UNIVERSITÉ DU
LUXEMBOURG

PhD-FSTM-2022-053

The Faculty of Science, Technology and Medicine

DISSERTATION

Defence held on 29 April 2022 in Luxembourg

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG

EN SCIENCES DE L'INGÉNIEUR

by

Thibault JACQUEMIN

Born on 02 October 1990 in Saint Germain en Laye, France

SMART CLOUD COLLOCATION: A UNIFIED WORKFLOW FROM CAD TO ENHANCED SOLUTIONS

Dissertation defence committee

Prof. Dr Stéphane P.A. Bordas, Dissertation Supervisor
Professor, Université du Luxembourg

Prof. Dr Andreas Zilian, Chairman
Professor, Université du Luxembourg

Dr Jack Hale
Research scientist, Université du Luxembourg

Prof. Dr Ivo Sbalzarini, Vice Chairman
Professor, Max Planck Institute of Molecular Cell Biology and Genetics

Dr Laurent Adam
R&D Director, Hexagon

Affidavit

I hereby confirm that the PhD thesis entitled "Smart cloud collocation: a unified workflow from CAD to enhanced solutions" has been written independently and without any other sources than cited.

Luxembourg, May 11th, 2022

Jacquemin

Thibault Jacquemin

Index

List of Figures

List of Tables

Summary

Computer Aided Design (CAD) software packages are used in the industry to design mechanical systems. Then, calculations are often performed using simulation software packages to improve the quality of the design. To speed up the development costs, companies and research centers have been trying to ease the integration of the computation phase in the design phase. The collocation methods have the potential of easing such integration thanks to their meshless nature. The geometry discretization step which is a key element of all computational method is simplified compared to mesh-based methods such as the finite element method.

We propose in this thesis a unified workflow that allows the solution of engineering problems defined by partial differential equations (PDEs) directly from input CAD files. The scheme is based on point collocation methods and proposed techniques to enhance the solution. We introduce the idea of "smart clouds". Smart clouds refer to point cloud discretizations that are aware of the exact CAD geometry, appropriate to solve a defined problem using a point collocation method and that contain information used to improve locally the solution.

We introduce a unified node selection algorithm based on a generalization of the visibility criterion. The proposed algorithm leads to a significant reduction of the error for concave problems and does not have any drawback for convex problems. The point collocation methods rely on many parameters. We select in this thesis parameters for the Generalized Finite Difference (GFD) method and the Discretization-Corrected Particle Strength Exchange (DC PSE) method that we deem appropriate for most problems from the field of linear elasticity. We also show that solution improvement techniques, based on the use of Voronoi diagrams or on a stabilization of the PDE, do not lead to a reduction of the error for all of the considered benchmark problems. These methods shall therefore be used with care. We propose two types of *a posteriori* error indicators that both succeed in identifying the areas of the domain where the error is the greatest: a ZZ-type and a residual-type error indicator. We couple these indicators to a h-adaptive refinement scheme and show that the approach is effective. Finally, we show the performance of Algebraic Multigrid (AMG) preconditions on the solution of linear systems compared to other preconditioning/solution methods. This family of preconditioners necessitates the selection of a large number of parameters. We assess the impact of some of them on the solution time for a 3D problem from the field of linear elasticity. Despite the performance of AMG preconditions, ILU preconditioners may be preferred thanks to their ease of usage and robustness to lead to a convergence of the solution.

Chapter 1

Introduction

1.1 Background and aim

In the industry, the design of mechanical systems very often begins by the creation of a Computer Aided Design (CAD) geometry. 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. Therefore, to minimize the development costs, both industry and academia have been trying to optimize this iterative process (design -> simulation -> design modification -> simulation . . .). In the field of solid mechanics, the finite element method is from far the most widely used simulation method. The meshless methods, and in particular the point collocation methods (also referred as collocation methods), have little commercial applications in this field.

The collocation methods have the advantage of being meshless. The meshing process, required by the finite element method for instance, may be time consuming and may requiring significant involvement of the user. The collocation methods are also flexible with regards to the node placement which is a great advantage for local refinement of the discretization.

We see a great potential of the collocation methods in easing the integration of the simulation phase in the design phase. However, this family of methods rely on many parameters which have a significant impact on the solution and should therefore be chosen carefully. In itself, this is a drawback compared to more parameter-robust methods, in particular the finite element method.

Based on these observations, we decided to study and propose in this thesis a unified workflow that allows future users of collocation methods to solve engineering problems from the field of linear elasticity with minimum effort. This unified workflow is based on a direct connection between the CAD geometry and the point collocation simulation. We focus primarily on two collocation methods: the Generalized Finite Difference (GFD) method and the Discretization-Corrected Particle Strength Exchange (DC PSE) method. We apply these methods to the field of linear elasticity. We start from CAD files, show how to convert them into a collocation model and study/propose methods to enhance the solutions obtained from the point collocation models. The workflow is based on the idea of "smart clouds". A "smart cloud" is a point cloud discretization of a domain where the points or nodes of the cloud contain specific information related to the base CAD geometry, to the boundary conditions and other information useful to improve locally the accuracy of the solution.

The collocation methods constitute a family of computational methods initiated by C. Runge in 1908 [1] with the finite difference method. These methods use nodes placed on the boundary of the domain and in the domain to solve a set of partial differential equations (PDEs). The field derivatives are approximated at each collocation center (or collocation node) as a function of the field values in the vicinity of the center.

Many collocation methods have been proposed. The most famous ones are the GFD method [2, 3, 4, 5, 6, 7], the Moving Least Squares (MLS) method [8, 9, 10], the Radial Basis Function Finite Difference (RBF-FD) method [11, 12, 13, 14, 15], the Smooth Particle Hydrodynamics (SPH) method [16] and the Reproduced Kernel Particle Method (RKPM) [17]. More recently, the DC PSE method was introduced in 2010 by Schrader et al. [18].

The idea of using the CAD geometry to solve problems defined by PDEs is at the heart of the Isogeometric Analysis (IGA) methods. These methods constitute a different class of numerical methods than collocation methods. They became rapidly popular after their first introduction in 2005 [19] and have been proven robust for problems of various nature. The IGA methods use the functions of the CAD geometry 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 introduced by Simpson et al. in 2012 [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, 22, 23, 24]. A collocation form of IGA has also been introduced [25]. We did not study IGA methods in this thesis. We focus on point collocation methods with the similar goal of having a tight connection between the CAD geometry and the simulation.

The solution of a problem defined by PDEs can be improved by the use of finer discretizations. However, such improvement is at the expense of the computational cost. Despite the constant progress of hardware, both in terms of memory and computation speed, the solution of complex engineering problems remains a bottleneck for companies and research groups. Therefore, the optimization of computational models and resources remains at the heart of simulation tasks. The aim of the proposed scheme is to minimize the size of the solved problems by improving as much as possible the quality of the solution. The selection of the nodes involved in the approximation of the field derivative impacts significantly the solution and the fill of the linear system matrix. They shall therefore be selected with care. Also, the parameters of the collocation methods play a key role in the quality of the approximation. These parameters need to be chosen to minimize the error. *A posteriori* error indication and adaptive refinement can be used to improve locally the solution where needed and is useful to minimize the number of degrees of freedom of a given problem. Finally, the solution of the linear system is the longest step of the simulation. Therefore, the solution scheme should be carefully selected depending on particular needs such as the robustness of the convergence to the solution, the memory usage of the solver or the available time to optimize the solver.

We published some of the content of this thesis in the articles "*Taylor-Series Expansion Based Numerical Methods: A Primer, Performance Benchmarking and New Approaches for Problems* *with Non-smooth Solutions*" [26] and "*A unified algorithm for the selection of collocation stencils for convex, concave, and singular problems*" [27]. We submitted a third article titled "*Smart cloud collocation: geometry-aware adaptivity directly from CAD* " for publication in a journal.

1.2 Thesis outline

We organized the chapters of this thesis with the aim of following the workflow of the solution of problems using point collocation methods. We present an overview of the content of each chapter in the following paragraphs.

Chapter 1: Introduction

The problems considered in this thesis belong to the field of linear elasticity. In the remainder of this first chapter, we present the governing equations of linear elasticity. We also present benchmark problems, for which analytical solutions are known. We used them in several chapters of the thesis to assess the impact of methods or parameters on the solution, based on exact error norms. We give the considered error norms in the last section of the chapter.

Chapter 2: Point collocation methods

We give in the second chapter a detailed description of two Taylor's series expansion based collocation methods: the GFD and the DC PSE methods. We take the specific case of a two-dimensional problem to facilitate the comprehension of the method. We also provide a side-by-side comparison of the methods in Appendix A. We compare the results obtained with the presented methods to results obtained with other collocation methods and to results obtained using the finite element method. Finally, we compare results obtained with the GFD method to results obtained with the finite element method for complex three-dimensional problems.

Chapter 3: CAD to smart cloud collocation

We introduce in the third chapter a method to generate efficiently smart clouds directly from CAD files. We show the impact of a parameter of the discretization method on the error and compare results obtained for problems discretized with the proposed method to results obtained for problems discretized using a Delaunay triangulation of the domain which is commonly used to generate point clouds.

Chapter 4: Stencil nodes selection: a unified algorithm

The selection of the collocation stencils is one of the key aspects of point collocation methods. We assess in the fourth chapter of the thesis how to best select the stencil size based on the position of collocation nodes in the domain. We show the impact of the visibility and diffraction criteria on the solution for a singular problem. We also introduce a unified node selection algorithm that can be applied to any convex, concave or singular problem, based on a generalization of the visibility criterion.

Chapter 5: Parameter selection and solution improvement methods

Point collocation methods rely on numerous parameters. We study in the fifth chapter the impact of some parameters of the GFD and DC PSE methods on the solution for a benchmark problem with a polynomial solution and another with a singular solution. We then show how Voronoi diagrams can be used as part of the GFD and DC PSE methods with the aim of improving the solution of problems solved with these methods. Finally, we assess the impact of a stabilization method on the solution for the considered benchmark problems.

Chapter 6: Smart cloud adaptivity

We focus on smart cloud adaptivity using *a posteriori* error estimation in the sixth chapter. We present first two error indicators that allow the identification of the areas of the domain where the error is the greatest. We then show how the smart cloud can be locally refined to converge efficiently to an accurate solution. Finally, we apply the smart cloud adaptive scheme to practical problems.

Chapter 7: Solution of collocation linear problems

The solution of the linear problem is often the longest step of the simulation for large problems. We present in the seventh chapter the bases of direct and iterative solvers and give the principle of the multigrid methods. Then, we show the impact of selected solvers on the solution time and on the number of iterations of the solver. We finally analyze in a sensitivity study the impact of parameters of a state of the art preconditioner applied to a problem solved using the GFD method.

Chapter 8: Conclusions and perspectives

We discuss the results that we obtained in this thesis in the eighth chapter, analyze them globally and give perspective for future research directions.

1.3 Governing equations

We focus in this thesis on problems from the field of linear elasticity. We present the partial differential equations associated with this class of problems in this section for the general case of 3D problems.

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:

$$\begin{aligned} &\boldsymbol{\nabla} \cdot \boldsymbol{\sigma} + \mathbf{b} = 0 \\ \text{or} \quad \forall i \in \{1,2,3\} \quad &\sigma_{ij,j} + b_i = 0. \end{aligned} \tag{1.1}$$

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 ϵ (kinematics):

$$\begin{aligned} &\epsilon = \frac{1}{2}\left(\boldsymbol{\nabla}\mathbf{u} + \boldsymbol{\nabla}\mathbf{u}^T\right) \\ \text{or} \quad \forall i \in \{1,2,3\} \quad &\epsilon_{ij} = \frac{1}{2}\left(u_{i,j} + u_{j,i}\right), \end{aligned} \tag{1.2}$$

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

$$\begin{bmatrix}\sigma_{11}\\\sigma_{22}\\\sigma_{33}\\\sigma_{23}\\\sigma_{13}\\\sigma_{12}\end{bmatrix}=\frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix}1-\nu&\nu&\nu&0&0&0\\\nu&1-\nu&\nu&0&0&0\\\nu&\nu&1-\nu&0&0&0\\0&0&0&1-2\nu&0&0\\0&0&0&0&1-2\nu&0\\0&0&0&0&0&1-2\nu\end{bmatrix}\begin{bmatrix}\epsilon_{11}\\\epsilon_{22}\\\epsilon_{33}\\\epsilon_{23}\\\epsilon_{13}\\\epsilon_{12}\end{bmatrix}. \quad (1.3)$$

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 $\epsilon_{23} = 0$).

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.

$$\begin{aligned}&\mathbf{u}=\mathbf{u^e} && \text{on} \quad \Gamma_D,\\&\boldsymbol{\sigma}\mathbf{n_N}=\mathbf{f^e} \quad \text{or} \quad \forall i\in\{1,2,3\} \quad \sigma_{ij}n_j=f_i^e && \text{on} \quad \Gamma_N.\end{aligned} \quad (1.4)$$

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). The von Mises criterion (noted σ_{vM}) is calculated using the equation:

$$\sigma_{\text{vM}}=\sqrt{\frac{(\sigma_{11}-\sigma_{22})^2+(\sigma_{22}-\sigma_{33})^2+(\sigma_{33}-\sigma_{11})^2+6\left(\sigma_{12}^2+\sigma_{23}^2+\sigma_{31}^2\right)}{2}}. \quad (1.5)$$

We use the individual stress components (e.g. σ_{11}, σ_{12}) and/or the von Mises stress to analyze the results obtained in this thesis.

1.4 Benchmark problems considered

We selected problems from the field of linear elasticity, for which analytical solutions are known, for benchmarking purposes. We considered four two-dimensional problems. Those are:

- a cylinder under internal pressure;
- an infinite body with a cylindrical hole under remote stress loading;
- an infinite plate with an elliptical hole under biaxial loading;
- a L-shape domain in mode I loading.

These problems have increasing levels of stress concentration. The models and boundary conditions are presented in Subsection 1.4.1. The exact solutions in terms of von Mises stress are presented in Subsection 1.4.2.

1.4.1 Models

2D cylinder

The model of the 2D cylinder under internal pressure is presented in Figure 1.1. Due to the symmetries of the problem in the Cartesian coordinate system, we modeled only a quarter of the cylinder. Symmetric boundary conditions are applied to the vertical edge on the left of the domain and to the horizontal edge at the bottom of the domain. Internal pressure loading is applied to the inner surface of the cylinder and stress-free surface boundary conditions to the outer surface. We present the well known solution of this problem in Figure 1.5 (a) for the von Mises stress σ_{VM}.

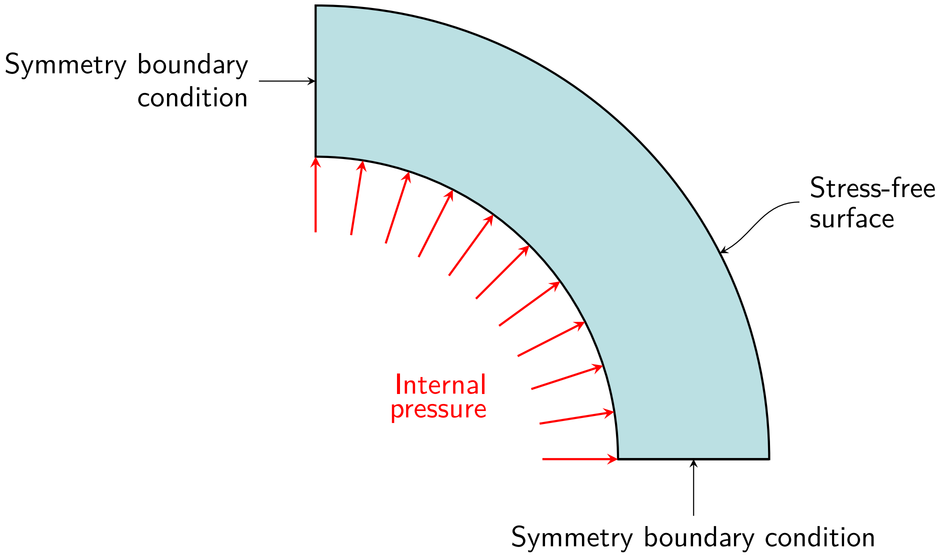


Figure 1.1: 2D model of cylinder under internal pressure. Symmetric boundary conditions are applied to the vertical and horizontal edges of the domain. A constant pressure loading is applied to the inner surface of the cylinder. A stress-free surface boundary condition is applied to the outer surface.

Infinite body with a cylindrical hole

The model of the infinite body with a cylindrical hole under remote stress loading is presented in Figure 1.2. Symmetry boundary conditions are applied to the vertical edge on the left and to the horizontal edge at the bottom. Stress-free surface boundary conditions are applied to the cylindrical hole. The displacement field of the exact solution is applied to the other boundaries of the domain. The exact solution to this problem is presented by Gould in Reference [28]. We show the solution for the von Mises stress σ_{VM} in Figure 1.5 (b).

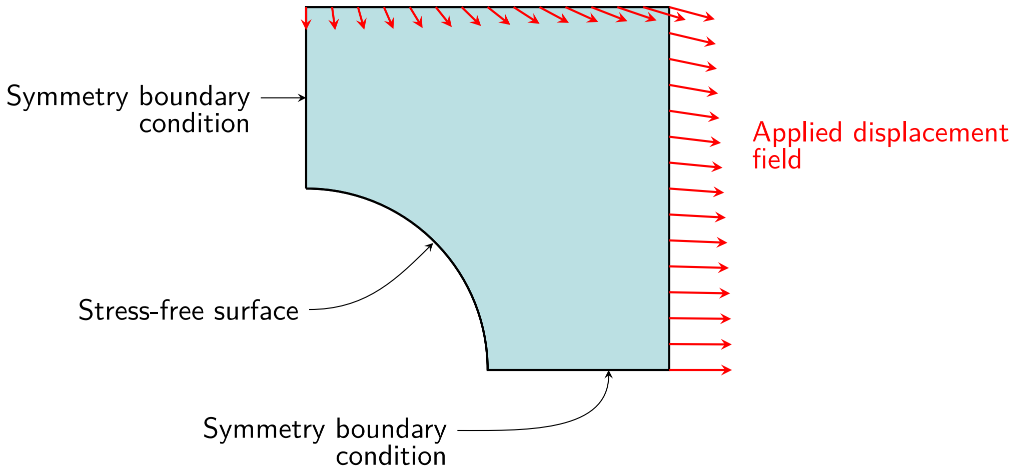

Symmetry boundary condition
Applied displacement field
Stress-free surface
Symmetry boundary condition

Figure 1.2: 2D model of an infinite body with a cylindrical 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. 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.

Infinite plate with an elliptical hole

We present the model of the infinite plate with an elliptical hole under biaxial loading in Figure 1.3. Symmetric boundary conditions are applied to the nodes on the vertical edge on the left of the model and stress-free surface boundary conditions to the elliptical hole. The displacement field of the exact solution is applied to the other boundaries of the domain. The exact solution to this problem is presented by X.L. Gao in Reference [29]. We show the solution for the von Mises stress σ_{VM} in Figure 1.5 (c).

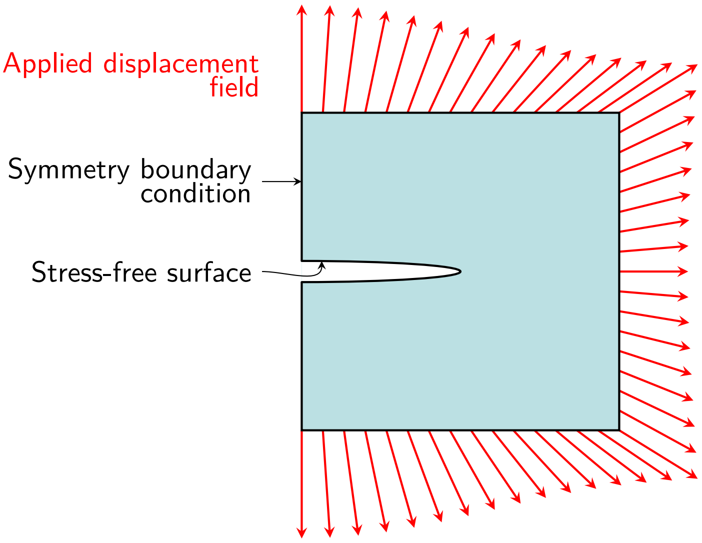


Figure 1.3: 2D model of an infinite plate with an elliptical hole under biaxial loading. Symmetric boundary conditions are applied to the vertical edge on the left. Stress-free surface boundary conditions are applied to the boundary of the elliptical hole. The displacement field of the exact solution is applied to the other boundaries of the domain.

L-shape

We present the model of a 2D L-shape domain in mode I loading in Figure 1.4. Stress-free boundary conditions are applied to the inner surface of the "L". The displacement field of the exact solution is applied to the other boundaries of the domain. The exact solution to this problem is presented by Babuška and Suri in Reference [30] and by Ainsworth and Senior in Reference [31]. We show the solution for the von Mises stress σ_{VM} in Figure 1.5 (d).

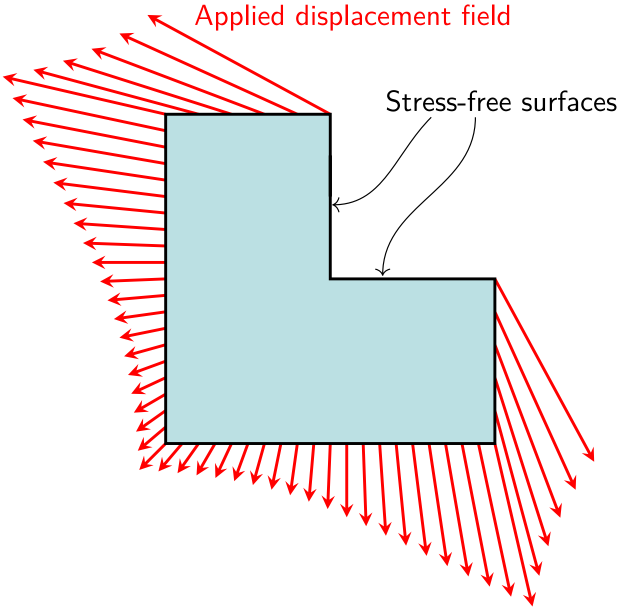
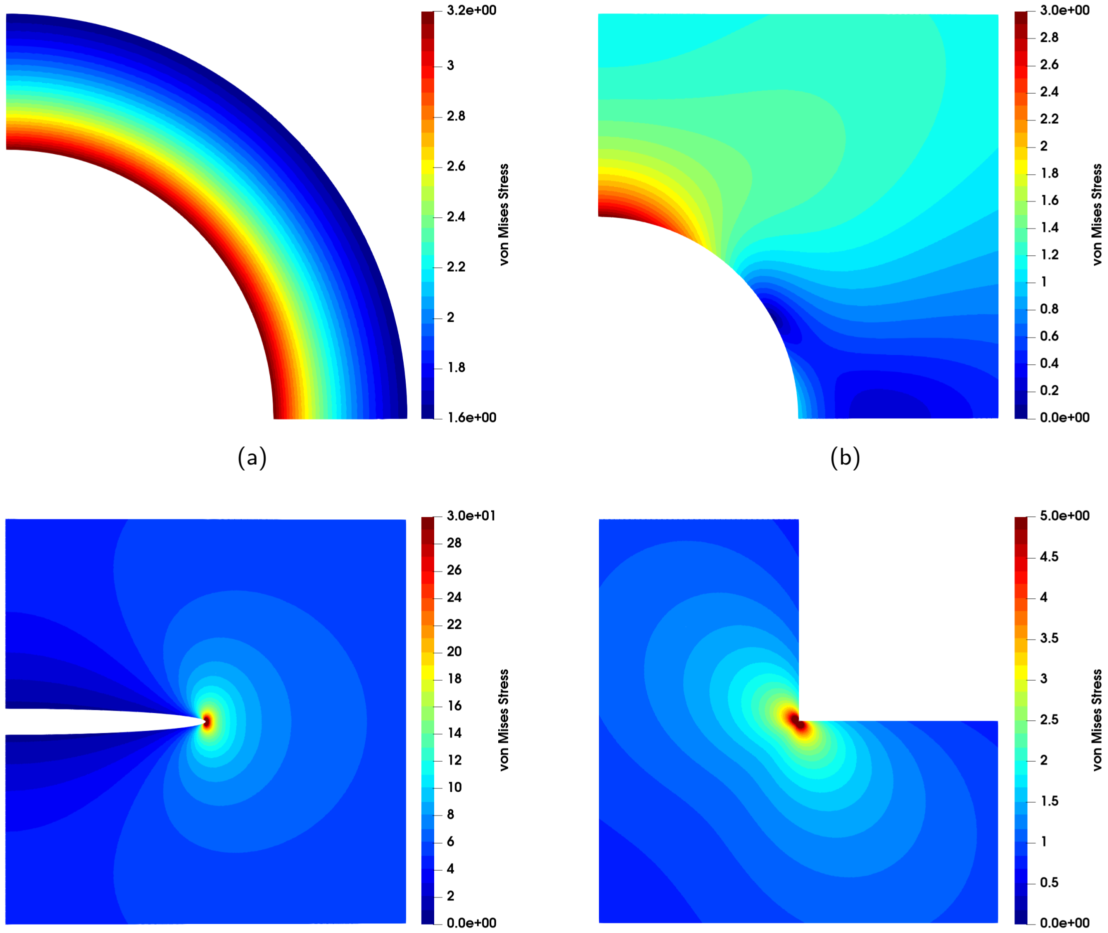


Figure 1.4: 2D model of L-shape domain in mode I loading. Stress-free surface boundary conditions are applied to the inner surface of the “L”. The displacement field of the exact solution is applied to the other boundaries of the domain.

1.4.2 Exact von Mises stress solutions

We present the exact von Mises stress solutions of the benchmark problems in Figure 1.5. We observe that the selected problems show different variations of the von Mises stress solution. The von Mises stress increases linearly from the outer surface to the inner surface of the cylinder. For the body with a cylindrical hole, the von Mises stress is maximum on the top surface of the hole. The von Mises stress increases sharply for the plate with an elliptical hole and for the L-shape domain close to the point of highest curvature and close to the singularity, respectively.



(a)

(b)

(c)

(d)

Figure 1.5: Exact von Mises stress solutions for the problems of the 2D cylinder (a), plate with a cylindrical hole (b), plate with an elliptical hole (c) and L-shape domain (d). To show the variation of the stress results over the domains, results are shown for the range [0-30] for the plate with an elliptical hole and [0-5] for the L-shape domain. 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 tends to infinity toward the re-entrant corner of the L-shape.

1.5 Exact error norms

We used error norms to assess the accuracy of results based on the exact solution, when available. For this purpose, we selected the L_2 weighted error norm (denoted L_2W), the L_2 relative error norm (denoted L_2R) and the L_∞ error norm. The L_2 weighted and relative error norms average out the error over all the collocation nodes. The L_∞ error norm corresponds to the maximum absolute error observed over the considered domain.

We considered errors on the stress field components and on the von Mises stress because these results are of greatest interest in the field of mechanical engineering.

At a collocation node $\mathbf{X_k}$, we denote the exact stress and approximated stress solutions $\sigma^e_{ij}(\mathbf{X_k})$ and $\sigma^h_{ij}(\mathbf{X_k})$, respectively. Considering a domain Ω discretized by n collocation nodes, the L_2 weighted error norm is calculated as per Equation (1.6). The L_2 relative error norm is calculated as per Equation (1.7) and the L_∞ error norm as per Equation (1.8).

$$L_2W(\sigma_{ij}) = \frac{\sqrt{\sum_{k=1}^{n}\left(\sigma^e_{ij}(\mathbf{X_k}) - \sigma^h_{ij}(\mathbf{X_k})\right)^2}}{n}. \tag{1.6}$$

$$L_2R(\sigma_{ij}) = \frac{\sqrt{\sum_{k=1}^{n}\left(\sigma^e_{ij}(\mathbf{X_k}) - \sigma^h_{ij}(\mathbf{X_k})\right)^2}}{\sqrt{\sum_{k=1}^{n}\sigma^e_{ij}(\mathbf{X_k})^2}}. \tag{1.7}$$

$$L_\infty(\sigma_{ij}) = \max_{k\in\Omega}\left(|\sigma^e_{ij}(\mathbf{X_k}) - \sigma^h_{ij}(\mathbf{X_k})|\right). \tag{1.8}$$

Chapter 2

Point collocation methods

2.1 General

Solving a problem using collocation methods consists in solving the set of partial differential equations (PDEs) only at collocation centers. Nodes spread over the domain are used to estimate the derivatives at the collocation centers. In most collocation methods, the equations are solved at the nodes. The strong form of the PDEs is considered.

The typical steps followed for solving a linear problem using a collocation method are as follows:

1. Selection of the collocation stencils;
2. Approximation the field derivatives based on the stencils;
3. Filling of the stiffness matrix and enforcement of the boundary conditions;
4. Solution of the linear problem;
5. Postprocessing of the results.

We describe in this chapter the theory of two Taylor's series expansion based collocation methods: the Generalized Finite Difference (GFD) and the Discretization-Corrected Particle Strength Exchange (DC PSE) method. We present the GFD method in Section 2.2 and the DC PSE method in Section 2.3. We compare both methods, for the case of a 1D problem, in Appendix A. We analyze the results obtained from three variations of the DC PSE method in Section 2.4. Finally, we compare the GFD and the DC PSE methods to two other popular collocation methods: the Moving Least Squares (MLS) method [8, 9, 32, 10] and to the Radial Basis function Finite Difference (RBF-FD) method [11, 12, 13, 14, 15]. We did not present the principles of these two other methods. These methods are presented in the aforementioned references and in Reference [26].

To facilitate the comprehension of the methods, we selected the case of a two dimensional problem in a Cartesian coordinate system. The spatial coordinates are denoted by x and y. The coordinates of a node $\mathbf{X}$ are then $\mathbf{X} = [x, y]^T$. The subscripts c and p are used to identify, respectively, the collocation node and a particle "p".

The first and second derivatives in the two spatial directions are denoted by: $\frac{\partial}{\partial x}$, $\frac{\partial}{\partial y}$, $\frac{\partial^2}{\partial x^2}$, $\frac{\partial^2}{\partial x \partial y}$, $\frac{\partial^2}{\partial y^2}$. In the general case, these derivatives are written as $D^{n_x,n_y} f(\mathbf{X_c})$, where n_x and n_y are, respectively, the derivation orders in the directions x and y.

The derivatives at a collocation center are typically approximated based on a defined support or stencil. The support is a set of nodes located in the vicinity of the collocation node. Figure 2.1 below shows the nodes of the domain Ω included in the support Ω_c of a collocation node $\mathbf{X_c}$. In 2D, the support is limited by a circle of radius R_c. All the nodes or some of the nodes within the disc of radius R_c are included in the support of $\mathbf{X_c}$ depending on the stencil selection technique.

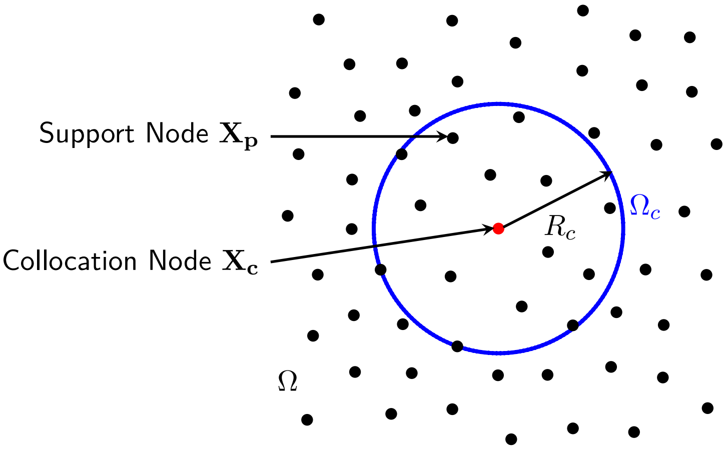


Figure 2.1: Support of a collocation node $\mathbf{X_c}$. Here, the support Ω_c is a disc of radius R_c.

2.2 Generalized Finite Difference method

2.2.1 Principle

The Finite Difference Method (FDM) is the most simple and one of the oldest methods for derivative approximation. It was introduced by C. Runge in 1908 [1]. 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 [6], Orkisz [7], Benito [33], Milewski [34] and many other contributors.

For the GFD method, the derivatives are calculated at collocation nodes $\mathbf{X_c} = [x_c, y_c]^T$ using a Taylor's series expansion of the unknown field. The field derivatives at $\mathbf{X_c}$ are computed in order to reproduce the known field values $f(\mathbf{X_{pi}})$ for a number of stencil points $\mathbf{X_{pi}} = [x_{pi}, y_{pi}]^T$. The number of selected points depends on the approximated derivative order.

