Computer-Aided Design 154 (2023) 103409

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Smart Cloud Collocation: Geometry-Aware Adaptivity Directly From CAD

Thibault Jacquemin [a], Pratik Suchde [a], Stéphane P.A. Bordas [a,b,*]

[a] University of Luxembourg, Institute of Computational Engineering, Maison du Nombre, 6 Avenue de la Fonte, Esch-sur-Alzette, L-4364, Luxembourg
[b] Department of Medical Research, China Medical University Hospital, China Medical University, Taichung, Taiwan

ARTICLE INFO

Article history:
Received 6 January 2022
Received in revised form 1 July 2022
Accepted 6 September 2022

Dataset link: https://gitlab.com/tjacquemin/smart-cloud

Keywords:
Smart cloud
Adaptive refinement
Generalized finite difference
Error indication
Linear elasticity

ABSTRACT

Computer Aided Design (CAD) is widely used in the creation and optimization of various industrial systems and processes. Transforming a CAD geometry into a computational discretization that be used to solve PDEs requires care and a deep knowledge of the selected computational method. In this article, we present a novel integrated collocation scheme based on smart clouds. It allows us to transform a CAD geometry into a complete point collocation model, aware of the base geometry, with minimum effort. For this process, only the geometry of the domain, in the form of a STEP file, and the boundary conditions are needed. We also introduce an adaptive refinement process for the resultant smart cloud using an *a posteriori* error indication. The scheme can be applied to any 2D or 3D geometry, to any PDE and can be applied to most point collocation approaches. We illustrate this with the meshfree Generalized Finite Difference (GFD) method applied to steady linear elasticity problems. We further show that each step of this process, from the initial discretization to the refinement strategy, is connected and is affected by the approach selected in the previous step, thus requiring an integrated scheme where the whole solution process should be considered at once.

1. Introduction

Computer Aided Design (CAD) software packages are used in many domains of engineering to design components of various nature. A new design is often proposed based on previous experience and knowledge. It can then be optimized using calculation and/or simulation tools to increase its performance, lower the manufacturing costs or for many other reasons. Reaching a satisfactory design often requires iterations. To minimize the development costs, both industry and academia have been trying to speed-up this iteration process (design –> simulation –> design modification –> simulation …) as much as possible.

Performing computer simulations directly from geometry can be a tedious task. We introduce in this article an integrated smart cloud collocation scheme. The scheme, based on point collocation, aims at simulating the behavior of components (the mechanical resistance of a solid part, for instance) directly from the designed CAD geometry. The proposed scheme minimizes the size of the solved problem by using *a posteriori* error indication and adaptive refinement. The smart cloud is geometry-aware. Each point of the smart cloud has additional information related to the CAD geometry and to the boundary conditions applied to the domain.

Many collocation schemes use meshes to generate point clouds. The approach presented in this work does not require this step. Mesh generation poses several additional constraints on the aspect ratio and shape of the elements which do not apply to point collocation methods. Therefore, skipping the mesh generation step could lead to a more robust domain discretization framework for point collocation methods.

Point collocation methods have been used for a long time and can be applied to smooth and non-smooth solutions and domains [1]. The selection of collocation stencil is a key aspect of this family of methods. To ease their application to all types of problems and domains, a unified approach was introduced in Ref. [2]. The finite difference method was the first such collocation method. It was introduced by C. Runge in 1908 [3]. A Cartesian grid was used to approximate the field derivatives, limiting the problems solved by this method to simple geometries. The method was then generalized in 1972 by Jensen [4]. He introduced the basis of the Generalized Finite Difference (GFD) method. The method was then successively developed by Liszka [5], Orkisz [6], Benito [7], Milewski [8] and many other contributors.

Recently, numerical methods based on point collocation have regained interest with their use in meshfree frameworks. The GFD method shows good performance compared to other point

* Corresponding author at: University of Luxembourg, Institute of Computational Engineering, Maison du Nombre, 6 Avenue de la Fonte, Esch-sur-Alzette, L-4364, Luxembourg.
E-mail address: stephane.bordas@alum.northwestern.edu (S.P.A. Bordas).

https://doi.org/10.1016/j.cad.2022.103409
0010-4485/

collocation methods [1] and was used in this work. The scheme presented in this article can however be readily applied, without any modification, to other collocation schemes such as the Moving Least Squares (MLS) method [9–11] or the Radial Basis Finite Difference (RBF-FD) method [12–15].

The idea of using the CAD geometry to solve problems defined by Partial Differential Equations (PDE) is at the heart of the Isogeometric Analysis (IGA) methods. These methods became rapidly popular after their first introduction in 2005 [16] and have been proven robust for problems of various nature. IGA methods use the basis functions of the CAD geometry representation as shape functions to approximate the field derivatives and solve PDEs over domains. The Isogeometric analysis boundary element method (IGABEM) is a popular IGA method successively developed by Politis et al. [17], Belibassakis et al. [18], Simpson et al. [19] and Ginnis et al. [20]. It combines the benefits of the IGA method (direct use of the functions of the CAD geometry) and the benefits of the Boundary Element Method (use of a discretization of the surface boundaries only). The clearest advantage of IGABEM is the possibility to solve PDEs without any mesh generation, which is particularly convenient for shape optimization [21–24]. A collocation form of IGA has also been introduced [25].

Using the functions of the geometry to solve a given problem can be seen both as a strength and a weak point of IGA computational methods. This assumes that the CAD geometry is suitable for analysis, which is not always the case. Point collocation methods are deemed quite "flexible" compared to element-based methods and IGA methods since there are no strong connections between the nodes. Furthermore, the discretization of the domain can be easily modified as needed during the simulation process to provide the best possible solution for a given computational cost.

Discretization adaptivity has always been of interest for point collocation methods. The performance of this approach in the framework of point collocation was shown in many articles for the GFD [8,26–28] or the RBF-FD [29–31] method.

Many point collocation schemes use a surface mesh as an input for discretizing the geometry. Therefore, for geometries composed of curves or non-plane surfaces, the input is an approximation of the exact geometry and any discretization adaptivity would not rely on the exact domain. In contrast to existing work, the smart cloud collocation scheme presented in this work uses the exact definition of a given geometry based on a CAD file. It can therefore be applied to most domains with the assurance that the exactness of the geometry is not lost as part of the refinement process. Such an approach was not present in previous publications, to the authors' knowledge. The proposed adaptive method uses new nodes, placed at key locations in the domain, to improve the solution. This approach is often referred to as *h*-adaptivity. The nodes of the initial point cloud are kept. This hierarchical approach implies that the point cloud does not need to be generated again before the next adaptive iteration step which saves computational effort.

Our work focuses on linear elasticity problems (2D and 3D) using the Generalized Finite Difference method. The smart cloud concept can also be applied to other types of elliptic problems and to most point collocation methods. The Generalized Finite Difference method is based on a Taylor's series expansion of the unknown field. We provide here after a brief description of this collocation method for the case of a 2D problem. For a complete description of the method, refer to Ref. [1]. Considering a function $f : \mathbb{R}^2 \rightarrow \mathbb{R}$, we can write the second order approximation of the Taylor's series expansion at a node $\mathbf{X_{pi}} = [x_{pi}, y_{pi}]^T$ near $\mathbf{X_c} = [x_c, y_c]^T$ as:

$$\begin{aligned} f_h(\mathbf{X_{pi}}) = & f(\mathbf{X_c}) + (x_{pi} - x_c)\frac{\partial f(\mathbf{X_c})}{\partial x} + (y_{pi} - y_c)\frac{\partial f(\mathbf{X_c})}{\partial y} \\ & + \frac{(x_{pi} - x_c)^2}{2!}\frac{\partial^2 f(\mathbf{X_c})}{\partial x^2} + (x_{pi} - x_c)(y_{pi} - y_c)\frac{\partial^2 f(\mathbf{X_c})}{\partial x \partial y} \\ & + \frac{(y_{pi} - y_c)^2}{2!}\frac{\partial^2 f(\mathbf{X_c})}{\partial y^2}. \end{aligned} \quad (1)$$

We wish to obtain an approximation of the field derivatives at $\mathbf{X_c}$ based on the field value at $\mathbf{X_c}$ and in the vicinity of $\mathbf{X_c}$. For this, we write Eq. (1) at a minimum of five nodes $\mathbf{X_{pi}}$ around $\mathbf{X_c}$. We obtain a system of equations that can be solved for the field derivatives $\frac{\partial f(\mathbf{X})}{\partial x}$, $\frac{\partial f(\mathbf{X})}{\partial y}$, $\frac{\partial^2 f(\mathbf{X})}{\partial x^2}$, $\frac{\partial^2 f(\mathbf{X})}{\partial x \partial y}$ and $\frac{\partial^2 f(\mathbf{X})}{\partial y^2}$. The system is solved in a weighted moving least square form if Eq. (1) is written at a number of nodes greater than the number of unknown field derivatives. The selection of the nodes (stencil) to be considered as part of the field derivatives approximation is a key parameter of the method. The distance criterion is the most simple and commonly used criterion. All the nodes located within a defined radius of the reference collocation node are selected. The visibility criterion introduced by Belytschko et al. [32] is an alternative for singular problems. The nodes $\mathbf{X_p}$ included in the stencil of a collocation node $\mathbf{X_c}$ are only the nodes for which the segment connecting $\mathbf{X_p}$ to $\mathbf{X_c}$ does not intersect the boundary of the domain. Jacquemin et al. [2] proposed to generalize this criterion to all concave problems based on a selected maximum acceptable intersection angle between the segment connecting $\mathbf{X_p}$ to $\mathbf{X_c}$ and the boundary of the domain.

The article is composed of two main sections. We present in the first one the method used to transform a CAD geometry into a smart cloud. We show in a second section how error indicators can be used to identify the zones of the domain where the error is the greatest. Once these zones are identified, we show how adaptive refinement can be used to converge efficiently to an accurate solution.

A key novelty in the present work is that the discretization of the domain and its adaptivity are based on the exact CAD geometry to minimize the user input in the simulation workflow and ensure that the domain is represented exactly throughout the refinement process.

The codes and all the input files used to generate the results, along with the results files, have been made open-source and are freely available at https://gitlab.com/tjacquemin/smart-cloud.

2. From CAD to smart cloud

2.1. General

Most numerical methods used to solve partial differential equations use some sort of discretization of the otherwise infinite dimensional mathematical problem. Either the differential operator is discretized (e.g., GFD, RBF-FD) or the unknown field is discretized (e.g., MLS, Finite Element). Collocation methods use nodes spread in the domain, on the boundary of the domain and sometimes even outside of the domain to approximate the differential operator. Nodes placed outside of the domain, called ghost nodes, can be used as additional degrees of freedom to enforce boundary conditions or balance the stencils of the collocation nodes located on, or close to the boundary of the domain. The concept of ghost cells or ghost nodes was described by Fedkiw et al. in 1999 in Ref. [33]. The approach was applied to fluid flow solvers but can be applied to other types of problems such as linear elasticity as presented in Ref. [34]. In this work, we considered only discretizations featuring nodes placed in the domain and on the boundary of the domain.

The design of a new mechanical system very often requires a model of the geometry. CAD software packages are used for

this purpose. Sophisticated user interfaces allow the design of complex geometries and the assembly of very large structures. Whilst many file formats are proprietary, the STEP file format (Standard for the Exchange of Product model) is a format defined by the international norm ISO 10303-21. Such a definition makes the STEP file format popular and most of the packages support it. It is mostly oriented toward 3D geometries but can also be used for 2D geometries. STEP files store the exact geometry of the domain using simple geometric features such as planes or conical surfaces but also B-spline surfaces and trimmed surfaces.

We present in this section a method to discretize a given geometry, provided in STEP file format, with the aim of solving a problem defined by a PDE over the domain using the Generalized Finite Difference point collocation method. We show how the smart cloud is generated from the discretization and from the CAD geometry. Finally, we analyze the impact of the selected parameters of the method.

2.2. Domain discretization

We used the library Open CASCADE Technology [35] to communicate with the STEP files and to get information about the exact geometry. Our algorithm is composed of the following steps:

1. Loading the information from the STEP file using Open CASCADE Technology;
2. Discretization of the boundaries of the domain;
3. Regular discretization of a rectangle or box enclosing the geometry;
4. Identification of the nodes of the rectangle or box included in the domain.

We presented these steps in Fig. 1 for the case of a 2D gear. We give more details about each of these steps in the paragraphs below.

Step 1. At the beginning of the discretization process key parameters, such as the bounding volume or the dimensions of the rectangle or box enclosing the geometry, are computed from the CAD file.

A CAD geometry is composed of multiple topological entities. Those are:

- solid;
- shell;
- face;
- edge loop;
- edge;
- vertex.

These entities are identified and used to discretize the boundaries of the domain and set the boundary conditions. Geometrical entities are associated with each topological entity. For instance, the geometrical entity associated with a face is a surface (e.g., plane, cylindrical surface, B-spline surface) and the geometrical entity associated with an edge is a curve (e.g., line, circle, B-spline curve). The library Open CASCADE that we use supports a large amount of geometric features. For the complete list of features, refer to the reference manual [35].

The discretization process requires the selection of a characteristic length noted h. h can be an input from the user or can be approximated based on a target number of nodes of the domain discretization. In this case, h is computed in this first step.

Step 2. The boundaries of the domain are discretized, based on the characteristic length h, ensuring that the distance between two adjacent boundary nodes is close to h. For 2D problems, all the edges of the domain are discretized using a fixed distance, close to h, between two consecutive nodes. The duplicated corner nodes are removed.

For 3D problems, we used a Delaunay triangulation of the boundary faces that compose the geometry. Generating such a mesh is robust since the boundary of the domain is composed of faces of simple geometry. The duplicated edge nodes are removed. We used the library Gmsh [36] to mesh the surfaces.

The exact normal vectors are computed, at each boundary collocation node, using the information about the exact geometry contained in the STEP file.

Step 3. The rectangle or box is discretized based on the characteristic length h. The nodes are placed regularly in the rectangle or box enclosing the geometry. The nodes can be organized in different regular forms, also called lattices. In 2D and 3D, the nodes can be placed following the Cartesian grid. In 2D, the rectangle can also be discretized using equilateral triangles. This corresponds to a hexagonal close-packed lattice in 3D. We selected these two lattices for 2D and 3D problems as they lead to the most uniform discretizations. Fig. 2 shows a comparison of the mentioned 2D and 3D lattices. Many other lattices could be considered. The rectangle or box could also be filled using the advancing front method [37].

Step 4. The final step of the discretization process requires the identification of the nodes which are outside of the domain. Multiple algorithms can be used for this purpose. The CAD file can be used directly to assess the position of a node in a domain. We used this approach, based on classes of the Open CASCADE [35] framework. For 2D problems, we compute the distance between a given vertex and the shapes that compose the domain using the class BRepExtrema_DistShapeShape of Open CASCADE. A null

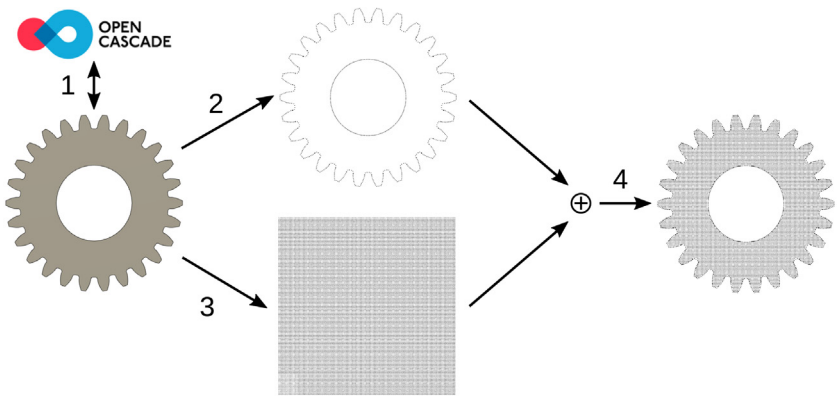


Fig. 1. Steps of the discretization of a domain from a CAD file using the library Open CASCADE Technology [35].

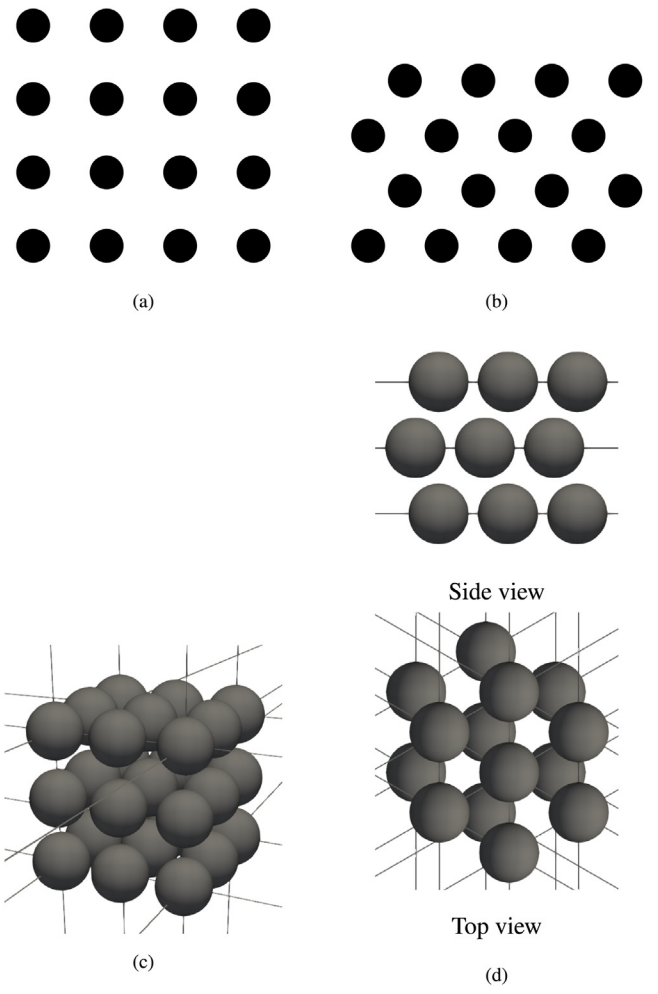


Fig. 2. Node arrangement configurations in 2D (a), (b) and 3D (c), (d). The subfigures (a) and (b) show the node arrangements for square and triangular lattices, respectively. The subfigures (c) and (d) show the node arrangements for cubic and hexagonal close-packed lattices, respectively.

distance means that the vertex is located on the considered shape. For 3D problems, we assess if a considered vertex is in the domain using the class BRepClass3d_SolidClassifier. The algorithms presented in Ref. [37] or in Ref. [2] are alternative algorithms that use boundary nodes and elements to decide upon the inclusion of nodes in the domain. Other algorithms such as the "crossing number" or the "winding number" methods, described in Refs. [38–40], can be used in 2D. For 3D problems, the Möller–Trumbore algorithm [41] or the AABB tree algorithm [42] can be used if the boundaries of the domain are triangulated surfaces. However, these alternative approaches are imprecise because they depend on an approximation of the boundary of the domain.

The positions of all the box nodes $\mathbf{X}$ with respect to a domain Ω do not need to be assessed. All the nodes located in a disk or sphere centered at a considered node $\mathbf{X_c}$ and of radius $\left\|\mathbf{X_c} - \mathbf{X_{pc}}\right\|_2$, where $\mathbf{X_{pc}}$ is the orthogonal projection of $\mathbf{X_c}$ on the boundary of the domain Γ_Ω, are located on the same side of the boundary as $\mathbf{X_c}$. This is illustrated by Fig. 3 for a node $\mathbf{X_i}$ located inside of the domain and a node $\mathbf{X_o}$ located outside of the domain. We use classes of the Open CASCADE [35] framework to perform the orthogonal projection. For 2D problems, we use the class Geom2dAPI_ProjectPointOnCurve. For 3D problems, we use the class GeomAPI_ProjectPointOnSurf.

The proximity of the interior nodes to nodes located on the boundary of the domain shall also be considered to avoid the ill conditioning of the system. This aspect is discussed in Section 2.4.

2.3. From discretization to smart cloud

Transforming the discretization into a smart cloud, which contains all the required information for its solution using a collocation model and for model adaptivity, is the final step. It consists primarily in the enforcement of the boundary conditions and in the addition of additional information about the geometry useful to improve the solution and adaptive refinement.

The topological entities are used to define the boundary conditions. The boundary conditions are most often defined on edges, for 2D problems, and on faces, for 3D problems. We defined in an input file the boundary condition associated to the topological entities of interest of the CAD geometry. The boundary conditions are transmitted from the topological entities to the collocation nodes during the discretization process. For nodes at the intersection between multiple CAD topological entities, we automatically select the boundary condition for each degree of freedom as follows. We first apply non-zero Neumann boundary conditions, then Dirichlet boundary conditions and, finally, homogeneous Neumann boundary conditions.

