Numerical analysis of thermo-fluid problems in 3D domains by means of the RBF-FD meshless method

Numerical analysis of thermo-fluid problems in 3D domains by means of the RBF-FD meshless method

Riccardo Zamolo, Davide Miotti and Enrico Nobile

Dipartimento di Ingegneria e Architettura, Università degli Studi di Trieste, via A. Valerio 10, 34127 Trieste, Italy

E-mail: rzamolo@units.it

Abstract. The use of CAE (Computer Aided Engineering) software, commonly applied to the design and verification of a great variety of manufactured products, is totally reliant on accurate numerical simulations. Classic mesh-based methods, e.g., Finite Element (FEM) and Finite Volume (FVM), are usually employed for such simulations, where the role of the mesh is crucial for both accuracy and time consumption issues. This is especially true for complex 3D domains which are typically encountered in most practical problems. Meshless, or meshfree, methods have been recently introduced in order to replace the usual mesh with much simpler node distributions, thus purifying the data structures of any additional geometric information. Radial Basis Function-Finite Difference (RBF-FD) meshless methods have been shown to be able to easily solve problems of engineering relevance over complex-shaped domains with great accuracy, with particular reference to fluid flow and heat transfer problems. In this paper the RBF-FD method is employed to solve heat transfer problems with incompressible, steady-state laminar flow over 3D complex-shaped domains. The required node distributions are automatically generated by using a meshless node generation algorithm, which has been specifically developed to produce high quality node arrangements over arbitrary 3D geometries. The presented strategy represents therefore a fully-meshless approach for the accurate and automatic simulation of thermo-fluid problems over 3D domains of practical interest.

1. Introduction

Traditional CAE (Computer Aided Engineering) software usually rely on classic mesh-based methods, e.g., Finite Element (FEM) [1] and Finite Volume (FVM)[2] for solving boundary value problems over generic 3D domains. These methods are based on well understood mathematical models, which have proved to be reliable and accurate in the hands of adequately experienced users. With time, their widespread adoption has brought to the surface a number of inherent limitations, for example the need for processing and storage of the so called connectivity information. The main consequences are [3]:

- the cost of the mesh creation and the size of the consequent data structure,
- the inability to allow large geometric deformation of the domain.

The first one refers to both computational and economical costs; indeed the generation of a mesh often require the intervention of an experienced human worker, and operator costs now outweigh those of CPU time for the computer [3]. Such an operation becomes necessary for instance when high level of accuracy is demanded and therefore some local refinement of the mesh is needed at certain spots of the domain. The second weakness becomes of interest in

many practical cases, for example when the solver is paired with some optimization algorithm and the geometry is deformed multiple times.

For those reasons the so-called meshless or meshfree methods have recently been proposed as an alternative to the mesh-based ones for the solution of partial differential equations over complex-shaped domains [3–6]. The common characteristic is the use of the so-called field nodes: at first a set of nodes is scattered over the domain, then shape functions are constructed for each internal node.

In this paper an implementation of the Radial Basis Function-generated Finite Difference (RBF-FD) meshless method [4, 7–11] is used to solve thermo-fluid problems over 3D geometries. The main advantages of this method is that no connectivity information is stored or processed, this implies the overcoming of all the limitations of the mesh-based methods enumerated so far. Furthermore, the employed node generation process lends itself to being automatically executed in parallel and therefore is particularly suitable for High Performance Computing applications. The employed code is developed using Julia programming language [12], which allows extensive code reuse and excellent computational performance already at the development stage.

In order to illustrate the numerical properties of the proposed method and its usability in the field of CFD simulations, several computations for a natural convection problem over a complex 3D geometry at a Rayleigh number $\mathrm{Ra}_D = 10^3$ will be presented.