2.2.2 Differential operator approximation

Considering a function $f : \mathrm{I\!R}^2 \to \mathrm{I\!R}$, the Taylor's series expansion of this function at $\mathbf{X_{pi}}$ in the vicinity of a collocation node $\mathbf{X_c}$ is written:

$$f(\mathbf{X_{pi}}) = \sum_{j=0}^{+\infty} \sum_{k=0}^{+\infty} \frac{\partial^{j+k} f(\mathbf{X_c})}{\partial x^j \partial y^k} \frac{(x_{pi} - x_c)^j}{j!} \frac{(y_{pi} - y_c)^k}{k!}. \tag{2.1}$$

For ease of notations, we write the second order approximation of the function f at the point $\mathbf{X_{pi}}$ near $\mathbf{X_c}$ as $f_h(\mathbf{X_{pi}})$. For $f_h(\mathbf{X_{pi}})$, Equation (2.1) becomes:

$$\begin{aligned} f_h(\mathbf{X_{pi}}) =& f(\mathbf{X_c}) + (x_{pi} - x_c)\frac{\partial f(\mathbf{X_c})}{\partial x} + (y_{pi} - y_c)\frac{\partial f(\mathbf{X_c})}{\partial y} \\ &+ \frac{(x_{pi} - x_c)^2}{2!}\frac{\partial^2 f(\mathbf{X_c})}{\partial x^2} + (x_{pi} - x_c)(y_{pi} - y_c)\frac{\partial^2 f(\mathbf{X_c})}{\partial x \partial y} \\ &+ \frac{(y_{pi} - y_c)^2}{2!}\frac{\partial^2 f(\mathbf{X_c})}{\partial y^2}. \end{aligned} \tag{2.2}$$

Equation (2.2) can be cast in a matrix form:

$$\begin{bmatrix} x_{pi}-x_c \\ y_{pi}-y_c \\ \frac{(x_{pi}-x_c)^2}{2!} \\ (x_{pi}-x_c)(y_{pi}-y_c) \\ \frac{(y_{pi}-y_c)^2}{2!} \end{bmatrix}^T \begin{bmatrix} \frac{\partial f(\mathbf{X_c})}{\partial x} \\ \frac{\partial f(\mathbf{X_c})}{\partial y} \\ \frac{\partial^2 f(\mathbf{X_c})}{\partial x^2} \\ \frac{\partial^2 f(\mathbf{X_c})}{\partial x \partial y} \\ \frac{\partial^2 f(\mathbf{X_c})}{\partial y^2} \end{bmatrix} = f_h(\mathbf{X_{pi}}) - f(\mathbf{X_c}). \tag{2.3}$$

In order to determine an approximation of the field derivative vector $\mathbf{Df}(\mathbf{X_c})$, shown in Equation (2.4),

$$\mathbf{Df}(\mathbf{X_c}) = \left[\frac{\partial f(\mathbf{X_c})}{\partial x}, \frac{\partial f(\mathbf{X_c})}{\partial y}, \frac{\partial^2 f(\mathbf{X_c})}{\partial x^2}, \frac{\partial^2 f(\mathbf{X_c})}{\partial x \partial y}, \frac{\partial^2 f(\mathbf{X_c})}{\partial y^2}\right]^T, \tag{2.4}$$

Equation (2.2) will be written for five nodes $\mathbf{X_{pi}}$ in the vicinity of $\mathbf{X_c}$, shown in Figure 2.2 for instance.

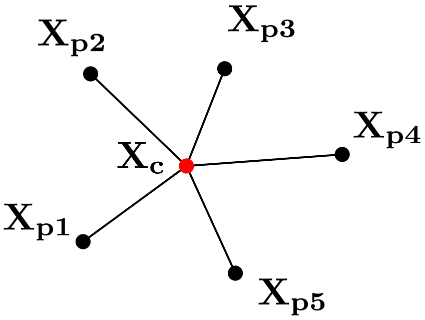


Figure 2.2: Five-node support of a collocation node $\mathbf{X_c}$.

The following linear system is obtained:

$$\begin{bmatrix} x_{p1}-x_c & y_{p1}-y_c & \frac{(x_{p1}-x_c)^2}{2!} & (x_{p1}-x_c)(y_{p1}-y_c) & \frac{(y_{p1}-y_c)^2}{2!} \\ x_{p2}-x_c & y_{p2}-y_c & \frac{(x_{p2}-x_c)^2}{2!} & (x_{p2}-x_c)(y_{p2}-y_c) & \frac{(y_{p2}-y_c)^2}{2!} \\ x_{p3}-x_c & y_{p3}-y_c & \frac{(x_{p3}-x_c)^2}{2!} & (x_{p3}-x_c)(y_{p3}-y_c) & \frac{(y_{p3}-y_c)^2}{2!} \\ x_{p4}-x_c & y_{p4}-y_c & \frac{(x_{p4}-x_c)^2}{2!} & (x_{p4}-x_c)(y_{p4}-y_c) & \frac{(y_{p4}-y_c)^2}{2!} \\ x_{p5}-x_c & y_{p5}-y_c & \frac{(x_{p5}-x_c)^2}{2!} & (x_{p5}-x_c)(y_{p5}-y_c) & \frac{(y_{p5}-y_c)^2}{2!} \end{bmatrix} \mathbf{Df}(\mathbf{X_c}) = \mathbf{F}(\mathbf{X_c}), \tag{2.5}$$

where:

$$\mathbf{F}(\mathbf{X_c}) = \begin{bmatrix} f_h(\mathbf{X_{p1}}) - f(\mathbf{X_c}) \\ f_h(\mathbf{X_{p2}}) - f(\mathbf{X_c}) \\ f_h(\mathbf{X_{p3}}) - f(\mathbf{X_c}) \\ f_h(\mathbf{X_{p4}}) - f(\mathbf{X_c}) \\ f_h(\mathbf{X_{p5}}) - f(\mathbf{X_c}) \end{bmatrix}. \tag{2.6}$$

Assuming that f_h is close to f in the vicinity of $\mathbf{X_c}$, the derivatives at the collocation node $\mathbf{X_c}$ can be approximated as a function of $f_h(\mathbf{X_c})$ and $f_h(\mathbf{X_{pi}})$ by solving the above system. If Equation (2.3) is written for more than five points $\mathbf{X_{pi}}$, the system is overdetermined. In that case, the derivatives at $\mathbf{X_c}$ leading to the minimum error can be determined using the moving least square method.

2.2.3 Overdetermined approximation

If Equation (2.3) is written at an arbitrary number of nodes m, the derivatives are determined using the moving least square method. A moving least square functional B is presented below for the two dimensional case for the general form in Equation (2.7) and the second order approximation in Equation (2.8). A weight function w is typically used to balance the contribution of each node in the approximation. While a wide range of functions can be used as weights, 3^{rd} and 4^{th} order splines are often preferred.

$$B(\mathbf{X_c}) = \sum_{i=1}^{m} w(\mathbf{X_{pi}} - \mathbf{X_c}) \Bigg[\sum_{j=0}^{+\infty} \sum_{k=0}^{+\infty} \frac{\partial^{j+k} f(\mathbf{X_c})}{\partial x^j \partial y^k} \frac{(x_{pi} - x_c)^j}{j!} \frac{(y_{pi} - y_c)^k}{k!} - f(\mathbf{X_{pi}}) \Bigg]^2. \tag{2.7}$$

$$\begin{aligned} B_h(\mathbf{X_c}) = \sum_{i=1}^{m} w(\mathbf{X_{pi}} - \mathbf{X_c}) \Big[& f(\mathbf{X_c}) - f(\mathbf{X_{pi}}) + (x_{pi} - x_c) \frac{\partial f(\mathbf{X_c})}{\partial x} + (y_{pi} - y_c) \frac{\partial f(\mathbf{X_c})}{\partial y} \\ & + \frac{(x_{pi} - x_c)^2}{2!} \frac{\partial^2 f(\mathbf{X_c})}{\partial x^2} + (x_{pi} - x_c)(y_{pi} - y_c) \frac{\partial^2 f(\mathbf{X_c})}{\partial x \partial y} \\ & + \frac{(y_{pi} - y_c)^2}{2!} \frac{\partial^2 f(\mathbf{X_c})}{\partial y^2} \Big]^2. \end{aligned} \tag{2.8}$$

The derivatives $\mathbf{Df}(\mathbf{X_c})$, that best approximate the known field values using the Taylor's series expansion, minimize $B_h(\mathbf{X})$ when:

$$\left.\frac{\partial B_h(\mathbf{X})}{\partial \mathbf{Df}(\mathbf{X})}\right|_{\mathbf{X}=\mathbf{X_c}} = 0. \tag{2.9}$$

Equation (2.9) can be written as a linear system of the form:

$$\mathbf{A}(\mathbf{X_c})\mathbf{Df}(\mathbf{X_c}) = \mathbf{E}(\mathbf{X_c})\mathbf{F}(\mathbf{X_c}). \tag{2.10}$$

For the two dimensional second order case, the matrices $\mathbf{A}(\mathbf{X_c})$, $\mathbf{E}(\mathbf{X_c})$ and $\mathbf{F}(\mathbf{X_c})$ are:

$$\mathbf{A}(\mathbf{X_c}) = \begin{bmatrix} m_{11} & m_{12} & \dots & m_{15} \\ m_{21} & m_{22} & \dots & m_{25} \\ \vdots & & & \vdots \\ m_{51} & m_{52} & \dots & m_{55} \end{bmatrix} \in \mathbb{R}^{5\times 5}, \tag{2.11}$$

$$\mathbf{E}(\mathbf{X_c}) = \begin{bmatrix} -m_{01} & m_{01,1} & \dots & m_{01,m} \\ -m_{02} & m_{02,1} & \dots & m_{02,m} \\ \vdots & & & \vdots \\ -m_{05} & m_{05,1} & \dots & m_{05,m} \end{bmatrix} \in \mathbb{R}^{5\times(\mathrm{m}+1)}, \tag{2.12}$$

$$\mathbf{F}(\mathbf{x_c}) = \begin{bmatrix} f(\mathbf{X_c}) & f(\mathbf{X_{p1}}) & f(\mathbf{X_{p2}}) & \dots & f(\mathbf{X_{pm}}) \end{bmatrix}^T, \tag{2.13}$$

where the moments $m_{jk,i}$ and m_{jk} correspond to:

$$\begin{aligned} m_{jk,i} &= w(\mathbf{X_{pi}} - \mathbf{X_c})P_{(j+1),i}(\mathbf{X_c})P_{(k+1),i}(\mathbf{X_c}), \\ m_{jk} &= \sum_{i=1}^{m} m_{jk,i}. \end{aligned} \tag{2.14}$$

The matrix $\mathbf{P}(\mathbf{X_c}) \in \mathrm{I\!R}^{6\times \mathrm{m}}$ is written as follows:

$$\mathbf{P}(\mathbf{X_c}) = \begin{bmatrix} 1 & 1 & \dots & 1 \\ (x_{p1} - x_c) & (x_{p2} - x_c) & \dots & (x_{pm} - x_c) \\ (y_{p1} - y_c) & (y_{p2} - y_c) & \dots & (y_{pm} - y_c) \\ \frac{(x_{p1}-x_c)^2}{2!} & \frac{(x_{p2}-x_c)^2}{2!} & \dots & \frac{(x_{pm}-x_c)^2}{2!} \\ (x_{p1} - x_c)(y_{p1} - y_c) & (x_{p2} - x)(y_{p2} - y_c) & \dots & (x_{pm} - x_c)(y_{pm} - y_c) \\ \frac{(y_{p1}-y_c)^2}{2!} & \frac{(y_{p2}-y_c)^2}{2!} & \dots & \frac{(y_{pm}-y_c)^2}{2!} \end{bmatrix}. \qquad (2.15)$$

The derivative vector $\mathbf{Df}(\mathbf{X_c})$ can then be determined as a function of $\mathbf{F}(\mathbf{X_c})$:

$$\mathbf{Df}(\mathbf{X_c}) = \mathbf{A}(\mathbf{X_c})^{\mathbf{-1}}\mathbf{E}(\mathbf{X_c})\mathbf{F}(\mathbf{X_c}). \qquad (2.16)$$

The approximated derivatives are determined by solving the linear system in Equation (2.16). These derivatives are, by definition, consistent with each other as they all participate in reproducing the unknown field values based on a Taylor's series expansion.

2.3 Discretization-Corrected Particle Strength Exchange method (DC PSE)

2.3.1 General DC PSE operator

The Particle Strength Exchange (PSE) method was introduced by Degond and Mas-Gallic in 1989 [35]. Initially developed to approximate the diffusion operator of the convection-diffusion equations, the method was generalized by Eldredge et al. in 2002 [36] in order to approximate any derivative order. The Discretization-Corrected Particle Strength Exchange (DC PSE) method was introduced by Schrader et al. [18] in 2010 in order to account for the discretization of the domain in the operator calculation. This allows removing the discretization error, which led to the name being "Discretization-Corrected".

The DC PSE method is based on a Taylor's series expansion of the unknown field. Convolution functions are used to select the approximated derivative terms. All the other unknown terms of the expansion are canceled out by the convolution functions. The Taylor's series expansions presented in Equation (2.1) and Equation (2.2) are convoluted by a function η over a domain Ω_c:

$$\int_{\Omega_c} f(\mathbf{X_p})\eta(\mathbf{X_p}-\mathbf{X_c})d\mathbf{X_p} = \sum_{i=0}^{+\infty}\sum_{j=0}^{+\infty}\int_{\Omega_c} \frac{\partial^{i+j} f(\mathbf{X_c})}{\partial x^i \partial x^j}\frac{(x_p-x_c)^i}{i!}\frac{(y_p-y_c)^j}{j!}\eta(\mathbf{X_p}-\mathbf{X_c})d\mathbf{X_p}. \tag{2.17}$$

The second order approximation of Equation (2.17) is written as follows:

$$\begin{aligned}\int_{\Omega_c} f_h(\mathbf{X_p})\eta(\mathbf{X_p}-\mathbf{X_c})d\mathbf{X_p} = &\int_{\Omega_c} f(\mathbf{X_c})\eta(\mathbf{X_p}-\mathbf{X_c})d\mathbf{X_p}\\ &+\int_{\Omega_c}\frac{\partial f(\mathbf{X_c})}{\partial x}(x_p-x_c)\eta(\mathbf{X_p}-\mathbf{X_c})d\mathbf{X_p}\\ &+\int_{\Omega_c}\frac{\partial f(\mathbf{X_c})}{\partial y}(y_p-y_c)\eta(\mathbf{X_p}-\mathbf{X_c})d\mathbf{X_p}\\ &+\int_{\Omega_c}\frac{\partial^2 f(\mathbf{X_c})}{\partial x^2}\frac{(x_p-x_c)^2}{2!}\eta(\mathbf{X_p}-\mathbf{X_c})d\mathbf{X_p}\\ &+\int_{\Omega_c}\frac{\partial^2 f(\mathbf{X_c})}{\partial x \partial xy}(x_p-x_c)(y_p-y_c)\eta(\mathbf{X_p}-\mathbf{X_c})d\mathbf{X_p}\\ &+\int_{\Omega_c}\frac{\partial^2 f(\mathbf{X_c})}{\partial y^2}\frac{(y_p-y_c)^2}{2!}\eta(\mathbf{X_p}-\mathbf{X_c})d\mathbf{X_p}.\end{aligned} \tag{2.18}$$

Equation (2.17) and Equation (2.18) can be simplified by introducing the moments $M_{i,j}(\mathbf{X_c})$ defined by:

$$M_{i,j}(\mathbf{X_c}) = \int_{\Omega_c}\frac{(x_p-x_c)^i}{i!}\frac{(y_p-y_c)^j}{j!}\eta(\mathbf{X_p}-\mathbf{X_c})d\mathbf{X_p}. \tag{2.19}$$

Assuming that the field is relatively smooth in Ω_c, the integration can be transformed into a discrete summation over the nodes of the domain. Constant volumes V_p are associated with each of the nodes of the domain. The moments then become:

$$M_{i,j}(\mathbf{X_c}) = \sum_{p\in\Omega_c} V_p\frac{(x_p-x_c)^i}{i!}\frac{(y_p-y_c)^j}{j!}\eta(\mathbf{X_p}-\mathbf{X_c}). \tag{2.20}$$

The volumes V_p associated with the particles p are hard to determine in the general case. Assuming a uniform distribution of the particles over the domain, these values are typically set to unity.

Equation (2.20) becomes:

$$M_{i,j}(\mathbf{X_c}) = \sum_{p\in\Omega_c} \frac{(x_p - x_c)^i}{i!} \frac{(y_p - y_c)^j}{j!} \eta(\mathbf{X_p} - \mathbf{X_c}). \tag{2.21}$$

Using these moments, Equation (2.17) and Equation (2.18), respectively, become:

$$\sum_{p\in\Omega_c} f_h(\mathbf{X_p})\eta(\mathbf{X_p} - \mathbf{X_c}) = \sum_{i=0}^{+\infty}\sum_{j=0}^{+\infty} \frac{\partial^{i+j} f(\mathbf{X_c})}{\partial x^i \partial x^j} M_{i,j}(\mathbf{X_c}), \tag{2.22}$$

$$\begin{aligned}\sum_{p\in\Omega_c} f_h(\mathbf{X_p})\eta(\mathbf{X_p} - \mathbf{X_c}) =& f(\mathbf{X_c})M_{0,0}(\mathbf{X_c}) + \frac{\partial f(\mathbf{X_c})}{\partial x} M_{1,0}(\mathbf{X_c}) \\ &+ \frac{\partial f(\mathbf{X_c})}{\partial y} M_{0,1}(\mathbf{X_c}) + \frac{\partial f^2(\mathbf{X_c})}{\partial x^2} M_{2,0}(\mathbf{X_c}) \\ &+ \frac{\partial f^2(\mathbf{X_c})}{\partial x \partial y} M_{1,1}(\mathbf{X_c}) + \frac{\partial f^2(\mathbf{X_c})}{\partial y^2} M_{0,2}(\mathbf{X_c}).\end{aligned} \tag{2.23}$$

The selection of an appropriate function η allows approximating the desired derivative $D^{k,l} f(\mathbf{X_c}) = \frac{\partial f^{k+l}(\mathbf{X_c})}{\partial^k x \partial^l y}$ by setting all the moments to zero except the one multiplying $D^{k,l} f(\mathbf{X_c})$, which is set to unity. Equation (2.22) can then be written:

$$\begin{cases} D^{k,l} f(\mathbf{X_c}) = \sum_{p\in\Omega_c} f_h(\mathbf{X_p})\eta(\mathbf{X_p} - \mathbf{X_c}) \\ \text{with} \quad M_{k,l}(\mathbf{X_c}) = 1 \\ \qquad\quad M_{i,j}(\mathbf{X_c}) = 0 \quad \text{if } (i,j) \neq (k,l). \end{cases} \tag{2.24}$$

2.3.2 The convolution function

The convolution function η needs to be chosen carefully to satisfy at each node of the domain the moment condition of Equation (2.24). Schrader et al. [37] performed a study of a wide range of functions. In general, the convolution function is composed of the product of two functions: the correction function K and the weight function w:

$$\eta(\mathbf{X}) = K(\mathbf{X})w(\mathbf{X}). \tag{2.25}$$

The correction function is typically derived from a polynomial or an exponential basis. For the case of a two dimensional problem, the polynomial basis $\mathbf{P}(\mathbf{X}) = [1, x, y, x^2, xy, y^2]^T$ can be selected. The weight function is a function that returns a scalar based on the distance to a defined origin. It has typically a compact support: the weights are null outside of a defined perimeter. For isotropic weight functions (functions with similar behavior in every direction), the support of a collocation node $\mathbf{X_c}$ is limited by a radius R_c. The normalized distance to the collocation node is written as s. For the node $\mathbf{X_p}$ within the support of $\mathbf{X_c}$, s is written as s_p and equals to:

$$s_p = \frac{\|\mathbf{X_p} - \mathbf{X_c}\|_2}{R_c}. \tag{2.26}$$

The shape of the weight function has a significant impact on the solution as it balances the contribution of each node of the support in the field derivative approximation. Three types of weight functions can be considered in particular. These are:

The exponential weight functions:

$$w(s) = \begin{cases} e(-s^{\alpha}\epsilon^{-2}) & \text{if } s \leq 1 \\ 0 & \text{if } s > 1, \end{cases} \tag{2.27}$$

where α is an exponent and ϵ is a shape parameter,

3$^{\text{rd}}$ order spline weight functions:

$$w(s) = \begin{cases} \frac{2}{3} - 4s^2 + 4s^3 & \text{if } s \leq 0.5 \\ \frac{4}{3} - 4s + 4s^2 - \frac{4}{3}s^3 & \text{if } 0.5 < s \leq 1 \\ 0 & \text{if } s > 1, \end{cases} \tag{2.28}$$

4th order spline weight functions:

$$w(s) = \begin{cases} 1 - 6s^2 + 8s^3 - 3s^4 & \text{if } s \leq 1 \\ 0 & \text{if } s > 1. \end{cases} \tag{2.29}$$

A typical convolution function, composed of a polynomial correction function $\mathbf{P}$ and a vector of coefficients $\mathbf{a}$, is written as:

$$\eta(\mathbf{X_p} - \mathbf{X_c}) = \mathbf{P}(\mathbf{X_p} - \mathbf{X_c})^T \mathbf{a}\ w(s_p). \tag{2.30}$$

In order for this convolution function to satisfy the moment condition, the polynomial order shall be of at least the derivation order.

2.3.3 Correction function calculation

The coefficient vector $\mathbf{a}$ is the solution of a linear system $\mathbf{A_M}(\mathbf{X_c})\mathbf{a} = \mathbf{B_M}$, where the left side of the equation corresponds to the moments calculation for the unknown convolution function η. The vector $\mathbf{B_M}$ corresponds to the moment condition which needs to be satisfied to obtain the desired derivative approximation. For instance, in order to approximate the derivative $D^{2,0}f(\mathbf{X_c})$, the system is:

$$\begin{cases} M_{0,0}(\mathbf{X_c}) = 0 & \Leftrightarrow \quad \sum\limits_{p\in\Omega_c} \mathbf{P}(\mathbf{X_p} - \mathbf{X_c})^T \mathbf{a} w(s_p) = 0 \\ M_{1,0}(\mathbf{X_c}) = 0 & \Leftrightarrow \quad \sum\limits_{p\in\Omega_c} (x_p - x_c)\mathbf{P}(\mathbf{X_p} - \mathbf{X_c})^T \mathbf{a} w(s_p) = 0 \\ M_{0,1}(\mathbf{X_c}) = 0 & \Leftrightarrow \quad \sum\limits_{p\in\Omega_c} (y_p - y_c)\mathbf{P}(\mathbf{X_p} - \mathbf{X_c})^T \mathbf{a} w(s_p) = 0 \\ M_{2,0}(\mathbf{X_c}) = 1 & \Leftrightarrow \quad \sum\limits_{p\in\Omega_c} \frac{(x_p - x_c)^2}{2!}\mathbf{P}(\mathbf{X_p} - \mathbf{X_c})^T \mathbf{a} w(s_p) = 1 \\ M_{1,1}(\mathbf{X_c}) = 0 & \Leftrightarrow \quad \sum\limits_{p\in\Omega_c} (x_p - x_c)(y_p - y_c)\mathbf{P}(\mathbf{X_p} - \mathbf{X_c})^T \mathbf{a} w(s_p) = 0 \\ M_{0,2}(\mathbf{X_c}) = 0 & \Leftrightarrow \quad \sum\limits_{p\in\Omega_c} \frac{(y_p - y_c)^2}{2!}\mathbf{P}(\mathbf{X_p} - \mathbf{X_c})^T \mathbf{a} w(s_p) = 0. \end{cases} \tag{2.31}$$

Considering the vector $\mathbf{Q}(\mathbf{X_c}, \mathbf{X_p}) = [1, (x_p - x_c), (y_p - y_c), \frac{(x_p - x_c)^2}{2!}, (x_p - x_c)(y_p - y_c), \frac{(y_p - y_c)^2}{2!}]^T$, the correction function basis $\mathbf{P}$ and the weight function w, the coefficients of the matrix $\mathbf{A_M} \in \mathbb{R}^{6\times 6}$ can be written:

$$A_{M(i,j)}(\mathbf{X_c}) = \sum_{p\in\Omega_c} Q_i(\mathbf{X_c}, \mathbf{X_p}) P_j(\mathbf{X_p} - \mathbf{X_c}) w(s_p). \tag{2.32}$$

Having solved the system of equations ($\mathbf{a} = \mathbf{A_M^{-1}}(\mathbf{X_c})\mathbf{B_M}$), the derivative $D^{2,0}f(\mathbf{X_c})$ can be approximated with Equation (2.24) as a function of $f_h(\mathbf{X_p})$, $p \in \Omega_c$. From a computational point of view, it shall be noted that the inversion of the matrix $\mathbf{A_M}(\mathbf{X_c})$ only needs to be performed once per collocation node $\mathbf{X_c}$. If the approximation of another derivative is required for the solution of the partial differential equation, only the moment condition set by the vector $\mathbf{B_M}$ is updated.

2.3.4 Identified variations of DC PSE method

We note from the DC PSE method presented above that setting the moment $M_{0,0}(\mathbf{X_c})$ to zero is not necessary. The value $f(\mathbf{X_c})$ is determined in the global problem, and does not need to be canceled by a null moment $M_{0,0}(\mathbf{X_c})$. Based on this remark, we identify three approaches that we labeled DCPSE0, DCPSE1 and DCPSE2.

- DCPSE0: $M_{0,0}(\mathbf{X_c})$ is set to zero (case presented above);
- DCPSE1: $M_{0,0}(\mathbf{X_c})$ is set to a constant value (e.g. 1), thereby reducing the sparsity of the global matrix;
- DCPSE2: The moment $M_{0,0}(\mathbf{X_c})$ is not introduced in the polynomial coefficient calculation. The dimension of the polynomial basis is then reduced by one. The matrix $\mathbf{A_M}$ belongs to $\mathbb{R}^{5\times 5}$.

We compare all of these methods in Section 2.4.

2.4 DC PSE variations comparison

The three variations of the DC PSE method are studied in this section. The results in terms of L_2 weighted error norms are compared in Figure 2.3 and in Figure 2.4, respectively, for the 2D cylinder and the 2D L-shape problems. The results are shown for the σ_{11} and σ_{12} stress components.

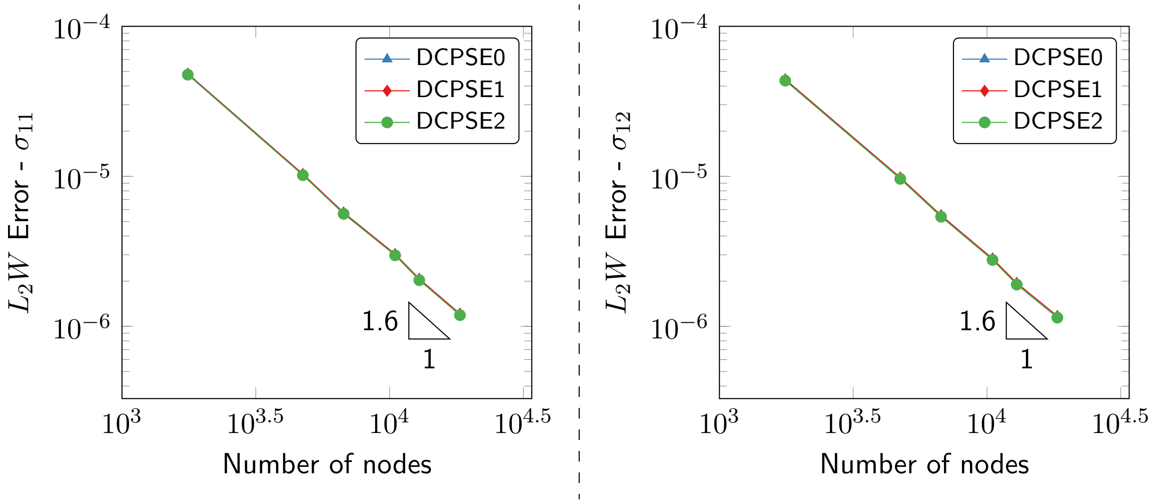


Figure 2.3: DC PSE method variations comparison - 2D Cylinder. L_2 weighted error as a function of the number of nodes in the model for the DCPSE0, DCPSE1 and DCPSE2 variations of the DC PSE method. No distinction can be observed from the results obtained with the different methods.

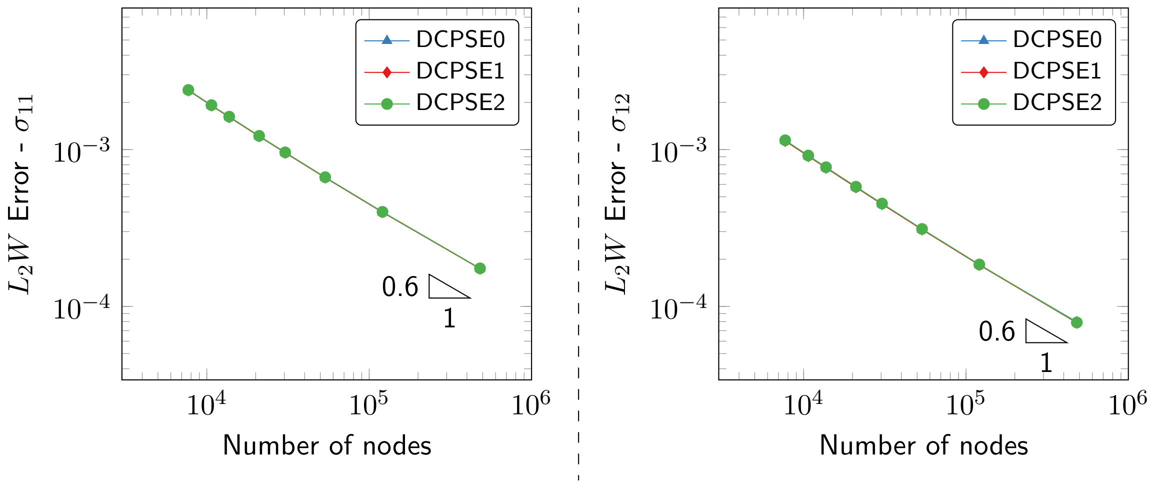


Figure 2.4: DC PSE method variations comparison - 2D L-Shape. L_2 weighted error as a function of the number of nodes in the model for the DCPSE0, DCPSE1 and DCPSE2 variations of the DC PSE method. No distinction can be observed between the different methods.

We observed from Figure 2.3 and Figure 2.4 that all three methods lead to very similar results. In order to quantify the difference, we presente the relative difference between the L_2 weighted error norm for the DCPSE0 and DCPSE1 variation of the DC PSE method and the L_2 weighted error norm for the DCPSE2 in Figure 2.5 and Figure 2.6. We denote this relative difference $\delta_{\text{DCPSE2}-L_2W}$.

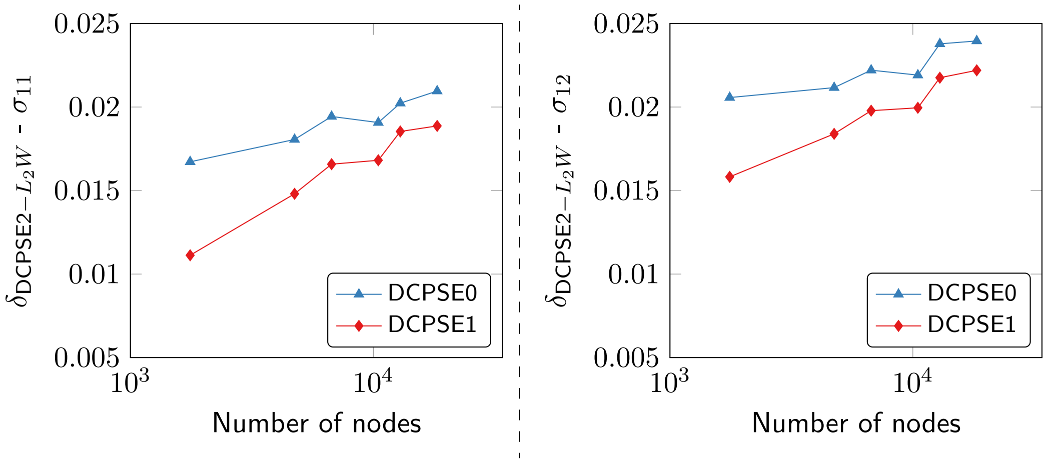


Figure 2.5: DC PSE method variations comparison - 2D Cylinder. Relative difference to the DCPSE2 L_2 weighted error for the DCPSE0 and DCPSE1 methods. The DCPSE2 methods leads to the lowest error followed by the DCPSE1 method.

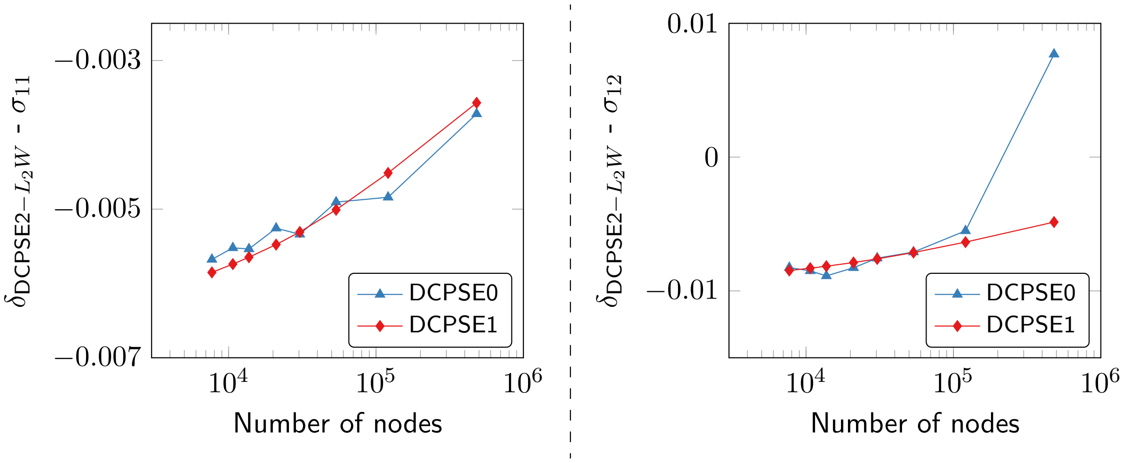


Figure 2.6: DC PSE method variations comparison - 2D L-Shape. Relative difference to the DCPSE2 L_2 weighted error for the DCPSE0 and DCPSE1 methods. The DCPSE1 method leads to the lowest error followed by the DCPSE0 method.

We observe from Figure 2.5 that the minimum error is obtained with the DCPSE2 method. The DCPSE1 method leads to a slightly lower error than the DCPSE0 method. The errors obtained with the DCPSE0 and DCPSE1 methods are between 1% and 2.5% larger than the errors obtained with the DCPSE2 method. For the 2D L-shape problem, the results presented in Figure 2.6 show that the DCPSE2 method leads to a larger error for most node densities. The DCPSE0 and DCPSE1 methods lead to similar errors.

We conclude from this study that the variations of the DC PSE method lead to very similar results. The assembly of the linear problem is slightly faster with the DCPSE2 method as the coefficients of the correction function are obtained with the inversion of a linear problem of a lower dimension. The DCPSE1 method leads in general to a lower error than the DCPSE0 method, and to an error lower than the DCPSE2 method for the 2D L-Shape problem. We selected this method for the comparison to other collocation methods presented in Section 2.5.