We explained in Section 2.2 that surface elements are used to discretize the boundary of the domain. In case of adaptive refinement, the smart nodes are used to carry pointers to the boundary conditions from the initial model to the refined models. This allows reducing the number of interactions between the collocation code and the geometry to the minimum, thus saving computational cost.

Each boundary node of the smart cloud has the following pieces of information:

- reference to the base CAD geometry;
- the exact normal vector;
- the boundary conditions applied to the considered node;
- pointers to the parent CAD edge or surface(s);
- pointers to the boundary conditions applied to the parent CAD edge or face(s);
- the connections to other boundary nodes if boundary elements are used to enforce the generalized visibility criterion [2] or to speed-up the refinement of the surface.

2.4. Threshold sensitivity analysis

Stencil nodes located close to the boundary of the domain should theoretically improve the quality of the field approximation. Boundary conditions are applied at the boundary nodes. At these nodes, the PDE cannot be enforced with classical collocation methods. Adding an inner node very close to a boundary node would allow enforcing both the PDE and boundary condition close to the boundary. However, this leads to ill-conditioning of the linear system solved, for the GFD collocation method and for most collocation methods, at these collocation centers to obtain an approximation of the field derivatives as a function of the field itself. To avoid ill-conditioning, we used a threshold ratio, denoted by t, to determine if an interior node, obtained from the regular discretization of the rectangle or box, should be included in the point cloud. A node $\mathbf{X_i}$ located in the domain Ω is included in the point cloud only if the closest boundary node $\mathbf{X_b}$ is located at a distance larger than the product th (i.e., $\mathbf{X_i} \in \Omega$ if $\|\mathbf{X_i} - \mathbf{X_b}\|_2 > th$).

We used two benchmark problems from the field of linear elasticity to assess the impact of the threshold on the error. The problems considered are: an infinite plate with an elliptical hole

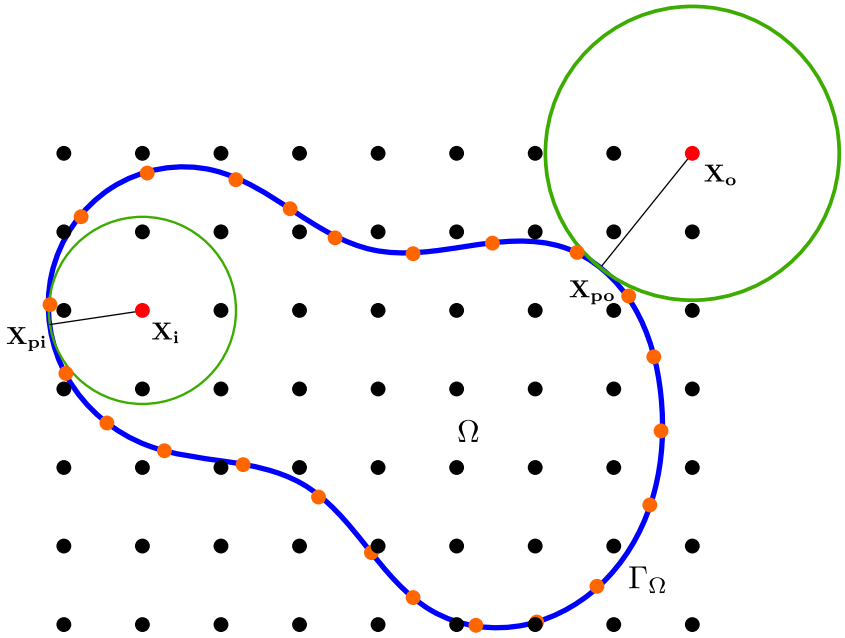


Fig. 3. Identification of the position of interior nodes with respect to the domain Ω. Considering a node $\mathbf{X_i}$ and its projection $\mathbf{X_{pi}}$ on the boundary of the domain Γ_Ω, all the nodes located within a disk or sphere of radius $\left\|\mathbf{X_i} - \mathbf{X_{pi}}\right\|_2$ are located in the domain Ω if the node $\mathbf{X_i}$ is located in the domain Ω. Similarly, all the nodes located within a disk or sphere of radius $\left\|\mathbf{X_o} - \mathbf{X_{po}}\right\|_2$ are located outside of the domain Ω if the node $\mathbf{X_o}$ is outside of the domain Ω.

and a infinite plate with a circular hole. The exact solution is known for each of the considered benchmark problems. The governing equations of linear elasticity and the problems considered are presented in Sections 3.1 and 3.2, respectively.

To assess the impact of the threshold value t on the solution of the benchmark problems, we selected threshold values ranging from 0.02 to 1.2. For the purpose of the sensitivity analysis, we generated coarse and fine discretizations of the considered benchmark problems (approximately 4500 and 30,000 nodes, respectively) based on the discretization method presented in Section 2.2. We selected a square lattice to discretize the rectangles (bounding box) which contain the geometries. We compared the results in terms of the l_2 relative error norm of the von Mises stress. The calculation of this norm is described in Section 3.2.

The results are presented in Fig. 4. The results show that the threshold has little impact on the error for the problem of a plate with an elliptical hole for threshold values lower than 1.0. The error varies by less than 9% for the coarse discretization in the threshold range 0.02–1.0. For the fine discretization, the error is lower by approximately 23% for a threshold of 0.8 than for a threshold of 0.5. We observe a sharp increase of the error for the threshold of 1.2. This result is expected, because a threshold larger than 1.2 means that there is a gap between the boundary of the domain and the closest inner nodes which is larger than the discretization characteristic length h. The variations are more important for the plate with a circular hole. A threshold of 0.3 leads to the lowest error for the fine discretization. A threshold of 0.02 leads to the lowest error for the coarse discretization. For both node densities, the lowest error is approximately 75% lower than the maximum error.

To better understand these results, we plot in Fig. 5 the maximum condition number of the linear systems solved as part of the field derivatives approximation as a function of the threshold value. We see that this condition number is minimum for the plate with a circular hole for threshold values ranging from 0.5 to 1.0 for the coarse discretization and from 0.7 to 1.0 for the fine discretization. For the problem of a plate with an elliptical hole, this condition number is little affected by the threshold value for the coarse discretization and minimum for the fine discretization for threshold values ranging from 0.3 to 1.0. An increase of the condition number is associated with an increase of the error. This can be observed in particular for the threshold of 1.2 for the problem of a plate with an elliptical hole and for the threshold of 0.3 for the fine discretization of a plate with a circular hole. The condition number of the stencil is closely related to the node selection algorithm. In this work, we selected the stencil nodes based on the distance criterion [4]. Close to concave boundaries of the domain, we used the visibility criterion with a threshold angle of 5.0° as presented in Ref. [2]. Based on these results, we selected in this work a threshold value of 0.3 as it leads to the lowest error for the plate with a circular hole problem and to the most significant error reduction compared to other threshold values. We decided not to investigate other node selection algorithms which could be more suitable for the lowest threshold values.

2.5. Discretization methods comparison

We compared the results obtained using the proposed discretization method to results obtained from discretizations generated using Gmsh [36]. Gmsh is a powerful mesh generator that regroups several meshing algorithms. Gmsh is suitable for both 2D and 3D problems. We selected unstructured meshing algorithms based on the Delaunay algorithm. We used the benchmark problems mentioned in Section 2.4 for the purpose of this comparison. We compare results for a square lattice and a triangular lattice discretization of the rectangle enclosing the geometry to results obtained from a discretization generated with Gmsh (Delaunay triangulation). We used a threshold value of 0.3 for the square and triangular lattice discretizations. The results in terms of the l_2 relative error of the von Mises stress are shown in Fig. 6.

The results show that the two discretization methods lead to similar errors for the plate with an elliptical hole. For the plate with a circular hole, the square and triangular lattice discretizations from CAD lead to errors approximately 40% lower than those obtained from the discretization obtained from a Delaunay triangulation of the geometry.

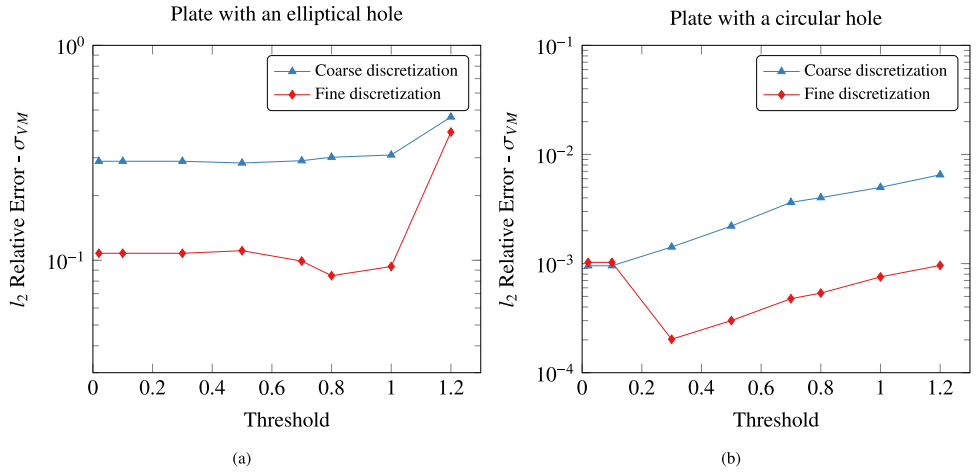


Fig. 4. Error for threshold values ranging from 0.02 to 1.2 for coarse (approx. 4500 nodes) and fine (approx. 30,000 nodes) discretizations for the plate with an elliptical hole problem (a) and for the plate with a circular hole problem (b). The error in terms of the l_2 relative error norm is presented for the von Mises stress noted σ_{VM}.

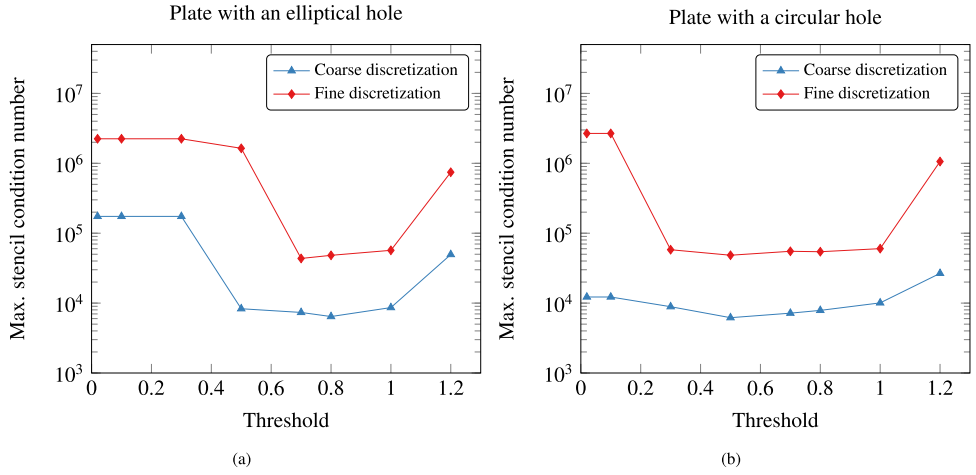


Fig. 5. Maximum stencil condition number for threshold values ranging from 0.02 to 1.2 for coarse (approx. 4500 nodes) and fine (approx. 30,000 nodes) discretizations for the plate with an elliptical hole problem (a) and for the plate with a circular hole problem (b).

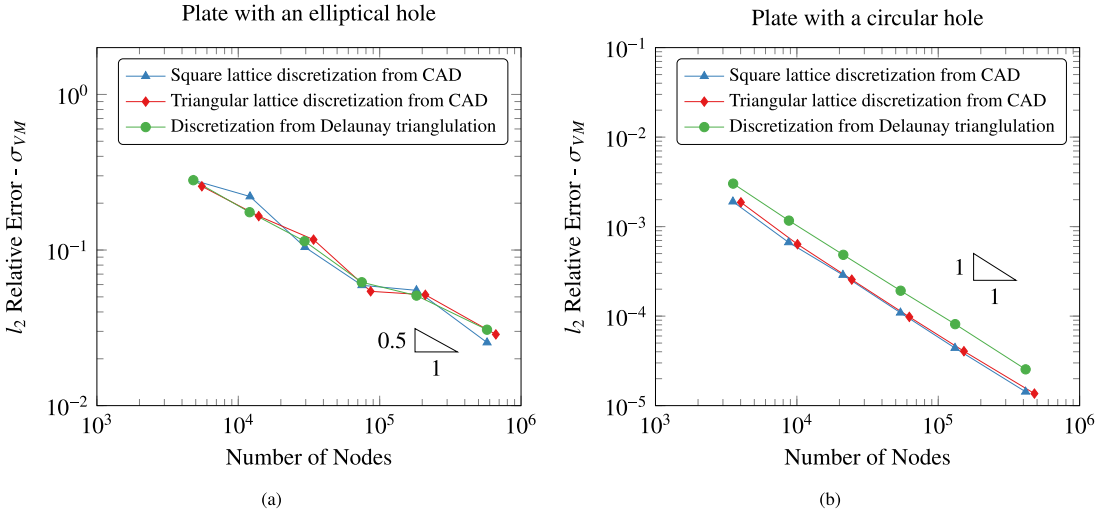


Fig. 6. Comparison of the error in terms of the l_2 relative error norm obtained from different discretization techniques (i.e., square or triangular lattice discretization from CAD and Delaunay triangulation generated using Gmsh). The results are presented for the plate with an elliptical hole problem (a) and for the plate with a circular hole problem (b). The three discretization methods lead to similar errors for the plate with an elliptical hole problem. The square and triangular lattice discretizations from CAD lead to similar results for the plate with a circular hole problem. The error obtained with these discretizations is lower than the error obtained from the Delaunay triangulation of the domain.

These results give confidence in the proposed discretization methods as they lead to results not far from the ones obtained from a discretization method based on a triangulation of the domain.

3. Model adaptivity from CAD

We show in this section how the CAD geometry can be effectively used in an adaptivity scheme using smart cloud discretizations presented in Section 2. We present results for problems from the field of linear elasticity solved using the GFD method. The governing equations for linear elasticity are introduced in Section 3.1. We used two 2D benchmark problems, for which analytical solutions are known, to assess the sensitivity of the parameters of the presented method on the quality of the error indicator and on the adaptive refinement scheme. The models considered are presented in Section 3.2. Then, we present in Section 3.3 two error indicators that we used to identify the zones where the error is the greatest. Finally, we show in Section 3.4 how error indicators are used to refine locally the domain and improve the convergence rate of the solution.

3.1. Governing equations

We present in this section the governing equations for linear elasticity for the general case of a 3D problem. This section is based on Ref. [2].

The equilibrium of a domain Ω subject to body forces $\mathbf{b}$ is expressed as a function of the stress tensor $\boldsymbol{\sigma}$ by Newton's second law. For static problems, the equilibrium equation is:

$$\begin{aligned} &\nabla \cdot \boldsymbol{\sigma} + \mathbf{b} = 0 \\ \text{or} \quad &\forall i \in \{1,2,3\} \quad \sigma_{ij,j} + b_i = 0. \end{aligned} \tag{2}$$

The equilibrium equations are expressed as a function of the displacement field $\mathbf{u}$ at each node of the domain using:

- the relationship between the displacement field and the strain field $\boldsymbol{\epsilon}$ (kinematics):

$$\begin{aligned} &\boldsymbol{\epsilon} = \frac{1}{2}\left(\nabla \mathbf{u} \otimes \nabla \mathbf{u}^T\right) \\ \text{or} \quad &\forall i \in \{1,2,3\} \quad \epsilon_{ij} = \frac{1}{2}\left(u_{i,j} + u_{j,i}\right), \end{aligned} \tag{3}$$

- Hooke's law which gives the relationship between the strain field and the stress field (presented here in Voigt form). This is the constitutive law:

$$\begin{bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ \sigma_{23} \\ \sigma_{13} \\ \sigma_{12} \end{bmatrix} = \frac{E}{(1+\nu)(1-2\nu)} \times \begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & 1-2\nu & 0 & 0 \\ 0 & 0 & 0 & 0 & 1-2\nu & 0 \\ 0 & 0 & 0 & 0 & 0 & 1-2\nu \end{bmatrix} \begin{bmatrix} \epsilon_{11} \\ \epsilon_{22} \\ \epsilon_{33} \\ \epsilon_{23} \\ \epsilon_{13} \\ \epsilon_{12} \end{bmatrix}. \tag{4}$$

The above equations can be used for 2D problems using either the plane stress assumption (i.e., $\sigma_{33} = 0$, $\sigma_{13} = 0$ and $\sigma_{23} = 0$) or the plane strain assumption (i.e., $\epsilon_{33} = 0$, $\epsilon_{13} = 0$ and $\epsilon_{33} = 0$).

Dirichlet and Neumann boundary conditions are respectively applied to the degrees of freedom of the collocation nodes located on the boundaries Γ_D and Γ_N. The known displacement field $\mathbf{u^e}$ is applied on Γ_D. An external pressure $\mathbf{f^e}$ is applied to the nodes located on Γ_N. The outer normal $\mathbf{n_N}$ allows the computation of the pressure at the nodes of Γ_N. Dirichlet and Neumann boundary conditions can be applied to different degrees of freedom of the same node.

$$\begin{aligned} &\mathbf{u} = \mathbf{u^e} && \text{on} \quad \Gamma_D \\ &\boldsymbol{\sigma}\mathbf{n_N} = \mathbf{f^e} \quad \text{or} \quad \forall i \in \{1,2,3\} \quad \sigma_{ij} n_j = f_i^e && \text{on} \quad \Gamma_N. \end{aligned} \tag{5}$$

3.2. Benchmark problems considered

We present in this section the benchmark problems that we considered as part of our analysis. We selected 2D problems with known solutions from the field of linear elasticity. These are:

- an infinite plate with an elliptical hole under biaxial loading;
- a infinite plate with a circular hole under remote stress loading.

The reference problems are said to be infinite because the boundary conditions are applied at an infinite distance from the

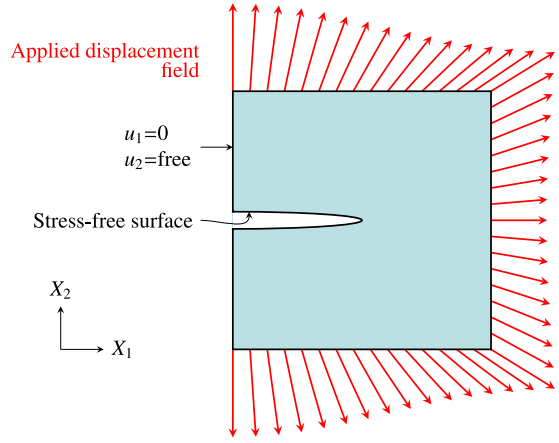


Fig. 7. 2D model of a plate with an elliptical hole under biaxial loading. Symmetry boundary conditions are applied to the vertical edge on the left. Considering a displacement field denoted u, this boundary condition corresponds to u_1=0, u_2=free. Stress-free surface boundary conditions are applied to the boundary of the elliptical hole. The displacement field of the exact solution is applied to the other boundaries of the domain. To improve the quality of the solution, three boundary nodes at the tip are considered as interior nodes.

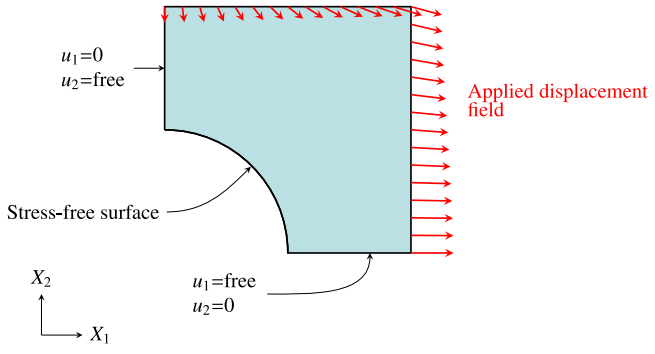


Fig. 8. 2D model of a plate with a circular hole under remote stress loading. Symmetry boundary conditions are applied to the vertical edge on the left and to the horizontal edge at the bottom. Considering a displacement field denoted u, these boundary conditions correspond respectively to u_1=0, u_2=free and u_1=free, u_2=0. Stress-free surface boundary conditions are applied to the boundary of the hole. The displacement field of the exact solution is applied to the other boundaries of the domain.

studied portion of the domain. We considered only portions of the domains, close to the holes, and applied boundary conditions corresponding to the exact solution. The domain considered and the boundary conditions applied are presented in Figs. 7 and 8, respectively, for the first and second problems. The analytical solutions to these problems are presented in Refs. [43,44], respectively. We show the exact solution of these problems in terms of von Mises stress in Fig. 9. The von Mises stress solution of the first problem varies rapidly at the point of highest curvature of the ellipse. The solution in terms of von Mises stress is smoother for the second problem. We used the l_2 relative error norm (denoted by l_2R) and the l_2 weighted error norm (denoted by l_2W) in this work to compare the results obtained to the reference solutions.