2. Governing equations

Let us consider an incompressible fluid with density ρ, kinematic viscosity ν, thermal diffusivity α, thermal conductivity k and volumetric temperature expansion coefficient β. The resulting nondimentional conservation equations of mass, momentum and energy are

$$\nabla \cdot \mathbf{u} = 0, \tag{1}$$

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \sqrt{\frac{\mathrm{Pr}}{\mathrm{Ra}_D}}\nabla^2 \mathbf{u} + \mathbf{k}T, \tag{2}$$

$$\frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T = \frac{1}{\sqrt{\mathrm{Pr} \cdot \mathrm{Ra}_D}}\nabla^2 T, \tag{3}$$

where $\mathbf{k}$ is the unit vector along the vertical direction z. In the above equations, length, velocity $\mathbf{u} = (u, v, w)$, time t, pressure p and temperature T are made dimensionless by taking respectively D, $U_0 = \sqrt{g\beta D \Delta T}$, D/U_0, ρU_0^2 and ΔT as reference quantities, where g is the gravitational acceleration. $\mathrm{Pr} = \nu/\alpha$ is the Prandtl number and $\mathrm{Ra}_D = \mathrm{Pr} \cdot g\beta D^3 \Delta T/\nu^2$ is the Rayleigh number. The chosen values for the presented computations are $\mathrm{Pr} = 0.71$ and $\mathrm{Ra}_D = 10^3$, leading to a steady-state solution with the presented geometry.

3. Numerical method

In order provide a better explanation of the RBF-FD meshless method, we can describe it as the succesive application of four steps:

(i) generation of node distributions,
(ii) RBF interpolation or function approximation,
(iii) collocation technique,
(iv) solution procedure.

Each of the aforementioned steps will be briefly presented as follows. Interested readers might find a more detailed discussion on the very same method in [13].

3.1. Node distributions

Node distributions are represented by a certain number N of 3D points fulfilling a prescribed spacing function s and which are required to be scattered within the domain Ω and on its boundary $\partial\Omega$, where the latter is assumed to be concretely available in the form of a `.stl` (stereolithography) file. This process can be further divided in two stages:

- generation of a volumetric node distribution within Ω (nodes are not placed on $\partial\Omega$ yet) that satisfies a certain spacing s on average,
- iterative refinement of the initial node distribution through a node-repel approach that provides a suitable node distribution also on $\partial\Omega$.

In the second step the nodes are moved according to mutual repulsion forces that arise between neighbors. Whenever one of them is pushed outside Ω, it is projected onto the nearest point on the boundary $\partial\Omega$. This projection operation is efficiently performed by exploiting an octree data structure [14, 15] for the boundary triangles of the stereolithography surface.

3.2. RBF Interpolation

The value of a generic function ζ at any point is approximated applying the Radial Point Interpolation Method (RPIM) using Radial Basis Functions (RBFs) augmented with polynomial terms [3]. This means that the value $\zeta(x)$ taken by such function at the point $x \in \Omega$ can be locally approximated by the quantity $\zeta^h(x)$, defined as follows:

$$\zeta(x) \approx \zeta^h(x) := \sum_{i=1}^{\bar{n}} \alpha_i \varphi(\|x - x_i\|_2) + \sum_{j=1}^{m} \beta_j p_j(x), \tag{4}$$

where:

- $\{\varphi(\|x - x_i\|_2) : i = 1, \ldots, \bar{n}\}$ is a set of $\bar{n}$ RBFs, each of which is associated to the corresponding node x_i. The set of the considered nodes is local, i.e. $x_1, \ldots, x_{\bar{n}}$ are the $\bar{n}$ nearest nodes to x. For brevity of notation, $\varphi(x, x_i)$ will be used instead of $\varphi(\|x - x_i\|_2)$.
- the multiquadratic Radial Basis Functions [16] are used, defined as $\varphi(r) := \sqrt{1 + (\varepsilon r)^2}$
- $\{p_j : j = 1, \ldots, m\}$ is a complete polynomial basis of degree P.

The calculation of the weights $\{\alpha_1, \ldots, \alpha_{\bar{n}}\}$ and $\{\beta_1, \ldots, \beta_m\}$ depends on the position of the stencil, that is, if some nodes of the stencil belongs to the boundary $\partial\Omega$, then boundary conditions must be taken into account. When none of the nodes belonging to the stencil lies on the boundary $\partial\Omega$, then the weights are estimated assuming that the values of the function u at the nodes of the stencil are known, this translates into a linear system of $\bar{n}$ equations of the form:

$$\zeta^h(x_i) = \zeta(x_i), \quad i = 1, \ldots, \bar{n} \tag{5}$$

where the definition of $\zeta^h(x)$ can be substituted at each row. In order to make the system solvable, other m equations are attained by enforcing the following additional orthogonality conditions:

$$\sum_{i=1}^{\bar{n}} \alpha_i p_j(x_i) = 0, \quad j = 1, \ldots, m \tag{6}$$

On the other hand, when some boundary nodes are included in the stencil, instead of assuming the true value of the function ζ in the corresponding equation, the appropriate boundary condition are imposed. In the case of Robin b.c. (i.e. $a\zeta + b\,\partial\zeta/\partial n = g$) applied to the i^{th} node of the stencil, the corresponding equation becomes:

$$a(x_i)\,\zeta^h(x_i) + b(x_i)\,\frac{\partial \zeta^h}{\partial n}(x_i) = g(x_i), \quad x_i \in \partial\Omega \tag{7}$$

where a, b, g are all known functions defined on the boundary. By substituting the definition of ζ^h given in Equation (4) and reordering the terms in order to isolate the unknown coefficients α's and β's, we get:

$$\sum_{k=1}^{\bar{n}} \alpha_k \left(a(x_i)\varphi(x_i, x_k) + b(x_i)\frac{\partial \varphi}{\partial n}(x_i, x_k) \right) + \sum_{j=1}^{m} \beta_j \left(a(x_i)p_j(x_i) + b(x_i)\frac{\partial p_j}{\partial n}(x_i) \right) = g(x_i) \quad (8)$$

We can now enforce the orthogonality conditions of Equation (6) and thus obtain a linear system of the form:

$$\underbrace{\begin{bmatrix} \boldsymbol{\Phi_{BC}} & \boldsymbol{P_{BC}} \\ \boldsymbol{P^T} & \boldsymbol{0} \end{bmatrix}}_{\boldsymbol{M}} \begin{Bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{\zeta} \\ \boldsymbol{g} \\ \boldsymbol{0} \end{Bmatrix} \quad (9)$$

where the block matrix $\boldsymbol{P}^T$ comes from Equation (6), $\{\alpha_1, \ldots, \alpha_{\bar{n}}, \beta_1, \ldots, \beta_m\}$ is the vector of unknowns and both $\boldsymbol{\Phi_{BC}}$ and $\boldsymbol{P_{BC}}$ come from Equation (4) but have boundary conditions (8) enforced at the appropriate rows.

3.3. Collocation Techinque

Given a linear partial differential equation $\mathcal{L}(\zeta) = f$ in the unknown field ζ, the RBF expansion (4) is made valid at each node x_i which does not lie on the boundary. The following sparse linear system is obtained:

$$\left[\boldsymbol{C}\right] \left\{\boldsymbol{\zeta}^h\right\} = \boldsymbol{q} - \boldsymbol{f} \quad (10)$$

where $\boldsymbol{\zeta}^h = \{\zeta^h(x_1), \ldots, \zeta^h(x_{N_I})\}$ is the vector of unknown scalar values $\zeta^h(x_i)$ evaluated at all N_I inner nodes, i.e. those contained in Ω, while the vector $\boldsymbol{q}$ comes from the enforcement of boundary conditions at the interpolant level.

3.4. Solution Procedure

At each time step, the computation of velocity, pressure and temperature through Equations (1)-(3) is decoupled using a projection scheme with a three-level Gear scheme for the time discretization while the buoyancy term is modeled with the Boussinesq approximation. A tentative velocity $\mathbf{u}^*$ is computed from the linearized nondimensional momentum equation:

$$\frac{3\mathbf{u}^* - 4\mathbf{u}^l + \mathbf{u}^{l-1}}{2\Delta t} + \mathbf{u}^l \nabla \mathbf{u}^* = -\nabla p^l + \sqrt{\frac{\mathrm{Pr}}{\mathrm{Ra}_D}} \nabla^2 \mathbf{u}^* + \mathbf{k} T^l, \quad (11)$$

where l is the time level and Δt is the nondimensional time step size.