2.5 GFD, DC PSE and other methods comparison

2.5.1 General

In this section, we compare the GFD and the DC PSE methods to the MLS, IMLS and RBF-FD collocation methods. We choose the same number of support nodes for all methods. We select a 3$^{\text{rd}}$ order spline weight function for the MLS method and the weight function presented in Equation (2.33), for the IMLS method, with $n = 4$ and $\epsilon = 10^{-15}$. We select an exponential radial basis function for the RBF-FD methods. We use the function defined in Equation (2.27) with $\alpha = 2$ and $\epsilon = 0.33$. The GFD and the DC PSE methods are based on the parameters obtained from the parametric study presented in Section 5.2 and summarized in Table 5.1. For reference purposes, results for the finite element method (FEM), obtained using the commercial software ABAQUS [38], are also included in the comparison.

$$w(s) = \begin{cases} e^{-s^2}(s^n - \epsilon)^{-1} & \text{if } s \leq 1 \\ 0 & \text{if } s > 1. \end{cases} \tag{2.33}$$

The FE results are extrapolated to the nodes. This allows the results from the FEM to be compared with the same error norms to the results obtained with the collocation methods. We used the same discretization as for the collocation model for the FE models. The adjacent nodes of the regular distribution have been grouped into bilinear quadrilateral elements with four integration points.

2.5.2 Results for the 2D cylinder under internal pressure

We observe from Figure 2.7 that the DCPSE1 method leads to the lowest error in terms of L_2 weighted and L_∞ error norms for the σ_{11} and σ_{12} stress components. The GFD method leads to a slightly higher error than the DC PSE method. The MLS and IMLS methods lead to very similar results. The IMLS method leads to an error constantly lower than the MLS method. Finally, the FEM and the RBF-FD method lead to the largest errors. The error obtained with the RBF-FD method does not monotonically decrease as the node density increases.

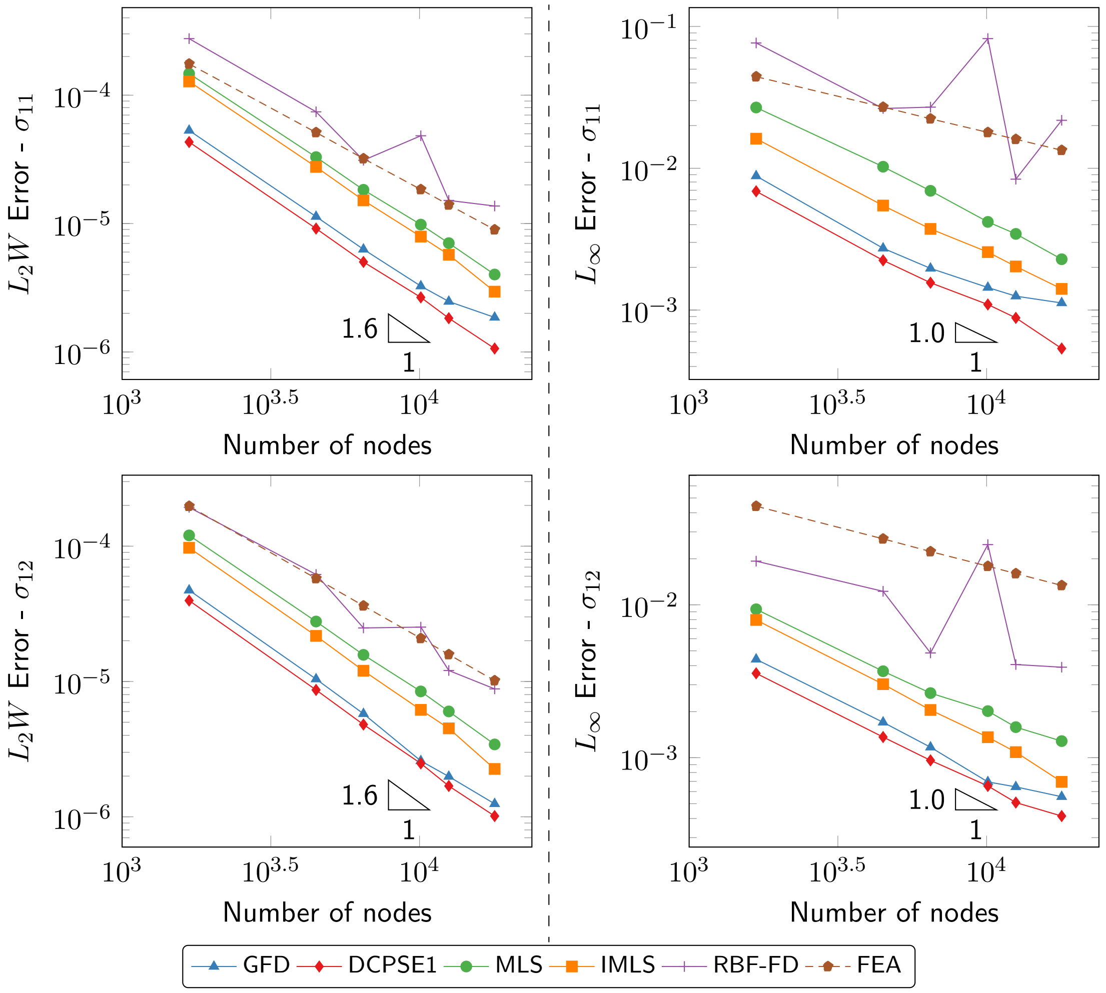


Figure 2.7: Methods comparison - 2D Cylinder. L_2 weighted (Left) and L_∞ (Right) errors as a function of the number of nodes for various collocation methods (i.e. GFD, DCPSE1, MLS, IMLS and RBF-FD) and for the FEM. The lowest error is obtained with the DCPSE1 method.

2.5.3 Results for the L-Shape problem

The different methods are also compared for the 2D L-shape problem. We present only the L_2 weighted error for this problem as the L_∞ error diverges to infinity as the distance to the singular node tends to zero. The results are presented in Figure 2.8.

We can see in Figure 2.8 that the results obtained with the FEM are the closest to the analytical solution. The MLS, IMLS, GFD and DC PSE methods lead to similar results. The trend is however opposite to the results presented in Figure 2.7. It shows how the methods are affected by a rapid change in the field solution. The MLS method is the method leading to the lowest error among the collocation methods. Finally, the RBF-FD is the method leading to the highest error.

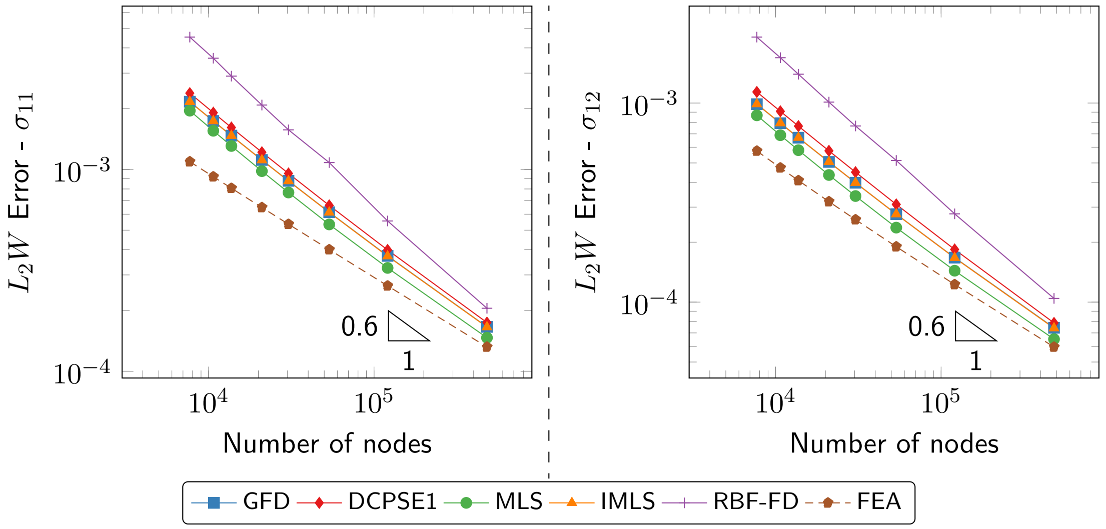


Figure 2.8: Methods comparison - 2D L-Shape. L_2 weighted errors as a function of the number of nodes for various collocation methods (i.e. GFD, DCPSE1, MLS, IMLS and RBF-FD) and for the FEM. The lowest error is obtained with the FEM. The collocation method leading to the lowest error is the MLS method.

2.5.4 Convergence rate and computational expense

We compare the methods in terms of convergence rate and solution time. We summarize the results in Table 2.1. We observe that the RBF-FD method has the lowest convergence rate for the 2D cylinder problem, and the largest convergence rate for the L-shape problem. The IMLS method shows the largest convergence rate for the 2D cylinder problem. The GFD method is the one having the lowest convergence rate for the L-shape problem.

The right column of Table 2.1 shows that the computation time for the MLS and IMLS method is significantly larger than the computation time for the other methods. This is due to the assembly step which is more time consuming for these methods, as a system of equations needs to be solved for each derivative.

Table 2.1: Comparison in terms of convergence rate and computation time of the considered point collocation methods.

Method	**Average L_2W convergence rate**		**Computation time [(1)]**
	2D Cylinder	2D L-Shape	
GFD	1.5109	0.6244	9.3s
DCPSE1	1.5592	0.6405	11.4s
MLS	1.5064	0.6249	19.7s
IMLS	1.5717	0.6279	20.5s
RBF-FD	1.0952	0.7401	10.1s

(1) Based on a 12,087 nodes 2D cylinder model solved with a direct solver.

The computation times presented in Table 2.1 can be split into four main steps. These steps are:

- Problem initialization;
- Matrix assembly;
- Solution of the linear problem;
- Postprocessing of the results.

The initialization step consists in loading the problem from the input file and searching for the node neighbors (the nodes to be included in the support of each collocation node). During the assembly step, all the derivatives of the unknown field are approximated as a function of the field, and the linear problem is assembled. For the solution of the linear problem, while direct solvers can be used for 2D problems of a reasonable size, iterative solvers should be used for large 3D problems, as the matrix of the linear problem is significantly denser than for 2D problems. Finally, the postprocessing step consists in the computation of the quantities of interest (e.g. stress components based on the displacement field). In this thesis, we used the linear solver MUMPS [39, 40] and the iterative solver PETSc KSP [41, 42]. We ran the analyses using a C++ code developed in-house. The code was run on a machine equipped with an Intel Xeon E5-1650 processor at 3.2 GHz. We used a single process and a single thread for the analyses.

For various node densities, the fraction of the total analysis time of the matrix assembly step and of the solution step is presented in Figure 2.9 and Figure 2.10 for the GFD and IMLS methods, respectively. We selected the L-shape problem for the purpose of this comparison. The duration of the problem initialization and of the postprocessing steps is negligible compared to the two other steps. Therefore, we did not present these steps in the figures. We also present the total analysis duration in these figures on a secondary axis. We selected the GFD and the IMLS methods because these methods are, respectively, the fastest and the slowest methods for the node density presented in Table 2.1.

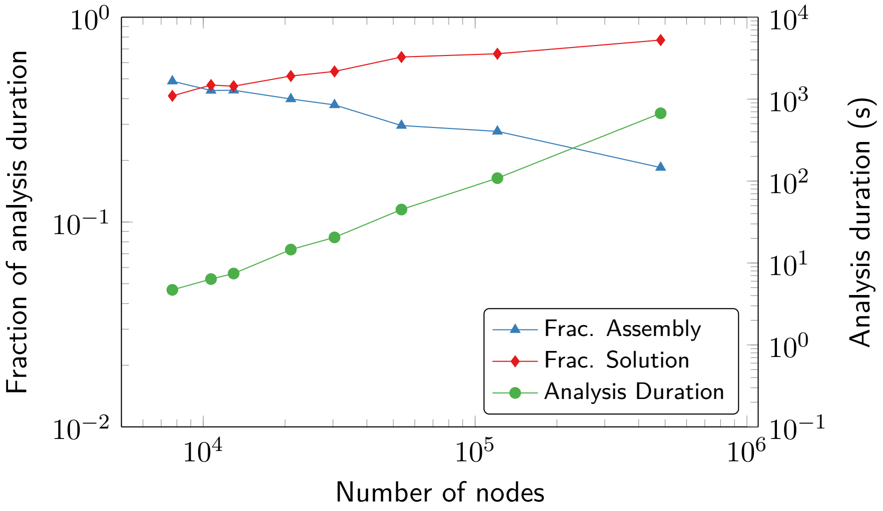


Figure 2.9: Computation time split and analysis duration - GFD method. Impact of the number of nodes on the fraction of the analysis spent in each step and total analysis time.

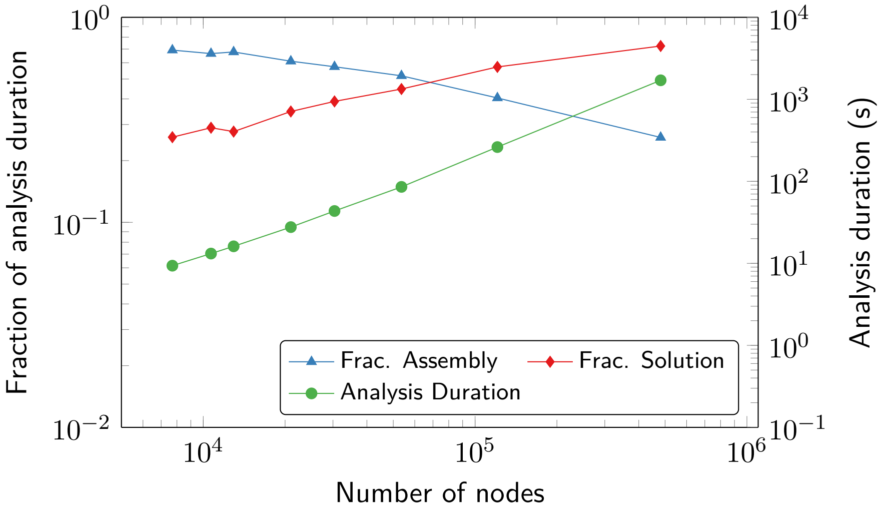


Figure 2.10: Computation time split and analysis duration - IMLS method. Impact of the number of nodes on the fraction of the analysis spent in each step and total analysis time.

We observe from Figure 2.9 and Figure 2.10 that for both methods the trend of the results is similar. The assembly step represents the largest proportion of the total analysis time for 2D problems of small dimensions. As the number of degrees of freedom increases, the fraction of the solution step (here with a direct solver) in the analysis time increases and becomes larger than the fraction of the assembly step. This result is expected as the assembly time increases linearly with the number of nodes while the solution time increases exponentially with the number of degrees of freedom. The increased computation effort required by the IMLS method during the assembly step is observed on this graph as the fraction of the analysis spent assembling the matrix is larger than for the GFD method.

2.6 GFD, FEA comparison for three-dimensional problems

We compared in Section 2.5 results obtained with the GFD and the DC PSE methods to results obtained with other methods for two benchmark problems. We compare in this section results obtained from a collocation method, the GFD method, to results obtained using Finite Element Analysis (FEA). We decided to present results for the GFD method only in this section since we have a greater focus on this method in this thesis.

In this section, we present the results from the stress analysis of various three dimensional problems. We compare the results in terms of von Mises stress obtained with the GFD method to results obtained with ABAQUS. For a consistent comparison, the same discretization is used for both methods. The results obtained from the FEA are extrapolated to the nodes. This tends to overestimate the error for the FEA method as the stresses are less accurate at the nodes than at the integration points. This allows, however, a comparison, at each node, of the results derived with the FE method to the results derived with the GFD method. Similarly to the results presented in Section 2.5, we selected the parameters of the GFD method based on the results of the parametric study presented in Section 5.2 and summarized in Table 5.1.

In order to obtain convergence of the 3D problems with the collocation method, a relatively large number of nodes is used. Such a high density is required to capture the details of the geometry.

Flange model

The first problem that we considered is an ISO flange. We modeled only a quarter of the flange due to the symmetries of the domain. We present the model and the various surfaces, on which boundary conditions are applied, in Figure 2.11. We considered two load cases for this model: an internal pressure loading and an axial traction imposed by the connected pipe. We present the boundary conditions associated with each load case in Table 2.2.

Figure 2.12 and Figure 2.13 show the von Mises stresses obtained with the GFD and the FE methods, respectively, for the internal pressure and traction load cases.

The results obtained from the GFD method and from the FEM are very close. In order to visually assess the difference between the two solutions, we present the difference between the von Mises stress results obtained from the GFD model and from the FE model in Figure 2.14 for both load cases.

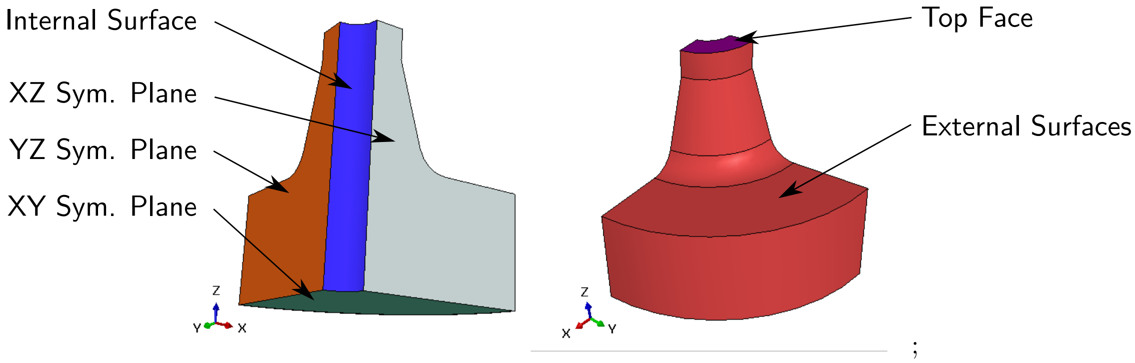


Figure 2.11: Flange model and boundary conditions.

Table 2.2: Boundary conditions applied to the flange for the pressure and displacement load cases. The surfaces are highlighted in Figure 2.11.

Surface	**Boundary Conditions**	
	Pressure Loading	Diplacement loading
XY Sym. Plane	Constrained in the Z direction	Constrained in the Z direction
XZ Sym. Plane	Constrained in the Y direction	Constrained in the Y direction
YZ Sym. Plane	Constrained in the X direction	Constrained in the X direction
Internal Surface	Constant pressure of 1.0	Stress free
External Surface	Stress free	Stress free
Top Face	Constrained in the Z direction	Applied displacement of 6.2e-04 in the Z direction

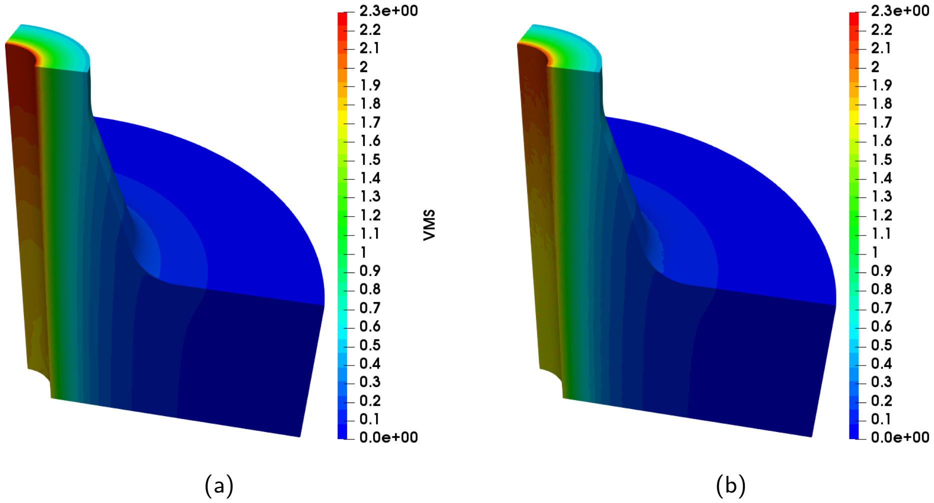


Figure 2.12: Flange ISO PN50 DN25 subject to internal pressure - von Mises stress results from the GFD method (a) and FEM (b) (548,648 nodes). The results from both models are very similar. The stress on the inner surface of the flange is larger for the GFD method.

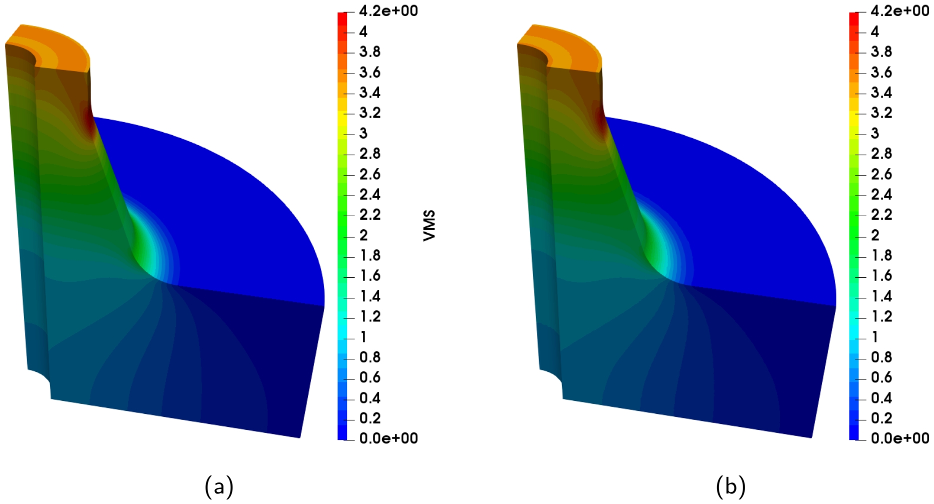
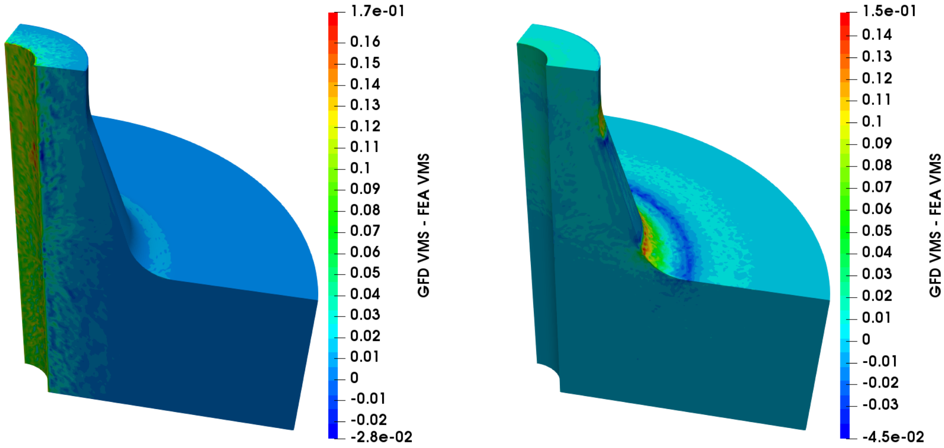


Figure 2.13: Flange ISO PN50 DN25 under traction - von Mises stress results from the GFD method (a) and FEM (b) (548,648 nodes). The results from both models are very similar. The stress in the neck of the flange is slightly larger for the GFD method.



(a)

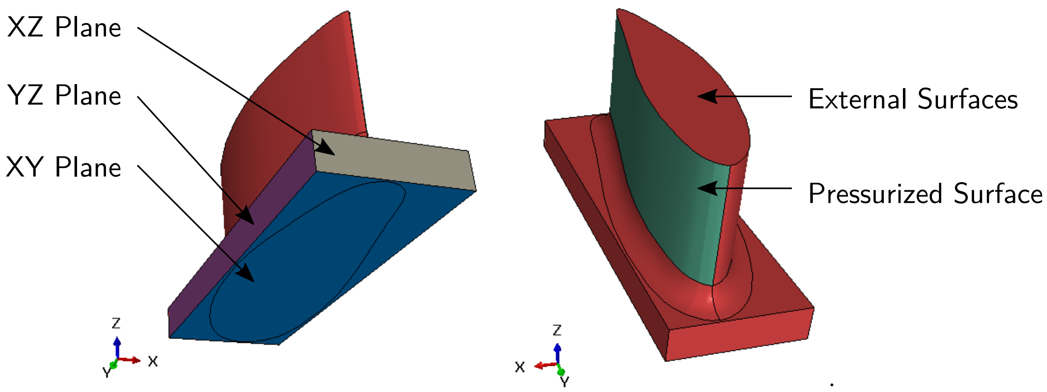
(b)

Figure 2.14: Flange ISO PN50 DN25 under traction - Difference between von Mises stress results obtained from GFD method and FEM for the internal pressure load case (a) and the traction load case (b) (548,648 nodes). The von Mises stress results are larger for the GFD method on the inner surface for the flange under internal pressure, and in the neck and in the cone bottom section for the flange under traction.

We observe from Figure 2.14(a) that the stress on the inner surface of the flange is larger for the GFD method. Figure 2.14(b) shows that the stress concentration in the neck and in the bottom of the conical section is larger for the GFD method.

Blade model

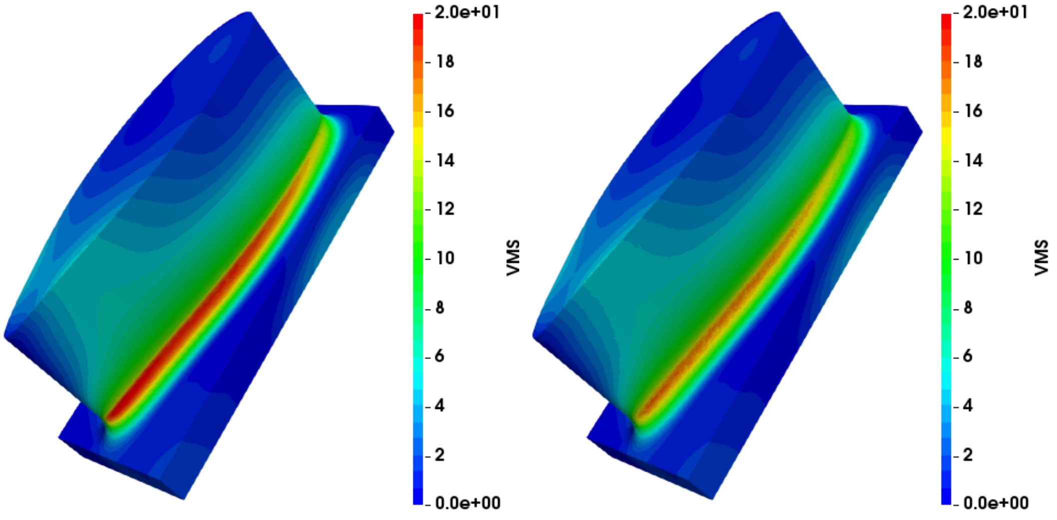
Figure 2.15 presents a simplified model of a high pressure blade. We show the surfaces on which the boundary conditions are applied in this same figure. We fixed the nodes in the planes YZ, XZ and XY, respectively, in the X, Y and Z directions. We applied a constant pressure resulting from a gas flow on the pressurized surface and considered the remaining surfaces of the blade stress free.



;

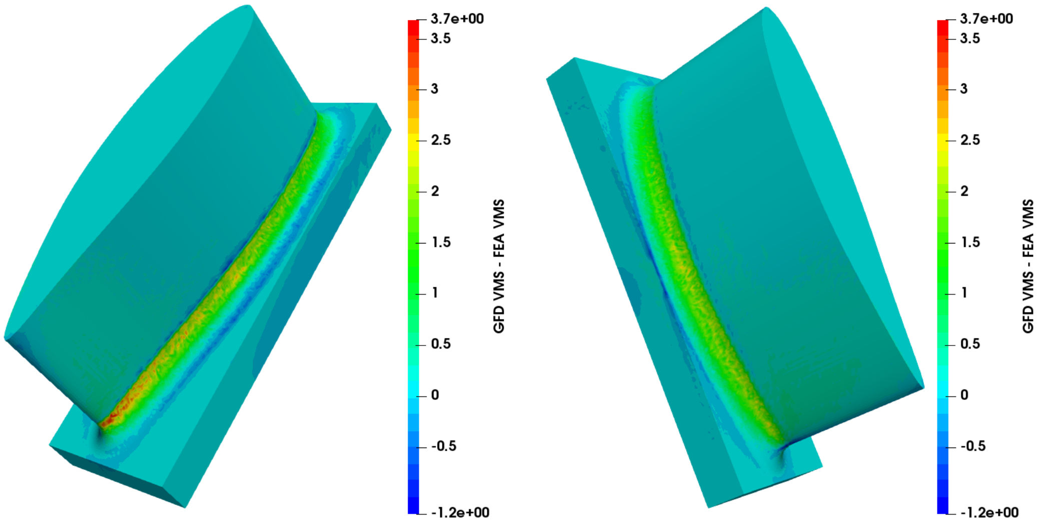
Figure 2.15: Blade model and boundary conditions.

Figure 2.16 shows the von Mises stress results for the GFD and FE methods. As for the flange problem, we present the difference between the two von Mises stress solutions in Figure 2.17.



(a) (b)

Figure 2.16: Simplified high pressure blade subjected to uniform pressure on one face - von Mises stress results obtained with the GFD method (a) and FEM (b) (484,238 nodes).



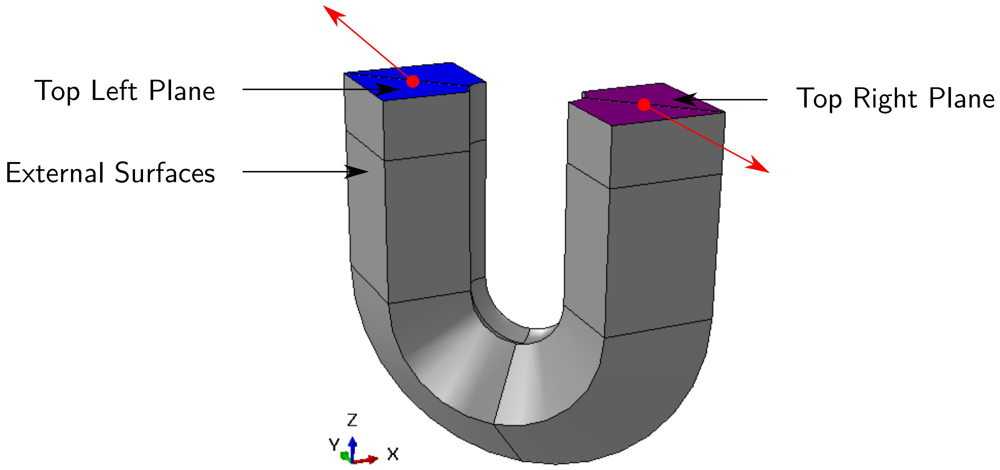
(a) (b)

Figure 2.17: Simplified high pressure blade subjected to uniform pressure on one face - Difference between von Mises stress results obtained with the GFD method and FEM (484,238 nodes). The stress concentration at the interface between the blade and the support is larger for the GFD model than for the FE model.

We observe from Figure 2.16 and Figure 2.17 that the stress concentration in the zone between the blade and the support is larger for the GFD model. The largest difference between the GFD and FE solutions is observed at this location.

Horseshoe model

In 2005, Hughes et al [19] solved the horseshoe problem using the IGA method. We reproduced this model and present it in Figure 2.18. We fixed the nodes of the top left plane in the X and Z directions. We applied a positive and a negative displacement in the Y direction to the nodes of the top left plane and of the top right plane, respectively. These displacements are equal in absolute value. We considered the external surfaces of the horseshoe stress free.

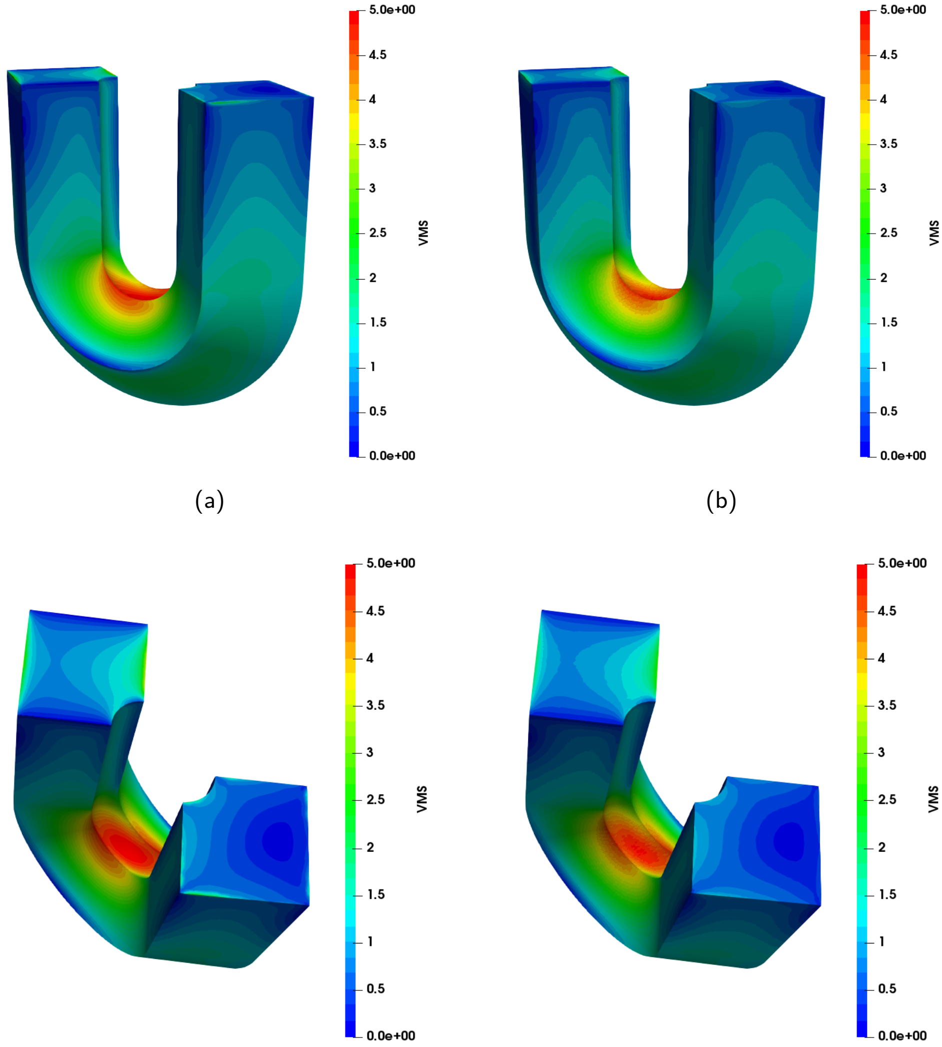


;

Figure 2.18: Horseshoe model and boundary conditions.