At a collocation node $\mathbf{X_k}$ the exact stress and approximated stress solutions are denoted $\sigma_{ij}^e(\mathbf{X_k})$ and $\sigma_{ij}^h(\mathbf{X_k})$, respectively. Considering a domain Ω discretized by n collocation nodes, the l_2 relative error norm is calculated as per Eq. (6). The l_2 weighted error norm is calculated as per Eq. (7).

$$l_2R(\sigma_{ij}) = \frac{\sqrt{\sum_{k=1}^{n}\left(\sigma_{ij}^e(\mathbf{X_k}) - \sigma_{ij}^h(\mathbf{X_k})\right)^2}}{\sqrt{\sum_{k=1}^{n}\sigma_{ij}^e(\mathbf{X_k})^2}}. \tag{6}$$

$$l_2W(\sigma_{ij}) = \frac{\sqrt{\sum_{k=1}^{n}\left(\sigma_{ij}^e(\mathbf{X_k}) - \sigma_{ij}^h(\mathbf{X_k})\right)^2}}{n}. \tag{7}$$

3.3. Error indicators

We describe in this section two types of error indicators that we used to assess the need for local refinement of the discretization:

- a ZZ-type error indicator;
- a residual-type error indicator.

We use the term "error indicator" in this article rather than the term "error estimator". Our methods only give an indication of the zones of the domain where the solution is expected to be the most imprecise rather than an estimation of the exact error. Therefore, the computed error should be considered relatively to the error computed at other locations of the domain rather than as an estimation of the true error. The considered indicators are described in the subsections below.

3.3.1. ZZ-type error indicator for the GFD method

The ZZ-type indicator refers to the class of error estimators introduced by Zienkiewicz and Zhu in 1987 [45] and extended by Bordas and Duflot [46,47] and Rodenas et al. [48] to enriched approximations. Zienkiewicz and Zhu used a moving least square approximation of the stress field (for linear elastic problems) computed at superconvergent points to estimate the error. The moving least square approximation is used to extrapolate the stress computed at the superconvergent points at nodes of a selected patch. ZZ-type error estimators can be understood as an indication of the smoothness of the computed stress field over the selected patch. If the stress field is smooth, the difference in terms of stress at the recovery nodes will be small. A sharp variation of the stress field leads to a large error. We build on this idea to define an error indicator in the framework of the GFD method.

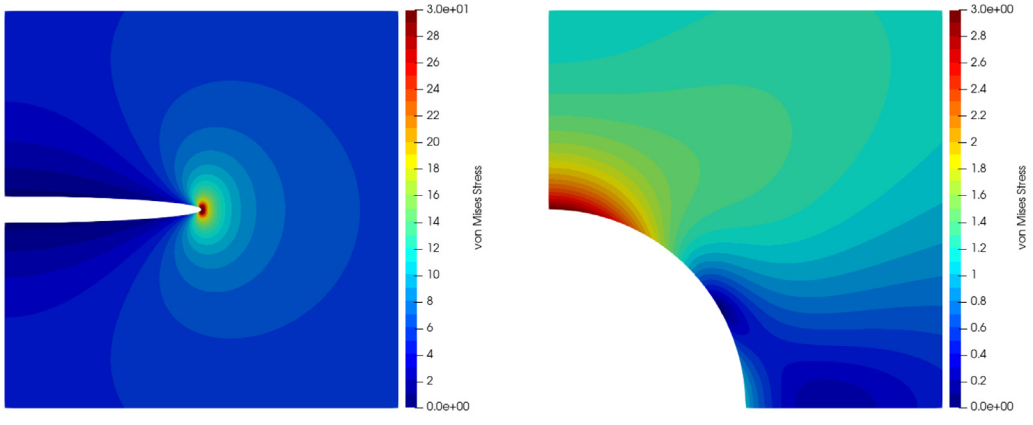
In many engineering problems, the stress field and in particular the von Mises stress, is the parameter of primary interest as it is a criterion associated with the onset of material yielding (von Mises plasticity). We considered this stress criterion to compute a ZZ-type error indicator. The von Mises criterion (noted σ_{vM}) is computed, for 2D plane stress problems, using the equation:

$$\sigma_{\text{vM}} = \sqrt{\frac{(\sigma_{11} - \sigma_{22})^2 + 6\sigma_{12}^2}{2}}. \tag{8}$$

The solution of a linear elastic problem using the GFD methods requires the estimation of the first and second derivatives of the displacement field $\mathbf{u}$ to solve Eq. (2) at the collocation nodes located in the domain and Eq. (5) at the collocation nodes located on the boundary of the domain. A detailed description of all the steps involved in the solution of a problem using the GFD method is presented in Ref. [1]. Once the system of equations is solved, the solution in terms of displacement field (usually at the collocation nodes) is used to compute the stress field using Eq. (3) and Eq. (4).

In this work, collocation is done at the nodes. This means that the partial differential equations and the boundary conditions are solved at the nodes. Therefore, the von Mises stress can be computed at each node of the domain.

We computed an indication of the error at each node of the domain. For this, we used the von Mises stress calculated at each node based on the classical GFD method and based on a smoothed (recovered) von Mises field computed using a moving least square approximation of the von Mises stress values obtained at each



(a) (b)

Fig. 9. Solutions in terms of von Mises stress for the problem of a plate with an elliptical hole under biaxial loading (a) and for the problem of a plate with a circular hole under remote stress loading (b). The maximum von Mises stress is 150 at the point of highest curvature of the elliptical hole for the problem of a plate with an elliptical hole. The von Mises stress decreases rapidly as the distance from the tip increases. We truncated the colorbar of the von Mises stress field to 30 to show the variations of the stress over the domain. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

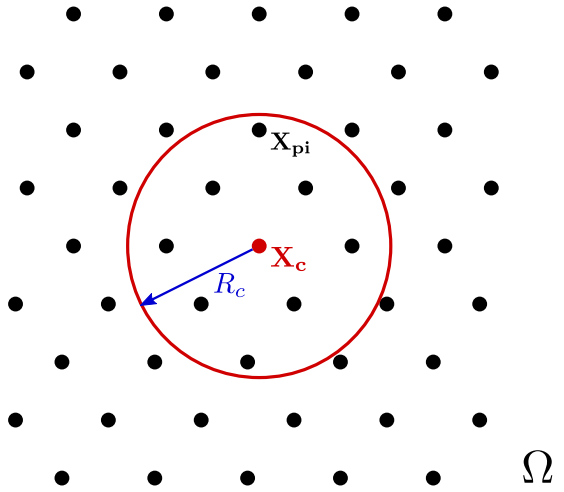


Fig. 10. Discretization of a portion of a domain Ω and identification of the nodes $\mathbf{X_{pi}}$, located within a support of radius R_c of a collocation node $\mathbf{X_c}$, involved in the computation of the smooth von Mises field $\sigma^s_{\text{vM}}(\mathbf{X_c})$.

collocation node. We base the indicator on patches composed of a relatively large number of nodes. The patch size is comparable to the size of the collocation stencil. We select a second order polynomial function and weight the contribution of the nodes of the patch using a radial weight function. We expect that this approach be more reliable and lead to a smoothest error indication than methods based on closest neighbor nodes although we did not compare in this article the proposed method to other indicators such as the ones proposed by Davydov [29] or Gavete [27] for instance.

More specifically, considering a domain Ω, the von Mises stress obtained at a collocation node $\mathbf{X_c}$ (noted $\sigma^c_{\text{vM}}(\mathbf{X_c})$) is compared to a moving least square approximation of the von Mises stress field at the collocation node $\mathbf{X_c}$ (noted $\sigma^s_{\text{vM}}(\mathbf{X_c})$). The moving least square approximation is calculated based on the von Mises stress $\sigma^c_{\text{vM}}(\mathbf{X_{pi}})$ computed at the support nodes $\mathbf{X_{pi}}$ of the collocation node $\mathbf{X_c}$ (see Fig. 10).

The moving least square approximation is computed using a second order polynomial basis $\mathbf{p}$ (or a polynomial of the same order as the GFD approximation) and a vector of coefficients $\mathbf{a}$ determined for each collocation node. For the case of a 2D problem, the polynomial basis $\mathbf{p}$ at a point $\mathbf{X} = [x, y]^T$ in the vicinity of $\mathbf{X_c} = [x_c, y_c]^T$ is:

$$\mathbf{p}(\mathbf{X}, \mathbf{X_c}) = \begin{bmatrix} 1 \\ (x - x_c) \\ (y - y_c) \\ (x - x_c)^2 \\ (x - x_c)(y - y_c) \\ (y - y_c)^2 \end{bmatrix}. \tag{9}$$

For a collocation node $\mathbf{X_c}$, the smooth von Mises stress field σ^s_{vM} is written:

$$\sigma^s_{\text{vM}}(\mathbf{X}, \mathbf{X_c}) = \mathbf{p}(\mathbf{X}, \mathbf{X_c})^T \mathbf{a}(\mathbf{X_c}). \tag{10}$$

The coefficients $\mathbf{a}(\mathbf{X_c})$ are computed to minimize the error between σ^c_{vM} and σ^s_{vM}. For a collocation node $\mathbf{X_c}$ which has m support nodes $\mathbf{X_{pi}}$ ($\mathbf{X_c}$ is not considered as a support node), we write the functional $B(\mathbf{X_c})$ presented in Eq. (11). The error is weighted by a function $w(\mathbf{X}, \mathbf{X_c})$ which depends on the support radius of the collocation node $\mathbf{X_c}$ and on a selected radial basis function.

$$B(\mathbf{X_c}) = \sum_{i=1}^{m} w(\mathbf{X_{pi}}, \mathbf{X_c}) \left(\mathbf{p}(\mathbf{X_{pi}}, \mathbf{X_c})^T \mathbf{a}(\mathbf{X_c}) - \sigma^c(\mathbf{X_{pi}}) \right)^2. \tag{11}$$

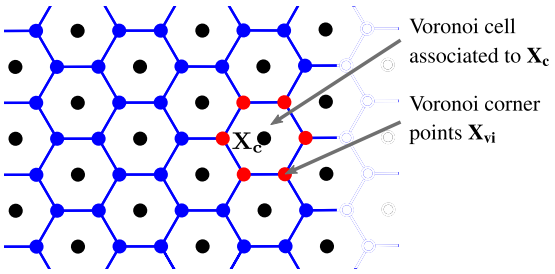


Fig. 11. Voronoi diagram for a set of collocation nodes. The residual-type error indicator at a collocation node $\mathbf{X_c}$ is computed based on the residual of the PDE at the corner points $\mathbf{X_{vi}}$ of the Voronoi cell associated to $\mathbf{X_c}$.

The error is minimized at each collocation node $\mathbf{X_c}$ when:

$$\frac{\partial B(\mathbf{X_c})}{\partial \mathbf{a}(\mathbf{X_c})} = 0. \tag{12}$$

This minimization problem can be transformed into a linear problem of the form:

$$\mathbf{A}(\mathbf{X_c})\,\mathbf{a}(\mathbf{X_c}) = \mathbf{E}(\mathbf{X_c})\,\mathbf{f}(\mathbf{X_c}). \tag{13}$$

For a polynomial basis of size q ($q = 6$ for a 2D second order basis), the matrices $\mathbf{A}(\mathbf{X_c})$, $\mathbf{E}(\mathbf{X_c})$ and $\mathbf{f}(\mathbf{X_c})$ are:

$$\mathbf{A}(\mathbf{X_c}) = \begin{bmatrix} m_{11} & m_{12} & \dots & m_{1q} \\ m_{21} & m_{22} & \dots & m_{2q} \\ \vdots & & & \vdots \\ m_{q1} & m_{q2} & \dots & m_{qq} \end{bmatrix} \in \mathbb{R}^{q \times q}, \tag{14}$$

$$\mathbf{E}(\mathbf{X_c}) = \begin{bmatrix} w(\mathbf{X_{p1}}, \mathbf{X_c})\mathbf{p}(\mathbf{X_{p1}}, \mathbf{X_c})_1 & \dots & w(\mathbf{X_{pm}}, \mathbf{X_c})\mathbf{p}(\mathbf{X_{pm}}, \mathbf{X_c})_1 \\ w(\mathbf{X_{p1}}, \mathbf{X_c})\mathbf{p}(\mathbf{X_{p1}}, \mathbf{X_c})_2 & \dots & w(\mathbf{X_{pm}}, \mathbf{X_c})\mathbf{p}(\mathbf{X_{pm}}, \mathbf{X_c})_2 \\ \vdots & & \vdots \\ w(\mathbf{X_{p1}}, \mathbf{X_c})\mathbf{p}(\mathbf{X_{p1}}, \mathbf{X_c})_q & \dots & w(\mathbf{X_{pm}}, \mathbf{X_c})\mathbf{p}(\mathbf{X_{pm}}, \mathbf{X_c})_q \end{bmatrix} \in \mathbb{R}^{q \times m}, \tag{15}$$

$$\mathbf{f}(\mathbf{X_c}) = \begin{bmatrix} \sigma^c(\mathbf{X_{p1}}) & \sigma^c(\mathbf{X_{p2}}) & \dots & \sigma^c(\mathbf{X_{pm}}) \end{bmatrix}^T, \tag{16}$$

where

$$m_{ij} = \sum_{k=1}^{m} w(\mathbf{X_{pk}}, \mathbf{X_c})\mathbf{p}(\mathbf{X_{pk}}, \mathbf{X_c})_i\, \mathbf{p}(\mathbf{X_{pk}}, \mathbf{X_c})_j. \tag{17}$$

$\mathbf{p}(\mathbf{X_{pk}}, \mathbf{X_c})_i$ refers to the ith component of the vector $\mathbf{p}(\mathbf{X_{pk}}, \mathbf{X_c})$.

The solution of Eq. (13) at $\mathbf{X_c}$ allows the computation of $\sigma^s_{\text{vM}}(\mathbf{X_c})$ and of an error indicator $e(\mathbf{X_c})$ as follows:

$$e(\mathbf{X_c}) = \left| \sigma^c(\mathbf{X_c}) - \sigma^s(\mathbf{X_c}) \right|. \tag{18}$$

We presented in this section a ZZ-type error indicator based on an assessment of the smoothness of the solution of the PDE. We present in the next section of the article a residual-type error indicator.

3.3.2. Residual-type error indicator

Residual-type error indicators are based on an estimation of the residual of the PDE at locations where it is not enforced as part of the solution process. This type of estimator has been widely used in literature: for example, in Ref. [49,50]. In this work, we selected the corners of the Voronoi cell surrounding a considered collocation node $\mathbf{X_c}$ to compute the residual of the PDE. The residual error at a collocation node $\mathbf{X_c}$ is calculated as the average of the residual of the PDE at each Voronoi center $\mathbf{X_{vi}}$ of the Voronoi cell associated to $\mathbf{X_c}$ (see Fig. 11).

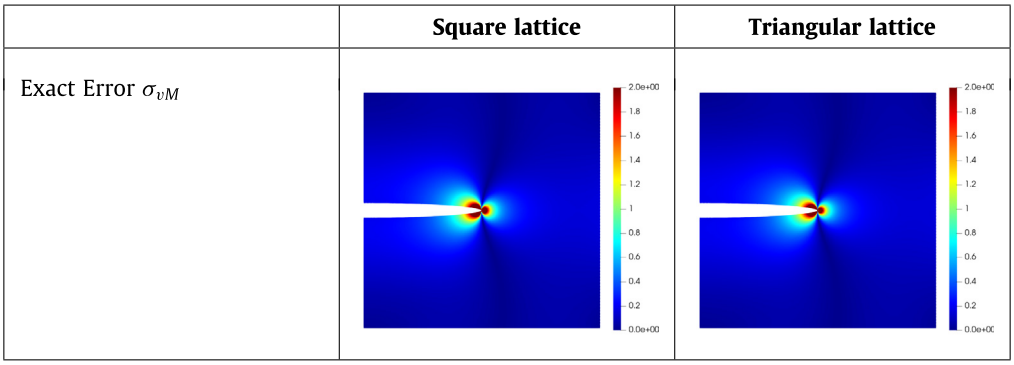


Fig. 12. Exact von Mises stress error for the problems of a plate with an elliptical hole under biaxial loading. The exact error is shown for a square lattice discretization of the domain (left) and for a triangular lattice discretization of the domain (right). We see that both node arrangements lead to the same pattern of the exact error.

The residual at each Voronoi center $\mathbf{X_{vi}}$ is approximated using the solution at the neighboring collocation nodes using the GFD method. For a collocation node $\mathbf{X_c}$ for which the associated Voronoi cell has q corner points $\mathbf{X_{vi}}$, we calculate the error indicator $e(\mathbf{X_c})$ as follows:

$$e(\mathbf{X_c}) = \frac{1}{q}\sum_{i=1}^{q} |\nabla \cdot \boldsymbol{\sigma}(\mathbf{X_{vi}}) + \mathbf{b}(\mathbf{X_{vi}})|. \tag{19}$$

3.3.3. Parameter variation and indicators comparison

We compare in this section the error obtained from the indicators presented in Sections 3.3.1 and 3.3.2. We show the impact of some parameters of the methods on the calculated error, compare the error indicators to the true error, and also compare the convergence rates.

We start by presenting a comparison of the spatial pattern of the error for the two benchmark problems considered. For this, we selected discretizations composed of approximately 200,000 nodes for the plate with an elliptical hole and 140,000 nodes for the plate with a circular hole.

The exact error for the regular nodes arrangements presented in Section 2 (i.e., square lattice and triangular lattice for 2D problems) is presented in Fig. 12 for the problem of a plate with an elliptical hole and in Fig. 13 for the problem of a plate with a circular hole.

We observe that, for both problems, the pattern of the exact error is the same for both discretization techniques. For the first problem, we see that the error is the highest close to the elliptical hole and in the region where the stress is the largest (see Fig. 9(a)). For the second problem, two regions can be identified, both close to the hole. A region at the top of the hole which corresponds to the region where the stress is the largest and a region in the middle of the considered portion of the hole. This second region corresponds to a region where the stress solution in terms of von Mises stress is the lowest and changes rapidly along the hole (see Fig. 9(b)).

We compared the results obtained for an "unweighted" indicator (i.e., $w(\mathbf{X}, \mathbf{X_c}) = 1$ in Eq. (11)) and for a "weighted" indicator. We selected a 4th order spline for the radial basis function w in Eq. (11). The equation of the spline is presented in Eq. (20).

$$w(s) = \begin{cases} 1 - 6s^2 + 8s^3 - 3s^4 & \text{if } s \leq 1 \\ 0 & \text{if } s > 1. \end{cases} \tag{20}$$

For a node $\mathbf{X}$ within a support of radius R_c of a collocation node $\mathbf{X_c}$, the weight function w based on the 4th order spline is:

$$w(\mathbf{X}, \mathbf{X_c}) = w(s) \quad \text{with} \quad s = \frac{\|\mathbf{X} - \mathbf{X_c}\|_2}{R_c}. \tag{21}$$

We observed little difference between the weighted and the unweighted indicators. We selected the weighted error indicator because we expect that such indicator be more appropriate in case large stencils are selected. The results that we present for the ZZ-type error indicator are, therefore, only based on weighted indicators.

We also compared the results obtained from an "indirect" computation of the indicator to results obtained from a "direct" computation of the indicator. The "indirect" computation of the indicator uses the computation of the individual components of the stress tensor $\sigma^s_{11}(\mathbf{X_c})$, $\sigma^s_{12}(\mathbf{X_c})$ and $\sigma^s_{22}(\mathbf{X_c})$ to compute $\sigma^s_{vM}(\mathbf{X_c})$. The "direct" computation of the indicator uses the computation of the von Mises stress components $\sigma^s_{vM}(\mathbf{X_{vi}})$ at each support nodes $\mathbf{X_{vi}}$ of $\mathbf{X_c}$ to compute $\sigma^s_{vM}(\mathbf{X_c})$. The results obtained from both approaches present a similar trend. Both approaches should be equivalent for smooth fields. Therefore, we decided to present only results obtained from the direct computation method in this article.