The tentative velocity $\mathbf{u}^*$ is then forced to satisfy the continuity equation (1) by means of an irrotational correction $\mathbf{u}^{l+1} = \mathbf{u}^* - \nabla\Phi$, leading to the Poisson equation $\nabla^2\Phi = \nabla \cdot \mathbf{u}^*$ in the auxiliary variable Φ. The pressure is then updated as $p^{l+1} = p^l + \Phi/\Delta t$ and the temperature is computed from the discretized nondimensional energy equation:

$$\frac{3T^{l+1} - 4T^l + T^{l-1}}{2\Delta t} + \mathbf{u}^{l+1} \nabla T^{l+1} = \frac{1}{\sqrt{\mathrm{Pr} \cdot \mathrm{Ra}_D}} \nabla^2 T^{l+1}. \quad (12)$$

The previous equations are discretized in space with the same RBF-FD scheme presented in Section 3.2. Sparse linear systems like the one in Equation (10) are attained for each component of velocity and for temperature.

In the current implementation such linear systems are preconditioned with an Incomplete LU factorization (ILU) [17] (package `IncompleteLU` in `Julia`) of the sparse matrix $\boldsymbol{C}$ whenever

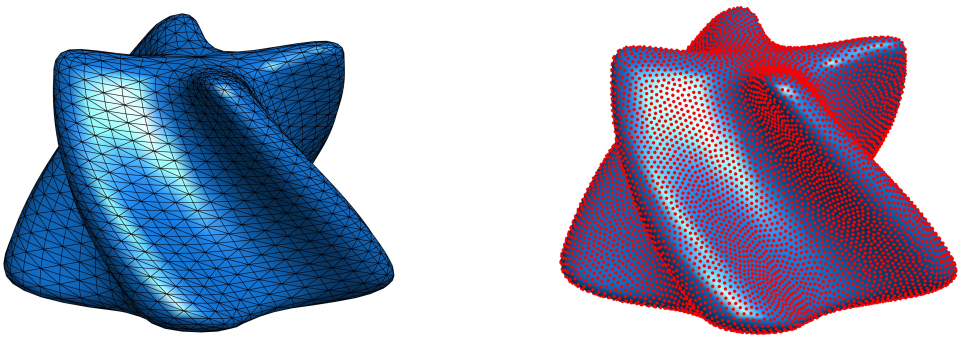

Figure 1. `.stl` surface of the 3D object (left) and distribution of the boundary nodes on the same object (right).

is needed. The Poisson equation for the velocity correction is also preconditioned through an ILU factorization which can be performed only once at the beginning of the simulation. ILU preconditioning is then followed by the application of the Biconjugate Gradient Stabilized Method [18] (package `IterativeSolvers` in `Julia`) using a relative error of 10^{-10}. The computational time required for each time step ranges from 2 seconds for $N \approx 100,000$ nodes and $P = 2$ to 18 seconds for $N \approx 500,000$ nodes and $P = 4$ on a laptop equipped with a quad-core Intel i7-6700HQ 2.6GHz processor with 16 GB of RAM memory.

3.5. Convergence Criterion

The convergence to steady-state is declared when the RMS time derivative for the computed velocity fields $\mathbf{u} = \{u, v, w\}$ and temperature field T becomes smaller than $5 \cdot 10^{-5}$, where the RMS time derivative $\dot{F}_{RMS}$ for a generic field F over the domain Ω is defined by:

$$\dot{F}_{RMS} = \sqrt{\frac{1}{\mu(\Omega)} \int_\Omega \left(\frac{\partial F}{\partial t} \right)^2 d\Omega} \approx \sqrt{\frac{1}{\sum_i s_i^3} \sum_i \left(\frac{F_i^{l+1} - F_i^l}{\Delta t} \right)^2 s_i^3} \tag{13}$$

where $\mu(\Omega)$ is the measure of the domain Ω and the index i runs over each internal node.

4. Geometry, boundary conditions and auxiliary computations

A complex-shaped 3D object with characteristic length D, represented in Figure 1, is enclosed in a cubic cavity with side length $W = 5D$. The 3D object is centered with respect to the cavity in the $x - y$ plane, while the distance between the bottom cavity wall and the lower surface of the object is D, as visible in Figure 4. The following boundary conditions are enforced:

- no-slip condition $\mathbf{u} = \mathbf{0}$ on all boundary surfaces;
- $T = 1$ on the surface of the 3D object;
- $T = 0$ on the inner walls of the cubic cavity;
- $\dfrac{\partial \Phi}{\partial \mathbf{n}} = 0$ on all boundary surfaces;