Figure 2.19 shows the von Mises stress results obtained from the GFD and FE models. The two figures on the left show two different views of the solution of the problem solved with the GFD method. The two figures on the right show the solution of the problem solved with the FEM. We observe from Figure 2.19 that the stress concentration in the bottom of the horseshoe is larger in the GFD solution. We also observe a higher stress concentration at the edges of the top left plane in the GFD solution.

It should be noted that the computation time was lower for the FE method than for the GFD method. This is due to the loading, which creates a singularity at the edges of the top left plane and of the top right plane. The impact of this singularity affects the GFD model more as it is solved by collocation (strong form of the PDE).



(a)

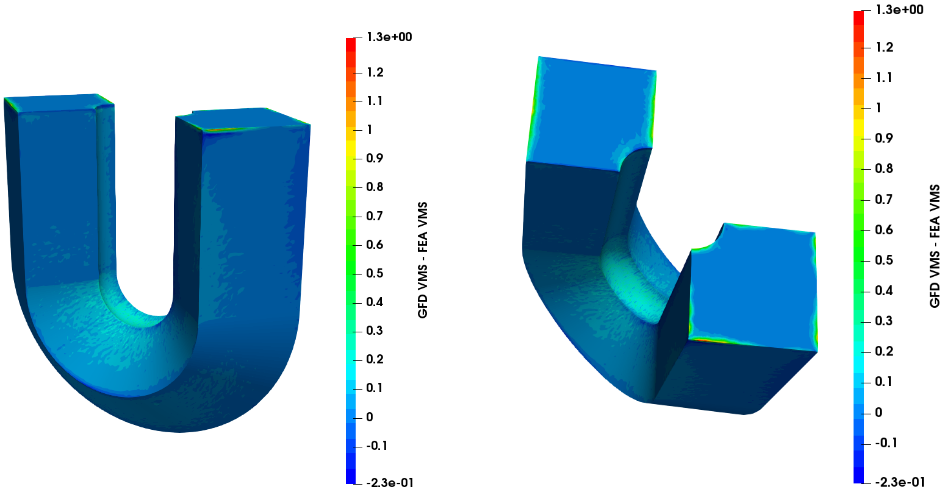
(b)

(c)

(d)

Figure 2.19: Horseshoe under shear loading - von Mises stress results obtained with the GFD method (a) (c) and FEM (b) (d) (521,326 nodes). The stress concentration in the inner surface of the horseshoe is slightly larger for the GFD method.

We present the difference between the two von Mises stress solutions in Figure 2.20. This figure confirms that higher stress concentrations are observed for the GFD model at the top plane's edges and in the bottom section of the horseshoe.



(a)

(b)

Figure 2.20: Horseshoe under shear loading - von Mises stress results obtained with the GFD method (a) and FEM (b) (521,326 nodes). The stress concentration in the inner surface of the horseshoe is slightly larger for the GFD method.

Fichera's corner model

We present the Fichera's corner model, analyzed in [43, 44, 45], in Figure 2.21. The characteristic planes, on which boundary conditions are applied, are highlighted and labeled in this figure. We fixed the nodes in the planes YZ, XZ and XY, respectively, in the X, Y and Z directions. We applied a uniform traction on the front face of the truncated cube. The rest of the surfaces are stress free. We did not include the internal corner nodes in the GFD model. These nodes lead to the divergence of the solution as the stress is infinite at these locations. We applied the visibility criterion, presented in Section 4.3, to this problem.

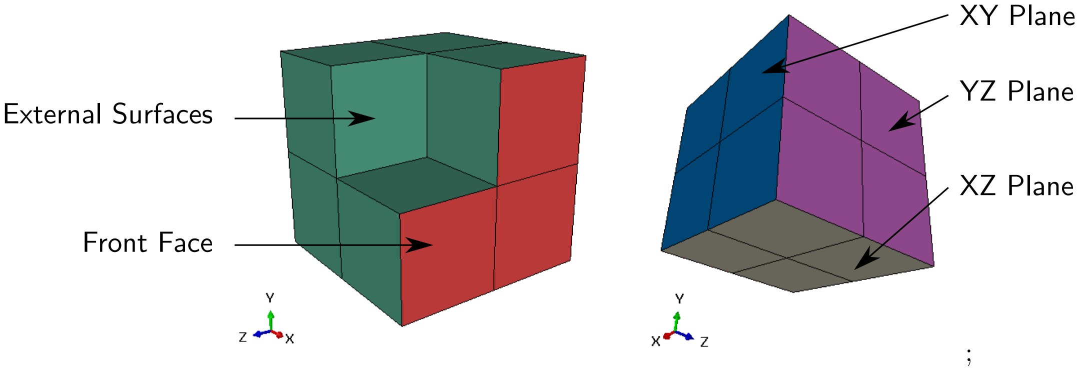
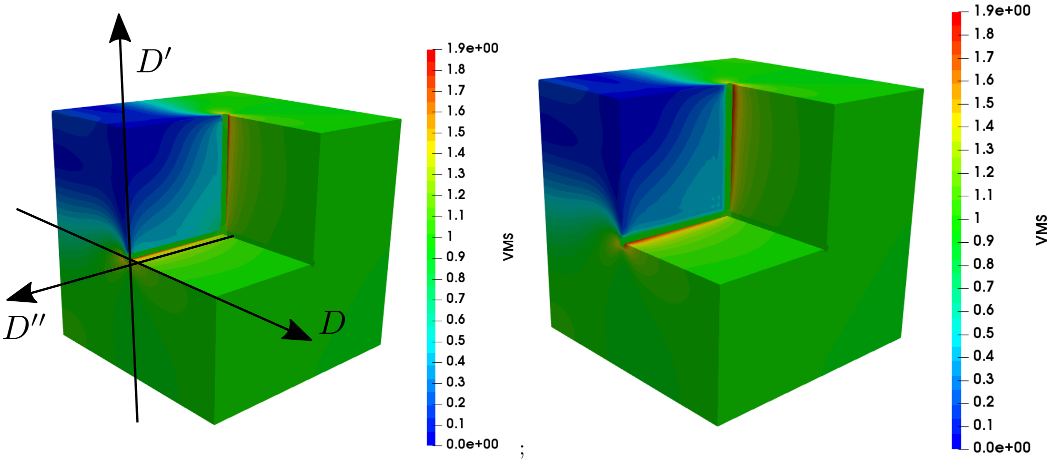


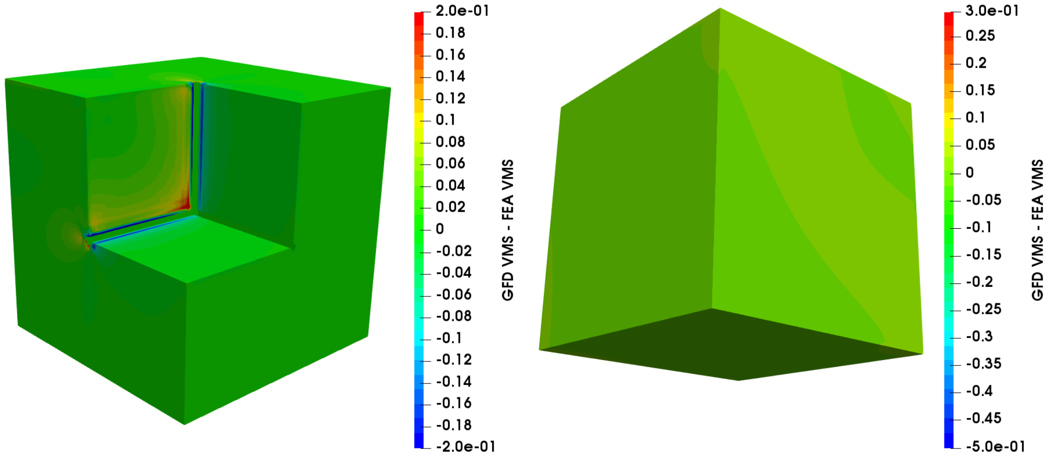
Figure 2.21: Fichera's corner model.

Figure 2.22 shows the von Mises stress results for the GFD and FE methods. We present the difference between the two von Mises stress solutions in Figure 2.23.



(a) (b)

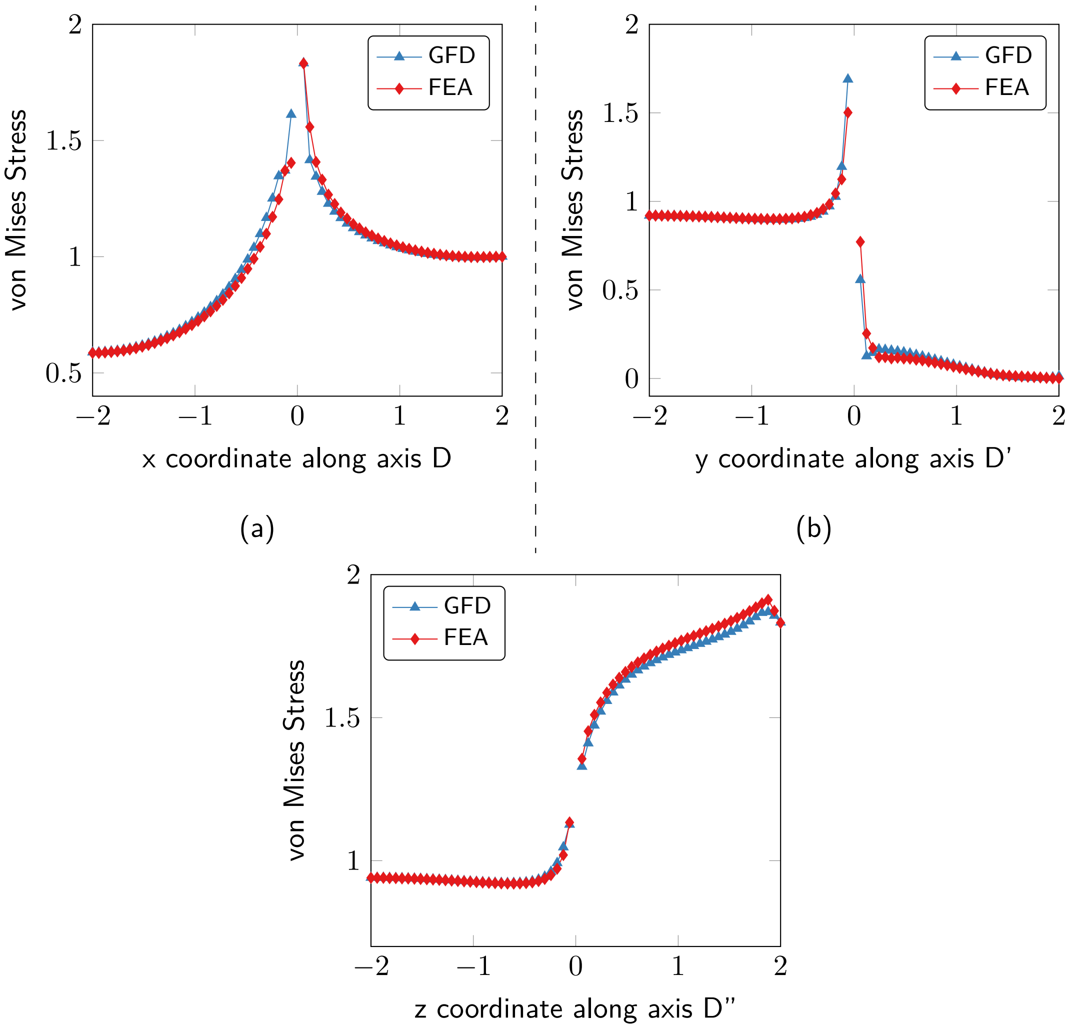
Figure 2.22: Fichera's corner subjected to a uniform traction on the front side - von Mises stress results obtained with the GFD method (a) and FEM (b) (264,726 nodes).



(a) (b)

Figure 2.23: Fichera's corner subjected to a uniform traction on the front side - Difference between von Mises stress results obtained with the GFD method and FEM (264,726 nodes). The stress concentration in the internal corners is larger for the FE model.

We observe from Figure 2.22 and Figure 2.23 that the stress concentration near the internal edges is larger for the FE model than for the GFD model. The GFD model leads to larger results only in the center of the corner. In order to visualize more precisely the results of this analysis, we plotted in Figure 2.24 von Mises stress results along the axes D, D' and D" shown in Figure 2.22(a). The axes D and D' follow the edge of the model, while the axis D" is slightly offset from the internal corner as results are not available at the corner nodes for the GFD method. The truncated cube has an edge length of 4. The coordinate of the internal corner is (0,0,0).



(a)

(b)

(c)

Figure 2.24: The von Mises stress results comparison along the axes D, D' and D", as presented in Figure 2.22(a). Depending on the considered axis, either the GFD method or the FE method leads to the maximum observed stress. The largest stress is observed in the subfigure (c) for the FE method.

Figure 2.24 shows that, depending on the considered axis, either the GFD method or the FE method lead to the largest results. For this model, the FE method is expected to give a higher stress concentration since the corner nodes have not been included in the GFD model but have been included in the FE model.

Discussion

We observe from the results presented in this section that the results obtained with the GFD method are very close to the results obtained with the FEM. The stress concentrations are slightly larger for the flange, the blade and the horseshoe problems when the GFD method is used. This might be due to the use of the strong form of the equations, which allows solving the loading equations on the boundary of the domain. In the FEM, the problem is solved in a weak form using an integration over the domain. For the Fichera's corner problem, we observe the largest von Mises stress concentration for the FE solution. This is due to the internal corner nodes which have not been included in the GFD method. The FEM is thus deemed more accurate.

2.7 Conclusion

We presented the theory of the GFD and DC PSE methods in this chapter. We assessed the impact of three identified variations of the DC PSE method on the error. We observed little difference between the results obtained from these method variations. We selected the DCPSE1 method as it led to a relatively low error for the benchmark problems considered.

We compared the results obtained with the GFD and DC PSE methods to results obtained with other collocation methods and results obtained with the finite element method. The GFD and DC PSE methods lead to the lowest error for the problem of the 2D cylinder under internal pressure. The FE method leads to the lowest error for the problem of the L-shape in mode I loading. The GFD method leads to results similar to those obtained with the FE method for three-dimensional problems. We observed that the GFD method leads to larger stress concentrations than the FE method for the problems with a non-singular solution.

Chapter 3

CAD to smart cloud collocation

3.1 General

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 chapter a method to discretize a given geometry, provided in a STEP file format, with the aim of solving it using a point collocation method. We show how the smart cloud collocation model is generated from the discretization and from the CAD geometry. Then, we analyze the impact of the selected parameters of the method and finally compare the results obtained from different discretization methods.

3.2 Domain discretization

We use the library Open CASCADE Technology [46] to communicate with the STEP files and to get information about the exact geometry. We propose a discretization algorithm 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 comprising the geometry;
4. Identification of the nodes of the rectangle or box included in the domain.

We present these steps in Figure 3.1 for the case of a 2D gear. We give more details about each of these steps in the paragraphs below.

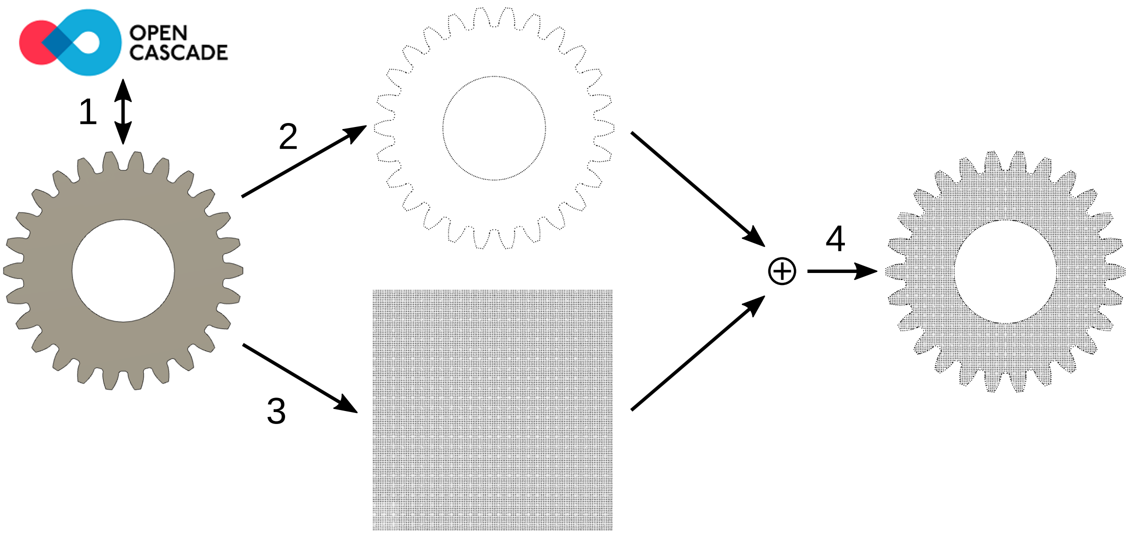


Figure 3.1: Steps of the discretization of a domain from a CAD file using the library Open CASCADE Technology [46].

Step 1

At the beginning of the discretization process key parameters, such as the bounding volume or the dimensions of the rectangle or box comprising 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. A geometrical entity is 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 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 the second 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, at the junction between adjacent edges, 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, at the junction between adjacent faces, are removed. We use the library Gmsh [47] to mesh the surfaces.

We compute the exact normal vectors, at each boundary collocation node, using the information about the exact geometry contained in the STEP file.

Step 3

The rectangle or box comprising the geometry is discretized based on the characteristic length h. 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. Figure 3.2 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 [48].

Step 4

The final step of the discretization process requires the identification of the rectangle or box 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 each node of the domain. We use this approach. The algorithms presented in reference [48] or in reference [27] 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 references [49, 50, 51], can be used in 2D. For 3D problems, the Möller-Trumbore algorithm [52] or the AABB tree algorithm [53] 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 position of all the rectangle or box nodes $\mathbf{X}$ with respect to a domain Ω do not need to be assessed. All the nodes located in a disc or sphere centered at a considered node $\mathbf{X_c}$ and of radius $\|\mathbf{X_c} - \mathbf{X_{pc}}\|_2$, where $\mathbf{X_{pc}}$ is the the projection of $\mathbf{X_c}$ on the boundary of the domain Γ_Ω, are

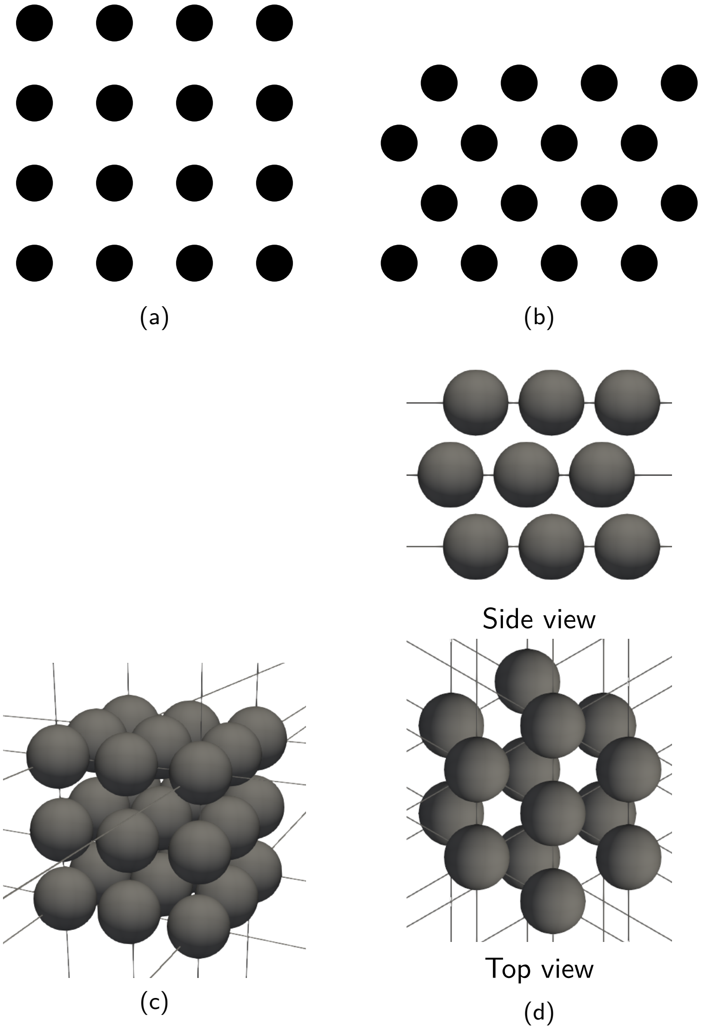


Figure 3.2: Node arrangement configurations in 2D (a), (b) and 3D (c), (d). The subfigures (a) and (b) show the node arrangements for square and triangular lattices, respectively. The subfigures (c) and (d) show the node arrangements for cubic and hexagonal close-packed lattices, respectively.

located on the same side of the boundary as $\mathbf{X_c}$. This is illustrated by Figure 3.3 for a node $\mathbf{X_i}$ located inside of the domain and a node $\mathbf{X_o}$ located outside of the domain. Such an approach speeds-up significantly the node identification process.

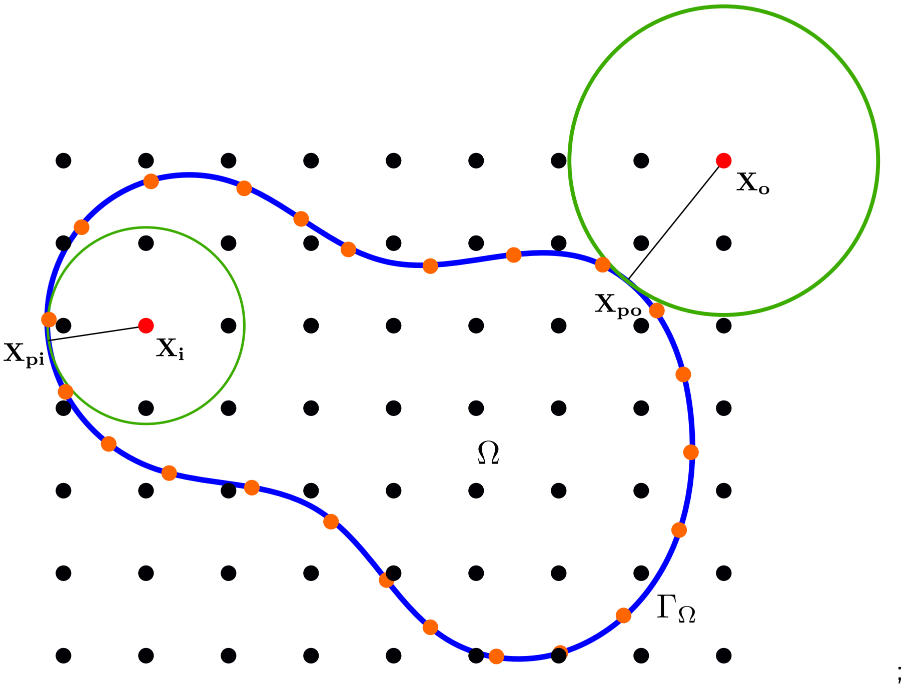


Figure 3.3: Identification of the position of interior nodes with respect to the domain Ω. Considering a node $\mathbf{X_i}$ and its projection $\mathbf{X_{pi}}$ on the boundary of the domain Γ_Ω, all the nodes located within a disc or sphere of radius $\left\|\mathbf{X_i} - \mathbf{X_{pi}}\right\|_2$ 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 disc or sphere of radius $\|\mathbf{X_o} - \mathbf{X_{po}}\|_2$ are located outside of the domain Ω if the node $\mathbf{X_o}$ is outside of the domain Ω.

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. We discuss this aspect in Section 3.4.

3.3 From discretization to smart cloud

Transforming the discretization into a smart cloud, which contains all the required information for its solution using a collocation method and for model adaptivity, is the final step. It consists primarily in the enforcement of the boundary conditions and in the addition of supplementary information about the geometry useful to improve the solution and for 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 define in an input file the boundary condition associated with 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 most relevant boundary condition for each degree of freedom. We apply first non-zero Neumann boundary conditions, then Dirichlet boundary conditions and, finally, homogeneous Neumann boundary conditions.

We explained in Section 3.2 that surface elements are used as part of the discretization of the process. In case of adaptive refinement, the smart nodes are used to carry the boundary condition information 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:

- the base CAD geometry;
- the exact normal vector;
- the boundary conditions;
- the parent CAD edge or surface(s);
- the boundary conditions applied to the parent CAD edge or surface(s);
- the connections to other boundary nodes if boundary elements are used to speed-up the refinement of the surface or enforce the generalized visibility criterion described in Section 4.4.

3.4 Threshold sensitivity analysis

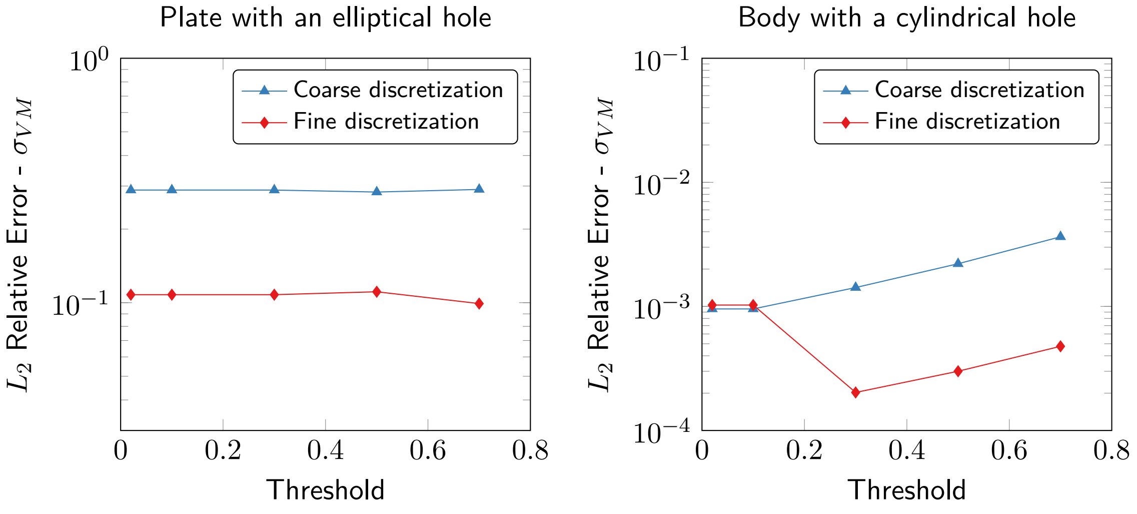
Stencil nodes located too close to each other may lead to ill-conditioning of the linear system solved, at each collocation center, to obtain an approximation of the field derivatives as a function of the field itself. We use a threshold ratio, denoted by t, to determine if an interior node, obtained 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 use two benchmark problems from the field of linear elasticity to assess the impact of the threshold on the error. The problems considered are: a plate with an elliptical hole and an infinite body with a cylindrical 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 1.3 and 1.4, 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 0.7. For the purpose of the sensitivity analysis, we considered coarse and fine discretizations of the benchmark problems (approximately 4,500 and 30,000 nodes, respectively), based on the discretization method presented in Section 3.2. We selected a square lattice to discretize the rectangles 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 1.5.

The results are presented in Figure 3.4. The results show that the threshold has little impact on the error for the problem of a plate with an elliptical hole. The error varies by less than 3% for the coarse discretization. For the fine discretization, the error is lower by approximately 10% for a threshold of 0.7. The variations are more important for the infinite body with a cylindrical 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.

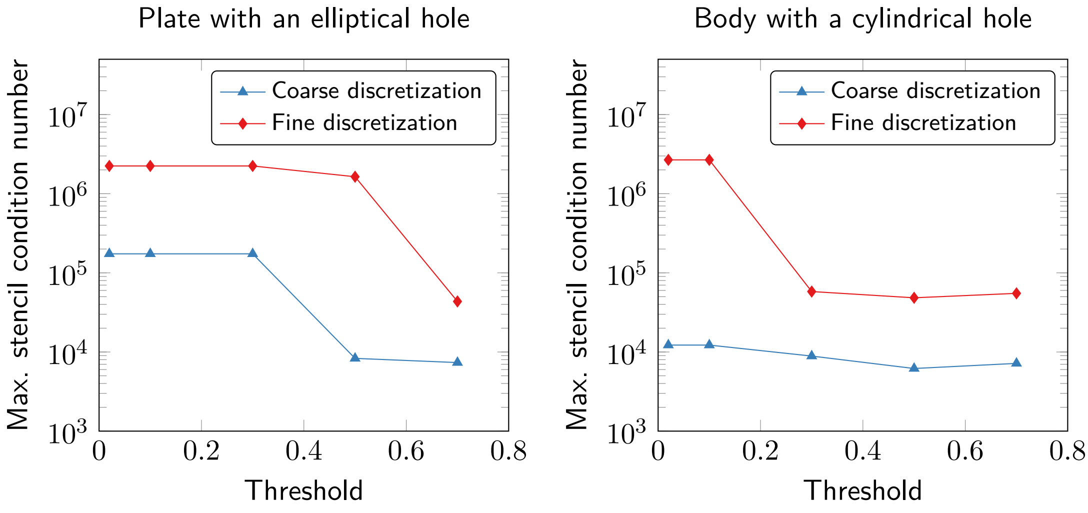
To better understand these results, we plot in Figure 3.5 the maximum condition number of the linear systems solved as part of the field derivatives approximation as a function of the threshold value. We see that the condition number increases as the threshold decreases. For the infinite body with a cylindrical hole, we observe a sharp increase in the condition number for the two lowest threshold values (i.e. 0.02 and 0.1). This increase is associated with an increase of the error. The condition number of the stencil is closely related to the node selection algorithm. In this thesis, we select the stencil nodes based on the distance criterion [4]. Based on the results presented in this section, we select a threshold value of 0.3 as it leads to the lowest error for the body with a cylindrical hole problem. We decide not to investigate other node selection algorithms which could be more suitable for the lowest threshold values.



(a)

(b)

Figure 3.4: Error for threshold values ranging from 0.02 to 0.7 for coarse (approx. 4,500 nodes) and fine (approx. 30,000 nodes) discretizations for the plate with an elliptical hole problem (a) and for the body with a cylindrical hole problem (b). The error in terms of the L_2 relative error norm is presented for the von Mises stress noted σ_{VM}.



(a)

(b)

Figure 3.5: Maximum stencil condition number for threshold values ranging from 0.02 to 0.7 for coarse (approx. 4,500 nodes) and fine (approx. 30,000 nodes) discretizations for the plate with an elliptical hole problem (a) and for the body with a cylindrical hole problem (b).

3.5 Discretization methods comparison

We compare in this section the results obtained using the proposed discretization method to results obtained from discretizations generated using Gmsh [47]. Gmsh is a powerful finite element mesh generator which generates primarily meshes based on a Delaunay triangulation of the geometry. Gmsh is suitable for both 2D and 3D problems. We use this code to generate uniform discretizations of the geometry by using only the nodes of the generated meshes. We consider the same benchmark problems as those mentioned in Section 3.4 for the purpose of this comparison (i.e. a plate with an elliptical hole and an infinite body with a cylindrical hole). We compare results for a square lattice and a triangular lattice discretization of the rectangle comprising the geometry. We use a threshold value of 0.3. The results in terms of the L_2 relative error of the von Mises stress are shown in Figure 3.6.

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

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.

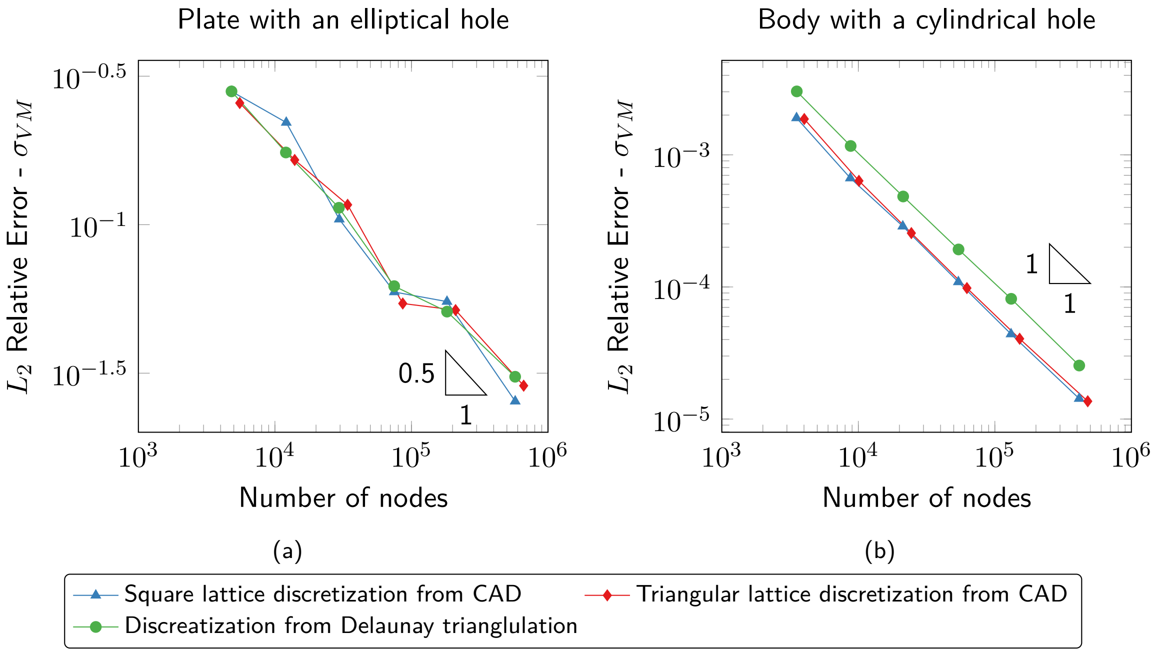


Figure 3.6: Comparison of the error in terms of the L_2 relative error norm obtained from different discretization techniques (i.e. square or triangular lattice discretization from CAD and Delaunay triangulation generated using Gmsh). The results are presented for the plate with an elliptical hole problem (a) and for the body with a cylindrical 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 body with a cylindrical hole problem. The error obtained with these discretizations is lower than the error obtained with the Delaunay triangulation of the domain.