Figs. 14 and 15 show error indicators obtained from the weighted configurations of the ZZ-type error indicator, considering a direct computation of $\sigma^s_{vM}(\mathbf{X_c})$ and error indicators obtained from the residual-type indicator. The results for each of these indicators are shown for both the square and triangular lattice discretizations. We presented the error indicator using a logarithmic color scale as it allows a better identification of the different error zones of the solution. The amplitude of the scale has been set constant for the ZZ-type error indicator and for the residual-type error indicator to facilitate the analysis of the results.

The results obtained for the plate with an elliptical hole show that the discretization method selected for the interior of the domain impacts significantly the pattern of the error indicator. For both discretization techniques, we can observe lines where the computed error indicator is lower. This phenomenon is the most significant for the square lattice discretization techniques. Such lines are not observed for the residual-type error indicator computed for the domain discretized using a triangular lattice.

The trend of the results obtained for the problem of a plate with a circular hole is the same as the trend of the results obtained for the problem of a plate with an elliptical hole. The discretization technique impacts the pattern of the error. The error pattern is the most uniform for the residual-type error indicator. It can be noticed, however, that the ZZ-type error indicator allows for the identification of a zone, close to the middle of the considered portion of the cavity, where the error is greater. The exact error presented in Fig. 13 shows that this zone corresponds to a zone where the error is significant. This zone is not identified with the residual-type error indicator. The zone at the top of the considered portion of the cavity is identified as a

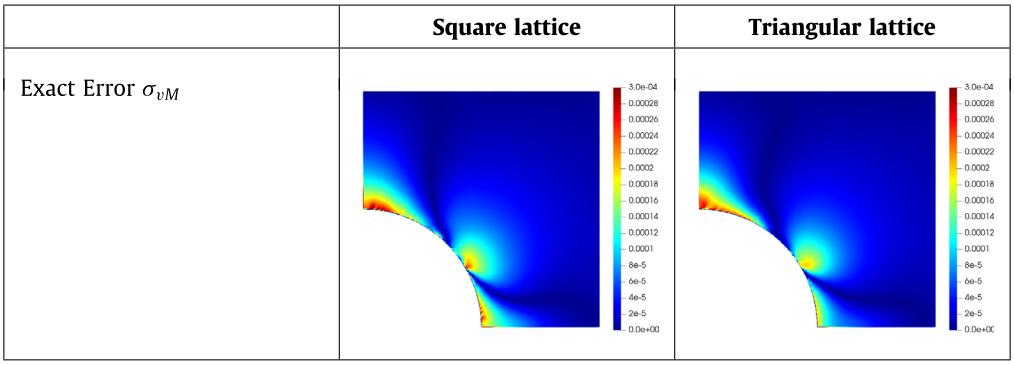


Fig. 13. Exact von Mises stress error for the problems of a plate with a circular hole under remote stress loading. The exact error is shown for a square lattice discretization of the domain (left) and for a triangular lattice discretization of the domain (right). We see that both node arrangements lead to the same pattern of the exact error.

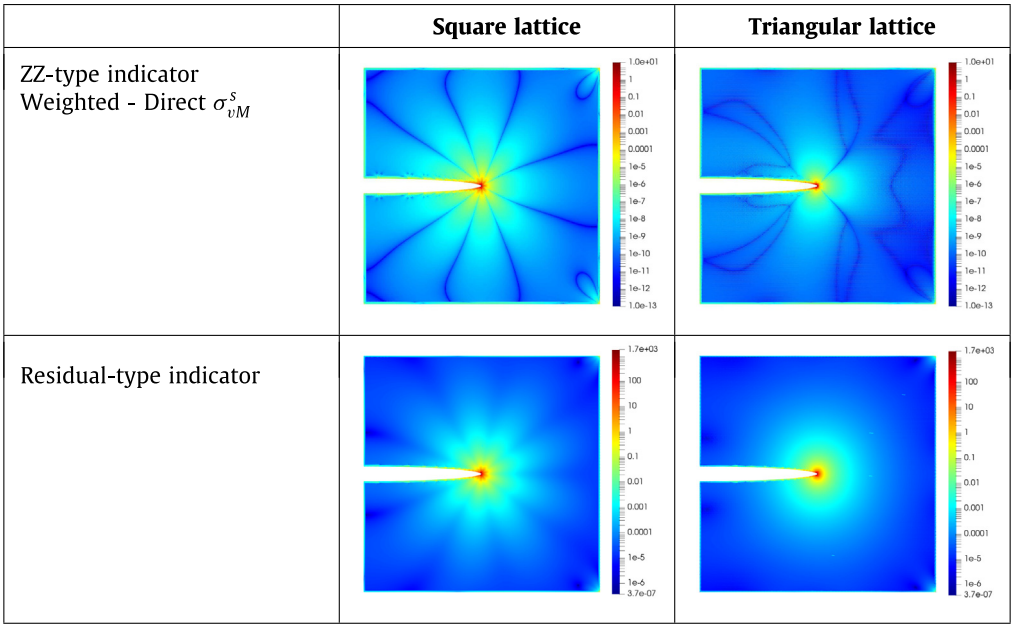


Fig. 14. Comparison of the error pattern for the ZZ-type error indicator and for the residual-type error indicator for the problem of a plate with an elliptical hole. The results are shown for square and triangular lattice discretizations of the interior of the domain. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

zone where the error indicator is high. However, the computed error indicators are not greater than in other zones close to the cavity. The gradient in this zone is not as high as in the middle of the considered portion of the cavity. This explains why the computed error indicators are not larger at this location of the cavity.

The trend of the results presented in Figs. 14 and 15 is in favor of the residual-type indicator because this indicator appears to be the least affected by the discretization of the geometry.

The computation of the ZZ-type error indicator necessitates the selection of a stencil size or radius R_c since we considered the distance criterion for the selection of the stencil nodes. The results presented in Figs. 14 and 15 were computed based on the same stencil as the stencil considered for the solution of the collocation problem. We present in Fig. 16 results showing the impact of the size of the stencil on the pattern of the error indicator. We considered two scaling factors applied to the size of the stencil selected for the solution of the collocation problem. We preferred applying the scaling factor to the number of selected stencil nodes rater than on the stencil radius R_c because our node selection algorithm is based on a target number of stencil nodes. We selected domains discretized based on a triangular lattice and a direct computation of σ^S_{vM} considering a weighted moving least square approximation.

The results presented in Fig. 16 show, for the problem of a plate with an elliptical hole, that the scaling factor has little impact on the pattern of the error indicator. The impact of the scaling factor is more significant for the problem of a plate with a circular hole. We observe that the intensity of the zones where the error is the lowest is more significant for a scaling factor of 0.8. In these zones, the error indicator does not represent well the pattern of the exact error and is expected to lead to an incorrect refinement of the domain. A scaling factor of 1.5 leads to similar results than the base case (i.e., scaling factor of 1.0 presented in Fig. 15).

3.3.4. Discussion

Both the ZZ-type error indicator and the residual-type error indicator enable the identification of zones of the point cloud where the error is the greatest. We observe that a discretization of the interior of the domain based on a triangular lattice leads to an error indicator which is less dependent on the geometry than

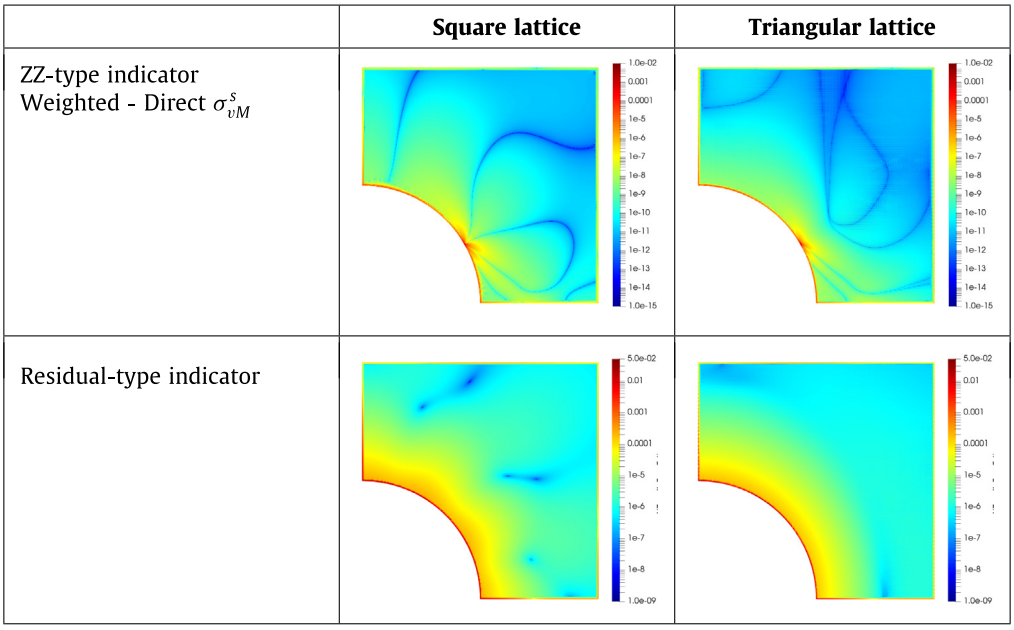


Fig. 15. Comparison of the error pattern for the ZZ-type error indicator and for the residual-type error indicator for the problem of a plate with a circular hole. The results are shown for square and triangular lattice discretizations of the interior of the domain. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

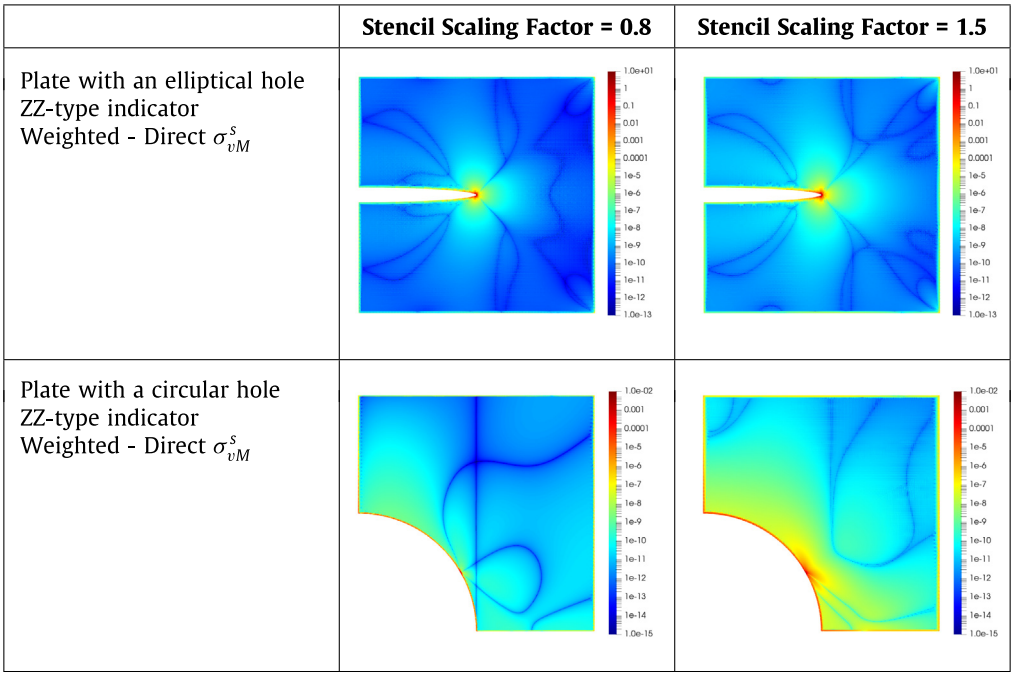


Fig. 16. Impact of the size of the stencil considered in the computation of the ZZ-type error indicator. Results are shown for both benchmark problems for two scaling factors applied to the selected stencil size considered in the solution of the collocation problem.

one based on a square lattice. This result is observed even though both discretization methods lead to similar exact error patterns as per the results presented in Figs. 12 and 13.

The weighted-direct computation of σ^s_{vM} is the configuration of the ZZ-type error indicator that leads to the best results although little difference is observed between the weighted and the unweighted indicators. The selection of a stencil larger or smaller than the one used as part of the solution of the global collocation model does not improve the indicator. The residual-type error indicator leads to smoother results which are not affected by the discretization of the geometry. However, the computational cost of this indicator is much greater than for the ZZ-type error indicator. Voronoi corner nodes are computed for all the nodes of the domain. The stencil of each Voronoi corner node needs to be determined and the derivatives approximated.

We show a comparison of the time required to compute the error indicators for the two considered problems in Fig. 17. We present these results in the form of a ratio of the indicator computation time to the time needed to assemble and solve the collocation problem. The results are indicative as they depend heavily on the method selected to solve the linear system and on the number of threads/processes involved in each step of

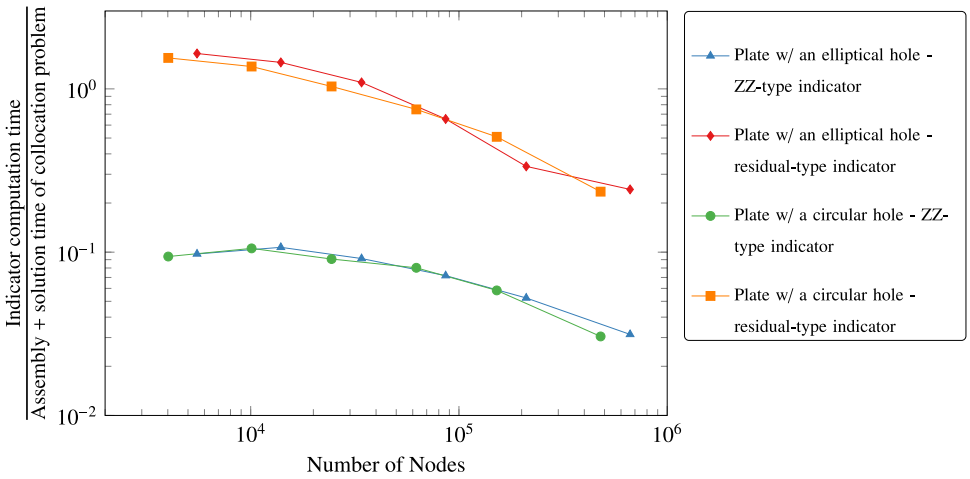


Fig. 17. Comparison of the ratio of the indicator computation time to the time needed to assemble and solve the collocation problem. Both problems lead to similar results. The computation time of the residual-type error indicator is approximately 10 times the computation time of the ZZ-type error indicator. The computation of the residual-type error indicator corresponds to between 165% and 23% of the assembly and solution time of the collocation problem. The computation of the ZZ-type error indicator corresponds to between 11% and 3% of the assembly and solution time of the collocation problem.

the solution process. The computation of the indicators at each node of the domain is independent from the computation of the indicator at other nodes of the domain. Therefore, both error indicators can be parallelized with no extra effort. We used two threads to assemble the linear system and compute the error indicator and solved the linear system using a LU factorization (one thread, one process). We see from Fig. 17 that the computation time of the residual-type error indicator is approximately 10 times the computation time of the ZZ-type error indicator. For both indicators, the ratio mostly decreases as the number of nodes increases. Such a result is expected since the computation of the error indicators scales linearly with the number nodes while the solution time of the collocation problem increases at an increasing rate when solved with LU factorization. The problem of the plate with an elliptical hole and the problem of a plate with a circular hole lead to similar results. The computation of the residual-type error indicator corresponds to between 165% and 23% of the assembly and solution time of the collocation problem. The computation of the ZZ-type error indicator corresponds to between 11% and 3% of the assembly and solution time of the collocation problem for the considered discretizations.

Based on these results, the ZZ-type indicator appears to be a better choice in terms of computational cost.

To complete the comparison of both error indicators, we computed ZZ-type and residual-type error indicators for the problem of a plate with an elliptical hole and the problem of a plate with a circular hole for various node densities obtained with a global refinement of the domains. We selected for both problems a discretization of the domain based on a triangular lattice. The results are presented in Figs. 18 and 19. The ZZ-type error indicator is based on a weighted-direct computation of σ^s_{vM}. We computed the exact error in terms of the l_2 relative error norm for the von Mises stress component and compared it to a l_2 relative error norm where the "smooth" von Mises stress field is considered as

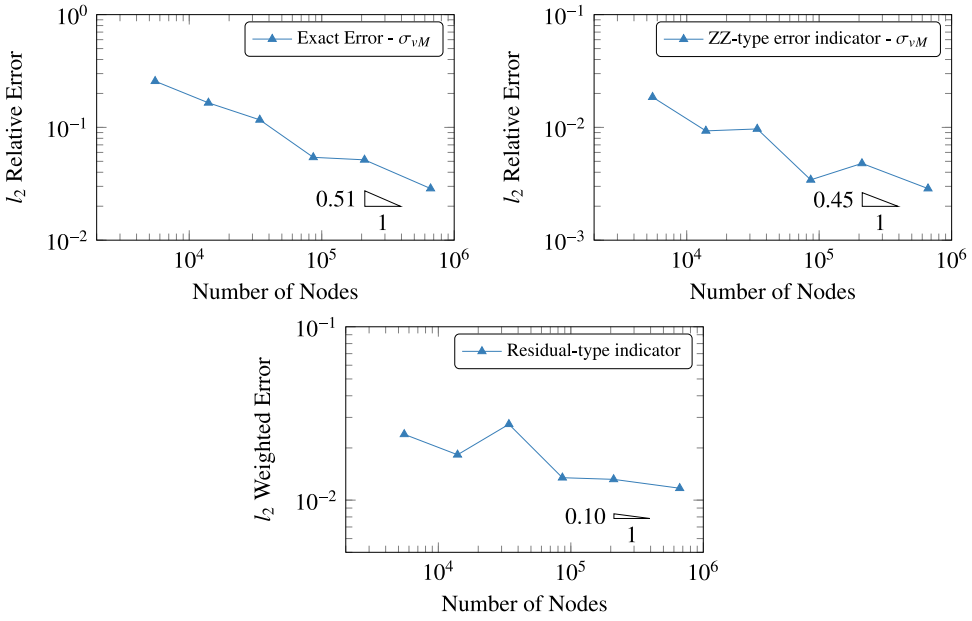


Fig. 18. Plate with an elliptical hole — Comparison of the exact and estimated error. The results are presented in terms of the l_2 relative error norm for the exact error on the von Mises stress and for the ZZ-type indicator of the error on the von Mises stress. The results are presented in terms of the l_2 weighted error norm for the residual-type indicator.

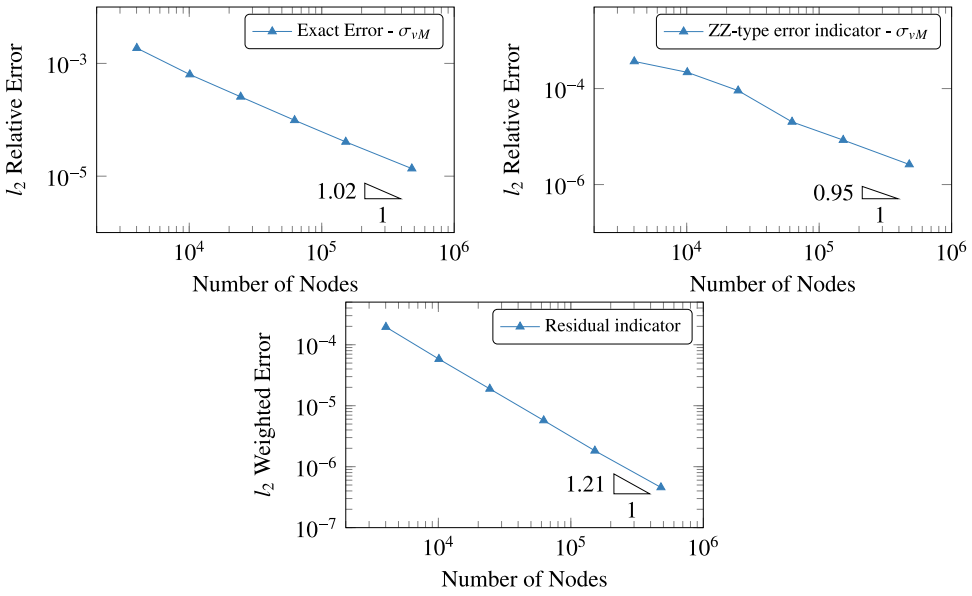


Fig. 19. Plate with a circular hole — Comparison of the exact and indicative error. The results are presented in terms of the l_2 relative error norm for the exact error on the von Mises stress and for the ZZ-type indicator of the error on the von Mises stress. The results are presented in terms of the l_2 weighted error norm for the residual-type indicator.

the reference solution. We also computed the l_2 weighted error norm of the residual-type indicator computed using Eq. (22). We selected this norm since a reference solution is not computed as part of the indicator.

$$l_2 W(e) = \frac{\sqrt{\sum_{k=1}^{n} \left(e\left(\mathbf{X_c}\right)\right)^2}}{n}. \tag{22}$$

We observe from the results presented in Fig. 18 and in Fig. 19 a convergence of both error indicators for both problems. The convergence rate is higher for the problem of a plate with a circular hole. The convergence rate of the l_2 weighted error norm computed for the residual-type indicator is 0.1 for the problem of a plate with an elliptical hole and 1.2 for the problem of a plate with a circular hole. This result is expected since the solution is smoother for this problem than for the problem of a plate with an elliptical hole. For both problems, the convergence rate of the l_2 relative error norm is similar when the exact or smooth von Mises stress components are considered.