In order to ensure the uniqueness of the calculated auxiliary variable Φ, an additional condition is imposed in the form of a Lagrange multiplier. The complete problem for Φ assumes

therefore the following form:

$$\begin{cases} \nabla^2\Phi + \lambda = \nabla \cdot \mathbf{u}^* & in\ \Omega \\ \dfrac{\partial \Phi}{\partial \mathbf{n}} = 0 & on\ \partial\Omega \\ \displaystyle\int_\Omega \Phi = 0 \end{cases} \tag{14}$$

Since boundary layers are expected at the surface of the object, the following spacing function is employed to obtain refined node distributions in the proximity of such geometry:

$$s(\mathbf{x}) = s_{max} - \frac{s_{max} - s_{min,j}}{\frac{d(\mathbf{x})}{\lambda} + 1} \tag{15}$$

where s_{max} is the asymptotic maximum spacing and $s_{min,j}$ is a minimum spacing attained on each triangle of the original `.stl` surface. $d(\mathbf{x})$ is the distance of point $\mathbf{x}$ from the closest `.stl` triangle, denoted in Equation (15) with the index j, and λ is a parameter used for controlling the transition from $s_{min,j}$ to s_{max}.
In the presented case the value of s_{max} is chosen in order to attain a sought number of nodes, then s_{min} is chosen to be $s_{min} = 0.8 s_{max}$ on vertical and bottom walls of the domain, $s_{min} = 0.65 s_{max}$ on the top wall and $s_{min} = 0.1 s_{max}$ on the surface of the object, while $\lambda = 0.1$. An example of a boundary node distribution over the object is depicted in Figure 1 for a total number of nodes $N \approx 185,000$.

The mean Nusselt number over the surface Γ of the considered object is given by:

$$\overline{\mathrm{Nu}}_D = \frac{1}{\mu(\Gamma)} \int_\Gamma \frac{\partial T}{\partial \mathbf{n}} d\Gamma \approx \frac{1}{\sum_b s_b^2} \sum_b \frac{\partial T}{\partial \mathbf{n}}\Big|_b s_b^2 \tag{16}$$

where the index b runs over each boundary node on Γ.

5. Results

In order to highlight the convergence properties of the employed RBF-FD approach, several simulations are carried out for different node distributions using a total number of nodes ranging from $N \approx 100,000$ nodes to $N \approx 740,000$ nodes. Three polynomial degrees $P = 2, 3, 4$ are employed. Convergence curves for $P = 2$ and $P = 3$ are shown in Figure 2 in terms of normalized RMS errors for the flow variables $\mathbf{u} = \{u, v, w\}$, T and p. The normalized RMS error $err(F)$ of a generic field F is defined as:

$$err(F) = \sqrt{\frac{1}{\mu(\Omega)} \int_\Omega \left(\frac{F - \bar{F}}{\max(F)} \right)^2 d\Omega} \tag{17}$$

where the reference solution $\bar{F}$ is a highly accurate solution obtained with polynomial degree $P = 4$ on a fine node distribution with $N \approx 545,000$ nodes.

The convergence curves depicted in Figure 2 show that the resulting order of accuracy for $P = 2$ ranges from 2.1 to 2.5 for each flow variable, while for $P = 3$ the resulting order of accuracy ranges from 3.2 to 3.7, as expected. The orders of accuracy for larger polynomial degrees $P > 3$ are therefore expected to be greater than or equal to P for the present problem, although this assumption could not be verified due to the large memory requirements for $P \geq 4$ on 3D problems with the present RBF-FD implementation, which makes it unsuitable for a personal (laptop) computer.