3.6 Conclusion

We introduced in this chapter a method to generate smart point clouds directly from CAD geometries without any intermediate step. The proposed method has a limited number of parameters: the node arrangement of the uniform discretization and the threshold allowing the inclusion of inner nodes close to the boundary of the domain. We selected a threshold value of 0.3 as it led to lowest error for one of the considered problems. We used this threshold in this thesis whenever this discretization algorithm was used in Chapter 6.

The comparison of results obtained from models that use the proposed discretization strategy to results obtained from models based on a Delaunay triangulation of the domain give confidence in the presented method.

Chapter 4

Stencil nodes selection: a unified algorithm

4.1 General

The selection of the stencil nodes considered in the field derivatives approximation is a key step of point collocation methods. Several techniques can be considered for this purpose. Schönauer [54] proposed a method based on the selection of rings of nodes around the collocation node. Kennett et al. [55] introduced a selection method for anisotropic point distributions. Seibold presented a method to ensure that the selected nodes lead to a positive stencil [56]. Davydov proposed an algorithm to select the support nodes based on the angles formed by the collocation node and two adjacent support nodes for 2D problems [15] and another algorithm based on the 8-octant criterion [57] for 3D problems. Even though many methods have been proposed, the most commonly used methods remain the distance criterion [4] and the quadrant criterion introduced by Liszka and Orkisz [6].

The distance criterion is illustrated by Figure 2.1. The stencil nodes associated with a collocation node $\mathbf{X_c}$ are the nodes $\mathbf{X_p}$ within a defined radius R_c. We define the stencil radius based on a selected target number of nodes and include all the nodes located at an equal distance from the collocation node. We assess the impact of the stencil size on the error in Section 4.2.

For singular problems, the selection of the stencil nodes, in the vicinity of the singularity, may be modified to improve the results. The visibility criterion and the diffraction criterion are two popular node selection criteria. The principle of these criteria is presented in Figure 4.8. We show the impact of these criteria on the solution in Section 4.3.

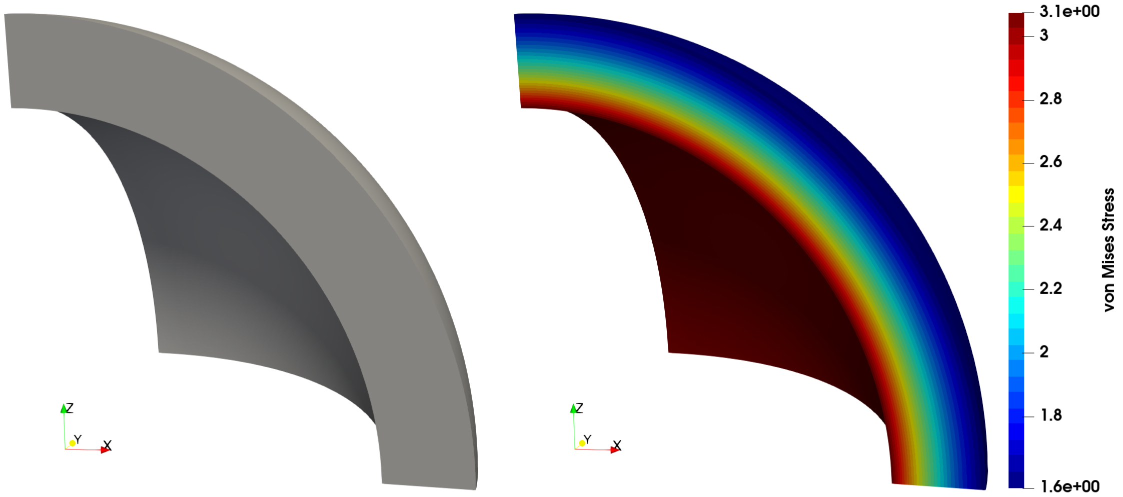
We show in Section 4.4 that the principle of the visibility criterion can be generalized to all concave problems and that it has a positive impact on the solution. We introduce in this section a unified algorithm for the selection of stencil nodes for all types of geometries (convex, concave, singular). We show results obtained with the proposed node selection algorithm and how the algorithm can be used in the context of model refinement.

4.2 Number of support nodes

The number of stencil nodes associated with a collocation node has to be of at least the number of approximated derivatives. In order to account for the node distribution, a larger number of nodes is often used. We analyze in this section the impact of the number of stencil nodes on the error in terms of L_2 and L_∞ error norms for two 2D benchmark problems: the 2D cylinder and the L-shape and for a 3D problem: the 3D sphere. The models and solutions of the 2D benchmark problems are presented in Section 1.4.1 and Section 1.4.2, respectively. We present in Figure 4.1 the model and the exact von Mises stress solution of the considered 3D benchmark problem. We analyze separately the impact of the number of stencil nodes for the GFD and the DC PSE methods.

4.2.1 GFD

In this subsection, we study the impact of the number of nodes in the support of a collocation node for the GFD method. We considered various combinations of inner node and boundary node support sizes. Results are presented in Figure 4.2, Figure 4.3 and Figure 4.4, respectively, for the 2D cylinder, the 2D L-shape and the sphere under internal pressure. The L_2 and the L_∞ norms are presented as a function of the number of support nodes for collocation nodes located on the



(a) (b)

Figure 4.1: 3D sphere under internal pressure. Uniform pressure is applied on the internal surface of the sphere. Symmetric boundary conditions are applied to the symmetry planes XY, XZ and YZ. Stress-free surface boundary conditions are applied to the external surface of the sphere. CAD model (a) and exact von Mises stress solution (b).

boundary. The results are presented for three sizes of inner node supports for the 2D problems and four sizes for the 3D problem.

The results in terms of L_2 and L_∞ errors present a similar trend for the σ_{11} and σ_{12} stress components. We can see that increasing the number of nodes in the inner nodes supports does not necessarily lead to an error reduction. The loss in terms of resolution is not compensated by the gain in solution smoothness. Increasing the number of support nodes for the boundary nodes steadily (and rapidly) reduces the error for the 2D cylinder problem. We observe an error reduction of a factor of approximately one hundred when the number of support nodes for boundary collocation nodes is increased from 13 to 18.

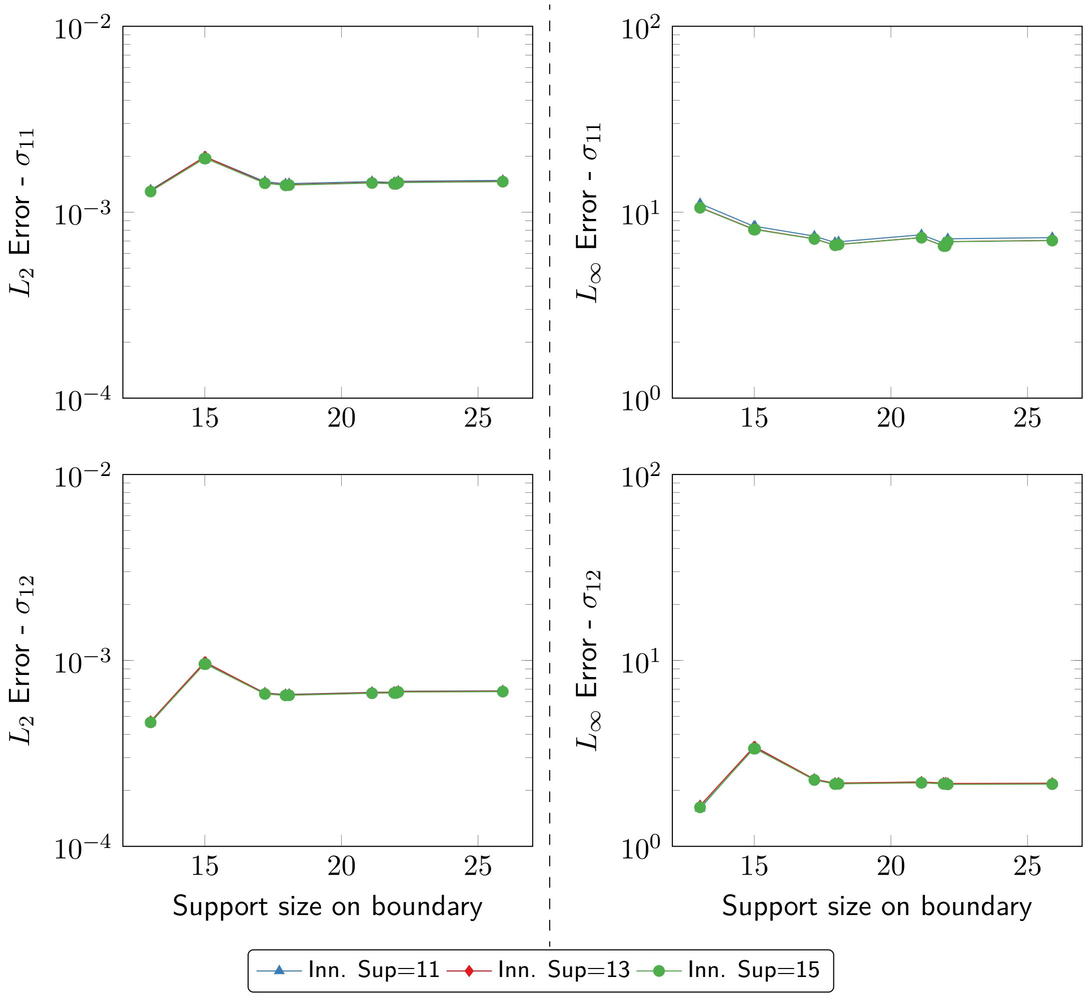


Figure 4.3: GFD Support Node Number Sensitivity - 2D L-Shape. L_2 (Left) and L_∞ (Right) errors for various combinations of inner nodes and boundary nodes support sizes. All the combinations of inner and boundary nodes support size lead to similar errors. This is due to the system being loaded via Dirichlet boundary conditions.

The number of support nodes on the boundary does not affect much the observed error for the 2D L-shape problem as the model is loaded via Dirichlet boundary conditions. The size of the inner nodes support also has little impact on the error for this problem.

Based on these results we select a combination of 11 support nodes for interior nodes and 19 support nodes for boundary nodes as it leads to a low error for both problems while maintaining the fill of the system matrix reasonably low.

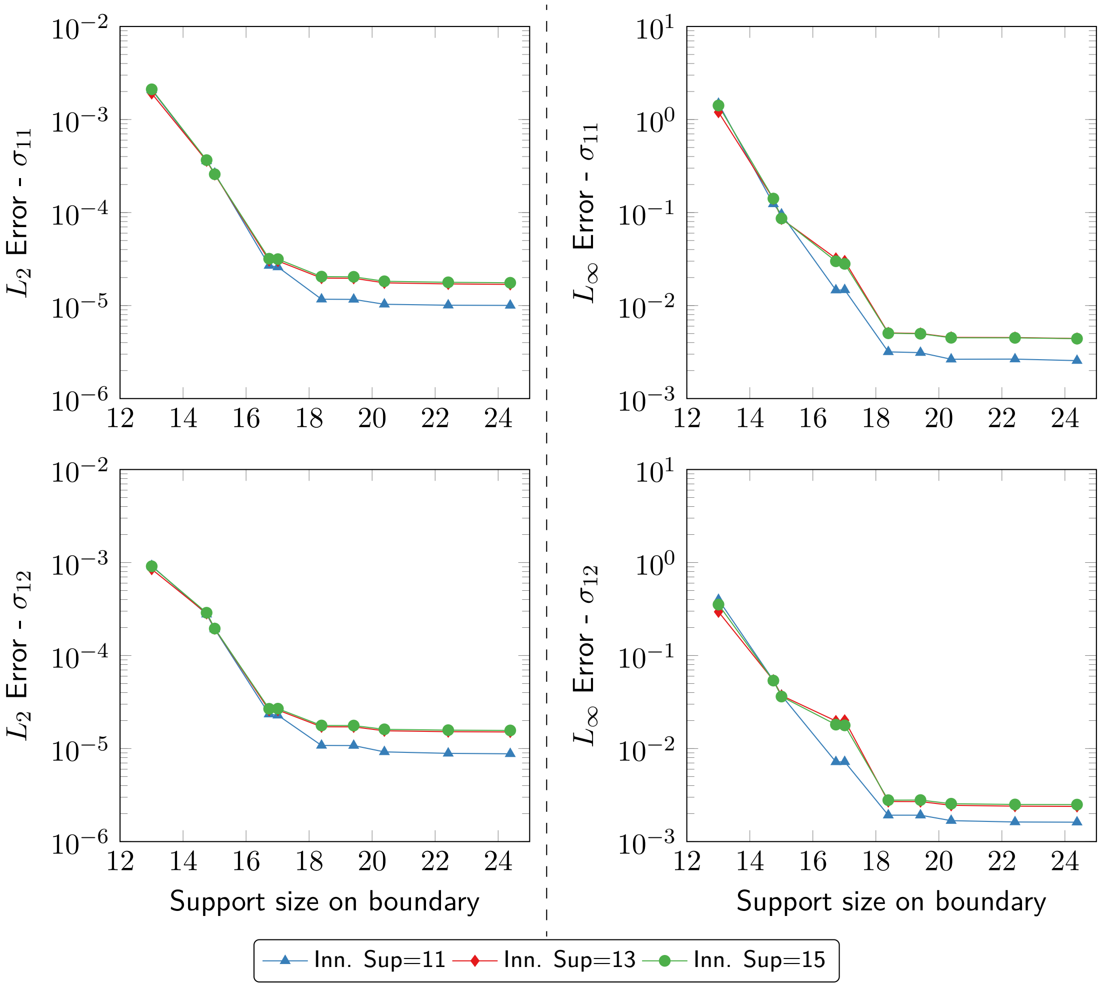


Figure 4.2: GFD Support Node Number Sensitivity - 2D Cylinder. L_2 (Left) and L_∞ (Right) errors for various combinations of inner nodes and boundary nodes support sizes. Inner collocation nodes with 11 support nodes lead to the lowest observed error. The error stops decreasing for boundary nodes supports larger than 18 nodes.

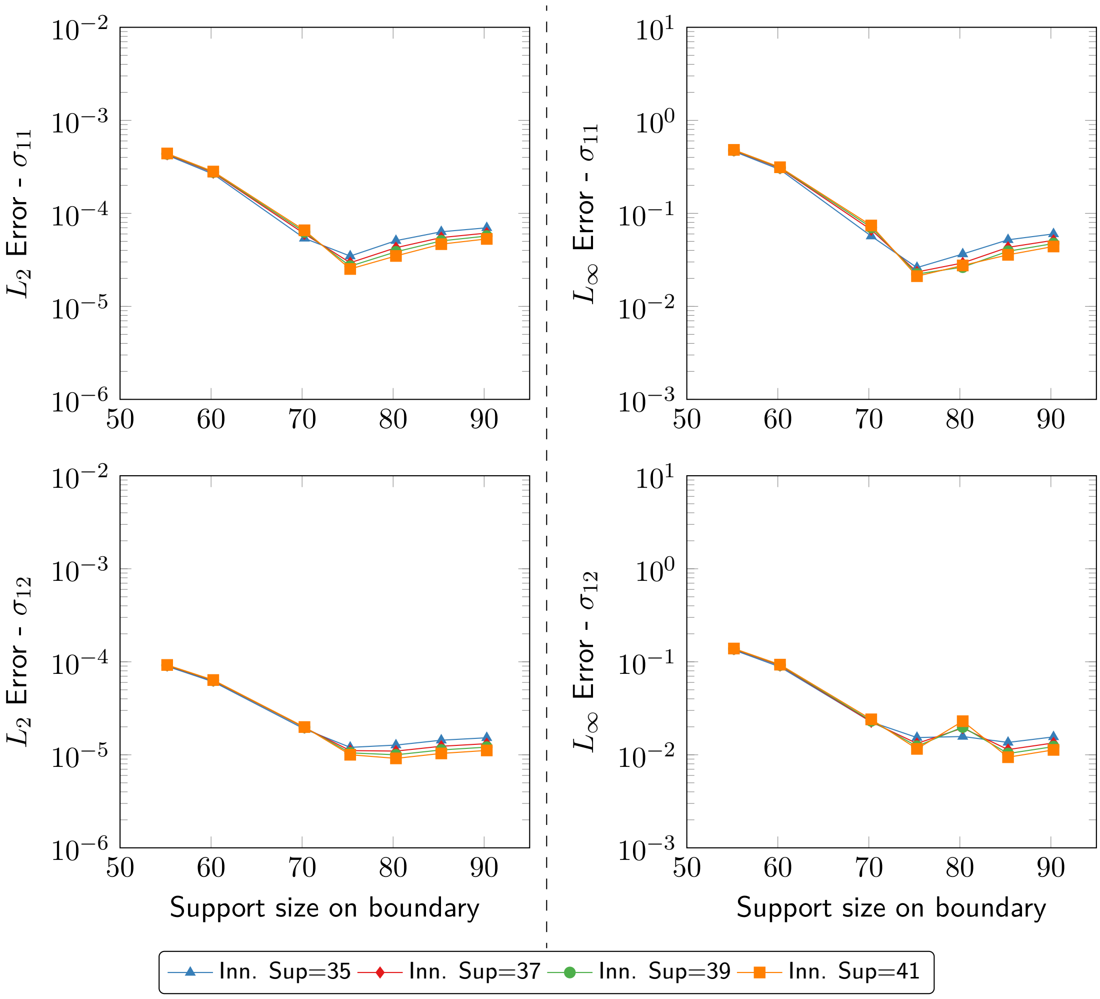


Figure 4.4: GFD Support Node Number Sensitivity - 3D Sphere. L_2 (Left) and L_∞ (Right) errors for various combinations of inner nodes and boundary nodes support sizes. Boundary collocation nodes with 75 support nodes lead to the lowest error. The size of the support of inner collocation nodes has little impact on the error.

We observed from Figure 4.4 that, for the 3D sphere, a minimum error is obtained for 75 support nodes for boundary collocation nodes. Increasing the size of the support from 55 to 75 for boundary nodes reduces in average by a factor 10 the observed error both in terms of L_2 and L_∞ norms. The number of support nodes for the collocation nodes located in the domain has a smaller impact on the error. 37 support nodes appears to be a reasonable choice as it leads to a low error while keeping the fill of the matrix reasonable.

The sparsity of the problem matrix is reduced when the number of support nodes increases. However, the impact of an increase in the number of support nodes for the boundary nodes is limited as it only affects a fraction of the nodes of the domain.

4.2.2 DC PSE

As for the GFD method, we present in this subsection the impact of the support size on the observed error for the DC PSE method. The results are presented in Figure 4.5, Figure 4.6 and Figure 4.7, respectively, for the 2D cylinder, the 2D L-shape and the sphere under internal pressure for various combinations of inner node and boundary node support sizes.

We observe from Figure 4.5 that an inner node support composed of 13 nodes leads almost for all the considered boundary support sizes to the minimum error. We also observe that an increasing number of nodes in the support of the boundary nodes reduces relatively steadily the error for the 2D cylinder. We observe an error reduction of a factor two when we increase the number of support nodes from 13 to 19 for most inner nodes support sizes.

We observe from Figure 4.6 that the number of support nodes on the boundary has little effect on the observed error for the L-shape problem. This is because the model is loaded via Dirichlet boundary conditions. The number of support nodes for inner collocation nodes has little impact on the error.

Based the results from Figure 4.5 and Figure 4.6, we select for the DC PSE method 13 support nodes for inner collocation nodes and 19 support nodes for boundary collocation nodes.

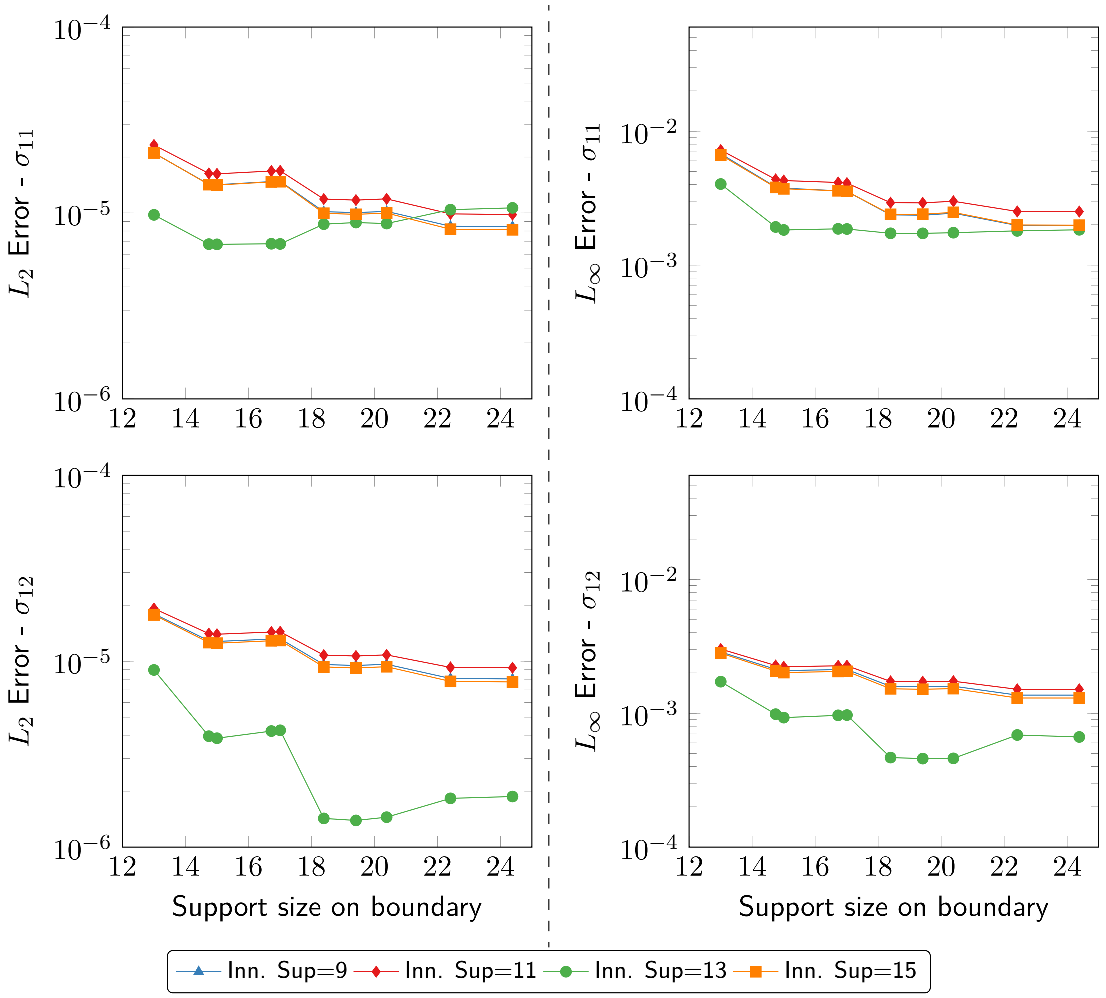


Figure 4.5: DC PSE Support Node Number Sensitivity - 2D Cylinder. L_2 (Left) and L_∞ (Right) errors for various combinations of inner nodes and boundary nodes support sizes. Inner collocation nodes with 13 support nodes lead relatively constantly to a low error. The error starts to increase for this case when the number of boundary nodes is larger than 19.

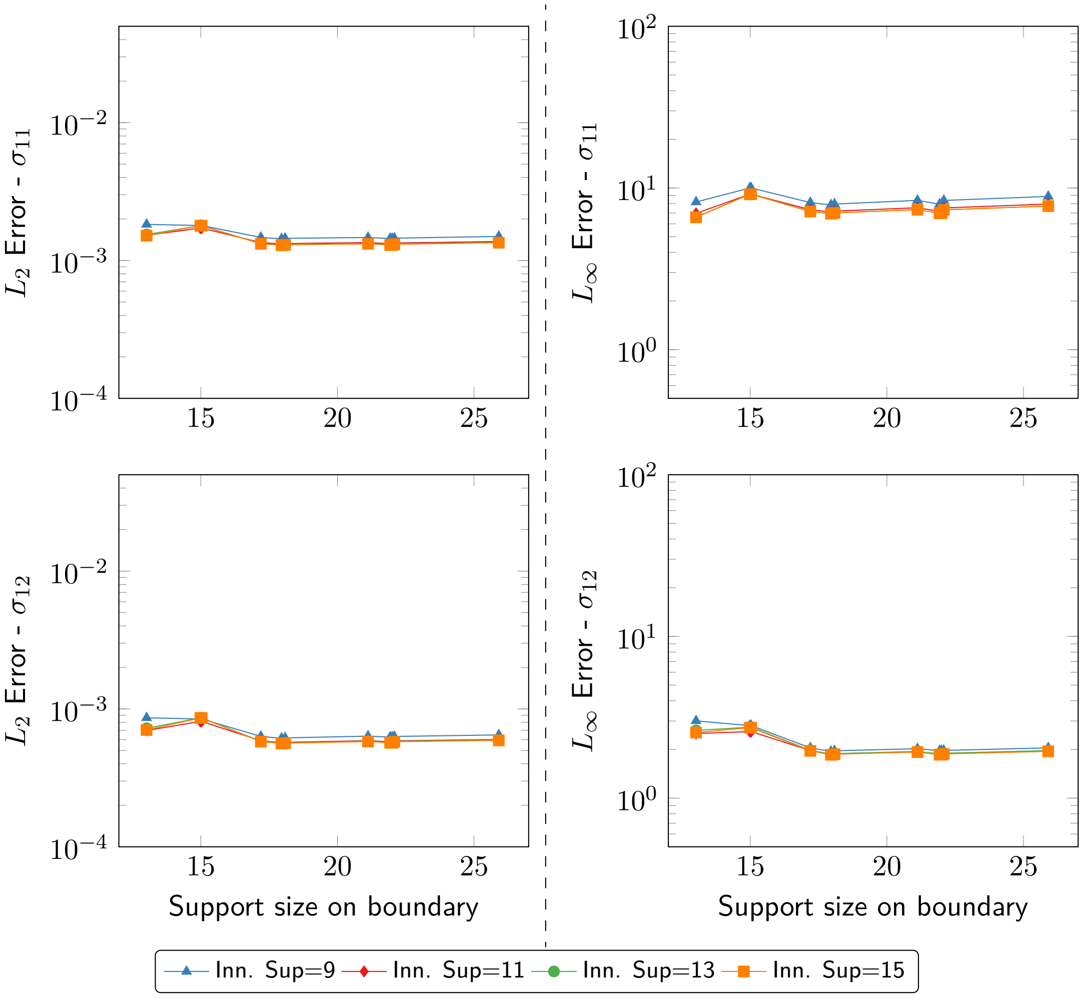


Figure 4.6: DC PSE Support Node Number Sensitivity - 2D L-Shape. L_2 (Left) and L_∞ (Right) errors for various combinations of inner nodes and boundary nodes support sizes. All the combinations of inner and boundary nodes support size lead to similar errors. This is due to the system being loaded via Dirichlet boundary conditions.

We observe from Figure 4.7 that, for the 3D sphere, the minimum error is reached for inner node supports composed of 37 nodes. The number of boundary support nodes has little impact on the error. Increasing the number of support nodes from 60 to 90 reduces by only 8% in average the observed error. Therefore, we select 75 support nodes for boundary collocation nodes, as for the GFD method, in order to keep the fill of the system matrix as low as possible while maintaining the error low.

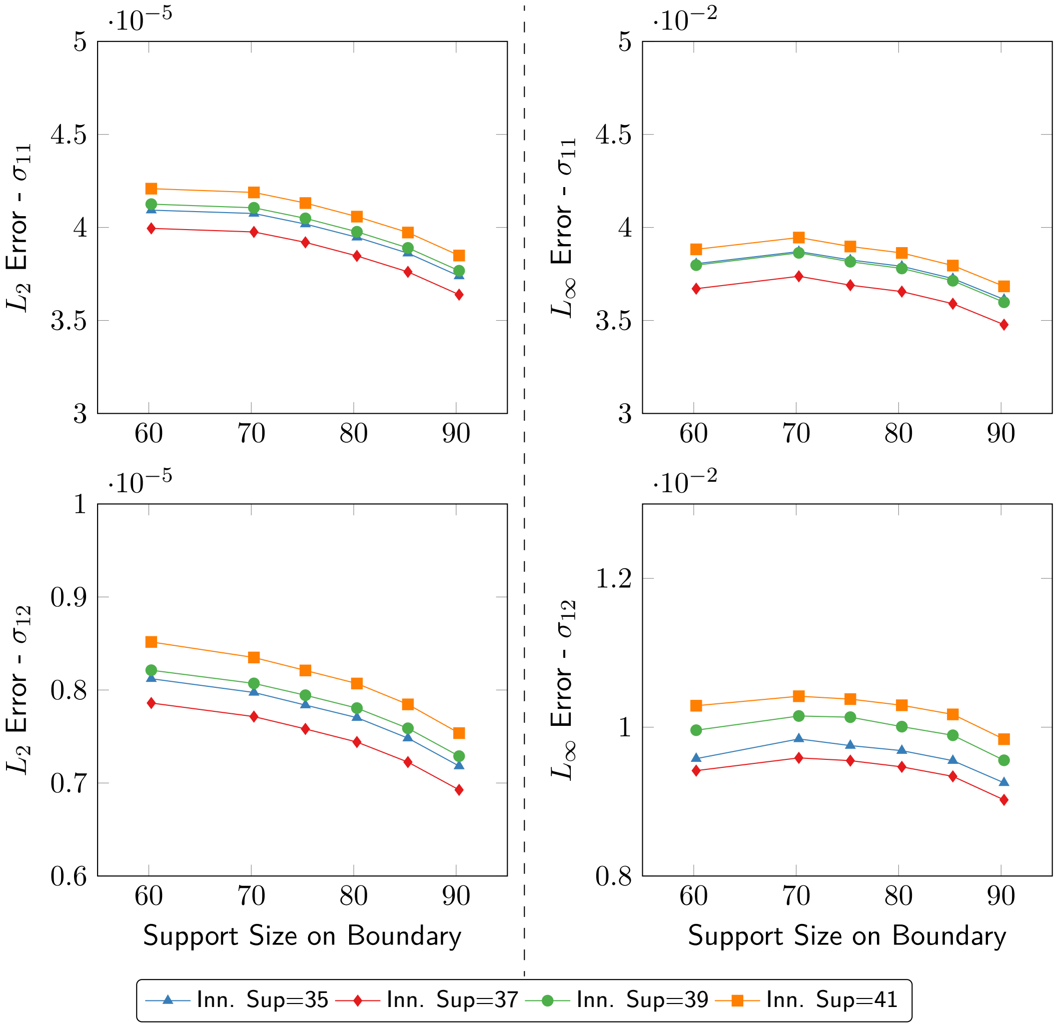


Figure 4.7: DC PSE Support Node Number Sensitivity - 3D Sphere. L_2 (Left) and L_∞ (Right) errors for various combinations of inner node and boundary node support sizes. Inner collocation nodes with 37 support nodes lead to the lowest observed error.

We use the results from this section as a basis for the simulations performed in this thesis. For some problems the stencil size is slightly increased to improve the solution. The ratio of boundary node support size to the inner node support size is maintained approximately the same for all the problems.

4.3 Visibility and diffraction criteria

4.3.1 Principle

The visibility criterion is a node selection method, introduced as part of the Element Free Galerkin (EFG) method [32] for singular problems. It has been proven effective for both meshless methods [32, 58, 59] and collocation methods [60, 61, 62]. The concept of the visibility criterion is presented in Figure 4.8 (a). Considering a domain Ω, only the support nodes $\mathbf{X_p}$ which are "visible" from the collocation nodes $\mathbf{X_c}$ are considered in the stencil. In other words, the segments connecting $\mathbf{X_c}$ to the support nodes $\mathbf{X_p}$ shall not intersect the boundary of the domain. This can be understood by the fact that, in most collocation methods, the field derivatives at a collocation node are approximated as a function of the field values at the support nodes. For this, the solution needs to be continuous and as smooth as possible on the collocation stencil. In theory, only the nodes "visible" from the collocation nodes satisfy these criteria. We show in this chapter that an acceptance intersection angle can be beneficial for concave problems.

The diffraction criterion and the transparency criterion are other criteria, introduced in 1996 by Organ [63], to select stencil nodes in the vicinity of a singularity. Both methods ensure a continuity of the weight functions at the singularity. In the context of fracture modelling based on the Element Free Galerkin Method, it was showed that the visibility criterion was leading to spurious crack extensions unlike the diffraction or transparency criteria [64]. The concept of the diffraction criterion is presented in Figure 4.8 (b). The nodes $\mathbf{X_p}$ in the "hidden" zone are included in the stencil of a node $\mathbf{X_c}$ only if the distance between the collocation node and the singularity $\mathbf{X_s}$ plus the distance between the singularity and the candidate stencil node is lower than the selected stencil radius R_c i.e. $\|\mathbf{X_s} - \mathbf{X_c}\|^2 + \|\mathbf{X_p} - \mathbf{X_s}\|^2 < R_c$. The computation of the weight function is based on this extended distance. The transparency criterion is based on the same idea. The distance between a node $\mathbf{X_c}$ and a candidate stencil node $\mathbf{X_p}$ located in the "hidden" zone is increased by a function which depends on the distance between the singularity $\mathbf{X_s}$ and the point of intersection between the segment connecting the considered node to the candidate stencil node and a boundary of the domain.

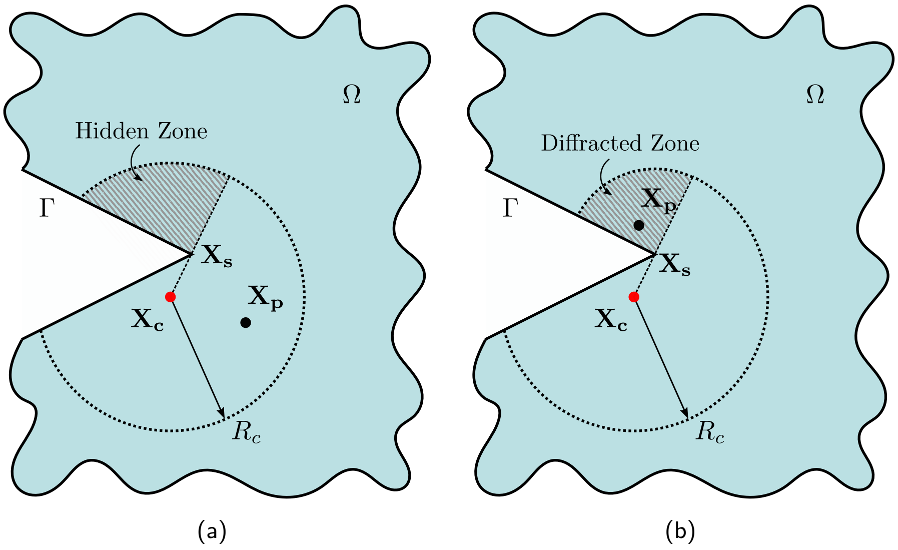


Figure 4.8: Concept of the visibility criterion (a) and diffraction criterion (b).