Based on the results presented in this subsection, we selected the ZZ-type error indicator, using a weighted-direct computation of σ^s_{vM}, as the main input to the discretization refinement scheme presented in the next section. We preferred this indicator rather than the residual-type error indicator because of its reduced computational expense and because of its simplicity.

3.4. Discretization refinement

We describe in this subsection a scheme to refine a point cloud based on the results obtained from *a posteriori* error indicators. We use the information of the smart cloud to place new nodes on the exact geometry of the domain and to set the boundary conditions of the updated collocation model.

A posteriori error indicators allow the identification of the areas of the domain where the error is expected to be the greatest. Several techniques can then be used to improve the solution in these zones. The most common ones are *h*- and *p*-adaptivity. *h*-adaptivity consists of an increase of the node density in the zone where the error is the greatest. Such type of adaptivity is the most commonly used in the literature. For instance, it was used by Benito [51], Davydov [29], Gavete [27] or Slak and Kosec [52] within the framework of point collocation. *h*-adaptivity has also be considered in literature based on geometric indicators [53,54]. *p*-adaptivity is another technique which consists in an increase of the order of the approximation. Liszka et al. and Duarte et al. made use of *p*-adaptivity, in the framework of meshless methods, as part of the hp-meshless cloud method [55,56]. Jancic et al. showed the benefits of *p*-refinement for a Poisson problem with a strong source within the domain [57] in the framework of the radial basis function-generated finite difference method (RBF-FD).

Our scheme is based on *h*-adaptivity. We selected this technique to be able to perform successive refinement iterations and reduce the observed error as much as possible. An *h*-adaptive scheme is based on the repetition of a succession of steps. The main steps that we followed are presented in the form of a pseudo code in Fig. 20.

The algorithm is composed of three main steps:

1. Identification of the refinement areas;
2. Placement of new nodes;
3. Generation of the updated collocation model

We describe these steps in the subsections below.

3.4.1. Identification of the refinement areas

The error indicators presented in Section 3 are computed at the collocation nodes. Therefore, we decided to identify the areas of the domain to be refined based on a selection of marked collocation nodes. The selection of these nodes is based on the computed error indicator. Different criteria can be used to determine the collocation nodes to be marked for local refinement of the domain.

An error indicator threshold could be selected by the user. However, such a threshold is problem specific and cannot be easily generalized to all problems. To help the definition of a node selection criterion, we presented the computed error indicator, sorted from the lowest error to the highest error in Fig. 21 for the considered benchmark problems. The presented error indicators are ZZ-type error indicators, computed based on the parameters presented in Section 3.3. Vertical bars are used to visualize the percentage of nodes in the different error zones.

We observe three distinct zones from the results presented in Fig. 21. A limited number of nodes have a very low error (approximately below 10^{-8} for both problems). A majority of the nodes have an error between 10^{-8} and 10^{-4} for the plate with an elliptical hole and between 10^{-8} and 10^{-5} for the plate with a circular hole. Finally, a limited number of nodes have an error larger than 10^{-4} and 10^{-5} for the plate with an elliptical hole and the plate with a circular hole, respectively. The zones of low and high error represent each approximately 5% of the total number of nodes. Based on these observations, we decided to select the nodes of highest error based on a defined fraction f of the total number of nodes n. Therefore, the fn nodes having the highest error are selected for local refinement. The impact of the selected threshold on the convergence rate is analyzed in a later section (see Section 3.4.4). This approach has similarities with the Dörfler marking strategy [58] used in the framework of adaptive finite element method [59].

In order to obtain as smooth a refinement pattern as possible, we also selected, for local refinement, all the stencil nodes associated with the selected nodes of highest error. The results presented in Fig. 22 show the benefits of this approach for the problems of a plate with an elliptical hole and for the plate with a circular hole. We presented first the pattern of the error indicator, based on a ZZ-type error indicator. Then, we present the nodes selected based on a fraction of the nodes of highest error. 10% of the nodes showing the largest computed error indicator are marked in red. Finally, we show all the nodes marked for local refinement based on the method described above (i.e., the nodes of highest error and their corresponding stencil nodes). We see that the boundaries of the zones marked for local refinement are smooth and correspond to the zones of highest computed error indicator.

3.4.2. Placement of new nodes

We presented in Section 3.4.1 how we select the zones for local refinement. Once these zones are identified, we add new nodes to the domain with the aim of placing them as far as possible from existing nodes to avoid ill-conditioning of the refined discretization. The refinement process consists in three steps:

1. Refinement of the boundaries of the domain;
2. Refinement of the interior of the domain;
3. Deletion of the nodes which are too close to other nodes.

Step 1: Refinement of the boundaries of the domain. The boundaries of the domain are refined first. We use boundary elements to discretize surfaces for 3D problems and facilitate the implementation of the visibility criterion as described in Ref. [2]. Those elements are used as part of the refinement process to

h-adaptivity algorithm

Initial discretization of the domain;

Application of boundary conditions and generation of the initial collocation input file;

While (Continue adaptive refinement = true)

  Solution of the collocation problem;

  Computation of the *a posteriori* error indicator;

  Assessment of the need for a new refinement iteration;

  if (Refine the discretization = true)

    Identification of the nodes of highest error for discretization refinement;

    Placement of new nodes in the areas of high error;

    Application of boundary conditions and generation of the updated collocation input file;

  else

    Continue adaptive refinement = false.

loop

Fig. 20. h-adaptivity algorithm considered for the presented adaptive refinement method.

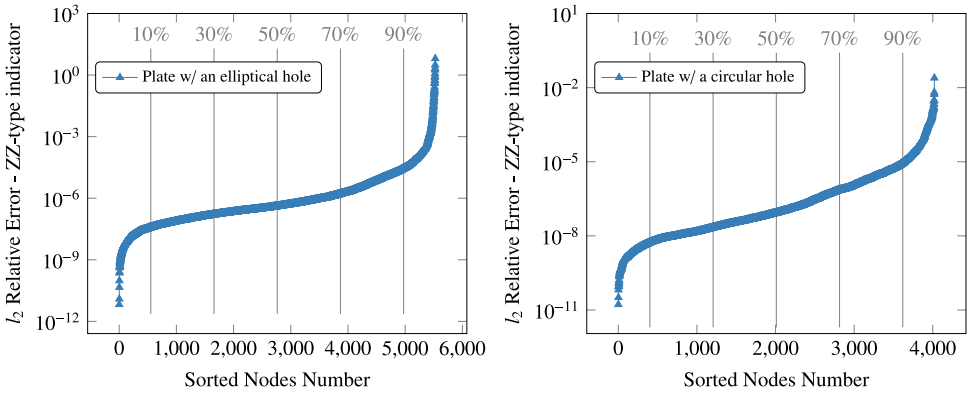


Fig. 21. Distribution of the error in terms of ZZ-type error indicator for the problem of a plate with an elliptical hole (left) and for the problem of a plate with a circular hole (right). Three distinct zones are observed on these graphs. Less than 10% of the nodes have a very low error (nearly zero). Approximately 80%–90% of the nodes have an error in the similar range (10^{-8}-10^{-4} for the plate with an elliptical hole, 10^{-8}-10^{-5} for the plate with a circular hole). Less than 10% of the nodes have an error much larger than the other nodes.

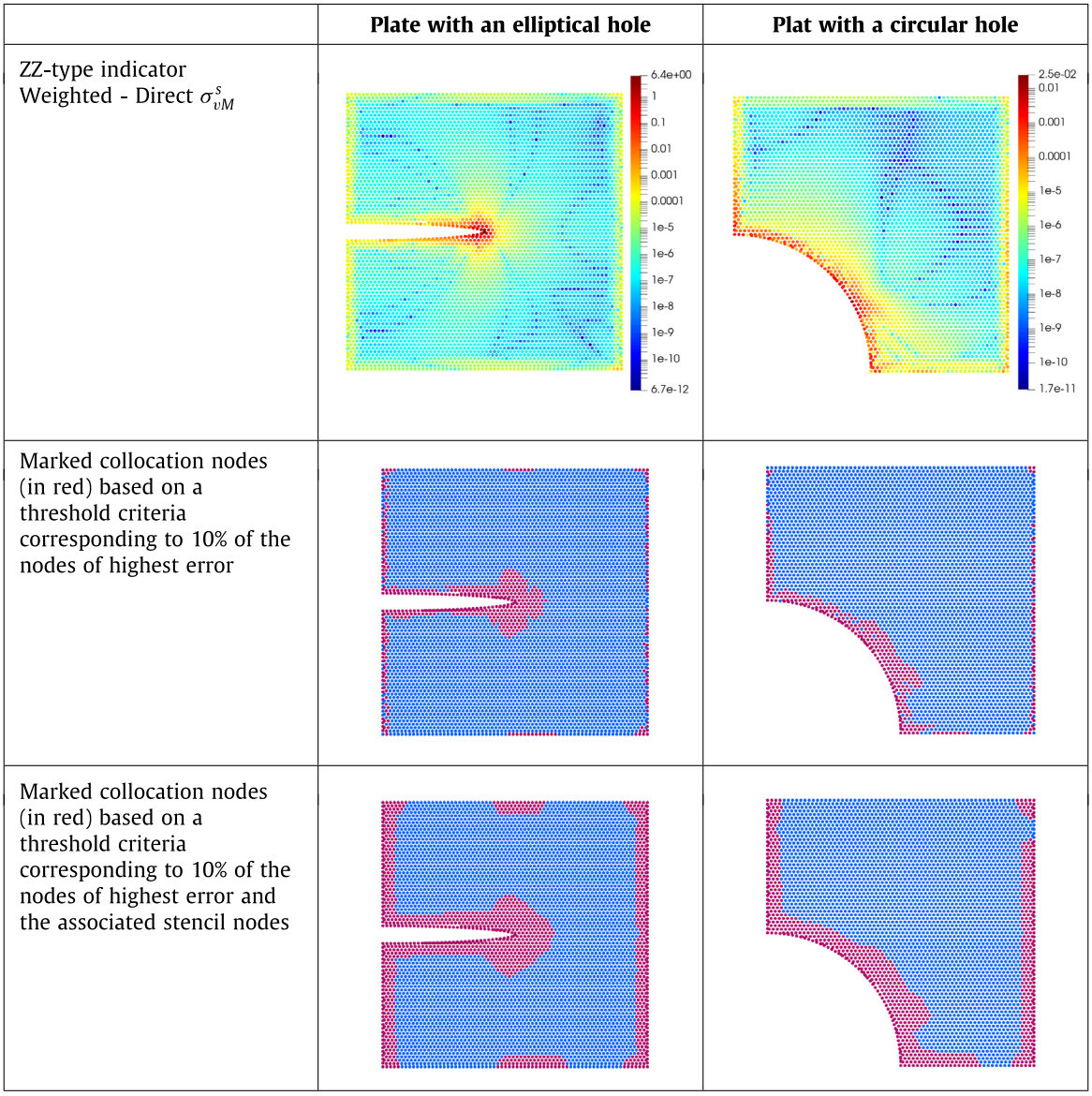


Fig. 22. Pattern of the ZZ-type error indicator and selection of the nodes for local refinement based on a fraction of the nodes of highest error and based on a fraction of the nodes of highest error and all the stencil nodes of these collocation nodes. The results are presented for the problems of a plate with an elliptical hole and for the plate with a circular hole.

facilitate the placement of new nodes far from existing nodes. The approach for 2D and 3D problems is different. We present both approaches below.

The elements connected to the boundary nodes marked for local refinement are selected for local refinement. In 2D, new nodes are added in the center of the elements selected for refinement, and then projected onto the CAD boundary of the domain. The elements are then split into two new surface elements. In 3D, the new nodes are added in the center of the edges of the elements selected for refinement, and then projected onto the CAD boundary of the domain. The refined surface elements are split into four new surface elements if the surface is discretized with triangular elements. The element normal vectors are then re-computed.

Fig. 23 shows a domain Ω with boundary Γ_Ω for the case of a 2D problem (left) and a 3D problem (right). A collocation node marked for local refinement is shown in red color. The edges of surface elements connected to the marked collocation node are shown by dashed lines in blue color. New boundary nodes are added in the vicinity of the marked collocation node and are shown in green color bounded by dotted lines. The nodes obtained are part of the refined discretization.

In 2D, the projection of new nodes onto the boundary of the domain consists only in a projection of the new node onto the edge of the domain. In 3D, the new boundary nodes, not located at the intersection between two faces, are projected onto the surface, parent to the boundary element. If some edges of the boundary elements are located at the intersection of faces of the domain, the new nodes placed in the middle of those edges are projected onto the intersection of faces. Those intersections often correspond to edges of the domain. Therefore, such a projection allows for an accurate refinement of the edges of the domain. This projection is illustrated by Fig. 24. The new node, in the middle of the refined edge, shown in red dotted line should be projected onto the edge at the intersection between the gray and orange

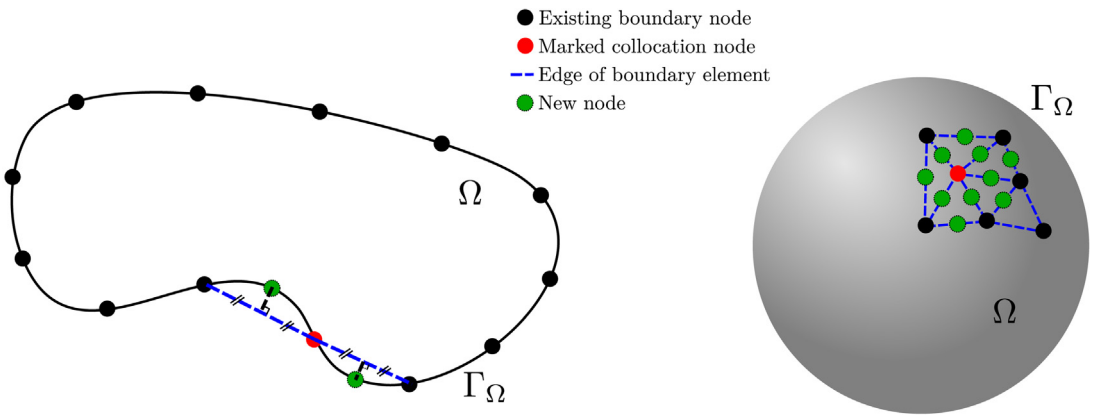


Fig. 23. Computation of the position of new boundary nodes. Collocation nodes marked for local refinement are shown in red color. New boundary nodes, shown in green color bounded by dotted lines, are added in the middle of the edges of the elements connected to the collocation node and projected onto the surface or edge of the domain. The obtained nodes are part of the refined discretization. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

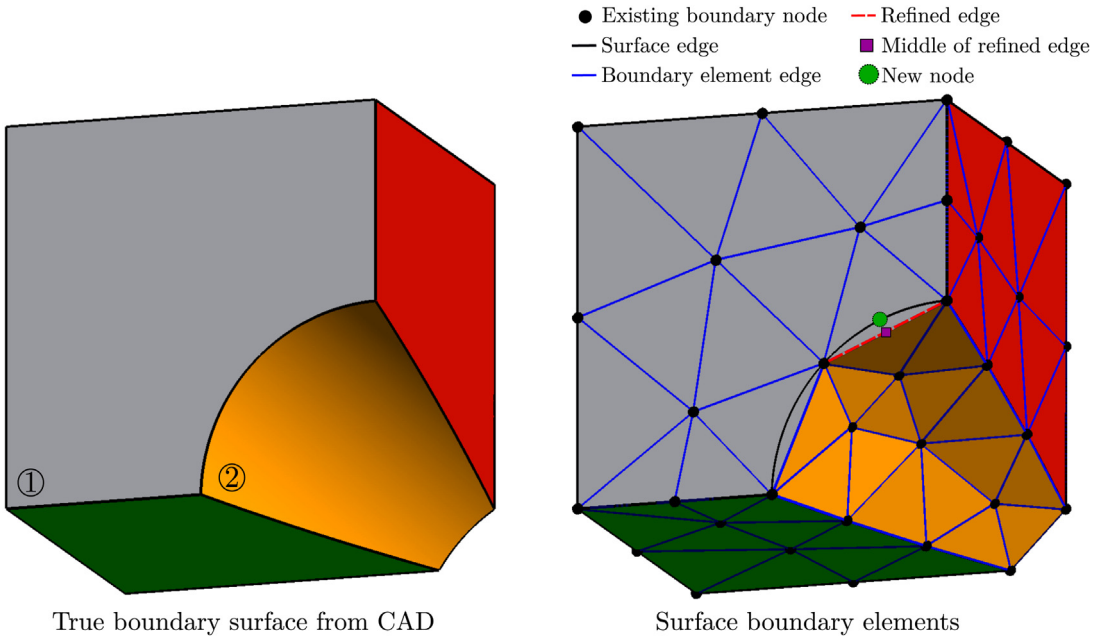


Fig. 24. 3D edge refinement. The true faces of the CAD geometry are shown on the left. The faces are discretized using boundary elements. The edge shown in the red dotted line is located at the intersection between two discretized surfaces. This edge is refined. The middle point of the edge, shown as a purple square, should be projected onto the edge at the intersection between the gray and orange surfaces, marked ① and ② respectively since a projection onto one of these surfaces would lead to a node outside of the domain.

surfaces, marked ① and ② respectively. A projection on either surface would lead to a node outside of the domain.

To facilitate the projection process, the reference of the CAD topological entities (edge or face) should be associated with each node of the smart cloud. For 3D problems, the reference of the parent edge should also be associated with the node of the smart cloud node located at the intersection between two surfaces.

Step 2: Refinement of the interior of the domain. The zones around the interior nodes marked for local refinement are considered once the new boundary nodes have been added to the refined discretization. Voronoi diagrams are used to place the new nodes as far as possible from existing nodes and new boundary nodes. In two dimensions, all the nodes located on the edges of a Voronoi cell are located at equal distance from two adjacent nodes. The corners of a Voronoi cell are located at equal distance from three adjacent nodes as shown in Fig. 25.

Voronoi diagrams are computed at all the collocation nodes marked for local refinement. The existing nodes of the discretization and the new boundary nodes are considered in the computation of the Voronoi diagrams. Only the Voronoi cells around the marked collocation nodes are of interest. All the corners of the selected cells are added to a list of new candidate nodes. The duplicated nodes are deleted from this list.

Step 3: Deletion of the nodes too close to another node. The final step of the node placement process consists in the deletion of the nodes too close from other nodes. For this, a characteristic length is computed at each node of the initial discretization. It corresponds to the minimum distance between the considered node and its adjacent closest node (also based on the initial discretization). The characteristic length associated with a new node of the domain is the characteristic length of its closest node

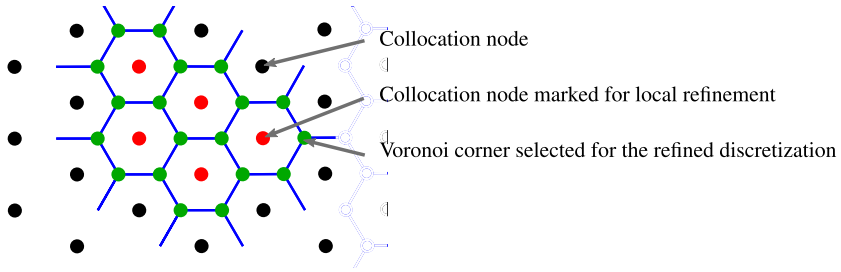


Fig. 25. Computed Voronoi diagram for a set of collocation nodes — Use for new node selection.

