less the duplicated sub-domain-interface points). If the RBF mapping is used also to accomplish the mesh motion of the fluid mesh, the same principle described before is applied: every processor will move only its own points and conformity of mesh motion at processor-interface is guaranteed. Furthermore, usually the domain decomposition is done in such a way that every partition has the same number of cells/points, thus very good scalability can be achieved. If the RBF map is used also to transfer the fluid stress to the structural solver in a conservative way through equation (25), this process has well can be easily parallelized. As described in the previous section, the fluid stress are defined on every face of the FSI interface. When the domain is partitioned, every processor will have just a part of the whole interface, but no duplication of faces will happen (only shared points are duplicated, not faces). In order to obtain the total stress to be transferred to the structure, it will be sufficient to evaluate the contribution of the local faces on the structures and then sum them up over every processor:

$$M_S \mathbf{\Sigma}_{S_n} = H^T_{F_f S_n} M_F \mathbf{\Sigma}_{F_f} = \sum_{k=1}^{\#procs} H^T_{F_{f_k} S_n} M_{F_k} \mathbf{\Sigma}_{F_{f_k}} \,. \tag{36}$$

To assess the parallel performances of the RBF implementation, a scalability test has been carried out evaluating equation (36) on different numbers of processors. In this test case, we have used 2846 control points and the the fluid mesh consisted of 79850 nodes. The results are reported in Table 4.

The scalability is almost perfect. The small degradation for 4-8 cores is due to the small size of the matrices involved, With these implementation tricks, the cost of the RBF mapping has been considerably reduced allowing it to be successfully adopted in many applications. In particular, some recent results of large scale wind/sail FSI simulations have been presented in [19]. For example in a transient FSI simulation of a sail, with a number of control points of the order of 3-5 thousands and a volume mesh of about 200k-300k elements per partitions, the RBF mesh motion has been used very successfully, resulting usually faster

Method	OpenFOAM	Boost	ATLAS
N=1925	34.54	19.20	3.08
N=2850	73.72	60.11	9.29

Table 3: Time, expressed in seconds, required to LU factorize and compute the inverse of a matrix with three different libraries.

# proc.	1	2	4	8
Time (s.)	0.88	0.44	0.24	0.13

Table 4: Time, expressed in seconds, required to perform the evaluation of equation (36), i.e. two matrix-vector multiplication, for different number of partitions.

than a classical harmonic-extension mesh motion solver and preserving more efficiently the quality of the mesh. If the size of the problem is further increased, mesh motion can be improved in terms of computational efficiency by resorting to interpolation techniques, such as the Inverse Distance Weighting [27], which do not require the solution of any linear system (see also [19]).

8 Conclusions

In this work, the problem of information transfer between different domains has been object of investigation, in particular in the framework of fluid-structure interaction problems. In the first section, a formal analysis of the coupling between FEM-FEM discretizations has been carried out, focusing both on the interpolation of data and the energy conservation property at the interface when non-conforming grids are used.

Then, we have considered the common scenario in which a finite-volume fluid solver is coupled with a finite-element structural solver, so that the interpolation at the interface should account also for different spatial discretizations and different arrangements of the degrees-of-freedom over the grids. The same conclusion on the way how to achieve conservation of energy at the interface found on the FEM-FEM case is obtained via an algebraic argument.

In the second part of the work, we have investigated different aspects regarding the adoption of the RBF technique in fluid-structure interaction problems. In particular, we have analyzed the interpolation properties of different RBF bases, verifying their behavior in terms of interpolation error and sensitivity on the shape factor r through different test cases.

In this framework, we have analyzed the energy conservation of RBF-based interpolation methods presented and assessed numerically the error. A comparison with a more classical geometrical interpolation scheme has shown that the proposed method greatly improves the energy conservation error.

The adoption of the RBF technique for the mesh motion required in the ALE approach has also been explored and, again, the behavior of different RBF bases was compared. An adequate implementation of the technique in a parallel environment showed that this technique can be successfully used for FSI simulation even when the dimension of the problem is relatively large. The results of our investigation show that the RBF technique can be successfully adopted in fluid-structure interaction problems.

References

[1] A. Beckert and H. Wedland. Multivariate interpolation for fluid-structure-interaction problem using radial basis functions. *Aerosp. Sci. Technol.*, pages pp. 125–134, 2001.

[2] F.M. Bos. *Numerical simulations of flapping foil and wing aerodynamics: Mesh deformation using radial basis functions.* PhD thesis, T.U. Delft, 2010.

[3] M. D. Buhmann. Radial basis functions. *Acta Numerica*, 9:1–38, 2000.

[4] M. D. Buhmann. *Radial Basis Functions: Theory and Implementations.* Cambridge University Press, 2003.

[5] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3d objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 67–76, New York, NY, USA, 2001. ACM.

[6] P. Causin, J. F. Gerbeau, and F. Nobile. Added-mass effect in the design of partitioned algorithms for fluid-structure problems. *Computer Methods in Applied Mechanical Engineering*, 194:pp. 4506–4527, 2005.

[7] Paolo Crosetto, Simone Deparis, Gilles Fourestey, and Alfio Quarteroni. Parallel Algorithms for Fluid-Structure Interaction Problems in Haemodynamics. *Siam Journal on Scientific Computing*, 33(4):1598–1622, 2011.

[8] Paolo Crosetto, Philippe Reymond, Simone Deparis, Dimitrios Kontaxakis, Nikolaos Stergiopulos, and Alfio Quarteroni. Fluid-structure interaction simulation of aortic blood flow. *Computers & Fluids*, 43(1):46 – 57, 2011.