We compare in Subsection 4.3.2 the results obtained from a stencil selection based on the visibility criterion and based on the diffraction criterion, in the vicinity of a singularity, for the problem of the 2D L-shape .

4.3.2 Results for a singular problem

In this subsection, we assess the impact of the support node selection criterion on the L_2 error for the L-shape problem. We compare results obtained with the visibility criterion and with the diffraction criterion to results where no criterion is considered (i.e. every node within the support radius of a collocation node is included in the support). We consider both the GFD and DC PSE methods and present results for each method in Figures 4.9 and 4.10, respectively.

We observe that, for both methods, the use of the visibility criterion leads to a significant error reduction compared to results where no criterion is applied. For the GFD method, the error reduction ranges from 50% to 20% depending on the node density. For the DC PSE method, the error reduction ranges from 55% to 25%. These results are expected as the singularity of the domain is more accurately captured with the visibility criterion. The diffraction criterion leads to an error increase ranging from a factor 2 to factor 10 for the GFD and the DC PSE methods. For

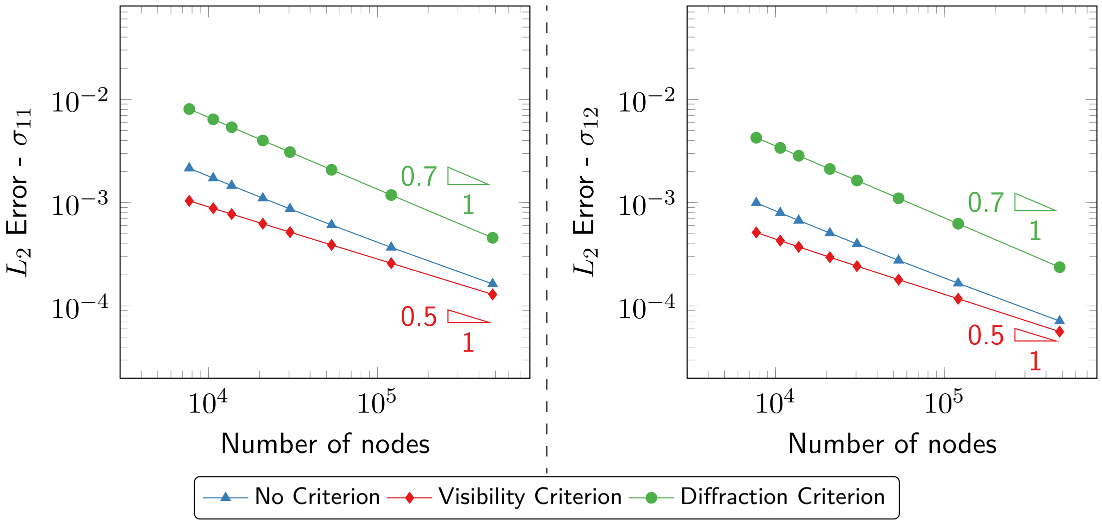


Figure 4.9: Support node selection results comparison - 2D L-Shape - GFD Method. L_2 error obtained with no node selection criterion, with the visibility criterion, and the diffraction criterion. The lowest error is observed with the visibility criterion.

both methods, the convergence rate is larger with the diffraction criterion. The visibility criterion appears to be a sensible choice for singular problems.

4.4 Unified node selection algorithm - Generalized visibility criterion

We saw the benefits of the visibility criterion for singular problems solved using a point collocation method in Section 4.3. We introduce in this section a unified algorithm that allows the selection of stencil nodes, for all types of problems (convex, concave or singular). The algorithm is based on the distance criterion and on generalization of the visibility criterion.

The node selection algorithm is split into multiple sub-algorithms: the general node selection algorithm is presented in Subsection 4.4.1 and the specific details of the algorithm for concave domains in Subsection 4.4.2. We present the algorithms in the form of flow charts and pseudo codes.

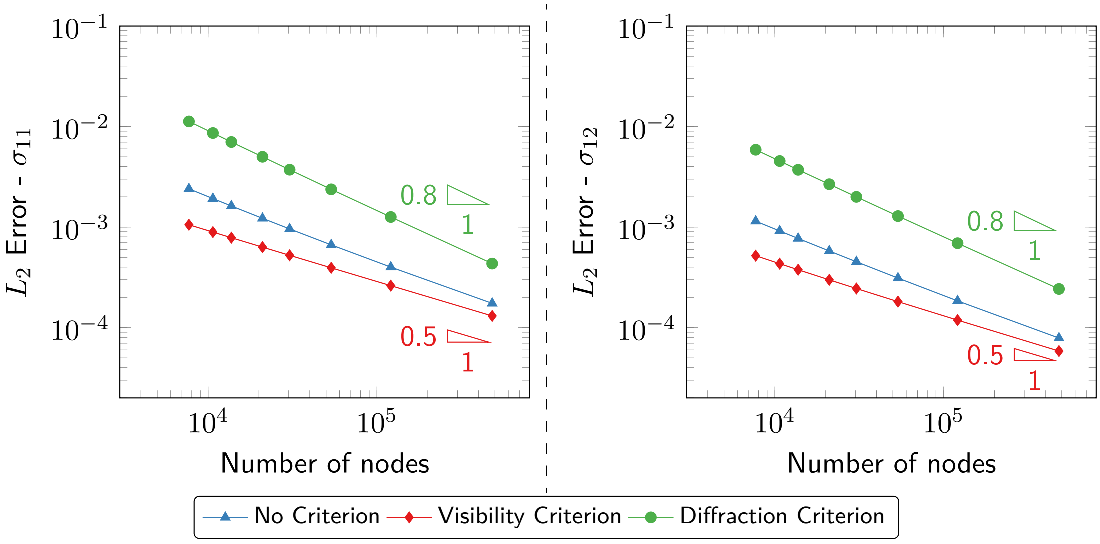


Figure 4.10: Support Node Selection Results Comparison - 2D L-Shape - DC PSE Method. L_2 error obtained with no node selection criterion, with the visibility criterion, and the diffraction criterion. The lowest error is observed with the visibility criterion.

4.4.1 Support node selection algorithm

We consider a domain Ω, limited by the boundary Γ for the purpose of the node selection algorithm as presented in Figure 4.11. The boundary of the domain is split into a concave boundary Γ_{cc} and a convex boundary Γ_{cv}. The nodes $\mathbf{X_p}$ to be included in the support of each collocation node $\mathbf{X_c}$ can be identified using the algorithm presented in Figure 4.12.

We consider the distance criterion in this thesis but the algorithm can also be used to determine the candidate nodes for other node selection techniques.

4.4.2 Algorithm details

We present in this section details of the node selection algorithm.

The number of support nodes N_c to be considered in the stencil of a collocation node $\mathbf{X_c}$ is selected based on the position of the collocation node in the domain (e.g. on the boundary of the domain or inside of the domain, based on the results presented in Section 4.2), and based on the regularity of the node placement. All the N_c time k nodes in the vicinity of $\mathbf{X_c}$, which

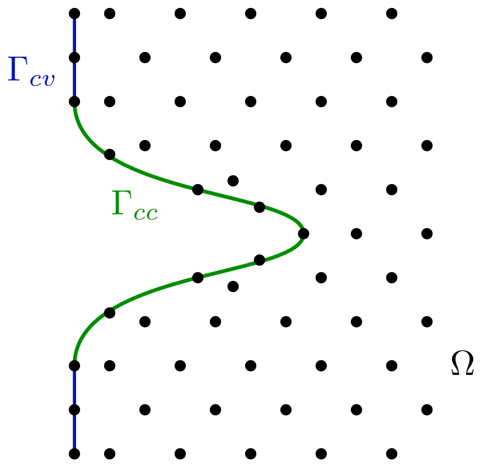


Figure 4.11: Concave domain Ω, defined by the boundary Γ_{cv} and Γ_{cc}.

could be included in the support, are considered as candidate nodes for the stencil. k is larger than 1.0 as some nodes might not be included in the support of the collocation node if they are not "visible" from $\mathbf{X_c}$ or if some nodes are located at the same distance from the collocation node. κ is a factor used to scale the support radius once the target number of support nodes has been identified. This allows the selection of all the nodes located at a distance R_c from $\mathbf{X_c}$. The algorithm allowing the selection of the support nodes near a concave boundary of the domain is presented in Figure 4.13. We provide more details about the steps 1 to 4 in the paragraphs below.

Step 1 - Details

In order to assess if a node is "visible" from a collocation node, the concave boundaries of the domain need to be known. Even if the collocation methods do not make use of elements to solve the partial differential equation problem, the use of boundary elements, in the concave regions of the domain, is a simple and efficient method to enforce the visibility criterion. Figure 4.14 shows a concave domain Ω. The concave boundary of the domain, Γ_{cc}, is discretized using boundary elements. The outer normals to the boundary elements are calculated. These normals allow positioning the inside of the domain relatively to the boundary and are used to assess if the segment connecting a collocation node to a support node interests the boundary. In case of an

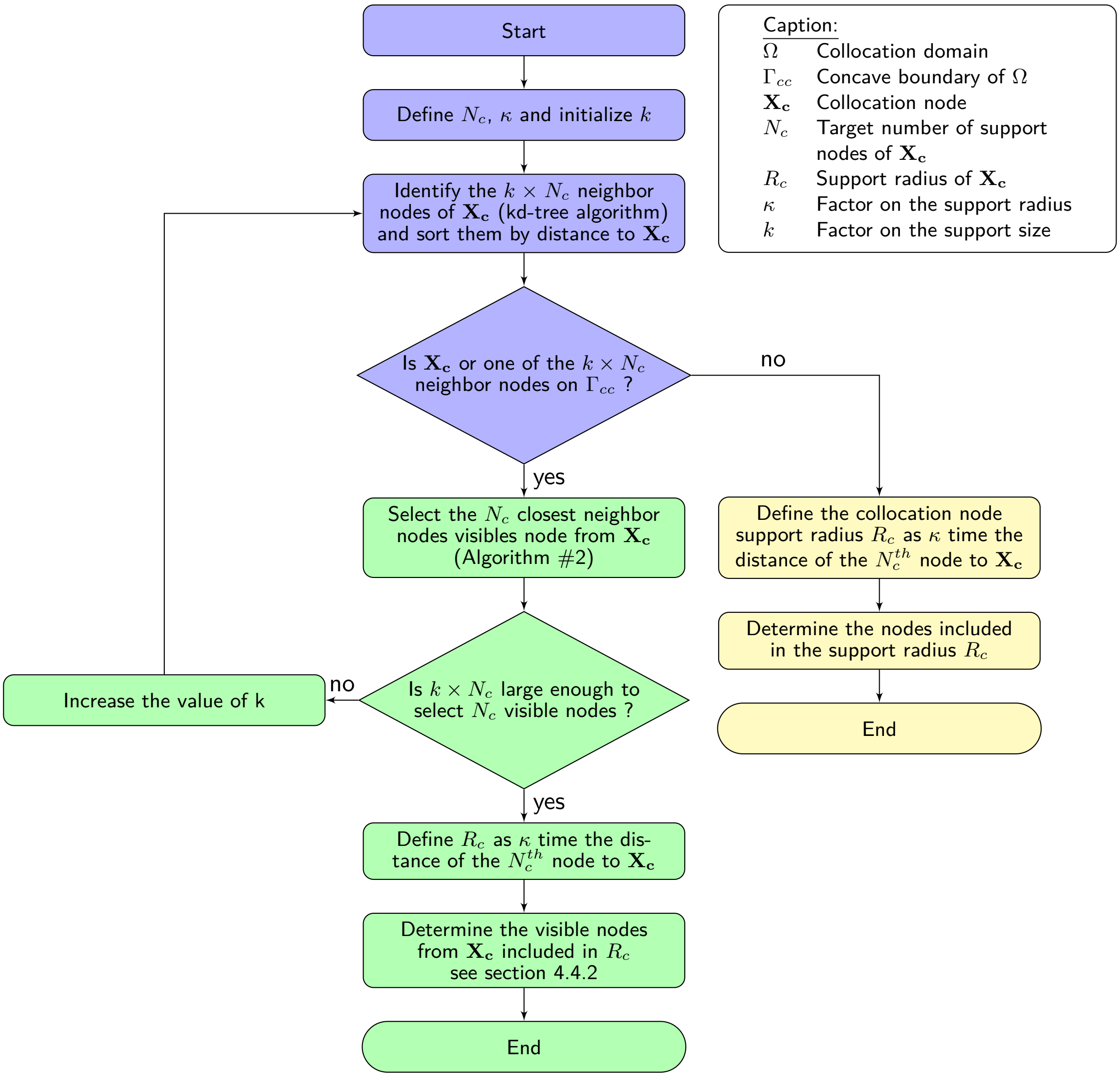

Start
Define N_c, κ and initialize k
Identify the $k \times N_c$ neighbor nodes of $\mathbf{X_c}$ (kd-tree algorithm) and sort them by distance to $\mathbf{X_c}$
Is $\mathbf{X_c}$ or one of the $k \times N_c$ neighbor nodes on Γ_{cc} ?
no
yes
Select the N_c closest neighbor nodes visibles node from $\mathbf{X_c}$ (Algorithm #2)
Is $k \times N_c$ large enough to select N_c visible nodes ?
no
Increase the value of k
yes
Define R_c as κ time the distance of the N_c^{th} node to $\mathbf{X_c}$
Determine the visible nodes from $\mathbf{X_c}$ included in R_c see section 4.4.2
End
Define the collocation node support radius R_c as κ time the distance of the N_c^{th} node to $\mathbf{X_c}$
Determine the nodes included in the support radius R_c
End
Caption:
Ω Collocation domain
Γ_{cc} Concave boundary of Ω
$\mathbf{X_c}$ Collocation node
N_c Target number of support nodes of $\mathbf{X_c}$
R_c Support radius of $\mathbf{X_c}$
κ Factor on the support radius
k Factor on the support size

Figure 4.12: Algorithm #1 flowchart for collocation support node selection. The central branch of the flowchart, in green, is the path followed for the collocations nodes near a concave boundary of the domain, if any. The right branch is the path for the nodes far from a concave boundary of the domain.

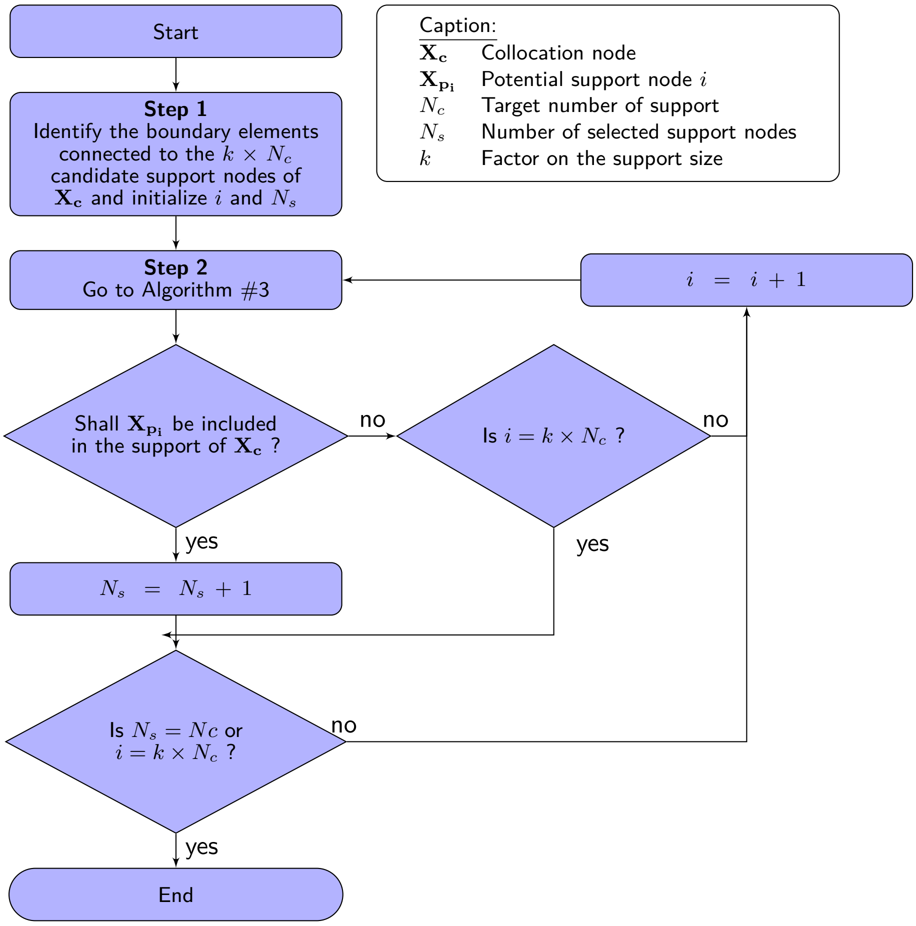

Start
Step 1
Identify the boundary elements connected to the $k \times N_c$ candidate support nodes of $\mathbf{X_c}$ and initialize i and N_s
Caption:
$\mathbf{X_c}$ Collocation node
$\mathbf{X_{p_i}}$ Potential support node i
N_c Target number of support
N_s Number of selected support nodes
k Factor on the support size
Step 2
Go to Algorithm #3
$i = i + 1$
Shall $\mathbf{X_{p_i}}$ be included in the support of $\mathbf{X_c}$?
no
Is $i = k \times N_c$?
no
yes
yes
$N_s = N_s + 1$
Is $N_s = Nc$ or $i = k \times N_c$?
no
yes
End

Figure 4.13: Algorithm #2 flowchart for support node selection near a concave boundary of the domain.

intersection, the angle between the segment and the crossed element can be calculated.

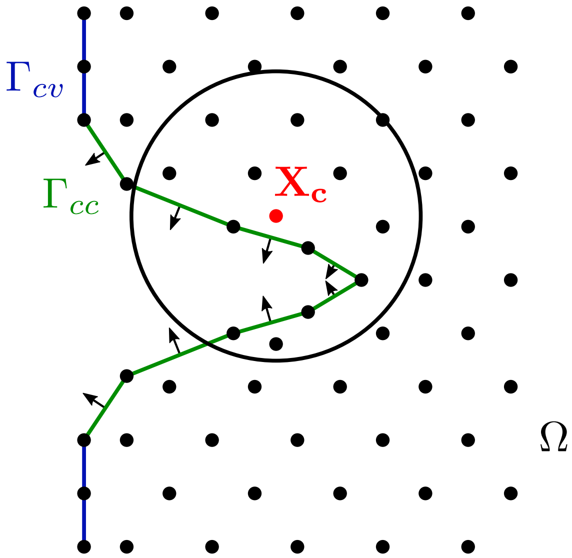


Figure 4.14: The concave boundaries of the domain are discretized by elements and the outer normal of each element is calculated.

Step 2 - Details

In this step, the inclusion of each candidate support node is assessed using the visibility criterion based on the Algorithm #3 presented in Figure 4.15.

Step 3 - Details

Step 3 involves three sub-steps which are described below.

a) Compute the parameters of the intersection between the segment connecting $\mathbf{X_c}$ to $\mathbf{X_{p_i}}$ and the considered boundary element. The coordinates of the intersection point can be calculated as a function of $\mathbf{X_c}$, $\mathbf{X_{p_i}}$ and of the element corner nodes ($\mathbf{X_1}$, $\mathbf{X_2}$ and $\mathbf{X_3}$ in Figure 4.16). For instance, for the 2D and 3D cases presented in Figure 4.16, the coordinates of the intersection points are calculated by solving the systems presented in Equation (4.1) and Equation (4.2) for the 2D and 3D cases, respectively.

```
Algorithm #3
For each concave boundary element identified in Step 1
  Go to Step 3.
  Case: 2D problem
    if ($0 < \alpha < 1$ and $0 < \beta < 1$ and abs($\mathbf{n_j} \cdot \mathbf{X_{cp_i}}$)$> \epsilon$)
      Do not include $\mathbf{X_{p_i}}$ in the support of $\mathbf{X_c}$. Exit from the loop.
    else if (($\alpha = 0$ or $\alpha = 1$) and $0 \leq \beta \leq 1$)
      Go to Step 4.
    else if (($\beta = 0$ or $\beta = 1$) and $0 \leq \alpha \leq 1$)
      Go to Step 4.
  Case: 3D problem
    if ($0 < \alpha < 1$ and $0 < \beta < 1$ and $0 < \gamma < 1$ and $\beta + \gamma < 1$ and abs($\mathbf{n_j} \cdot \mathbf{X_{cp_i}}$)$> \epsilon$)
      Do not include $\mathbf{X_{p_i}}$ in the support of $\mathbf{X_c}$. Exit from the loop.
    else if (($\alpha = 0$ or $\alpha = 1$) and $0 \leq \beta \leq 1$ and $0 \leq \gamma \leq 1$ and $\beta + \gamma \leq 1$)
      Go to Step 4.
    else if (($\beta = 0$ or $\beta = 1$) and $0 \leq \alpha \leq 1$ and $0 \leq \gamma \leq 1$ and $\beta + \gamma \leq 1$)
      Go to Step 4.
    else if (($\gamma = 0$ or $\gamma = 1$) and $0 \leq \alpha \leq 1$ and $0 \leq \beta \leq 1$ and $\beta + \gamma \leq 1$)
      Go to Step 4.
end
```

Figure 4.15: Algorithm #3 for the cases of 2D and 3D problems.

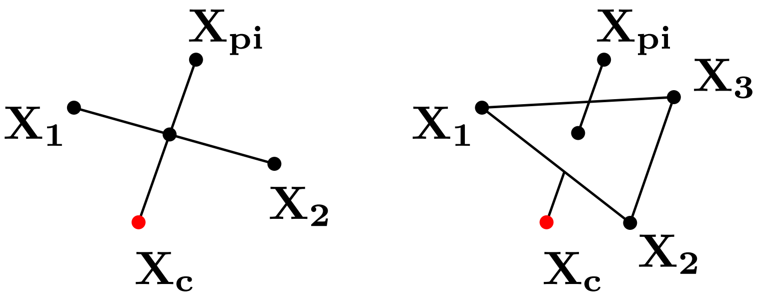


Figure 4.16: Intersection of a segment connection a collocation node $\mathbf{X_c}$ and a neighbor node $\mathbf{X_{p_i}}$ with a boundary element for the 2D (left) and the 3D (right) cases.

$$\begin{bmatrix} x_c - x_{p_i} & x_2 - x_1 \\ y_c - y_{p_i} & y_2 - y_1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} x_c - x_1 \\ y_c - y_1 \end{bmatrix} \tag{4.1}$$

$$\begin{bmatrix} x_c - x_{p_i} & x_2 - x_1 & x_3 - x_1 \\ y_c - y_{p_i} & y_2 - y_1 & y_3 - y_1 \\ z_c - z_{p_i} & z_2 - z_1 & z_3 - z_1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} x_c - x_1 \\ y_c - y_1 \\ z_c - z_1 \end{bmatrix} \tag{4.2}$$

b) Compute the scalar product of the vector $\mathbf{X_{cp_i}} = \frac{\mathbf{X_{p_i}} - \mathbf{X_c}}{\left\|\mathbf{X_{p_i}} - \mathbf{X_c}\right\|^2}$ with the normal $\mathbf{n_j}$ of the identified boundary elements.

c) Define a threshold ϵ, based on an acceptable intersection angle between the boundary element and the segment connecting the collocation node to the support node. ϵ is expressed as a function of a threshold angle θ corresponding to the angle between the element and the vector $\mathbf{X_{cp_i}}$. We have $\epsilon = \cos\left(\frac{\pi}{2} - \theta\right)$.

Step 4 - Details

Identify the elements connected to the point of intersection between the segment connecting $\mathbf{X_c}$ to $\mathbf{X_{p_i}}$ and the considered element. Use the Algorithm #4 presented in Figure 4.17 to assess if the node $\mathbf{X_{p_i}}$ shall be included in the support of $\mathbf{X_c}$.

```
Algorithm #4
Initialize Count1, Count2 and n to zero.
**For each** boundary element $j$ connected to the point of intersection
  n = n + 1
  **if** ($\alpha < 0.5$ and $\mathbf{n_j} \cdot \mathbf{X_{cp_i}} > \epsilon$)
    Count1 = Count1 + 1
  **else if** ($\alpha \geq 0.5$ and $\mathbf{n_j} \cdot \mathbf{X_{cp_i}} < -\epsilon$)
    Count2 = Count2 + 1
**end**
**if** (Count1 = n or Count2 = n)
  Do not include $\mathbf{X_{p_i}}$ in the support of $\mathbf{X_c}$.
```

Figure 4.17: Algorithm #4 for the cases of 2D and 3D problems.

4.4.3 Visibility threshold sensitivity

The node selection algorithm introduced in Subsections 4.4.1 and 4.4.2 uses a visibility threshold angle parameter to determine if a node, located in the vicinity of a concave or singular region of a domain, should be included in the stencil of a collocation node. We analyze in this subsection the impact of the visibility threshold angles (denoted θ) on the solution. We compare results obtained from collocation models solved using threshold angles ranging from 0.0° to 40° to results obtained without consideration of the generalized visibility criterion (distance criterion only).

We selected three 2D concave problems for this purpose: one with a concavity of low curvature, another one with a concavity of high curvature and a singular problem. The problems considered are:

- a cylinder under internal pressure;
- an infinite plate with an elliptical hole under biaxial loading;
- a L-shape domain in mode I loading.

The models and the associated exact solutions are presented in Section 1.4.

The results for the 2D cylinder, for the infinite plate with an elliptical hole and for the L-shape domain are respectively presented in Figure 4.18, Figure 4.19 and in Figure 4.20.

We see from the results presented in Figure 4.18, for the 2D cylinder, that the visibility criterion leads to an increase of the observed error in terms of L_2 relative norm when a threshold angle of 0.0° is used. With a threshold angle of 1.0° and 5.0°, the observed error is very close to the reference case where the visibility criterion is not considered.

When a threshold angle of 0.0° is selected, the nodes on the concave boundary of the domain are excluded from the stencils of the collocation nodes also located on a concave boundary of the domain. The results show that, for a model having a concavity of low curvature such as the 2D cylinder, the inclusion of these nodes in the stencils (when a threshold angle of 1.0° or 5.0° is selected) leads to a lower error than the exclusion of these nodes.

The results presented in Figure 4.19, for the infinite plate with an elliptical hole, show that the use of the visibility criterion almost always leads to a reduction of the observed error compared to the reference case where the visibility criterion is not considered. With a threshold angle of 0.0°, the error reduction is not constant. For most node discretizations, the results obtained with threshold angles of 1.0° and 5.0° lead to the smallest error. An error reduction is observed for the relatively coarse discretizations when a threshold angle of 40° is selected. No error reduction is observed for the finer discretizations for such a threshold angle.

The exclusion of some nodes from the stencil of collocation nodes located near concave boundaries

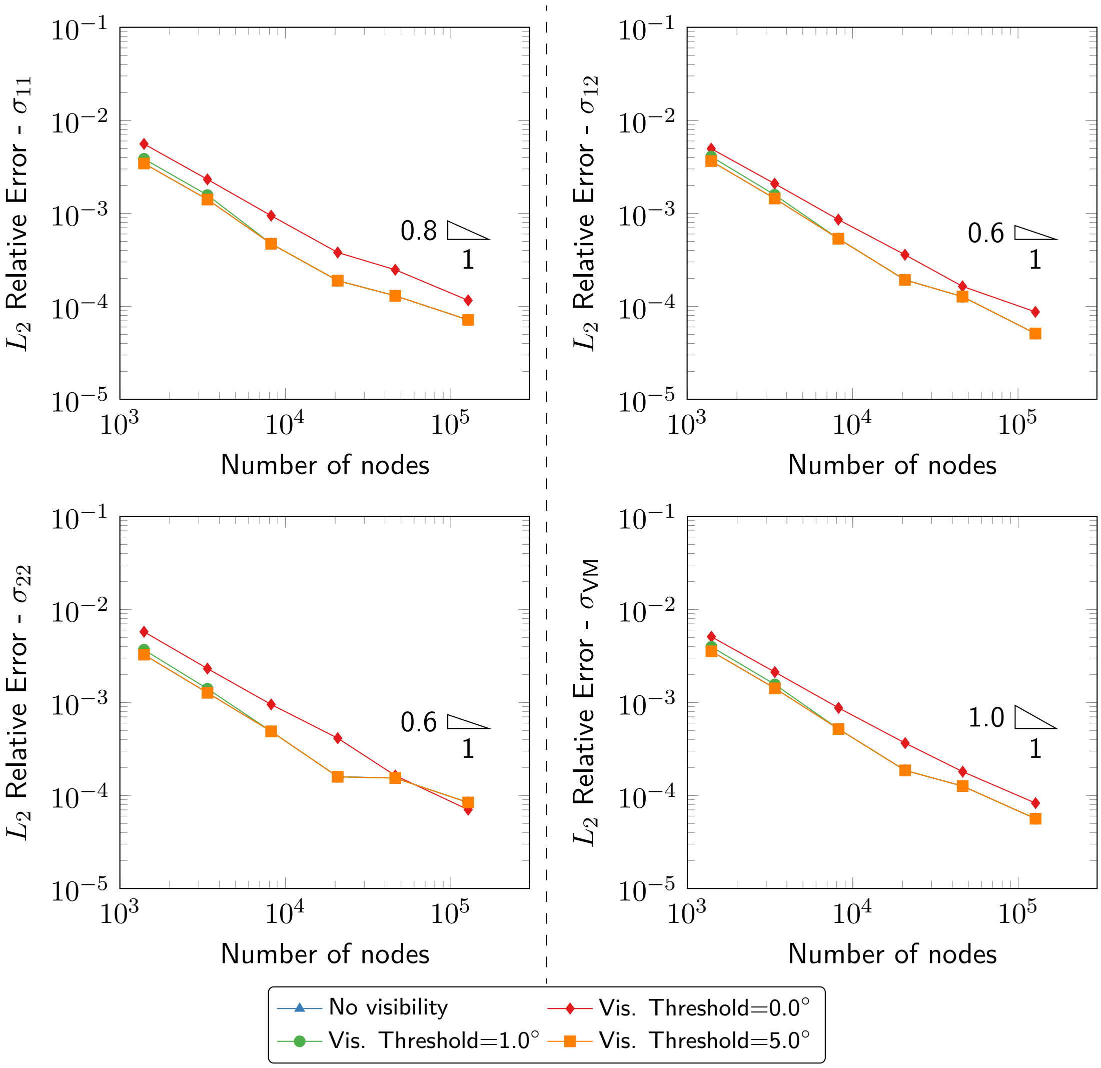


Figure 4.18: 2D cylinder - Comparison of the error between a case without consideration of the visibility criterion and cases with consideration of the visibility criterion for different threshold angles. The error in terms of L_2 relative norm is presented for the σ_{11}, σ_{12} and σ_{22} stress components and for the von Mises stress noted σ_{VM}.

of the domain allows capturing more precisely the geometry of the domain, especially in the areas having a concavity of high curvature. The fact that a threshold angle of 0.0° leads to a higher error than a threshold angle of 1.0° or 5.0° shows that the inclusion of some boundary nodes in the areas having a concavity of low curvature allows a reduction of the error, as for the 2D cylinder problem. As expected, the selection of a large threshold angle of 40° leads to a higher error than with a smaller threshold angle. The concave boundary of the domain is less accurately captured since some segments connecting collocation nodes to support nodes intersect the boundary of the domain. The observed error remains below the error for the model where the visibility criterion is not considered.

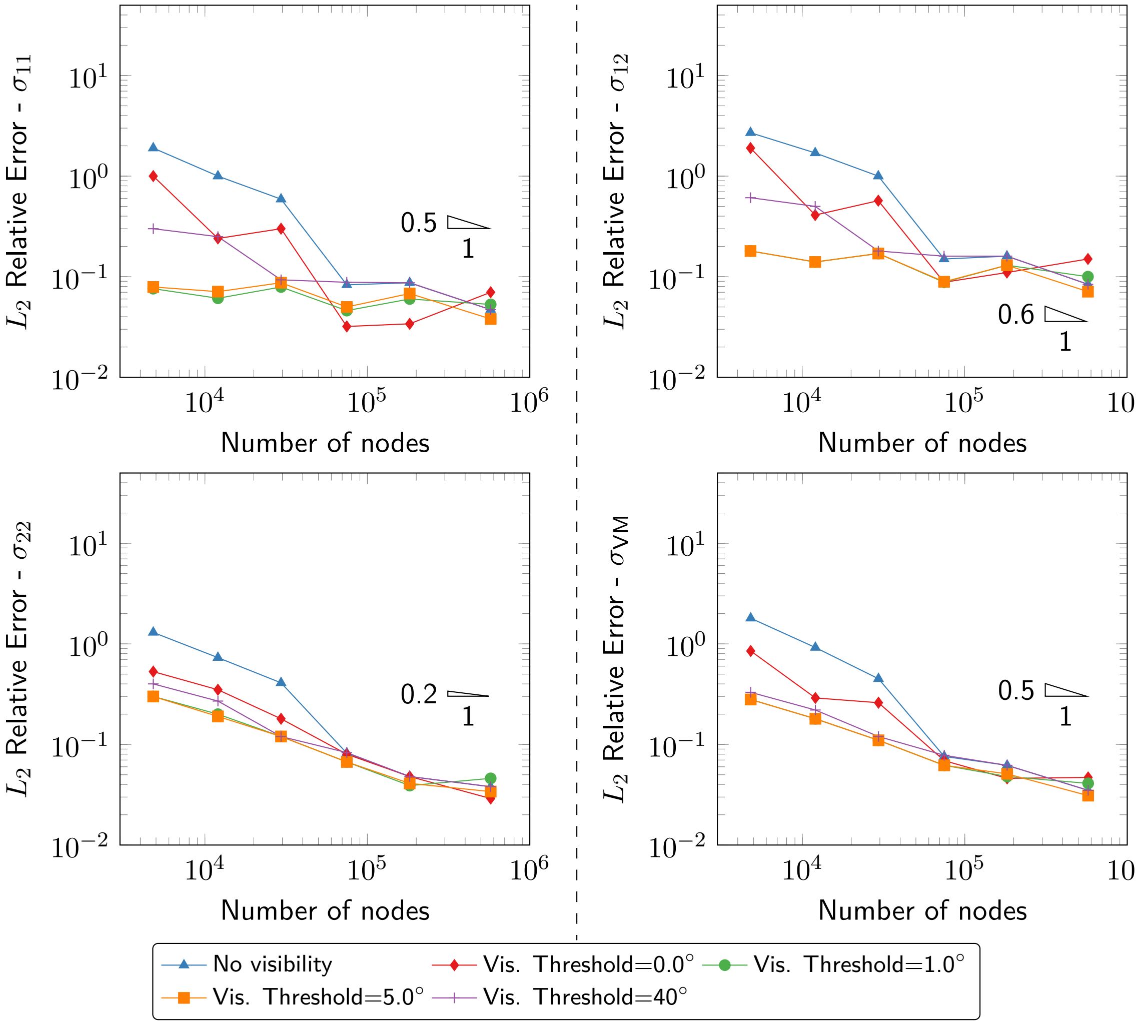


Figure 4.19: Infinite plate with an elliptical hole - Comparison of the error with and without consideration of the visibility criterion. The error in terms of L_2 relative norm is presented for the σ_{11}, σ_{12} and σ_{22} stress components and for the von Mises stress noted σ_{VM}.