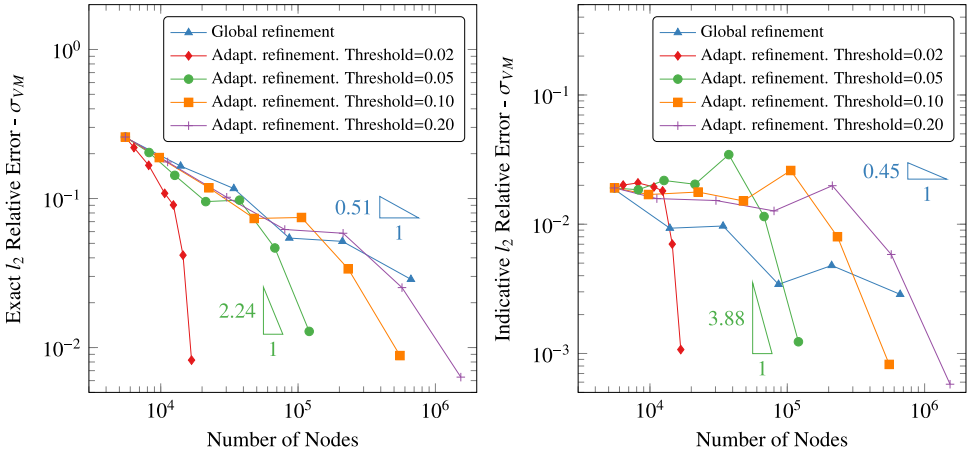


Fig. 26. Exact and indicative l_2 relative error norm for the problem of a plate with an elliptical hole. Results from adaptive refinement based on a ZZ-type error indicator for adaptive threshold ratio between 0.02 and 0.20 are compared results obtained from a global refinement of the domain discretization.

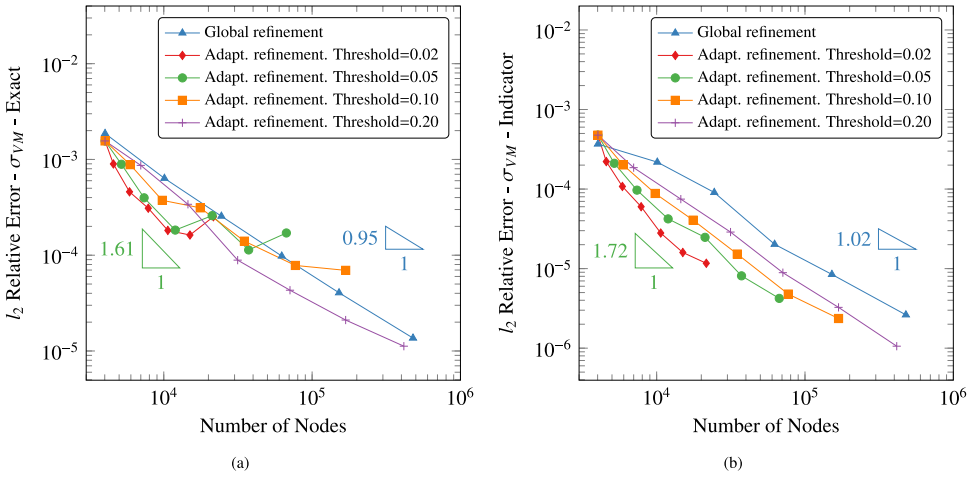


Fig. 27. Exact and indicative l_2 relative error norm for the problem of a plate with a circular hole. Results from adaptive refinement based on a ZZ-type error indicator for adaptive threshold ratio between 0.02 and 0.20 are compared results obtained from a global refinement of the domain discretization.

from the initial discretization. This minimum distance is noted h_{Loc}. The minimum allowed inter-node distance, noted h_{Min}, is the ratio of h_{Loc} to a factor α:

$$h_{\text{Min}} = \frac{h_{\text{Loc}}}{\alpha}. \tag{23}$$

We selected $\alpha = 3$ in our work.

3.4.3. Generation of the updated collocation model

We presented in Section 3.4.2 a method to place new nodes on the boundary of the domain and in the domain based on a selection of marked collocation nodes. The generation of the updated collocation model is the next step.

The process is the same as the one described in Section 2.3. The boundary nodes of the smart cloud have references to their parent edge or surface and also references to the boundary conditions associated to these entities. New boundary nodes can be located at the intersection of multiple surfaces. In this case, we apply first the boundary conditions of the surface associated to stress loading boundary conditions, if any. Then, we set the Dirichlet boundary conditions and finally the stress-free boundary conditions.

3.4.4. Node selection threshold sensitivity

We assessed in this section the impact of the node selection threshold on the convergence of the error using model adaptivity. We considered node selection threshold values between 0.02 and 0.2 based on the trend of the results presented in Fig. 21. The results are presented in Figs. 26 and 27 for the benchmark problems introduced in Section 3.2. We compared the results in terms of "exact" l_2 relative error norm and in terms of "indicative" l_2 relative error norm. We also compared the results obtained from the adaptive refinement technique presented in this article to results obtained using global refinement (uniform refinement of the domain).

The results show that the convergence rate is the largest for a threshold of 0.02 for the problem of a plate with an elliptical hole. After six iterations of iterative refinement, the error obtained is more than three times as low as those obtained with the largest node density considered for the case of global refinement. Such a solution is obtained with approximately 40 times fewer nodes. For the problem of a plate with a circular hole, a threshold of 0.02 leads to a non monotonic error reduction. A node selection threshold of 0.2 leads to results similar to the results obtained using a global refinement strategy. We selected a threshold of 0.05 for the problems solved in the next sections of the article. This value is a good compromise between rapid convergence of the solution and non monotonic error reduction.

3.4.5. Node relaxation

The addition of new nodes in the domain leads to regions of relatively high node density and regions of relatively low node density. Node relaxation can be used to smooth-out transitions between these regions. It consists in the application of a Laplace smoothing operator to nodes of the point cloud to obtain a uniform discretization. We used the relaxation method of the library Medusa for this purpose [60].

Rather than applying node relaxation to the entire domain, we preferred a local approach to smooth-out only the transitions between fine and coarse regions. At each node of the refined domain, we select the 30 nearest neighbor nodes for 2D problems. Then, we count the number of nodes in the "fine" region of the domain. The nodes marked for adaptive refinement and the new nodes are considered located in the "fine" region of the domain. Node relaxation is performed if between 20% and 80% of the nearest neighbor nodes belong to the "fine" region of the domain.

The radius R_{rc} is the distance between the farthest selected neighbor node and the considered collocation node $\mathbf{X_c}$. All the neighbor nodes located at a distance between R_{rc} and $\frac{2}{3}R_{rc}$ are considered as boundary nodes, and fixed in position, during the relaxation process. The position of the nodes is updated once relaxation is performed. The nodes moved are no longer considered part of the "fine" region of the domain. This process is repeated for all the nodes of the discretization. The relaxation process is illustrated by Fig. 28.

We show in Figs. 29 and 30 the impact of node relaxation on the convergence of the adaptivity scheme and on the error indicator.

We observe from these figures that node relaxation improves the convergence rate of the error for both benchmark problems considered. This result is expected since the node discretization is more uniform. We notice, however, that the trend is reversed for the two finest discretizations considered in Fig. 30. The error for these two discretizations is larger than the error obtained for the previous step of the considered iterative refinement schemes. This means that the change in node density of the discretizations leads to imbalanced stencils that negatively affect the solution. For these two discretizations, we cannot explain why the adaptive iterative scheme considering node relaxation leads to a larger error than the scheme that does not consider node relaxation. The improvement is, however, not significant and its usefulness may be considered debatable given the computational effort it necessitates. We did not investigate this approach further in the context of the problems solved in the next section of the article.

3.5. Practical applications

To show the applicability of our method to more complicated test cases, we present in this section additional results for practical 2D and 3D problems. The selected problems are:

- a gear coupled to a shaft (2D);
- a closed cylinder subject to pressure (3D).

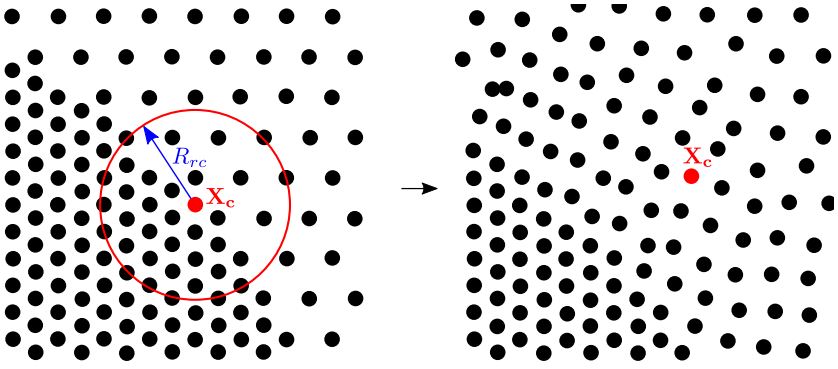


Fig. 28. Point cloud before (left) and after (right) node relaxation. Node relaxation is performed locally around collocation nodes such as $\mathbf{X_c}$ over the domain of radius R_{rc}. Relaxation is performed when between 20% and 80% of the nodes within R_{rc} are located in "fine" discretization regions.

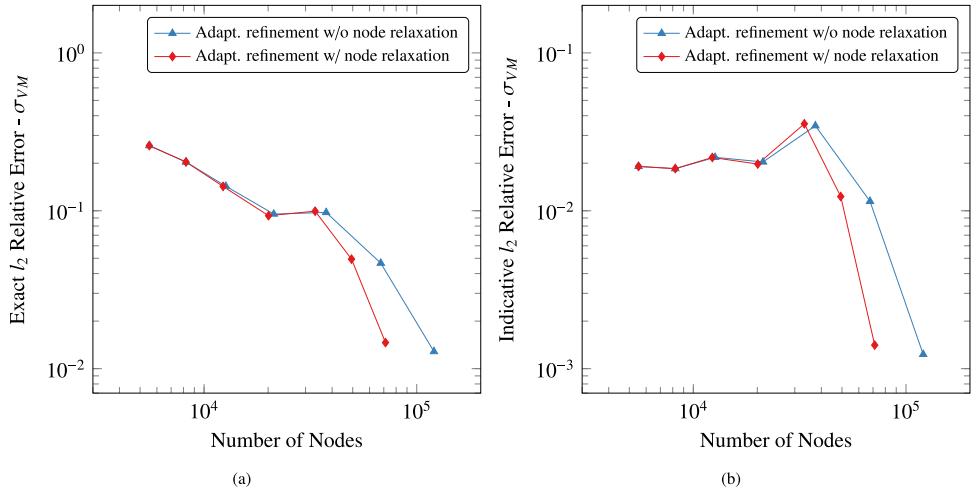


Fig. 29. Comparison of the error with and without node relaxation for the problem of a plate with an elliptical hole. The exact and indicative l_2 relative error norms are shown in subfigures (a) and (b), respectively. A node selection threshold ratio of 0.05 is selected. The error obtained with node relaxation is lower than the one obtained without node relaxation. The trend of the results is similar for both cases.

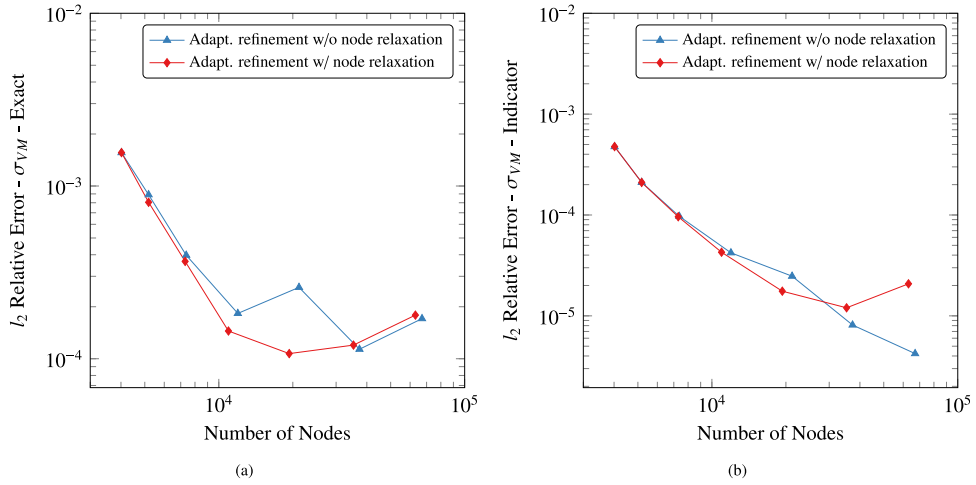


Fig. 30. Comparison of the error with and without node relaxation for the problem of a plate with a circular hole. The exact and indicative l_2 relative error norms are shown in subfigures (a) and (b), respectively. A node selection threshold ratio of 0.05 is selected. The error obtained with node relaxation is lower than the one obtained without node relaxation for the four first iteration steps. The trend of the error indicator is closer to the trend of the exact error for the case of adaptive refinement with node relaxation.

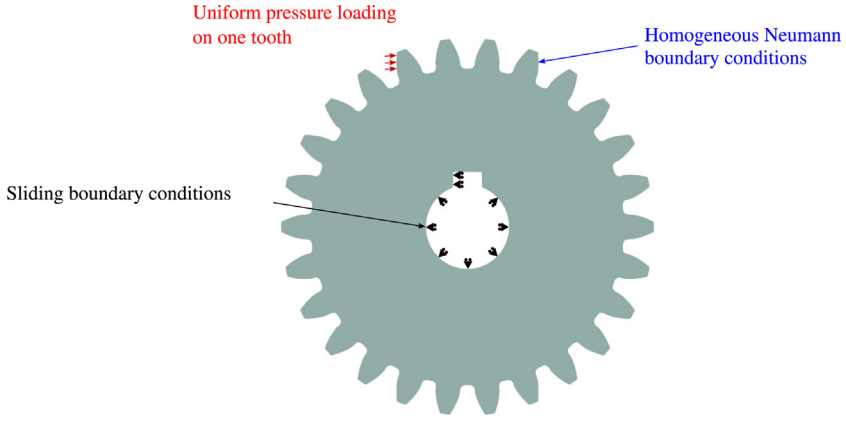


Fig. 31. Boundary conditions applied to model a gear coupled to a shaft by a key.

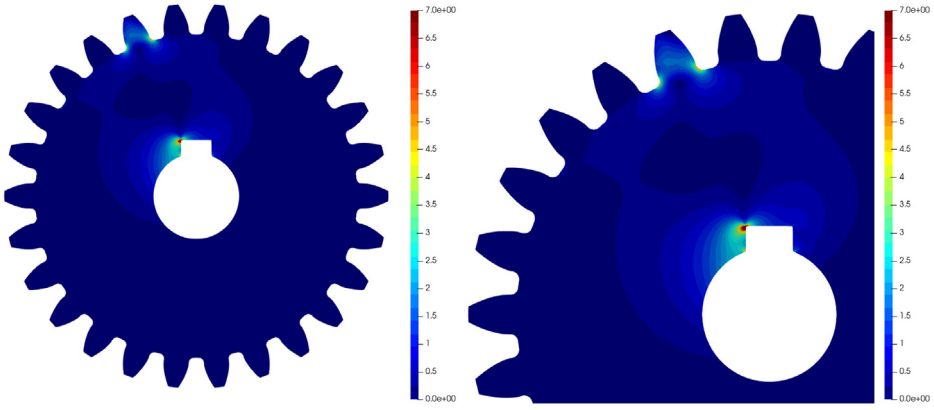


Fig. 32. Gear coupled to a shaft — Solution in terms of von Mises stress obtained from a finite element model composed of 132,665 nodes and 262,193 linear triangular elements. The results are shown for the stress range 0–7 for comparison purposes.

We generated initial discretizations of the domains, directly from the CAD geometry, using a triangular lattice and a hexagonal close-packed lattice for the 2D and the 3D problems, respectively. We used a threshold of 0.3 to select the nodes close to the boundary of the domain that shall be deleted.

We performed several model adaptivity iterations. We used the ZZ-type error indicator, based on the parameters presented in Section 3.3, to determine the areas of the domain where the error is likely to be high. We selected the areas to be refined based on a threshold ratio of 0.05. Finally, we placed new nodes on the boundaries of the domain and in the domain based on the method presented in Section 3.4.

3.5.1. Gear coupled to a shaft

We present in Fig. 31 the geometry of a gear, coupled to a shaft by a key. The gear is loaded by uniform pressure on a tooth. We used a finite element solution of the test case as a reference solution which we computed using code_aster [61] for this purpose. The finite element reference solution to this problem is shown in Fig. 32. We observed three areas of stress concentration: the top left corner of the groove and both roots of the loaded tooth.

We show in Fig. 33 the evolution of the point cloud and of the results in terms of von Mises stress through four adaptive refinement iterations. We show a general view of the domain and a detailed view of the top left corner of the groove where the stress concentration is the greatest. The results are shown for the stress range 0–7 for comparison purposes.

We observe from this figure that new nodes are placed at the roots of all the teeth at the first refinement iteration. The tooth subject to pressure loading is the tooth being the most refined. The area around the groove is also refined successively at each refinement iteration. We see from the third and fourth iterations that areas at the interface between coarse and fine discretization zones are also being refined. The zones correspond to areas of the domain where the discretization is not uniform leading to larger values of the computed error indicator.

The von Mises stress field obtained after four iterations of adaptive refinement is smooth and close to the von Mises stress field of the reference finite element solution presented in Fig. 32. We compare both solutions in greater detail in Fig. 34. We focus on the stress field on the left side of the groove and in the loaded tooth. In the area around the groove, we see no difference in the stress field obtained from both methods. At the root of the loaded tooth, we see a slightly larger stress concentration for the smart cloud collocation solution. The smart cloud has a much higher node density at the root of the loaded tooth than the finite element discretization. The inter-node distance is approximately 0.018 in the dense region of the smart cloud. For the finite element discretization, the inter-node distance is approximately 0.15. This tends to explain the larger computed von Mises stress values.

We present in Fig. 35 the computed error indicator for the results obtained from the initial discretization and the four refined discretizations. The error indicator is shown in terms of the l_2 relative error. We see that the value of the error increases at the first iteration and then reduces relatively steadily. The error reduction between the initial discretization and the final refinement iteration is approximately 63%.

3.5.2. Closed cylinder subject to pressure

We present in Fig. 36 the geometry of a closed cylinder subject to uniform pressure loading on the top of its closed end and the associated boundary condition. The bottom of the closed cylinder is fixed in the direction normal to the closed end of the cylinder. We considered only a quarter of the cylinder to reduce the computational cost. We applied Dirichlet boundary conditions in the directions normal to the surface on the symmetry planes XZ and YZ.

We show the results obtained for the initial discretization and for two refinement iterations in Fig. 37. To facilitate the analysis of the results, we also present in this figure the results for a thick "slice" of the domain. This allows a closer view of a portion of the domain. The "slice" is the intersection of a box and the geometry. The position of the box is shown in Fig. 38. We focus on the fillet between the closed top and the cylinder since this is the zone where the von Mises stress is the largest. The results are shown for the stress range 0–37 for comparison purposes.

We observe from Fig. 37 that new nodes are placed, at each iteration, in two areas of the domain. The first area is the fillet, where the stress concentration is the highest. The second area is the center of the top surface. It corresponds to the area where the displacement is the greatest. The refinement of the domain is relatively symmetric.