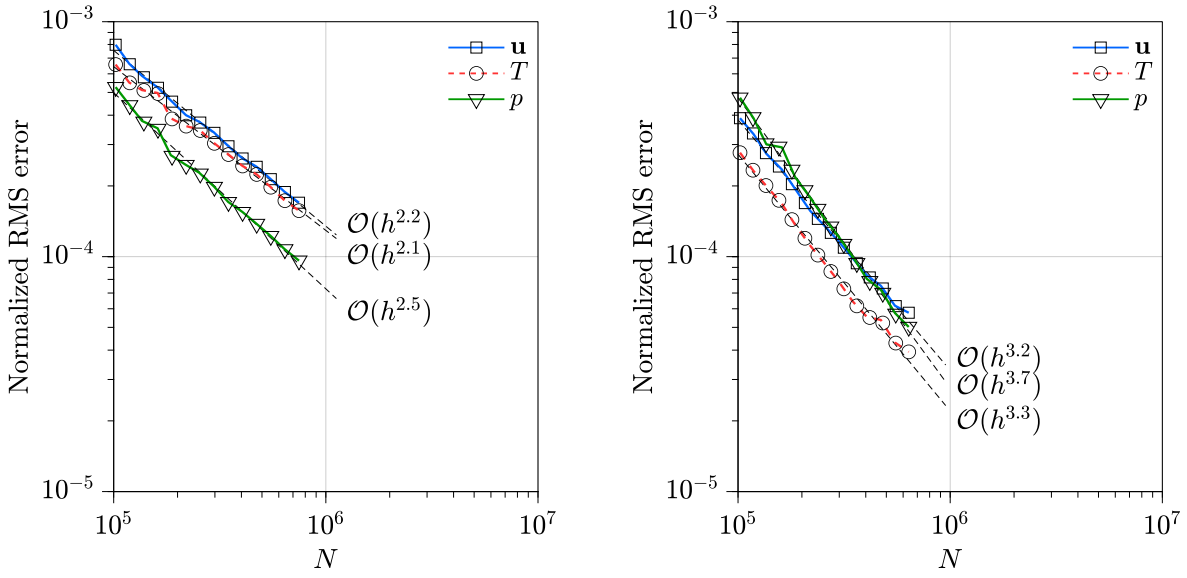

Figure 2. Convergence curves for the flow variables $\mathbf{u} = \{u, v, w\}$, T and p for $P = 2$ (left) and $P = 3$ (right).

Table 1. Mean Nusselt number.

	$\overline{\mathrm{Nu}}_D$			
N	$P = 2$	$P = 3$	$P = 4$	Fluent
100,000	3.85	3.91	3.87	-
116,666	3.88	3.92	3.89	4.36*
136,110	3.89	3.95	3.92	-
158,794	3.95	4.01	3.98	-
185,258	3.97	4.02	4.00	-
216,133	4.00	4.06	4.03	4.36*
252,154	4.03	4.08	4.06	-
294,178	4.05	4.10	4.08	-
343,206	4.08	4.13	4.12	-
400,404	4.10	4.15	4.14	-
467,136	4.12	4.17	4.16	4.37*
544,989	4.13	4.18	4.17	-
635,816	4.15	4.20	-	-
741,782	4.18	-	-	-

* using inflation layers with patch-conforming mesh from .stl model

The computed values of the mean Nusselt number $\overline{\mathrm{Nu}}_D$ for different number of nodes N and polynomial degrees $P = 2, 3, 4$ are reported in Table 1, where mean Nusselt numbers computed with Ansys Fluent 2020R1 are also reported for comparison. In the latter case inflation layers (Wed6 elements) via patch-conforming meshing are employed at the object surface. Despite the refined node distribution, i.e., reduced node spacing near the object surface, the $\overline{\mathrm{Nu}}_D$ values