The results presented in Figure 4.20, for the L-shape problem, show that the use of the visibility criterion leads to a significant error reduction compared to the reference case where the visibility criterion is not considered. For this problem, the selected threshold angle does not impact the results. This result is expected since a threshold angle in the order of 0-5° has little impact on the selection of the support nodes in the singular region since the angle of the boundary at the singularity is of 90°.

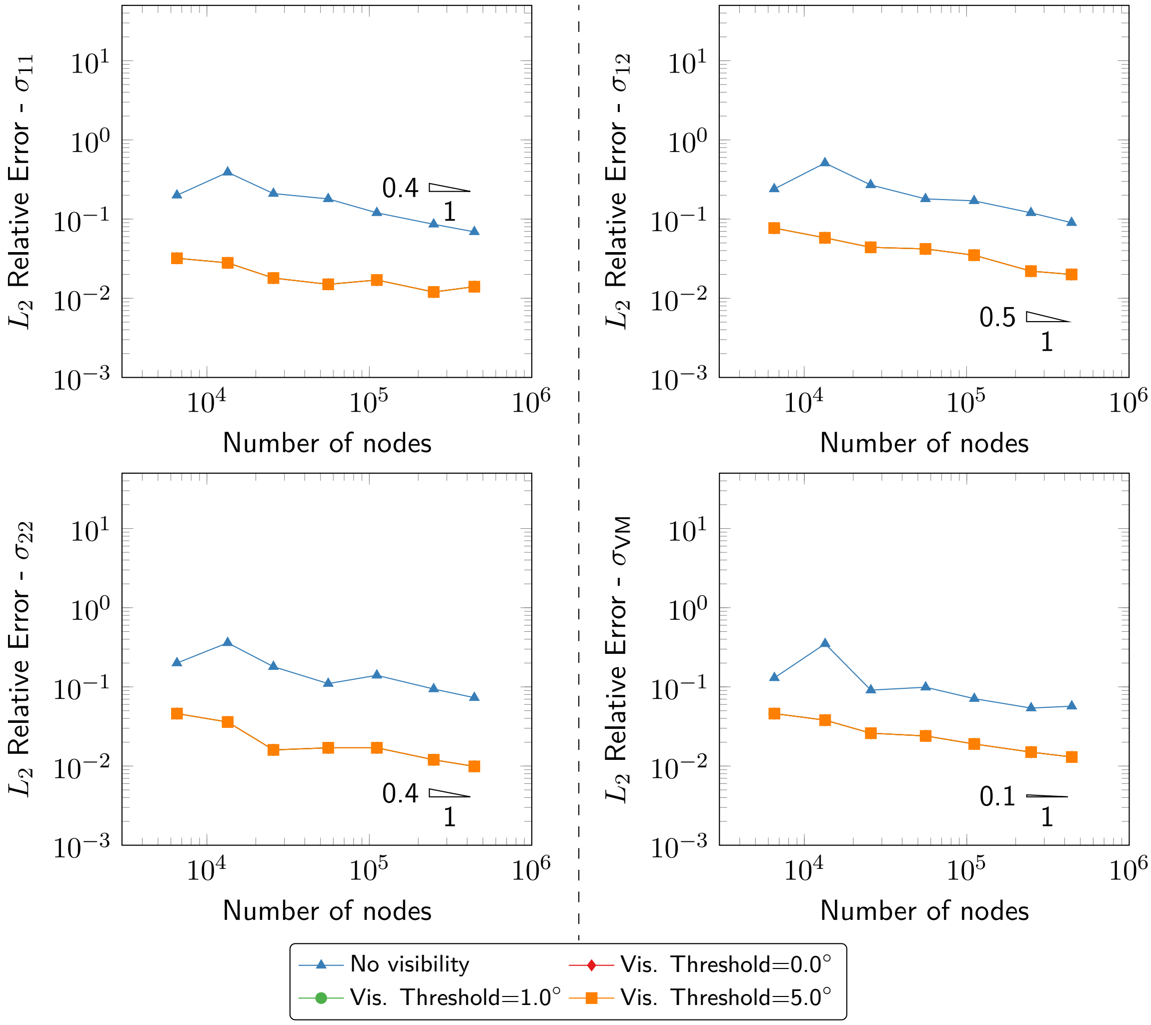


Figure 4.20: L-shape - Comparison of the error with and without consideration of the visibility criterion. The error in terms of L_2 relative norm is presented for the σ_{11}, σ_{12} and σ_{22} stress components and for the von Mises stress noted σ_{VM}. The observed error is the same for the visibility thresholds considered.

The results presented in Figure 4.18 differ from the results presented in Figure 4.19 and 4.20 since, for the 2D cylinder, the consideration of the visibility criterion with a threshold angle of

0.0° leads to an increase of the error. This can be explained by the smoothness of the solution for this problem, even in the concave region. We observe that, for this problem, the inclusion of adjacent boundary nodes in the stencil of collocation nodes located on the concave boundary improves the approximation of the PDE at this node.

We showed in this section that the visibility criterion can be applied to concave problems with high and low curvatures and to singular problems. The selection of a threshold angle is important to ensure that, for all problems, the use of the visibility criterion leads to an error reduction or does not lead to an increase of the error. In this thesis we select a threshold angle of 5.0° as it leads to the lowest error for the 2D cylinder problem. Such a threshold angle leads to a significant error reduction compared to the reference case which does not consider a node selection based on the visibility criterion, for the problem of an infinite plate with an elliptical hole for most discretizations.

4.4.4 Additional results for 3D problems

General

We analyze in this subsection results obtained with and without the use of the generalized visibility criterion introduced in Subsections 4.4.1 and 4.4.2 for three 3D problems for which analytical solutions are known. These problems are the 3D extensions of the problems considered in 2D:

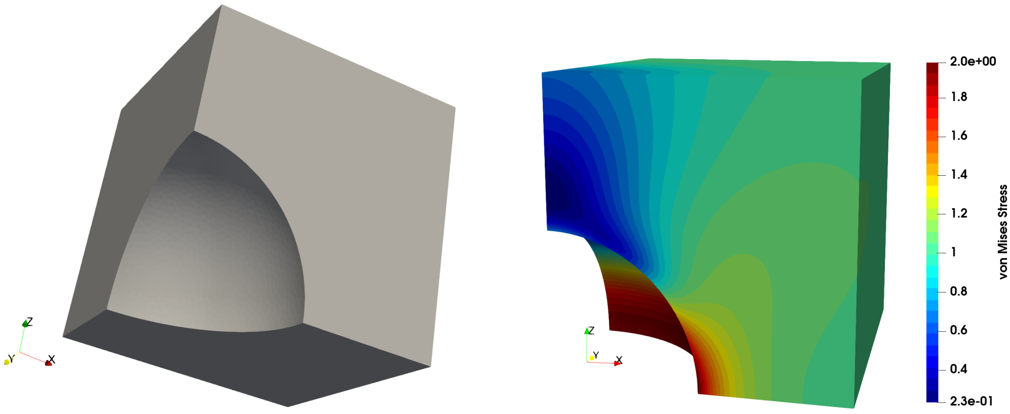
1. an infinite body with a spherical cavity under remote stress loading;
2. an infinite 3D plate with an elliptical hole under biaxial loading;
3. a 3D L-shape in mode I loading.

The first problem has a concavity of low curvature, the second, a concavity of high curvature and the last one, a singularity. For each problem, we show a comparison in terms of L_2 relative error norm of the results from models without and with consideration of the visibility criterion in the collocation stencil selection. A threshold angle of 5.0° is considered for the models with the visibility criterion, based on the results presented in the Subsection 4.4.3. The comparison is performed for a coarse and a fine model.

Infinite body with a spherical cavity

We present the model of an infinite body with a spherical cavity in Figure 4.21 (a). The infinite body is subject to a uniform stress loading $\sigma_{33} = \sigma_0$ far from the cavity. The solution to this problem is presented in a number of articles [65, 66] and is shown in Figure 4.21 (b) in terms of von Mises stress.

We present the L_2 relative error norm for the σ_{11}, σ_{12}, σ_{13}, σ_{33} and σ_{VM} stress components in Table 4.1. The results for the stress components σ_{22} and σ_{23} are not presented because of the symmetries of the problem.



(a)

(b)

Figure 4.21: Infinite body with a spherical cavity under remote stress loading. Symmetric boundary conditions are applied to the symmetry planes XY, XZ and YZ. Stress-free surface boundary conditions are applied to the internal surface of the spherical cavity. The displacement field of the exact solution is applied to the other boundaries of the domain. CAD model (a) and exact von Mises stress solution (b).

Infinite 3D plate with an elliptical hole

We present the model of a 3D infinite plate with an elliptical hole under biaxial loading in Figure 4.22. The solution to this problem is presented in Subsection 1.4.2 for the 2D case. Symmetric boundary conditions are applied to plane YZ. Stress-free surface boundary conditions are applied to the boundary of the elliptical hole and to the top surface of the plate. The bottom surface of the plate is fixed in the Z direction. The displacement field of the exact solution is applied to the other boundaries of the domain.

We present the L_2 relative error norm in Table 4.2 for the σ_{11}, σ_{12} and σ_{22} stress components and for the von Mises stress σ_{VM}. We do not present the results for the stress components σ_{13}, σ_{23} and σ_{33} since those stress components are null.

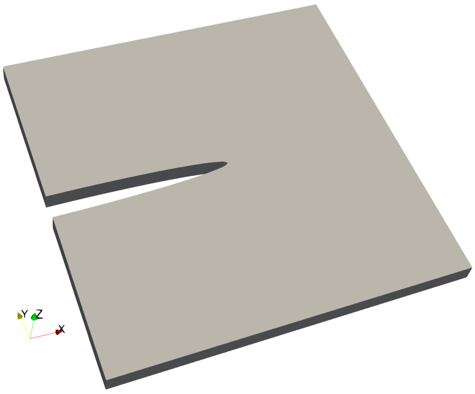


Figure 4.22: Infinite 3D plate with an elliptical hole. Symmetric boundary conditions are applied to plane YZ. Stress-free surface boundary conditions are applied to the boundary of the elliptical hole and to the top surface of the plate. The plate is fixed in the Z direction on the bottom surface of the plate. The displacement field of the exact solution is applied to the other boundaries of the domain.

3D L-shape

We present the model of a 3D L-shape in mode I loading in Figure 4.23. The solution to this problem is presented in Subsection 1.4.2 for the 2D case. Stress-free boundary conditions are applied to the inner surface of the "L". The bottom and top surfaces of the body are fixed in the Z direction. The displacement field of the exact solution is applied to the other boundaries of the domain.

We present the L_2 relative error norm in Table 4.3 for the σ_{11}, σ_{12}, σ_{22} and σ_{33} stress components and for the von Mises stress σ_{VM}. The results for the stress components σ_{13} and σ_{23} since those stress components are null.

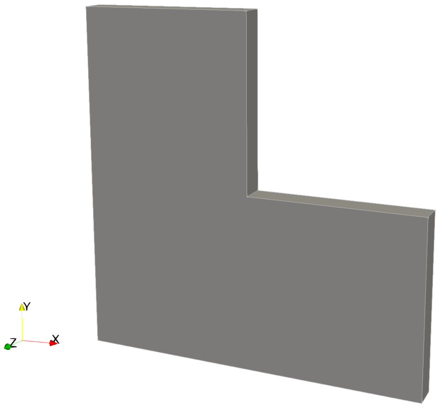


Figure 4.23: 3D L-shape. Stress-free boundary conditions are applied to the inner surface of the “L”. The bottom and top surfaces of the body are fixed in the Z direction. The displacement field of the exact solution is applied to the other boundaries of the domain.

Results

We present the results for the problem of an infinite body with a spherical cavity in Table 4.1. We can see from these results that only a slight error reduction is observed when the visibility criterion is used. A small error increase is observed for the σ_{13} stress component for the coarse model. The error is reduced by a factor of 3 approximately, between the coarse and the fine model.

The results obtained for the infinite body with a spherical cavity are similar to the results obtained for the 2D cylinder. The use of the visibility criterion, in conjunction with a threshold angle of 5.0°, does not lead to an important decrease or increase of the error. With such a threshold angle, the selected collocation stencils are very close with and without consideration of the visibility criterion.

We present the results for the problem of an infinite 3D plate with an elliptical hole in Table 4.2. We can see from these results that an important error reduction is achieved when the visibility criterion is used. The error is reduced by a factor of 6 approximately, for the coarse model and by a factor 19 for the fine model. The error is reduced by a factor of 3 approximately, between the coarse and the fine model, when the visibility criterion is used.

As expected, these results are similar to the ones obtained for the 2D plate. In the areas of the model where the concavity is of high curvature, the visibility criterion allows capturing mode

Table 4.1: Comparison of the results in terms of L_2 relative error norm for the problem of an infinite body with a spherical cavity without and with consideration of the visibility criterion for a coarse and a fine model.

Parameter	Coarse model (45,251 nodes)		Fine model (183,281 nodes)	
	w/o visibility criterion	w/ visibility criterion	w/o visibility criterion	w/ visibility criterion
σ_{11}	1.200E-01	1.134E-01	3.583E-02	3.453E-02
σ_{12}	1.171E-01	1.015E-01	3.544E-02	3.421E-02
σ_{13}	5.877E-02	5.964E-02	2.387E-02	2.378E-02
σ_{33}	1.088E-02	9.885E-03	3.851E-03	3.663E-03
σ_{VM}	1.116E-02	9.948E-03	3.791E-03	3.681E-03

accurately the geometry of the model.

Table 4.2: Comparison of the results in terms of L_2 relative error norm for the problem of an infinite 3D plate with an elliptical hole without and with consideration of the visibility criterion for a coarse and a fine model.

Parameter	Coarse Model (34,038 nodes)		Fine Model (168,116 nodes)	
	w/o visibility criterion	w/ visibility criterion	w/o visibility criterion	w/ visibility criterion
σ_{11}	7.112E+00	1.087E+00	5.887E+00	2.280E-01
σ_{12}	8.995E+00	2.095E+00	5.515E+00	6.252E-01
σ_{22}	4.722E+00	9.264E-01	4.818E+00	3.459E-01
σ_{VM}	1.073E+01	1.650E+00	9.752E+00	5.179E-01

We present the results for the 3D L-shape problem in Table 4.3. We can see from these results that an important error reduction is achieved when the visibility criterion is used. The error is approximately reduced by a factor 37 for the coarse model and by a factor 6 for the fine model. The error is reduced by a factor nearing 1.6 between the coarse and the fine model when the visibility criterion is used.

Table 4.3: Comparison of the results in terms of L_2 relative error norm for the 3D L-shape problem without and with consideration of the visibility criterion for a coarse and a fine model.

Parameter	Coarse model (28,890 nodes)		Fine model (119,683 nodes)	
	w/o visibility criterion	w/ visibility criterion	w/o visibility criterion	w/ visibility criterion
σ_{11}	6.109E+00	1.943E-01	6.762E-01	1.579E-01
σ_{12}	5.039E+00	2.082E-01	4.937E-01	1.284E-01
σ_{22}	4.527E+00	2.035E-01	6.383E-01	1.479E-01
σ_{33}	1.484E+01	3.445E-01	2.091E+00	3.138E-01
σ_{VM}	9.410E+00	2.514E-01	9.263E-01	1.596E-01

4.4.5 Discussion

The results obtained for the 3D problems are consistent with those obtained for the 2D problems. A significant error reduction is observed for the infinite plate with an elliptical hole and for the L-shape problem, both in 2D and in 3D, when the visibility criterion is considered. As expected, the error reduction is more significant for the L-shape problem due to the presence of a sharp corner.

For the case of an infinite plate with an elliptical hole (2D and 3D) and for the infinite body with a spherical cavity problem, the error reduction achieved when the visibility criterion is used tends to decrease as the number of nodes in the domain and on the boundaries of the domain increases. This result is expected as the angle between two adjacent surface elements decreases for these problems when the node density increases. This result is not observed for the 2D L-shape problem because the angle between surface elements on each side of the singularity is not affected by the discretisation. For the 3D L-shape, the error reduction achieved when the visibility criterion is used is smaller for the fine model than for the coarse model. This is attributed to the important error observed for the coarse model without consideration of the visibility criterion.

For the 2D cylinder under internal pressure, the results are only slightly affected by the use of the visibility criterion if a threshold angle of 1.0° or 5.0° is used. A threshold angle of 5.0° leads to the smallest error. For the infinite plate with an elliptical hole and for the 2D L-shape problems, threshold angles of 1.0° and 5.0° lead to significant error reductions. Based on these results, we believe that the selection of a threshold angle of 5.0° is a reasonable choice for most problems.

The problems presented in the previous section were used to benchmark the proposed method against exact solutions to determine the most suitable threshold parameter. To confirm the suitability of the approach to engineering problems, we applied the method to a more complex problem: the loading of a gear coupled to a shaft by a key. The problem is approximated by a 2D plane stress model presented in Figure 4.24. We considered very small radii at the corners of the groove. We modeled the coupling between the shaft and the gear by a sliding boundary condition between the gear and the shaft and between the gear and the key. We applied a uniform pressure loading on a face of a tooth. We compared the results in terms of von Mises stress from a coarse discretization (33,404 nodes) and a fine discretization (109,858 nodes) with and without use of the visibility criterion. We selected a visibility threshold of 5.0°. The results are compared to results obtained from a very fine finite element model composed of 132,665 nodes and 262,193 linear triangular elements. The finite element problem is solved with the code_aster [67].

We present the results in terms of von Mises stress from the finite element model in Figure 4.25. The results show an important stress concentration at the left corner of the keyway. The fillets on each side of the loaded tooth are also subject to a stress concentration but of a much lower magnitude. We present the results in terms of von Mises stress obtained from the collocation models in Figure 4.26. For the coarse model, the results obtained with the visibility criterion are very close to the results obtained from the finite element model. The zone of high stress concentration obtained from the model without consideration of the visibility criterion is more important. This is understood by the less accurate representation of the geometry in the vicinity of the corners of the keyway. For the fine model, little difference can be observed between the collocation results with and without consideration of the visibility criterion. This result is similar to what was observed from the results presented in Subsections 4.4.3 and 4.4.4.

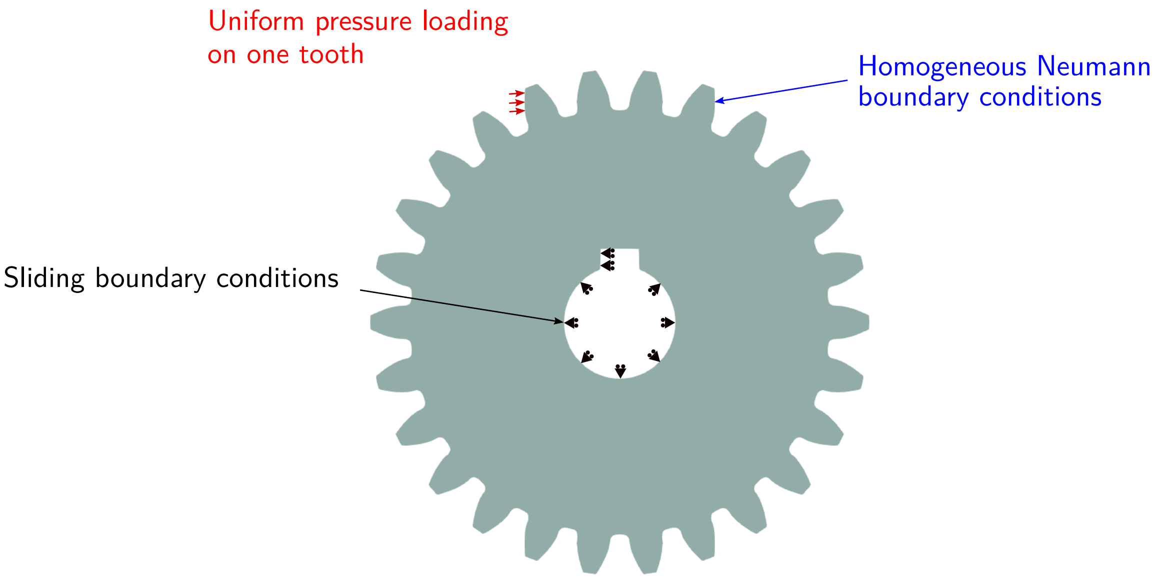
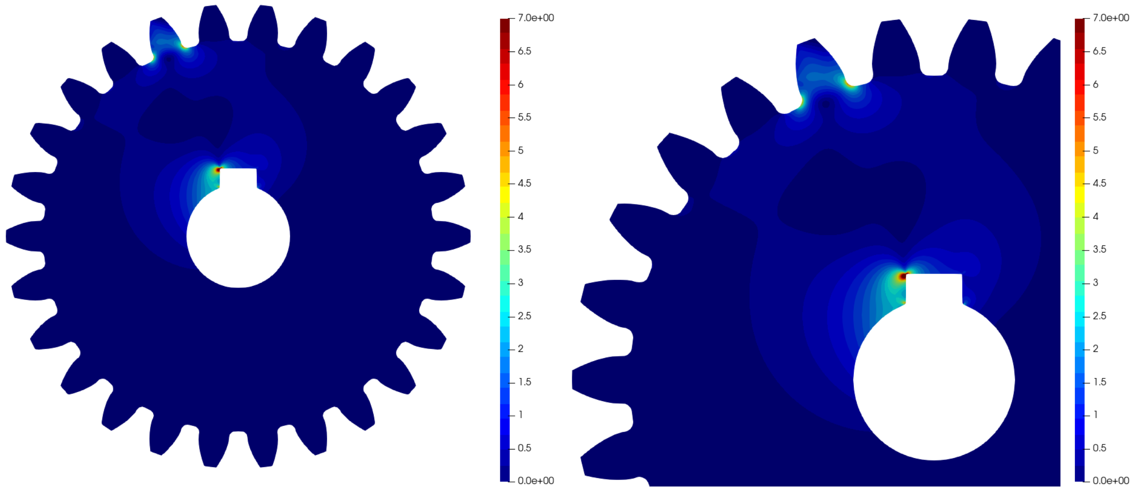
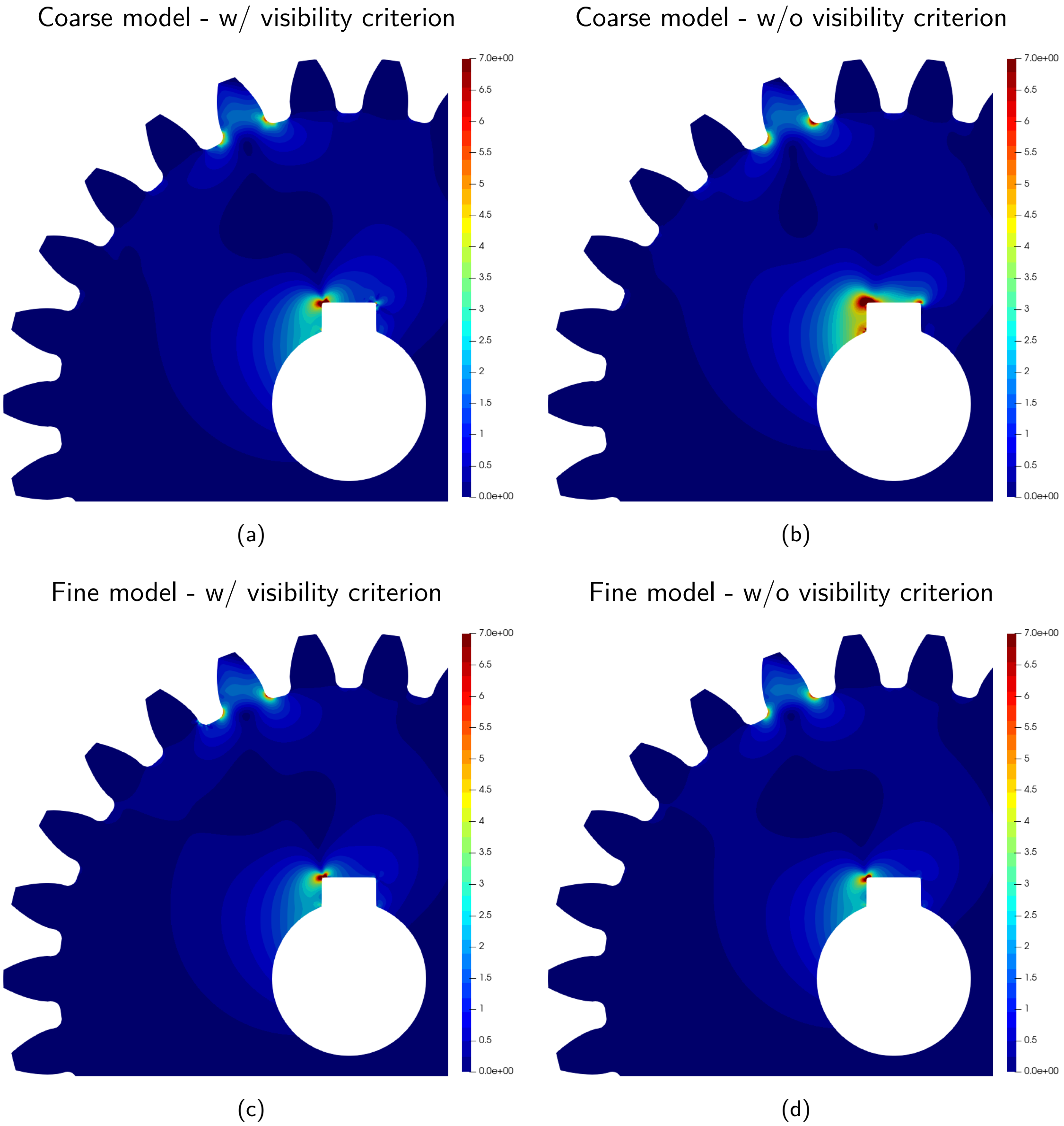

Uniform pressure loading on one tooth
Homogeneous Neumann boundary conditions
Sliding boundary conditions

Figure 4.24: Boundary conditions applied to model a gear coupled to a shaft by a key.



(a) (b)

Figure 4.25: 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.


Coarse model - w/ visibility criterion

Coarse model - w/o visibility criterion

(a)

(b)

Fine model - w/ visibility criterion

Fine model - w/o visibility criterion

(c)

(d)

Figure 4.26: Gear coupled to a shaft - Solution in terms of von Mises stress from collocation with and without use of the visibility criterion for coarse (33,404 nodes) and fine (109,858 nodes) models. The results are shown for the stress range 0-7 for comparison purposes.

4.4.6 Use of the algorithm for model refinement

The purpose of the algorithm presented in Section 4.4 is to assess if a potential support node is "visible" from a collocation node (i.e. to assess if the segment connecting a collocation node to a support node intersects the boundary of the domain with an angle smaller than a defined threshold). The algorithm can also be used to assess if a node is located inside or outside a given domain.

In the context of model refinement, additional nodes are added to the domain. Algorithm #2 presented in Figure 4.13 can be used to assess if a node is located inside or outside a given domain. The nodes added to the model can be considered as new collocation nodes noted $\mathbf{X_{cN}}$ (see Figure 4.27). For each added node, a k-d tree is used to locate the closest node. Depending on the regularity of the model, more than one node can be considered. If the segment connecting $\mathbf{X_{cN}}$ to $\mathbf{X_p}$ is considered as intersecting the boundary based on the condition stated in Algorithms #3 and #4 then the new collocation node is not in the domain. Note that depending on the refinement method, boundary elements might be needed over the whole domain and not only along the convex boundaries as for the visibility criterion algorithm.

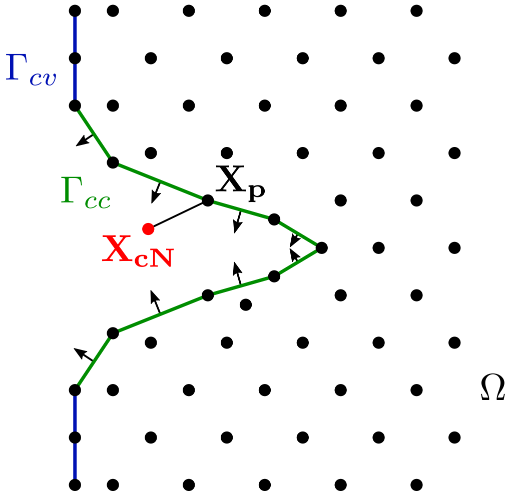


Figure 4.27: Assessment of the location of the potential new collocation node X_{cN} with respect to the domain Ω.

4.5 Conclusion

The selection of the stencil size is one of the major parameters of point collocation methods. The stencil size should be minimized to maximize the sparsity of the linear problem matrix. However, this should be done with care to avoid affecting too much the quality of the approximation. We showed the importance of increasing the stencil size for collocation nodes located on the boundary of the domain. This leads to a significant reduction of the error.

We showed in Section 4.3 the important impact of the use of the visibility criterion for a singular problem. We generalized this approach to all type of problems in Section 4.4 and showed that the approach has a meaningful impact on the solution for concave problems. We showed that a threshold angle parameter of the proposed method allows a more robust node selection. We studied the impact of this threshold angle for geometries comprising concavities of high and low curvature, relatively to the node spacing, and singularities. We showed that the use of the visibility criterion, in conjunction with a threshold angle of 5.0°, leads to a reduction of the error or does not affect the solution for problems having concavities of low curvature. We also showed for the model of a gear coupled to a shaft by a key that the use of the visibility criterion leads to a stress field closer to one obtained from a very fine finite element model. Therefore we recommend that a threshold angle of 5.0° be used as a starting point for most problems. The selected threshold can then be adjusted if needed.

Based on the proposed stencil sizes and on the provided node selection algorithm, one can readily assess a new convex, concave or singular geometry, using a collocation method. The error is minimized in the concave areas of the domain by the use of the generalized visibility criterion.

Chapter 5

Parameter selection and solution improvement methods

5.1 General

We investigate in this chapter ways to reduce the error of problems solved using the GFD and DC PSE methods. We focus first, in Section 5.2 on the optimization of parameters intrinsic to the methods. Then, in Section 5.3, we assess how Voronoi diagrams can be used to improve the results obtained from collocation models based on the GFD and DC PSE methods. Finally, in Section 5.4, we apply a stabilization method, developed for the Finite Point Method, to the GFD and DC PSE methods and analyze the impact on the solution.

5.2 Parametric study

We performed a parametric study, for the GFD and the DC PSE methods, to assess the impact of the various parameters on the solution and reduce as much as possible the error. We aim at selecting a single set of parameters that can be applied to any problem without knowing a priori the solution type.

We consider the following parameters:

- the selected weight function;
- the selected correction function basis for the DC PSE method.

The problems of a 2D cylinder under internal pressure and of a 2D L-shape domain in mode I loading are considered for this study. We used a regular distribution of 5,372 nodes and 13,735 nodes for these problems, respectively. We selected the DCPSE1 variation of the DC PSE method for the purpose of this sensitivity study.

5.2.1 Weight function sensitivity

GFD

Various functions can be considered for the weight function introduced in Equation (2.7) and Equation (2.14). The 3^{rd} and 4^{th} order splines are the preferred types (see Equation (2.28) and Equation (2.29)). In order to vary the shape of the weight functions, we composed the splines with the following power function:

$$W(s) = (w(s))^{\gamma}, \tag{5.1}$$

where w is the spline function and W is the modified weight function. We compareed the results obtained for power parameters γ between 0.4 and 1.2. We also considered a linear weight function for comparison purposes. The equation of this function is given by

$$w(s) = \begin{cases} 1 - s & \text{if } s \leq 1 \\ 0 & \text{if } s > 1. \end{cases} \tag{5.2}$$

The shapes of the considered functions are presented in Figure 5.1.

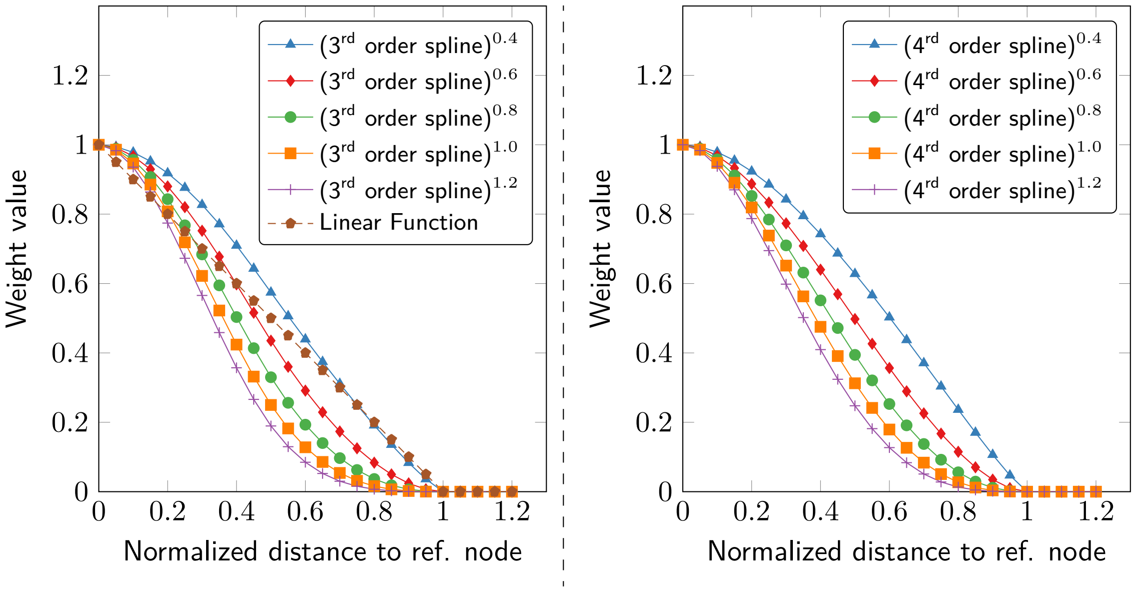


Figure 5.1: GFD Weight Functions: 3rd order spline (left) and 4th order spline (right) for power parameters ranging from 0.4 to 1.2. The linear weight function is also given for reference purposes.