[9] A. de Boer, M.S. van der Schoot, and H. Bijl. Mesh deformation based on radial basis function interpolation. *Computers & Structures*, 85(11-14):784 – 795, 2007.

[10] A. de Boer, A. H. van Zuijlen, and H. Bijl. Review of coupling methods for non-marching meshes. *Computer Methods in Applied Mechanics and Engineering*, 196:pp. 1515–1525, 2007.

[11] J. Donea, S. Giuliani, and J.P. Halleux. An arbitrary Lagrangian-eulerian finite element method for transient dynamic fluid-structure interactions. *Computer Methods in Applied Mechanics and Engineering*, 33:pp. 689–723, 1982.

[12] C. A. Felippa, K.C. Park, and C. Farhat. Partitioned analysis of coupled mechanical systems. *Computer Methods in Applied Mechanics and Engineering*, 190(24-25):3247 – 3270, 2001.

[13] M. Heil, A. Hazel, and J. Boyle. Solvers for large-displacement fluid–structure interaction problems: segregated versus monolithic approaches. *Computational Mechanics*, 43:91–101, 2008. 10.1007/s00466-008-0270-6.

[14] T. J. R. Hughes, W. K. Liu, and T. K. Zimmermann. Lagrangian-Eulerian finite element formulation for incompressible viscous flows. *Computer Methods in Applied Mechanics and Engineering*, 29(3):pp. 329–349, 1981.

[15] H. Jasak. *Error analysis and estimation for the finite volume method with applications to fluid flows.* PhD thesis, Imperial College (London), 1995.

[16] B. Karlsson. *Beyond the C++ Standard Library: An Introduction to Boost.* Addison Wesley Professional, August 2005.

[17] Ulrich Küttler and W. Wall. Fixed-point fluid–structure interaction solvers with dynamic relaxation. *Computational Mechanics*, 43:61–72, 2008. 10.1007/s00466-008-0255-5.

[18] M. Lesoinne and C. Farhat. Geometric conservation laws for flow problems with moving boundaries and deformable meshes, and their impact on aeroelastic computations. *Computer Methods in Applied Mechanics and Engineering*, 134:pp. 71–90, 1996.

[19] M. Lombardi. *Numerical simulation of a sailing boat: free surface, fluid structure interaction and shape optimization.* PhD thesis, EPFL, 2012.

[20] M. Lombardi, M. Cremonesi, A. Giampieri, N. Parolini, and A. Quarteroni. A strongly coupled fluid-structure interaction model for wind-sail simulation. In *4th High Performance Yacht Design Conference*, Auckland, New Zealand, 2012. Submitted.

[21] F. Nobile. *Numerical approximation of fluid-structure interaction problems with application to haemodynamics.* PhD thesis, EPFL, 2001.

[22] S. Piperno, C. Farhat, and B. Larrouturou. Partitioned procedures for the transient solution of coupled aroelastic problems part i: Model problem, theory and two-dimeosional application. *Computer Methods in Applied Mechanical Engineering*, 124:pp. 79–112, 1995.

[23] A. Quarteroni and A. Valli. *Domain Decomposition Methods for Partial Differential Equations.* Clarendon Press, 1999.

[24] Alfio Quarteroni and Luca Formaggia. Mathematical modelling and numerical simulation of the cardiovascular system. In N. Ayache, editor, *Computational Models for the Human Body*, volume 12 of *Handbook of Numerical Analysis*, pages 3 – 127. Elsevier, 2004.

[25] G. Turk and J. F. O'Brien. Modelling with implicit surfaces that interpolate. *ACM Transactions on Graphics*, 21:855–873, 2002.

[26] R. C. Whaley, A. Petitet, and J. J. Dongarra. Automated empirical optimization of software and the ATLAS project. *Parallel Computing*, 27(1–2):3–35, 2001.

[27] J. Witteveen. Explicit mesh deformation using inverse distance weighting interpolation. In *Proceedings of 47th AIAA Aerospace Sciences Meeting*, 2009.

MOX Technical Reports, last issues

Dipartimento di Matematica "F. Brioschi",
Politecnico di Milano, Via Bonardi 9 - 20133 Milano (Italy)

40/2012 Lombardi, M.; Parolini, N.; Quarteroni, A.
Radial basis functions for inter-grid interpolation and mesh motion in FSI problems

39/2012 Ieva, F.; Paganoni, A.M.; Ziller, S.
Operational risk management: a statistical perspective

38/2012 Antonietti, P.F.; Bigoni, N.; Verani, M.
Mimetic finite difference approximation of quasilinear elliptic problems

37/2012 Nobile, F.; Pozzoli, M.; Vergara, C.
Exact and inexact partitioned algorithms for fluid-structure interaction problems with finite elasticity in haemodynamics

36/2012 Canuto, C.; Verani, M.
On the Numerical Analysis of Adaptive Spectral/hp Methods for Elliptic Problems

35/2012 Pigoli, D.; Aston, J.A.D.; Dryden, I.L.; Secchi, P.
Distances and Inference for Covariance Functions

34/2012 Menafoglio, A.; Dalla Rosa, M.; Secchi, P.
A Universal Kriging predictor for spatially dependent functional data of a Hilbert Space

33/2012 Motamed, M.; Nobile, F.; Tempone, R.
Analysis and computation of the elastic wave equation with random coefficients

32/2012 Formaggia, L.; Fumagalli, A.; Scotti A.; Ruffo, P
A reduced model for Darcy s problem in networks of fractures

31/2012 Bonizzoni, F.; Buffa, A; Nobile, F:
Moment equations for the mixed formulation of the Hodge Laplacian with stochastic data