We compare in Fig. 39 the results obtained from the refined point cloud to reference finite element solutions obtained using

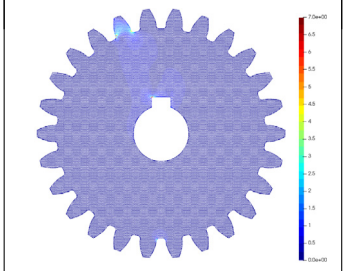
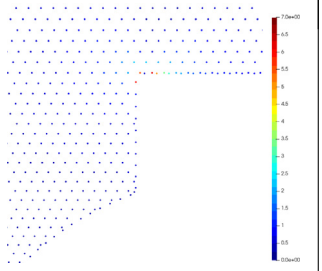
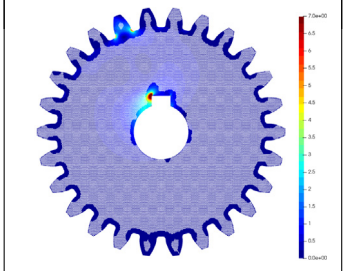
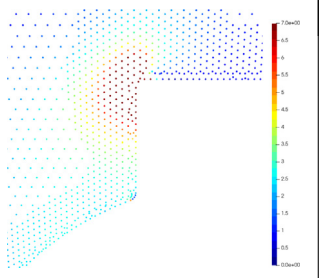
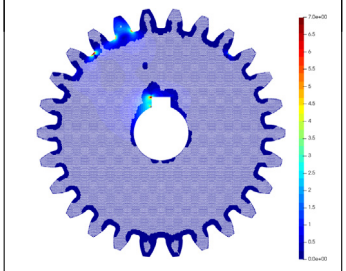
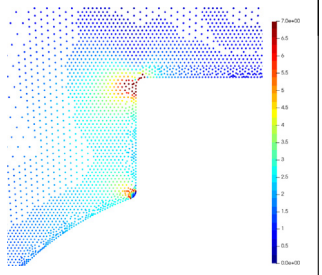
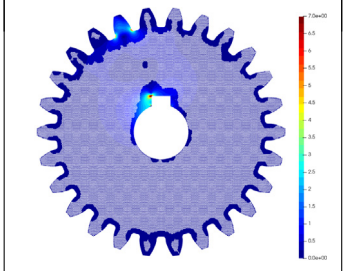
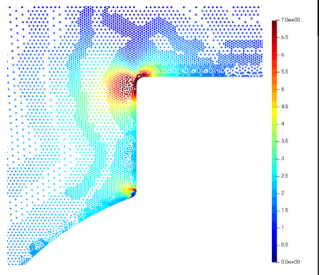
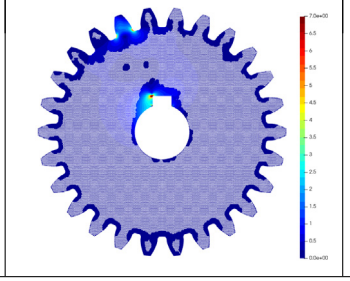
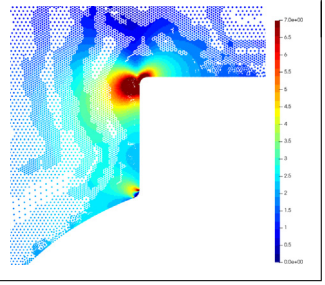
	Overall view	Groove detail view
Initial discretization 46,868 nodes		
Refinement iteration #1 59,544 nodes		
Refinement iteration #2 78,004 nodes		
Refinement iteration #3 104,527 nodes		
Refinement iteration #4 143,763 nodes		

Fig. 33. Gear coupled to a shaft — Evolution of the discretization of the domain and of the solution in terms of von Mises stress through four iterations of adaptive refinement. The results are shown for the stress range 0–7 for comparison purposes.

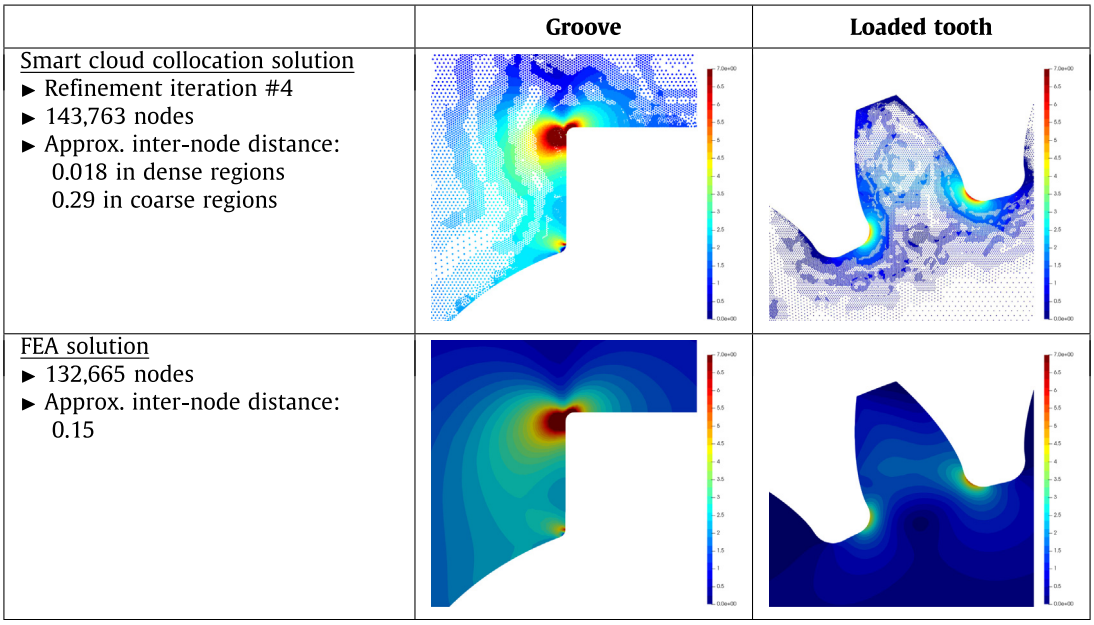


Fig. 34. Comparison of the results, in terms of von Mises stress, based on the smart cloud adaptive collocation scheme after four adaptive refinement iterations to results from the reference finite element solution. The comparison focuses on the groove and on the loaded tooth which are the areas where the stress concentration is the most important.

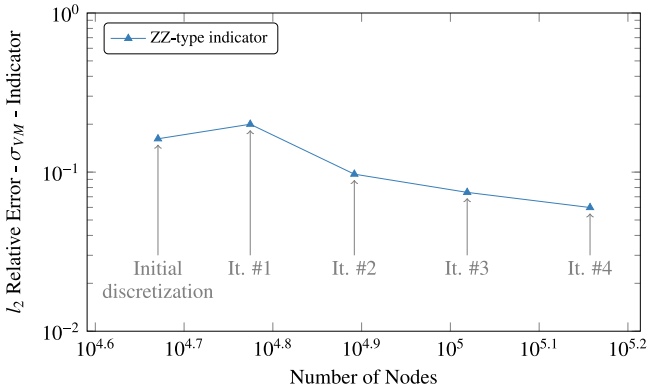


Fig. 35. Evolution of the error indicator for the four iterations of adaptive refinement for the problem of the gear coupled to a shaft.

code_aster [61]. We used a uniform finite element mesh composed of 166,1933 nodes and 900,039 linear tetrahedra for the purpose of this comparison. The discretization of the domain is uniform. The smart cloud collocation scheme leads to a larger stress concentration in the fillet than the finite element method solution obtained from the uniform mesh. The von Mises stress in the fillet is approximately 34–37 units for the smart cloud collocation solution and approximately 22–28 units for the finite element solution. The node density in the fillet is larger for the adapted smart cloud than for the finite element discretization, because of the two adaptive refinement iterations. The inter-node distance is approximately 0.0077 in the fillet for the smart cloud discretization after two refinement iterations. It is approximately 0.011 for the finite element discretization. We generate a second finite element mesh with a similar node density in the fillet region of the domain as the one of the smart cloud after two steps of adaptive refinement. The inter-node distance is approximately 0.007 units in the fillet region and 0.013 in the other regions of the domain. The results obtained from this "focused" mesh are very similar to the results obtained from the adapted smart cloud.

We present in Fig. 40 the computed error indicator for the initial discretization and the two discretization obtained after two refinement iterations. The error indicator is shown in terms of the l_2 relative error norm. We see that the value of the error does not decrease much between the results obtained from the initial discretization and from the discretization after two refinement steps. The error reduction is approximately 6%. A very slight increase of the error indicator is observed between the first and second refinement steps.

4. Conclusion

We presented in this article an integrated smart cloud collocation scheme to solve mechanics problems directly from CAD geometries. We used model adaptivity to speed up convergence of the results, while minimizing the input of the user.

The nodes of the smart cloud contain all the information required for optimum solution of the collocation problem (e.g., reference to the exact geometry, boundary conditions, connections to other boundary nodes). This information is key for adaptive refinement.

We used boundary-type elements to discretize the boundary of the domain. This approach is fast and robust since it relies on the discretization of the simple geometric features that compose the CAD domain. Boundary-type elements are also used to improve the accuracy of the solution using a generalization of the visibility criterion for point collocation methods. A larger number of interior nodes can be generated efficiently using regular node lattices such as triangular lattice for 2D domains or hexagonal close-packed lattice for 3D domains. The CAD file is used to select the nodes of the regular discretization which are in the domain.

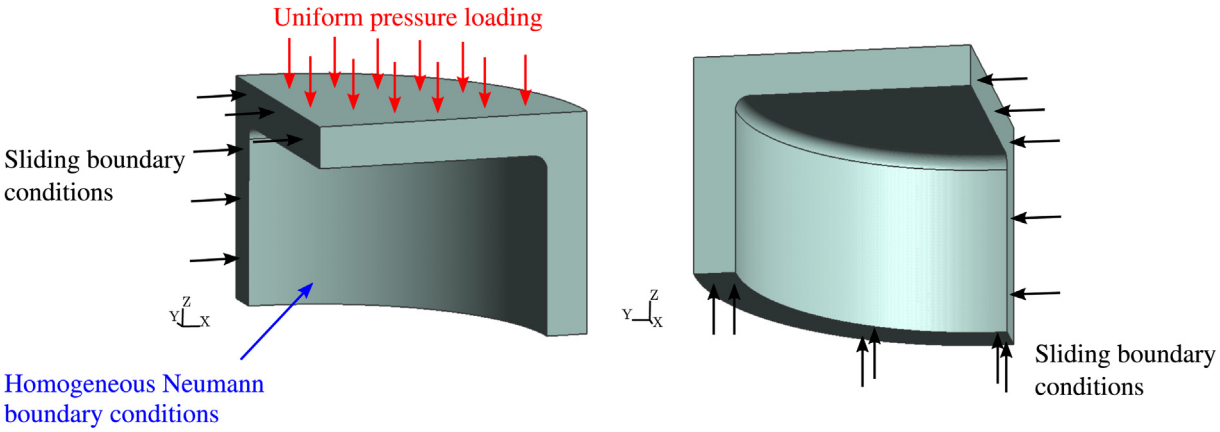


Fig. 36. Closed cylinder subject to pressure loading on the top. Uniform pressure loading is applied on the top surface. The displacement is limited in the directions normal to the surface on the symmetry planes XZ and YZ on the bottom surface. Stress-free boundary conditions are applied to the internal and external surface of the cylinder.

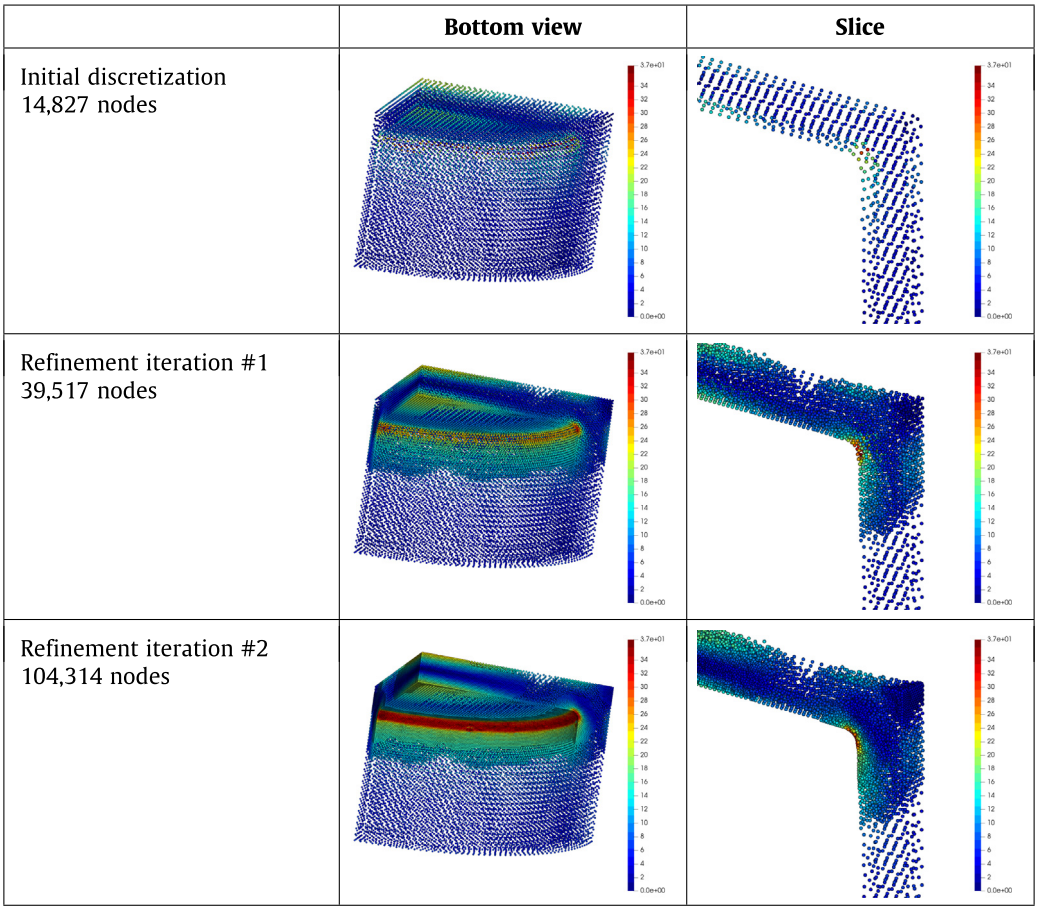


Fig. 37. Closed cylinder subject to pressure — Evolution of the discretization of the domain and of the solution in terms of von Mises stress through two iterations of adaptive refinement. The results are shown for the stress range 0–37 units for comparison purposes.

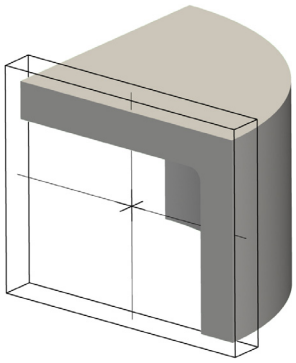


Fig. 38. Selection of a thick "slice" of the domain to allow a closer analysis of the discretization and of the results.

The proposed method only relies on the selection of a threshold parameter which is used to assess if interior nodes that are close to boundary nodes, are to be included in the collocation model or not. We showed that a threshold of 0.3 leads to good results.

Model adaptivity from CAD using the smart cloud collocation scheme is based on two key steps: error indication and cloud refinement. We presented two error indicators in this article: a ZZ-type error indicator and a residual-type error indicator. Their computation can be parallelized easily. Both allow a successful identification of the areas of the domain where the error is the greatest. We did not select the residual-type error indicator for our adaptive refinement method because of its complexity and computational cost. The adaptive refinement method presented in this article is based on the initial discretization of the CAD geometry. New nodes are added to the initial discretization to generate a refined discretization. Boundary elements are used to speed up the refinement process of the boundary. Once a new node is identified, it is projected onto the actual CAD geometry, thus ensuring that the point cloud discretization remains true to the original CAD geometry, and is not dependent on the element resolution. The required pieces of information about the CAD geometry and/or boundary conditions are added to the new nodes.

Two aspects of the proposed method should be further investigated: the quality of the error indicator and the convergence of the linear system for very large 3D problems. The error indicator considered here allows a proper identification of the refinement areas but cannot be considered as a good estimator. Even if an increase of the error indicator is a sign of an increase of the exact error, we do not consider it as a reliable estimator. The convergence of the linear systems obtained from adapted 3D clouds was difficult to attain. The condition of the matrices is an aspect that should be analyzed in greater detail to speed-up the convergence to the solution.

The use of CAD files at the heart of a collocation method is a great advantage to reduce the steps of the analysis to a minimum and ensure that the key features of the geometry are not lost during the refinement process, in the context of model adaptivity. These aspects are particularly true for domains with significant curvature, re-entrant corners or singularities.

To conclude, the work we have done to date on numerical methods, both within a strong-form framework [1,2,54,62] and within isogeometric boundary element approaches [19,21–24,63–66] makes us hopeful that, with some significant work from the community, those approaches may have a strong impact on the way we simulate and optimize engineering systems.

With the advent of imaging approaches, in particular LiDAR and photogrammetry, solving directly from point clouds and adapting them to the simulation that is required becomes an urgent matter. Imaging approaches have been thought out to improve and optimize visual rendering of different scenes. Significant work remains to be done to make those point clouds suitable for physics-based simulations. Based on our experience, we suggest the following research directions in this nascent field:

- [Goal-oriented defeaturing] Simplify smart point clouds through goal-oriented defeaturing such as done by Rahimi et al. [67] and references therein [68,69];
- [Goal-oriented error estimation and adaptivity] Minimize the error on a given quantity of engineering interest through goal-oriented error estimation for smart point clouds [70,71];
- [Smart clouds to CAD] Simplify smart point clouds into CAD primitives using classification machine learning approaches [72–76];
- [Properties identification from photogrammetry] Infer parameter values based on texture and color from photogrammetric images [77–79];
- [Solution process] Develop preconditioners and parallelization schemes for smart point clouds [80–82];
- [Integrated simulation pipeline] Develop open source pipelines based on libraries such as Open CASCADE Technology [35], VTK [83], CGAL [84], Medusa [60], Voro++ [85];
- [Multi-scale model reduction approaches] Defeature complex point clouds in order to accelerate simulations using multi-scale, domain decomposition and model order reduction approaches.

We will be pursuing the above directions, both in the context of computational engineering and computational archaeology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that support the findings of this study are openly available in https://gitlab.com/tjacquemin/smart-cloud.

Acknowledgments

Thibault Jacquemin and Stéphane P.A. Bordas would like to acknowledge funding from the Luxembourg National Research Fund (INTER/FWO/15/10318764), and from the European Union's Horizon 2020 research and innovation program for the DRIVEN project (grant agreement No 811099). Pratik Suchde would like to acknowledge support from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie Actions grant agreement No. 892761 "SURFING".

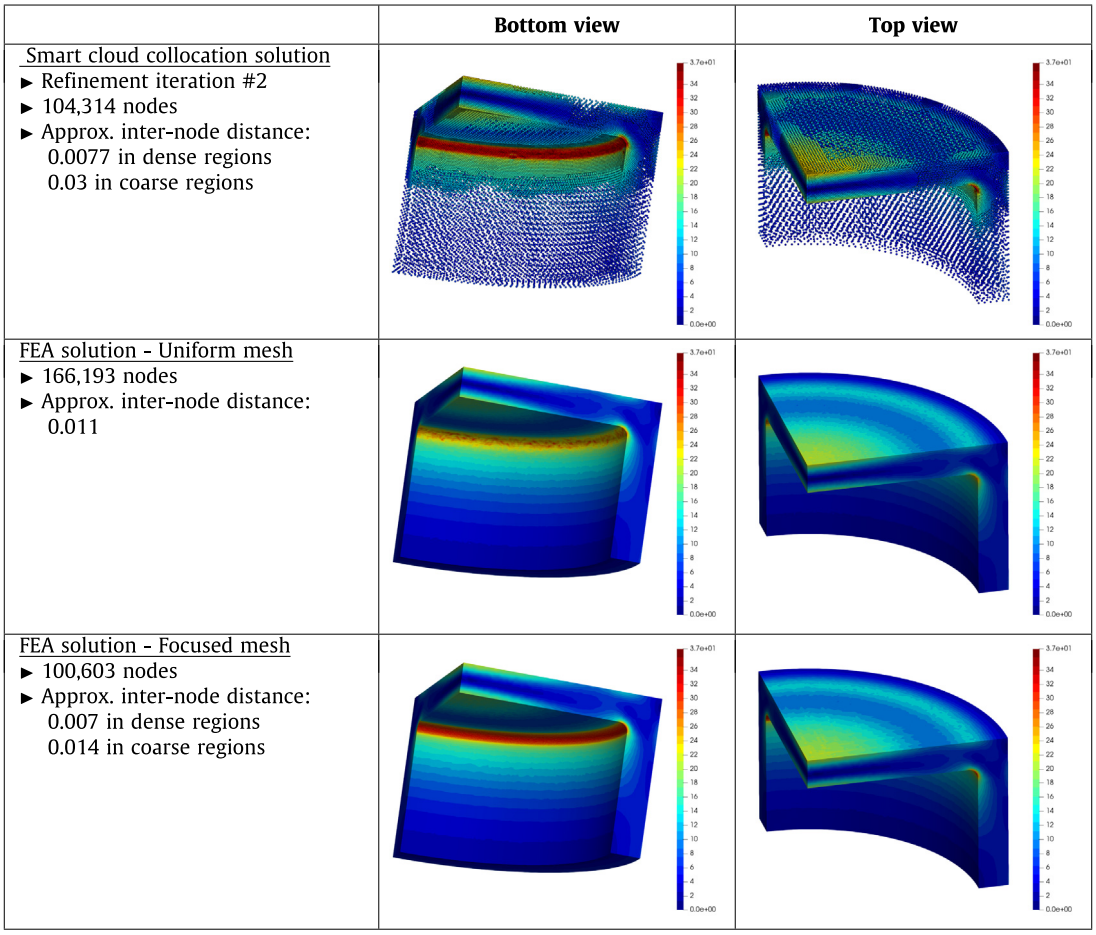


Fig. 39. Comparison, in terms of von Mises stress, of results obtained with the smart cloud method after two adaptive refinement iterations to results from reference finite element solutions.