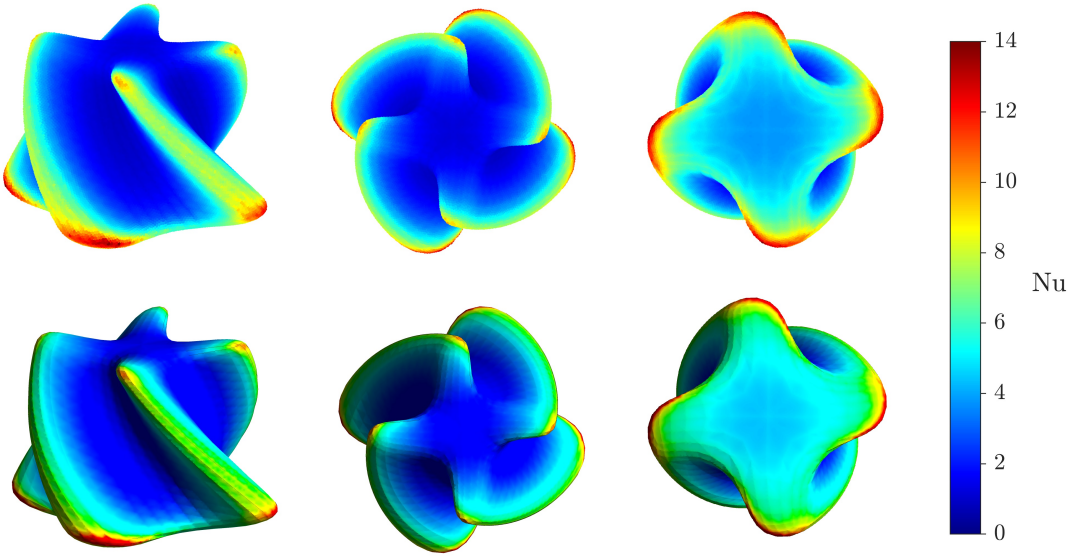

Figure 3. Visual comparison of contour plots of local Nusselt number at the object surface: RBF-FD (top row) vs. Ansys Fluent (bottom row).

show a slow convergence to the reference value $\overline{\mathrm{Nu}}_D = 4.37$, regardless of the polynomial degree P. This is mainly due to the fact that the generated node distributions are isotropic, i.e., node spacing is independent upon the spatial directions, therefore boundary layers are not efficiently solved as with inflation layers.

A graphical, qualitative comparison between RBF-FD ($P = 3$, $N \approx 640,000$ nodes) and Ansys Fluent ($470,000$ elements) is depicted in Figure 3 in terms of local Nusselt number at the object surface. The comparison highlights a very good agreement, to graphical accuracy, in the surface distribution of the local Nu number between the computed results and the reference ones. Figure 4 depicts the computed temperature field on the $x - z$ plane at $y = W/2 = 2.5$ together with the velocity vector plot projected on the same plane, highlighting the classic natural convection flow originated by the temperature differential between the object surface and the cavity walls.

At last, due to its relevance for practical applications, we report a comparison between the computational times required by the geometrical discretizations of the employed computational domain. The node generation for the RBF-FD meshless approach, as described in subsection 3.1, required 4s for the octree phase and an average of $5\mathrm{s}/10^5$ nodes for the iterative refinement phase, while the average time for the mesh generation in ANSYS Meshing was $6\mathrm{s}/10^5$ cells.

6. Conclusions

In this work the RBF-FD meshless method is applied to the simulation of heat transfer problems with incompressible laminar flow over a 3D complex-shaped domain.

The major strength of the proposed method lies in its potential convenience for the end user. The node generation algorithm is capable of coping with very complex geometries without any need for human intervention, furthermore, by choosing the order of the polynomial basis P, it is possible to decide the order of accuracy of the method (see Figure 2). While the code still is at the development stage, the good level of parallelization allowed by the Julia programming language already allows an efficient solution of large systems of equations (e.g. $N \approx 750,000$),