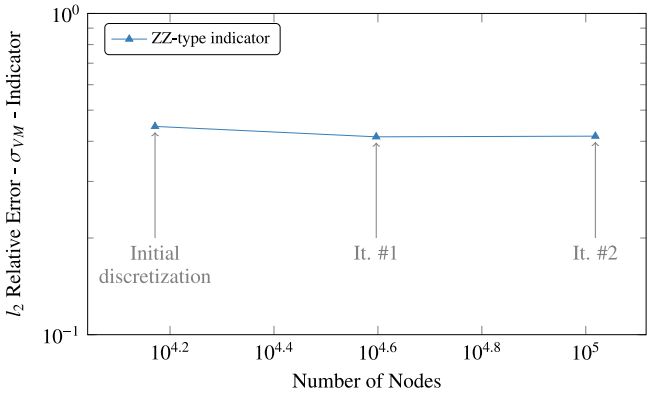


Fig. 40. Evolution of the error indicator for the two iterations of adaptive refinement for the problem of the closed cylinder subject to pressure.

References

[1] Jacquemin T, Tomar S, Agathos K, Mohseni-Mofidi S, Bordas S. Taylor-series expansion based numerical methods: A primer, performance benchmarking and new approaches for problems with non-smooth solutions. Arch Comput Methods Eng 2019. http://dx.doi.org/10.1007/s11831-019-09357-5.

[2] Jacquemin T, Bordas SPA. A unified algorithm for the selection of collocation stencils for convex, concave and singular problems. Internat J Numer Methods Engrg 2021. http://dx.doi.org/10.1002/nme.6703.

[3] Runge C. Z Math U Phys 1908;50:255.

[4] Jensen P. Finite difference techniques for variable grids. Comput Struct 1972;2(1–2):17–29. http://dx.doi.org/10.1016/0045-7949(72)90020-x.

[5] Liszka T, Orkisz J. The finite difference method at arbitrary irregular grids and its application in applied mechanics. Comput Struct 1980;11(1–2):83–95. http://dx.doi.org/10.1016/0045-7949(80)90149-2.

[6] Orkisz J. Finite difference method (Part III). In: Handbook of computational solid mechanics. Springer-Verlag; 1998, p. 335–432.

[7] Benito J, Ureña F, Gavete L. Influence of several factors in the generalized finite difference method. Appl Math Model 2001;25(12):1039–53. http://dx.doi.org/10.1016/s0307-904x(01)00029-4.

[8] Milewski S. Meshless finite difference method with higher order approximation—applications in mechanics. Arch Comput Methods Eng 2012;19(1):1–49. http://dx.doi.org/10.1007/s11831-012-9068-y.

[9] Shepard D. A two-dimensional interpolation function for irregularly-spaced data. In: Proceedings of the 1968 23rd ACM national conference. ACM Press; 1968, p. 517–524. http://dx.doi.org/10.1145/800186.810616.

[10] Lancaster P, Salkauskas K. Surfaces generated by moving least squares methods. Math Comp 1981;37(155):141. http://dx.doi.org/10.1090/s0025-5718-1981-0616367-1.

[11] Oñate E, Idelsohn S, Zienkiewick O, Taylor R. A finite point method in computational mechanics. Application to convective transport and fluid flow. Internat J Numer Methods Engrg 1996;39(22):3839–66. http://dx.doi.org/10.1002/(sici)1097-0207(19961130)39:22<3839::aid-nme27>3.0.co;2-r.

[12] Kansa E. Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics—i surface approximations and partial derivative estimates. Comput Math Appl 1990;19(8–9):127–45. http://dx.doi.org/10.1016/0898-1221(90)90270-t.

[13] Kansa E. Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics—II solutions to parabolic, hyperbolic and elliptic partial differential equations. Comput Math Appl 1990;19(8–9):147–61. http://dx.doi.org/10.1016/0898-1221(90)90271-k.

[14] Driscoll T, Fornberg B. Interpolation in the limit of increasingly flat radial basis functions. Comput Math Appl 2002;43(3–5):413–22. http://dx.doi.org/10.1016/s0898-1221(01)00295-4.

[15] Davydov O, Oanh DT. On the optimal shape parameter for Gaussian radial basis function finite difference approximation of the Poisson equation. Comput Math Appl 2011;62(5):2143–61. http://dx.doi.org/10.1016/j.camwa.2011.06.037.
[16] Hughes T, Cottrell J, Bazilevs Y. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Comput Methods Appl Mech Engrg 2005;194(39–41):4135–95. http://dx.doi.org/10.1016/j.cma.2004.10.008.
[17] Politis C, Ginnis AI, Kaklis PD, Belibassakis K, Feurer C. An isogeometric BEM for exterior potential-flow problems in the plane. In: 2009 SIAM/ACM joint conference on geometric and physical modeling on - SPM '09. ACM Press; 2009, http://dx.doi.org/10.1145/1629255.1629302.
[18] Belibassakis KA, Gerostathis TP, Kostas KV, Politis CG, Kaklis PD, Ginnis AI, et al. A BEM-isogeometric method with application to the wavemaking resistance problem of ships at constant speed. In: Volume 6: Ocean engineering. ASMEDC; 2011, http://dx.doi.org/10.1115/omae2011-49159.
[19] Simpson R, Bordas S, Trevelyan J, Rabczuk T. A two-dimensional isogeometric boundary element method for elastostatic analysis. Comput Methods Appl Mech Engrg 2012;209–212:87–100. http://dx.doi.org/10.1016/j.cma.2011.08.008.
[20] Ginnis AI, Feurer C, Belibassakis K, Kaklis PD, Kostas K, Gerostathis TP, et al. A CATIA ship-parametric model for isogeometric hull optimization with respect to wave resistance. 2015.
[21] Lian H, Simpson R, Bordas S. Stress analysis without meshing: Isogeometric boundary-element method. Proc Inst Civ Eng - Eng Comput Mech 2013;166(2):88–99. http://dx.doi.org/10.1680/eacm.11.00024.
[22] Simpson R, Scott M, Taus M, Thomas D, Lian H. Acoustic isogeometric boundary element analysis. Comput Methods Appl Mech Engrg 2014;269:265–90. http://dx.doi.org/10.1016/j.cma.2013.10.026, URL https://www.sciencedirect.com/science/article/pii/S0045782513002788.
[23] Peng X, Atroshchenko E, Kerfriden P, Bordas S. Isogeometric boundary element methods for three dimensional static fracture and fatigue crack growth. Comput Methods Appl Mech Engrg 2017;316:151–85. http://dx.doi.org/10.1016/j.cma.2016.05.038.
[24] Lian H, Kerfriden P, Bordas S. Shape optimization directly from CAD: An isogeometric boundary element approach using T-splines. Comput Methods Appl Mech Engrg 2017;317:1–41. http://dx.doi.org/10.1016/j.cma.2016.11.012.
[25] Auricchio F, da Veiga LB, Hughes T, Reali A, Sangalli G. Isogeometric collocation methods. Math Models Methods Appl Sci 2010;20(11):2075–107. http://dx.doi.org/10.1142/s0218202510004878.
[26] Benito JJ, Urena F, Gavete L, Alonso B. A posteriorierror estimator and indicator in generalized finite differences. Application to improve the approximated solution of elliptic PDEs. Int J Comput Math 2008;85(3–4):359–70. http://dx.doi.org/10.1080/00207160601167052.
[27] Gavete L, Gavete ML, Ureña F, Benito JJ. An approach to refinement of irregular clouds of points using generalized finite differences. Math Probl Eng 2015;2015:1–9. http://dx.doi.org/10.1155/2015/283757.
[28] Suchde P, Kuhnert J. A meshfree generalized finite difference method for surface PDEs. Comput Math Appl 2019;78(8):2789–805. http://dx.doi.org/10.1016/j.camwa.2019.04.030, URL http://www.sciencedirect.com/science/article/pii/S0898122119302469.
[29] Davydov O, Oanh DT. Adaptive meshless centres and RBF stencils for Poisson equation. J Comput Phys 2011;230(2):287–304. http://dx.doi.org/10.1016/j.jcp.2010.09.005.
[30] Oanh DT, Davydov O, Phu HX. Adaptive RBF-FD method for elliptic problems with point singularities in 2D. Appl Math Comput 2017;313:474–97. http://dx.doi.org/10.1016/j.amc.2017.06.006, URL https://www.sciencedirect.com/science/article/pii/S0096300317304186.
[31] Slak J. Adaptive RBF-FD method [Ph.D. thesis], Univerza v Ljubljani, Fakulteta za matematiko in fiziko; 2020.
[32] Belytschko T, Lu Y, Gu L. Element-free Galerkin methods. Internat J Numer Methods Engrg 1994;37(2):229–56. http://dx.doi.org/10.1002/nme.1620370205.
[33] Fedkiw RP, Aslam T, Merriman B, Osher S. A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method). J Comput Phys 1999;152(2):457–92. http://dx.doi.org/10.1006/jcph.1999.6236.
[34] Runnels B, Agrawal V, Zhang W, Almgren A. Massively parallel finite difference elasticity using block-structured adaptive mesh refinement with a geometric multigrid solver. J Comput Phys 2021;427:110065. http://dx.doi.org/10.1016/j.jcp.2020.110065.
[35] OpenCASCADE: Open CASCADE technology, 3D modeling & numerical simulation, https://dev.opencascade.org/.
[36] Geuzaine C, Remacle J-F. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. Internat J Numer Methods Engrg 2009;79(11):1309–31. http://dx.doi.org/10.1002/nme.2579.
[37] Löhner R, Oñate E. An advancing front point generation technique. Commun Numer Methods Eng 1998;14(12):1097–108. http://dx.doi.org/10.1002/(sici)1099-0887(199812)14:12<1097::aid-cnm183>3.0.co;2-7.
[38] Shimrat M. Algorithm 112: Position of point relative to polygon. Commun ACM 1962;5(8):434. http://dx.doi.org/10.1145/368637.368653.
[39] Chinn W, Steenrod N. First concepts of topology. The Mathematical Association of America; 1966, http://dx.doi.org/10.5948/upo9780883859339.
[40] O'Rourke J. Computational geometry in C second edition (Cambridge tracts in theoretical computer science (Paperback)). Cambridge University Press; 1998, URL https://www.xarg.org/ref/a/0521649765/.
[41] Möller T, Trumbore B. Fast, minimum storage ray-triangle intersection. J Graph Tools 1997;2(1):21–8. http://dx.doi.org/10.1080/10867651.1997.10487468.
[42] Alliez P, Tayeb S, Wormser C. 3D fast intersection and distance computation. In: CGAL user and reference manual. 5.0. CGAL Editorial Board; 2019, URL https://doc.cgal.org/5.0/Manual/packages.html.
[43] Gao X-L. A general solution of an infinite elastic plate with an elliptic hole under biaxial loading. Int J Press Vessels Pip 1996;67(1):95–104. http://dx.doi.org/10.1016/0308-0161(94)00173-1.
[44] Gould P. Introduction to linear elasticity. New York: Springer; 2013.
[45] Zienkiewicz OC, Zhu JZ. A simple error estimator and adaptive procedure for practical engineerng analysis. Internat J Numer Methods Engrg 1987;24(2):337–57. http://dx.doi.org/10.1002/nme.1620240206.
[46] Bordas S, Duflot M. Derivative recovery and a posteriori error estimate for extended finite elements. Comput Methods Appl Mech Engrg 2007;196(35):3381–99. http://dx.doi.org/10.1016/j.cma.2007.03.011, URL https://www.sciencedirect.com/science/article/pii/S0045782507001417.
[47] Duflot M, Bordas S. A posteriorierror estimation for extended finite elements by an extended global recovery. Internat J Numer Methods Engrg 2008;76(8):1123–38. http://dx.doi.org/10.1002/nme.2332.
[48] Ródenas JJ, Duflot M, Bordas S, Giner E, González-Estrada OA, Fuenmayor FJ. Comparison of recently developed recovery type discretization error estimators for the extended finite element method. In: 8th World congress on computational mechanics (WCCM8). 5th. European congress on computational methods in applied sciences and engineering. CINME; 2008.
[49] Driscoll T, Heryudono A. Adaptive residual subsampling methods for radial basis function interpolation and collocation problems. Comput Math Appl 2007;53:927–39.
[50] Oh J. Adaptive meshfree methods for partial differential equations [Ph.D. thesis], University of Southern Mississippi; 2018.
[51] Benito J, Ureña F, Gavete L, Alvarez R. An h-adaptive method in the generalized finite differences. Comput Methods Appl Mech Engrg 2003;192(5–6):735–59. http://dx.doi.org/10.1016/s0045-7825(02)00594-7.
[52] Slak J, Kosec G. Adaptive radial basis function–generated finite differences method for contact problems. Internat J Numer Methods Engrg 2019;119(7):661–86. http://dx.doi.org/10.1002/nme.6067.
[53] Suchde P, Kuhnert J. A fully Lagrangian meshfree framework for PDEs on evolving surfaces. J Comput Phys 2019;395:38–59. http://dx.doi.org/10.1016/j.jcp.2019.06.031, URL https://www.sciencedirect.com/science/article/pii/S0021999119304395.
[54] Suchde P, Kuhnert J, Schröder S, Klar A. A flux conserving meshfree method for conservation laws. Internat J Numer Methods Engrg 2017;112(3):238–56. http://dx.doi.org/10.1002/nme.5511.
[55] Liszka T, Duarte C, Tworzydlo W. Hp-meshless cloud method. Comput Methods Appl Mech Engrg 1996;139(1–4):263–88. http://dx.doi.org/10.1016/s0045-7825(96)01086-9.
[56] Duarte CA, Oden JT. H-p clouds—anh-p meshless method. Numer Methods Partial Differential Equations 1996;12(6):673–705. http://dx.doi.org/10.1002/(sici)1098-2426(199611)12:6<673::aid-num3>3.0.co;2-p.
[57] Jancic M, Slak J, Kosec G. P-refined RBF-FD solution of a Poisson problem. In: 2021 6th International conference on smart and sustainable technologies. 2021, p. 01–6.
[58] Dorfler W. A convergent adaptive algorithm for Poisson's equation. SIAM J Numer Anal 1996;33(3):1106–24, URL http://www.jstor.org/stable/2158497.
[59] Bulle R, Hale JS, Lozinski A, Bordas SPA, Chouly F. Hierarchical a posteriori error estimation of Bank-Weiser type in the FEniCS project. 2021, ArXiv, arXiv:2102.04360.
[60] Medusa: Coordinate Free Mehless Method implementation. Ghost nodes (theory) — Medusa: Coordinate free mehless method implementation. 2019.
[61] Electricité de France. Finite element *code_aster*, analysis of structures and thermomechanics for studies and research. Open source on https://www.code-aster.org.
[62] Suchde P, Kuhnert J, Tiwari S. On meshfree GFDM solvers for the incompressible Navier–Stokes equations. Comput & Fluids 2018;165:1–12. http://dx.doi.org/10.1016/j.compfluid.2018.01.008, URL http://www.sciencedirect.com/science/article/pii/S0045793018300070.
[63] Lian H, Kerfriden P, Bordas SPA. Implementation of regularized isogeometric boundary element methods for gradient-based shape optimization in two-dimensional linear elasticity. Internat J Numer Methods Engrg 2016;106(12):972–1017. http://dx.doi.org/10.1002/nme.5149.

[64] Chen L, Lian H, Liu Z, Chen H, Atroshchenko E, Bordas S. Structural shape optimization of three dimensional acoustic problems with isogeometric boundary element methods. Comput Methods Appl Mech Engrg 2019;355:926–51. http://dx.doi.org/10.1016/j.cma.2019.06.012.

[65] Chen L, Lu C, Lian H, Liu Z, Zhao W, Li S, et al. Acoustic topology optimization of sound absorbing materials directly from subdivision surfaces with isogeometric boundary element methods. Comput Methods Appl Mech Engrg 2020;362:112806. http://dx.doi.org/10.1016/j.cma.2019.112806, URL https://www.sciencedirect.com/science/article/pii/S004578251930698X.

[66] Chen L, Zhang Y, Lian H, Atroshchenko E, Ding C, Bordas S. Seamless integration of computer-aided geometric modeling and acoustic simulation: Isogeometric boundary element methods based on Catmull-Clark subdivision surfaces. Adv Eng Softw 2020;149:102879. http://dx.doi.org/10.1016/j.advengsoft.2020.102879, URL https://www.sciencedirect.com/science/article/pii/S0965997820305779.

[67] Rahimi N, Kerfriden P, Langbein FC, Martin RR. CAD model simplification error estimation for electrostatics problems. SIAM J Sci Comput 2018;40(1):B196–227. http://dx.doi.org/10.1137/16m1078641.

[68] Thakur A, Banerjee AG, Gupta SK. A survey of CAD model simplification techniques for physics-based simulation applications. Comput Aided Des 2009;41(2):65–80. http://dx.doi.org/10.1016/j.cad.2008.11.009.

[69] Danglade F, Pernot J-P, Véron P. On the use of machine learning to defeature CAD models for simulation. Comput-Aided Des Appl 2013;11(3):358–68. http://dx.doi.org/10.1080/16864360.2013.863510.

[70] Prudhomme S, Oden J. On goal-oriented error estimation for elliptic problems: Application to the control of pointwise errors. Comput Methods Appl Mech Engrg 1999;176(1):313–31. http://dx.doi.org/10.1016/S0045-7825(98)00343-0, URL https://www.sciencedirect.com/science/article/pii/S0045782598003430.

[71] Chamoin L, Legoll F. Goal-oriented error estimation and adaptivity in MsFEM computations. Comput Mech 2021;67(4):1201–28. http://dx.doi.org/10.1007/s00466-021-01990-x.

[72] Aghighi F, Ebadati O, Aghighi H. Classification of LiDAR points cloud using Markov random field and machine learning techniques. Iran J Remote Sens GIS 2018;9(2):41–60.

[73] Aghighi F, Aghighi H, Ebadati OM. Conditional random fields for airborne lidar point cloud classification in urban area. Eng J Geospat Inf Technol 2020;7(4):139–56.

[74] Li L, Sung M, Dubrovina A, Yi L, Guibas LJ. Supervised fitting of geometric primitives to 3D point clouds. In: 2019 IEEE/CVF conference on computer vision and pattern recognition. IEEE; 2019, http://dx.doi.org/10.1109/cvpr.2019.00276.

[75] Angles B. Geometric modeling with primitives [Ph.D. thesis], Université Paul Sabatier -, Toulouse III; 2019.

[76] Saporta T, Sharf A. Unsupervised recursive deep fitting of 3D primitives to points. Comput Graph 2021. http://dx.doi.org/10.1016/j.cag.2021.10.020, URL https://www.sciencedirect.com/science/article/pii/S0097849321002314.

[77] Zhan Q, Liang Y, Xiao Y. Color-based segmentation of point clouds. In: ISPRS Laser Scanning Workshop, vol. 38, 2009.

[78] Awrangjeb M, Zhang C, Fraser CS. Automatic extraction of building roofs using LIDAR data and multispectral imagery. ISPRS J Photogram Remote Sens 2013;83:1–18. http://dx.doi.org/10.1016/j.isprsjprs.2013.05.006.

[79] Alshawabkeh Y. Linear feature extraction from point cloud using color information. Heritage Sci 2020;8(1). http://dx.doi.org/10.1186/s40494-020-00371-6.

[80] Balay S, Abhyankar S, Adams M, Brown J, Brune P, Buschelman K, et al. PETSc users manual. Tech. rep. ANL-95/11 - Revision 3.9, Argonne National Laboratory; 2018.

[81] Falgout RD, Yang UM. Hypre: A library of high performance preconditioners. In: Sloot PMA, Hoekstra AG, Tan CJK, Dongarra JJ, editors. Computational science — ICCS 2002. Berlin, Heidelberg: Springer Berlin Heidelberg; 2002, p. 632–41.

[82] The Trilinos Project Team. The trilinos project website.

[83] Schroeder W. The visualization toolkit : An object-oriented approach to 3D graphics. Clifton Park, N.Y: Kitware; 2006.

[84] The CGAL Project. CGAL user and reference manual, 5.3.1. CGAL Editorial Board; 2021, URL https://doc.cgal.org/5.3.1/Manual/packages.html.

[85] Rycroft C. VORO++: A three-dimensional Voronoi cell library in C++. Chaos 2009;19(4):041111. http://dx.doi.org/10.1063/1.3215722.