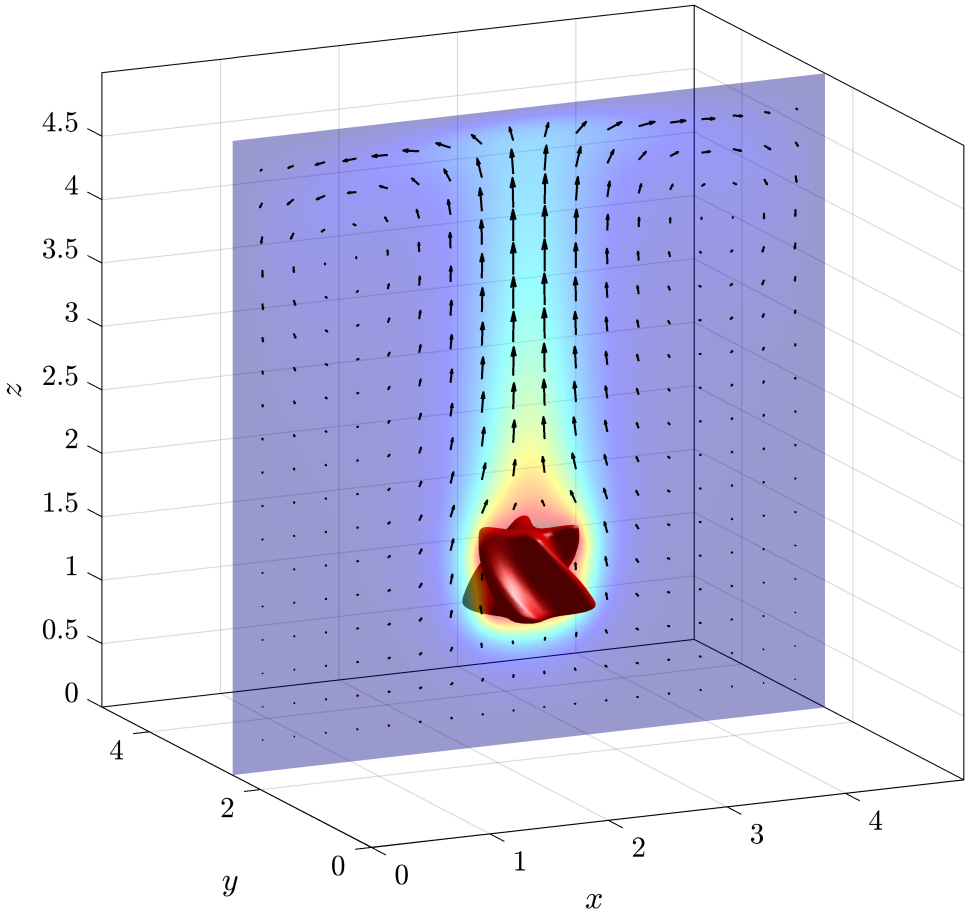

Figure 4. Temperature contour plot and velocity vector plot projected on the $x-z$ plane (T ranges between $T=0$ on the cube faces and $T=1$ on the object surface).

thus leading to reliable results in terms of accuracy also on a personal computer.

Future developments of the code will be aimed at addressing its limitations with the presented implementation, such as the need for directional node refinement close to boundary walls, and adding important features such as turbulence models.

We believe that the RBF-FD method has the potential to reach and outperform the accuracy allowed by the best traditional CAE solvers and to make the design process far more convenient and democratic by overcoming the need to generate a mesh.

References

[1] Zienkiewicz O C 1977 *The finite element method* vol 3 (McGraw-hill London)
[2] Versteeg H K and Malalasekera W 2007 *An introduction to computational fluid dynamics: the finite volume method* (Pearson education)
[3] Liu G R and Gu Y T 2005 *An introduction to meshfree methods and their programming* (Springer Science & Business Media)
[4] Fornberg B and Flyer N 2015 *A primer on radial basis functions with applications to the geosciences* (SIAM)
[5] Liu G R 2009 *Meshfree methods: moving beyond the finite element method* (Taylor & Francis)
[6] Li H and Mulay S S 2013 *Meshless methods and their numerical properties* (CRC press)
[7] Divo E and Kassab A J 2007 *J. Heat Transf.* **129** 124–136
[8] Zamolo R and Nobile E 2019 *Numer. Heat Tr. B-Fund.* **75** 19–42

[9] Šarler B and Vertnik R 2006 *Comput. Math. Appl.* **51** 1269–1282
[10] Waters J and Pepper D W 2015 *Numer. Heat Tr. B-Fund.* **68** 185–203
[11] Kosec G and Slak J 2020 *AIP Conference Proceedings* **2293** 420094
[12] Bezanson J, Edelman A, Karpinski S and Shah V B 2017 *SIAM Rev.* **59** 65–98
[13] Miotti D, Zamolo R and Nobile E 2021 *Energies* **14** 1351
[14] Frey P and George P 2013 *Mesh Generation: Application to Finite Elements* ISTE (Wiley) ISBN 9781118623824 URL `https://books.google.it/books?id=TZQokVbCwGIC`
[15] de Berg M, Cheong O, van Kreveld M and Overmars M 2008 *Computational Geometry: Algorithms and Applications* (Springer Berlin Heidelberg) ISBN 9783540779742 URL `https://books.google.it/books?id=9nhHUZpKzeEC`
[16] Hardy R L 1971 *J. Geophys. Res.* **76** 1905–1915
[17] Saad Y 2003 *Iterative Methods for Sparse Linear Systems* (Philadelphia, Pennsylvania: SIAM) chap 10: Preconditioning Techniques, pp 297–368 2nd ed
[18] van der Vorst H 1992 *SIAM J. Sci. Comput.* **13**(2) 631–644