We present the L_2 and L_∞ errors obtained with the considered weight functions in Figure 5.2 and Figure 5.3 for the 2D cylinder and for the 2D L-shape, respectively. We also present the error for the linear weight function in these figures for comparison purposes, even though a power parameter is not used in this function.

We see from Figure 5.2 that within the range [0.6; 0.9] the power parameter has little impact on the error. In this range, the type of spline used does not significantly impact the error either. We observe from the results presented in Figure 5.3 that the power parameter has little impact on the error in terms of L_2 norm. A linear weight function leads to similar results as the spline weight function is the range of power parameters [0.3;1.1]. In terms of L_∞ error norm, the linear function leads to a lower error than the spline functions in the range of power parameters [0.4;1.2].

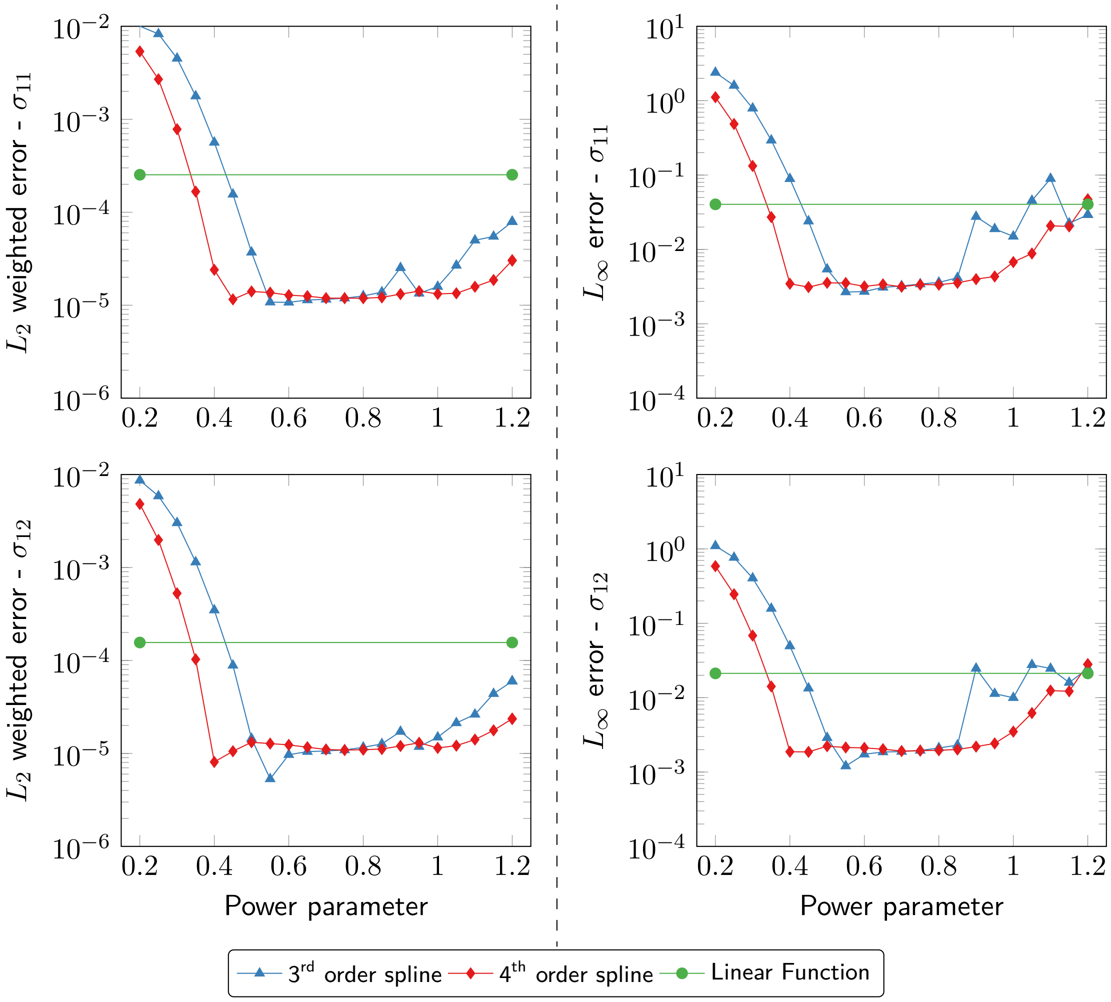


Figure 5.2: GFD weight sensitivity - 2D Cylinder - Results in terms of L_2 weighted (left) and L_∞ (right) error norms obtained with 3rd and 4th order splines weight functions composed with a power function of various exponents. Comparison to results obtained with the linear weight function for reference purposes. The 4th order spline consistently leads to a low error.

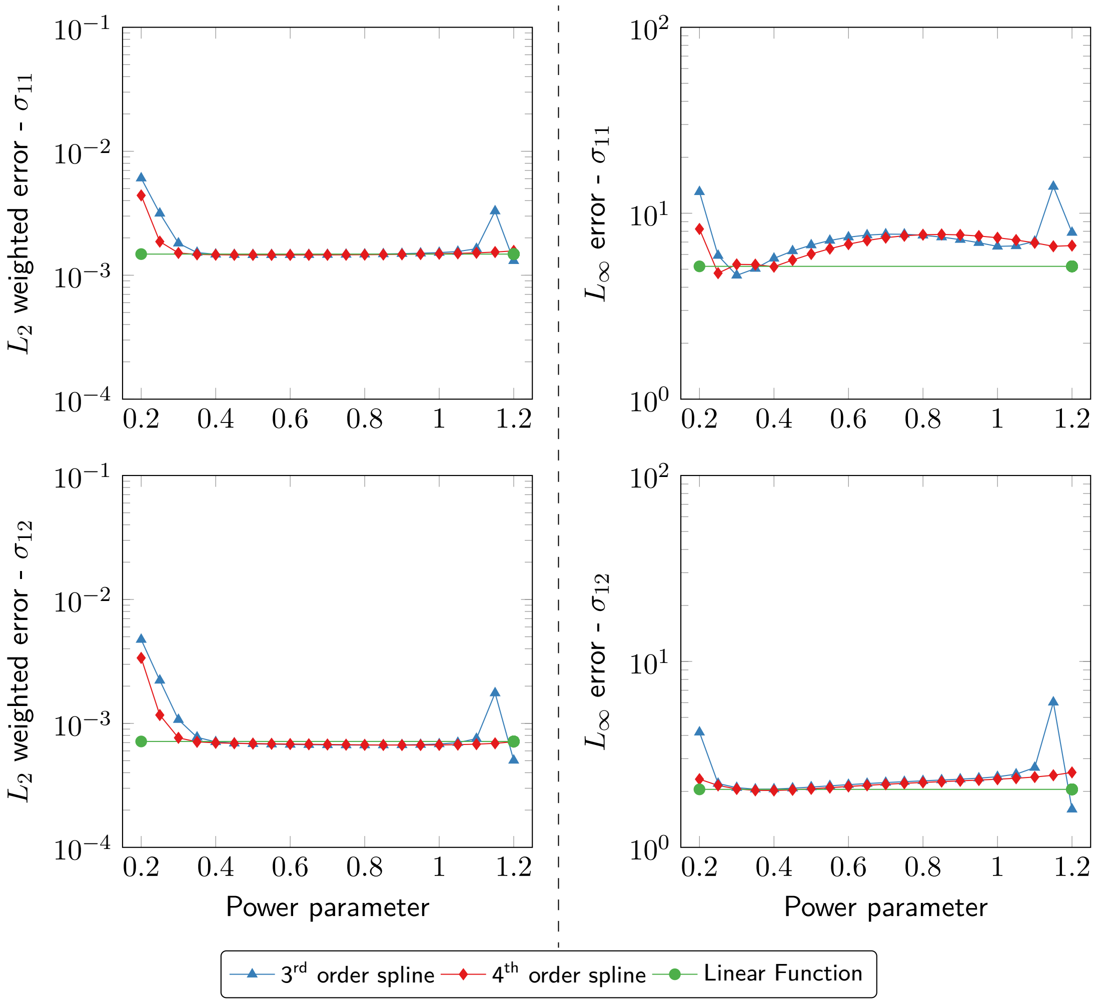


Figure 5.3: GFD weight sensitivity - 2D L-Shape - Results in terms of L_2 weighted (left) and L_∞ (right) error norms obtained with 3rd and 4th order splines weight functions composed with a power function of various exponents. Comparison to results obtained with the linear weight function for reference purposes. The 4th order spline leads to a low error than the 3rd order spline for both stress components and both error norms. The linear function leads to the lowest error in terms of L_∞ error norm.

A 4th order spline with a power parameter of 0.75 appears to be a reasonable choice as it leads to a minimum error for both considered problem. It leads to a low error for the 2D cylinder without being too close to the rapid error increase that we observe when the power parameter decreases below 0.6. It also leads to a reasonably low error for the singular problem both in terms of L_2 and L_∞ error norms. We select a unique set of parameters for both problems in order to be applied to a wide variety of problems in the field of linear elasticity.

DC PSE

We assess in this subsection the influence of the selected weight function on the error for the DC PSE method. We compare results obtained with the exponential weight function presented in Equation (2.27) to results obtained with 3$^{\text{rd}}$ and 4$^{\text{th}}$ order splines, presented in Equations (2.28) and (2.29) respectively. We consider various combinations of shape parameters (ϵ) and exponents (α). We present in Figure 5.4, the profile of three exponential weight functions, along with the two splines functions.

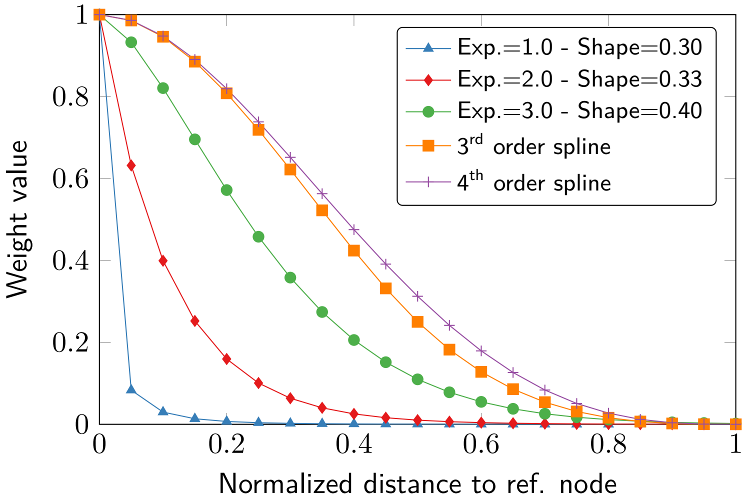


Figure 5.4: DC PSE Weight Functions: Profiles of typical exponential and spline functions used as weight functions in the DC PSE approximation.

We present in Figure 5.5 and Figure 5.6, the error in terms of L_2 and L_∞ norms for the σ_{11} and σ_{12} stress components for various combinations of exponents and shape parameters. We also present the error obtained with the 3^{rd} and 4^{th} order splines on the graphs for comparison purposes, even though the shape parameter does not apply for these functions.

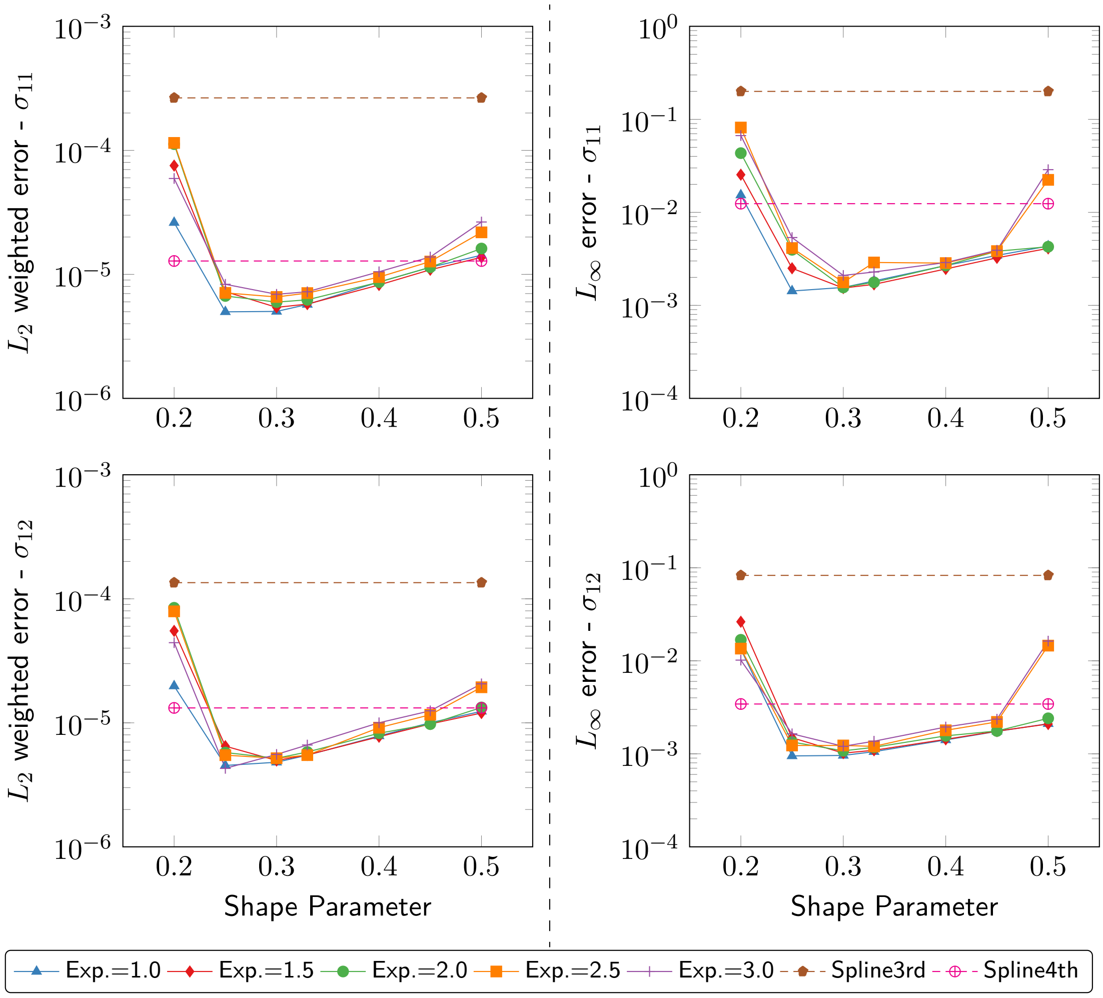


Figure 5.5: DC PSE weight sensitivity - 2D Cylinder - Results in terms of L_2 weighted (left) and L_∞ (right) error norms obtained with shape parameters ranging from 0.2 to 0.5 for exponential functions of various exponents. Comparison to results obtained with 3^{rd} and 4^{th} order splines. A shape parameter of 0.3 leads to a low error for all the exponents considered.

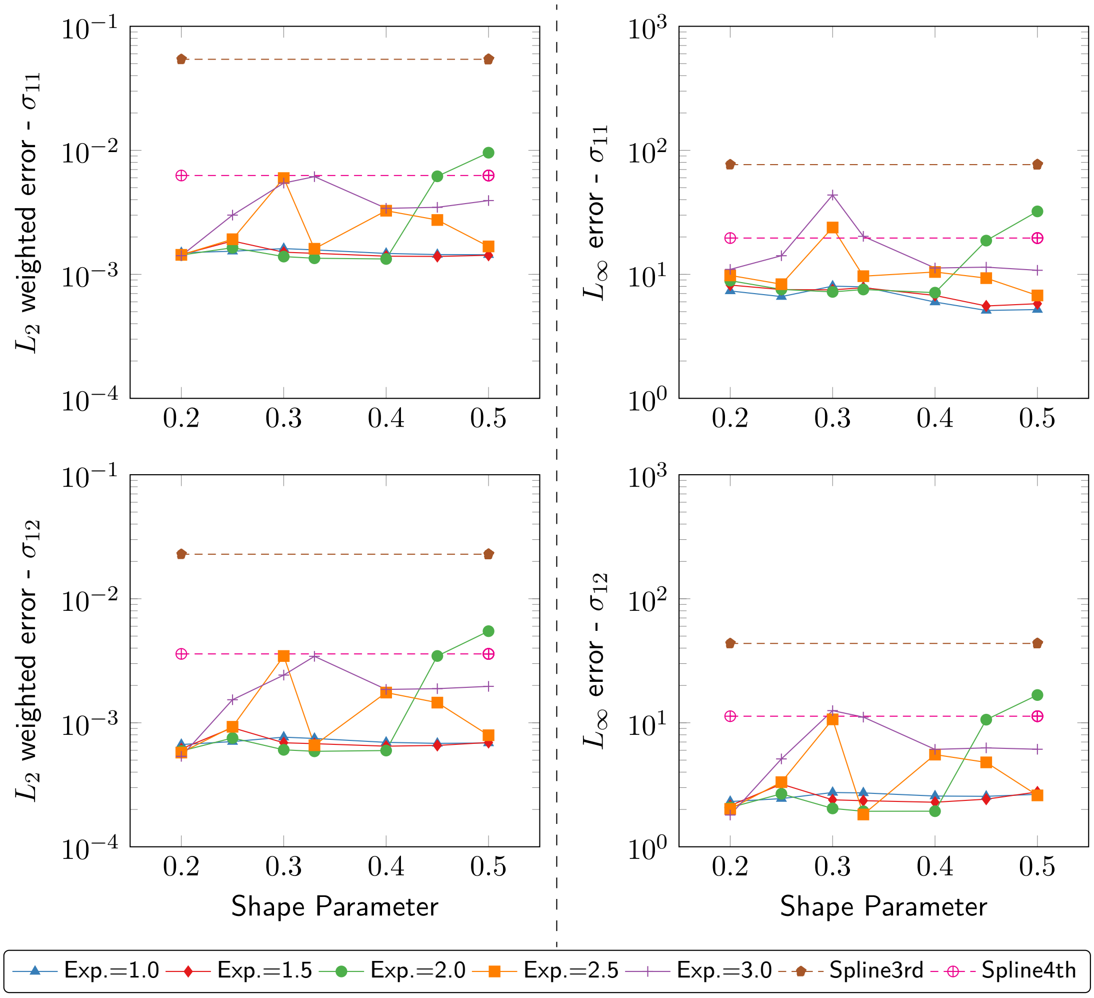


Figure 5.6: DC PSE weight sensitivity - 2D L-Shape - Results in terms of L_2 weighted (left) and L_∞ (right) error norms obtained with shape parameters ranging from 0.2 to 0.5 for exponential functions of various exponents. Comparison to results obtained with 3^{rd} and 4^{th} order splines. An exponent of 2.0 and a shape parameter of 0.33 lead to the lowest error in terms of L_2 norm.

The analysis of Figure 5.5 and Figure 5.6 shows that the exponential weight functions lead to smaller errors than the spline functions for shape parameters between 0.25 and 0.45 and for both problems. The combination of an exponent of 1.0 and a shape parameter of 0.25 leads to the smallest error for the 2D cylinder. A shape parameter of 0.30, associated with an exponent of 1.0, leads to similar results for this problem without being too close to a rapid increase in the observed error.

The analysis of the results for the 2D L-shape shows that an exponent of 2.0 associated with a shape parameter of 0.33 leads relatively constantly to the lowest error.

We select a single set of parameters that can be applied to most of the problems for which the solution is a priori unknown. This set of parameters is: a shape parameter of 0.30 and an exponent of 1.0. This combination leads to a more significant error reduction than the set of parameters leading to the minimum error for the 2D L-shape problem.

5.2.2 DC PSE correction function

The basis functions used to build the correction function in the DC PSE method can be selected from various function types. The most common bases are the polynomial and the exponential bases. For the case of a two-dimensional problem, the polynomial basis is $\mathbf{P} = [1, x, y, x^2, xy, y^2]^T$, and the exponential basis: $\mathbf{P} = [1, e^x, e^y, e^{2x}, e^{x+y}, e^{2y}]^T$. In order for these bases to be independent from the node density, the functions are scaled according to the support radius. For a node $\mathbf{X_{pi}}$ in the support of the collocation node $\mathbf{X_c}$, the scaling parameters are $Sx_i = \frac{x_c - x_{pi}}{R_c}$ and $Sy_i = \frac{y_c - y_{pi}}{R_c}$, where R_c is the support radius of the collocation node $\mathbf{X_c}$. The polynomial basis becomes:

$$\mathbf{P} = \left[1, Sx_i, Sy_i, {Sx_i}^2, Sx_iSy_i, {Sy_i}^2\right]^T. \tag{5.3}$$

We present the errors obtained for each correction function bases in Figure 5.7 for the 2D cylinder problem and in Figure 5.8 for the 2D L-shape problem.

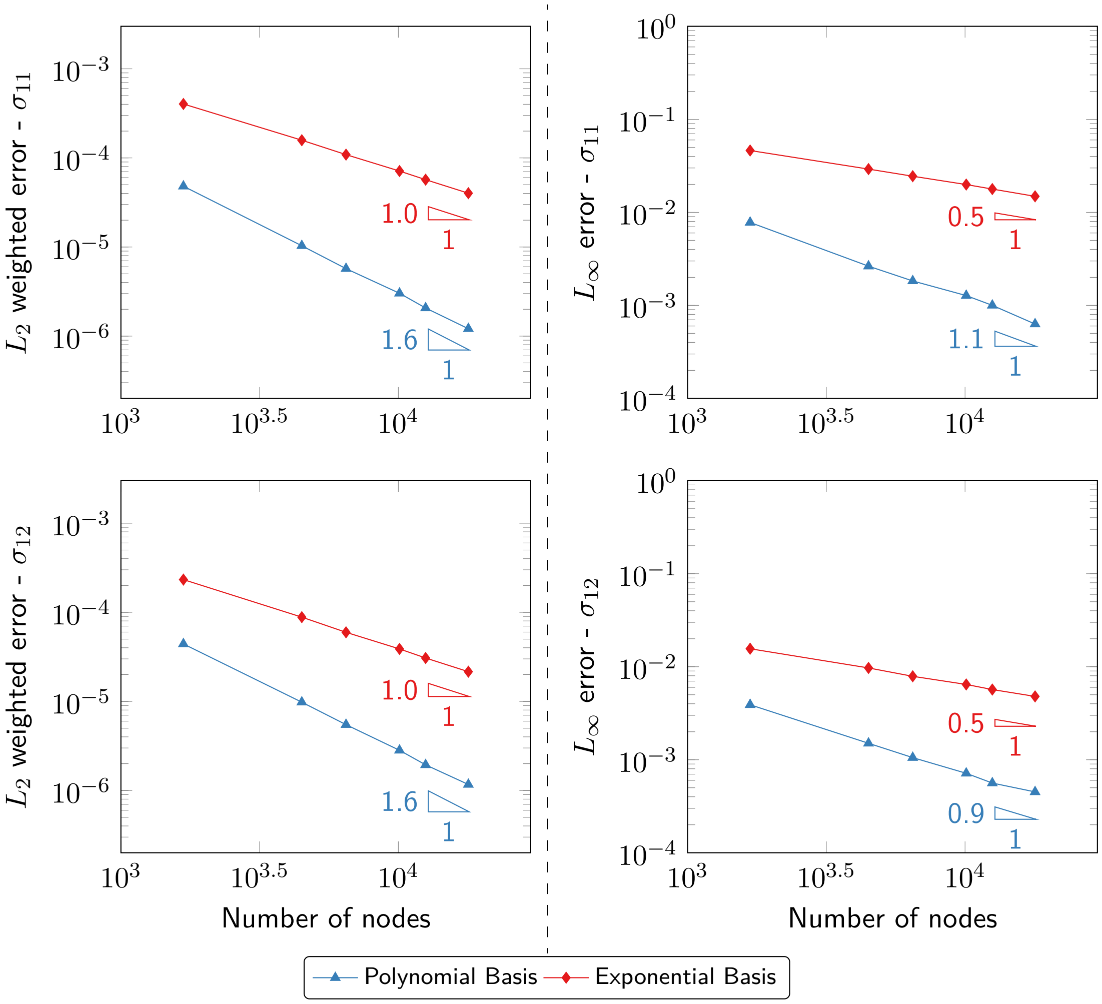


Figure 5.7: 2D Cylinder - Results in terms of L_2 weighted (Left) and L_∞ (Right) error norms for polynomial and exponential basis functions as a function of the number of nodes in the model. The use of a polynomial basis leads to a lower error and a faster convergence.

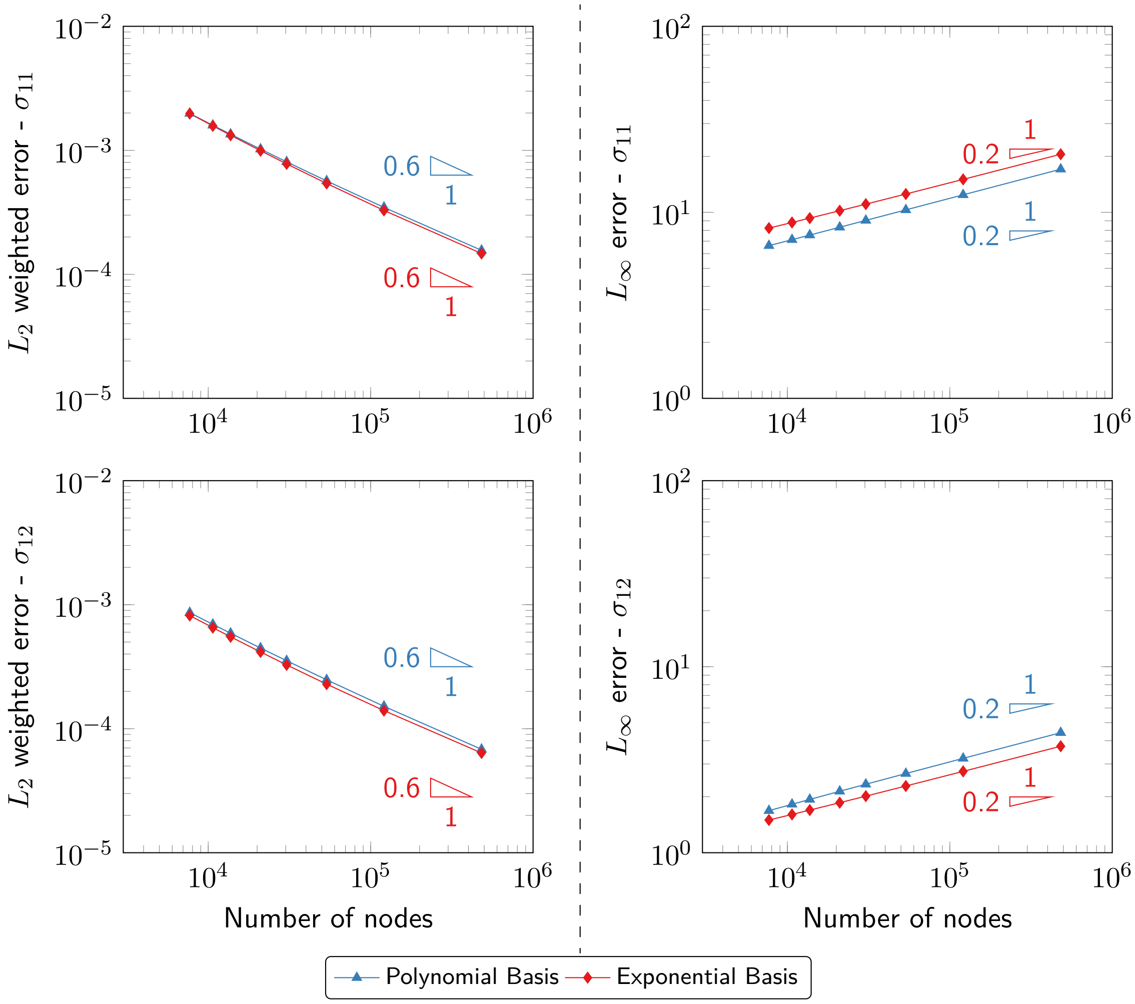


Figure 5.8: 2D L-Shape. Results in terms of L_2 weighted (Left) and L_∞ (Right) error norms for polynomial and exponential basis functions as a function of the number of nodes in the model. Both function bases lead to similar convergence rates. The use of an exponential basis leads to a slightly lower error for the L_2 weighted error norm. For the L_∞ error norm, the polynomial basis leads to the lowest error for the σ_{11} stress component while the exponential basis leads to the lowest error for the σ_{12} stress component.

We observe from Figure 5.7 that the polynomial function basis constantly leads to a much lower error than the exponential basis. Depending on the number of nodes, the error increases by a factor between 5 and 30 when the exponential basis is used.

In Figure 5.8 we see that both correction function bases lead to similar results. In terms of L_2 weighted error norm the exponential basis leads to a reduction of around 5% compared to the results where a polynomial basis is used.

Based on the results presented in this subsection, we prefer the polynomial basis as it leads to a much higher error reduction than the exponential basis for the considered problems. We expect that this basis function gives a reasonably low error for most of the problems from the field of linear elasticity.

5.2.3 Results summary

Based on the results presented in the above subsections, we present in Table 5.1 our selected set of parameters that are most likely to lead to a minimum error, while maintaining the computational expense reasonably low. We expect that these parameters lead to a low error for a wide variety of linear elasticity problems, including singular problems. We use these parameters for most of the problems solved in this thesis.

Table 5.1: Selected parameters for the GFD and DC PSE methods based on the results of the parametric study.

Parameter	**GFD**	**DC PSE**
Weight Function Type	4^{th} order spline	Exponential
Weight Function Parameter	$\gamma = 0.75$	$\alpha = 1$, ϵ=0.30
Correction Function	N/A	Polynomial
Size of Inner Nodes Support (2D/3D)	11/37	13/37
Size of Boundary Nodes Support (2D/3D)	19/75	19/75

5.3 Use of a Voronoi diagram in collocation

5.3.1 General

In this section we assess if using a Voronoi diagram on the support of a collocation node helps reduce the error of the considered point collocation methods.

A Voronoi diagram is a partition of a selected region of a domain over which nodes are distributed.

A cell is associated with each node. The boundaries of the cell are defined so that all the points contained in it are closer to the cell reference node than to any other node of the domain. Figure 5.9 shows a 2D Voronoi diagram drawn on the support of an inner node of the domain. The nodes are shown in red color and the boundary of the support in blue color. The Voronoi cells are limited by gray lines. Sukumar [68] and Zhou et al. [69], respectively, used Voronoi diagrams for node selection, and body integration.

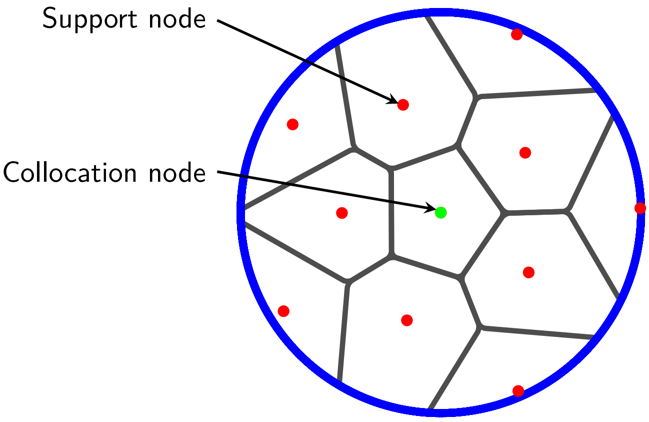


Figure 5.9: 2D Voronoi diagram on the disc support of a collocation node. The cells associated with each node are delimited by gray lines, while the boundary of the support is drawn in blue.

5.3.2 Application to the GFD and DC PSE methods

GFD

The principle of the GFD method is presented in Section 2.2. When more nodes than derivatives are present in the node support, a moving least square approximation is used to determine the field derivatives that best fit the distribution. The contribution of each node to the least square approximation is weighted by a function which only depends on the distance between the reference node and the support node. We use a Voronoi diagram to determine an additional weight based on the spatial arrangement of the nodes. This weight is the area or volume v of the considered

Voronoi cell and is multiplied with the distance based weight w. Equation (2.14) becomes:

$$m_{ij} = \sum_{k=1}^{m} w(\mathbf{X_{pk}} - \mathbf{X_c}) v(\mathbf{X_{pk}}^c) P_{ik}(\mathbf{X_c}) P_{jk}(\mathbf{X_c}). \tag{5.4}$$

DC PSE

One of the key aspects of the DC PSE method presented in Section 2.3 is the convolution of the Taylor's series expansion with a correction function η. The domain integral is transformed into a discrete summation with a volume V_p associated with each particle $\mathbf{X_p}$ of the support. In a first approximation, all V_p values are set to unity. In order to improve the accuracy of the method, we use a Voronoi diagram to set V_p equal to the volume of the Voronoi cell associated with each node $\mathbf{X_p}$.

5.3.3 Results

In this subsection, we present results, obtained based on the approaches presented in Subsection 5.3.2, for the 2D cylinder and 2D L-shape problems. The results from a model where Voroni weights are used and a model where these weights are not considered are compared. We consider two types of node distributions: a structured and a free node distribution. The structured node distribution is created using a constant angle and radius increment for the 2D cylinder. For the 2D L-shape, a grid-type arrangement is used. The free node distribution uses a Delaunay triangulation of the domain for both problems. The two types of node arrangements are presented in Figure 5.10 for the 2D cylinder problem.

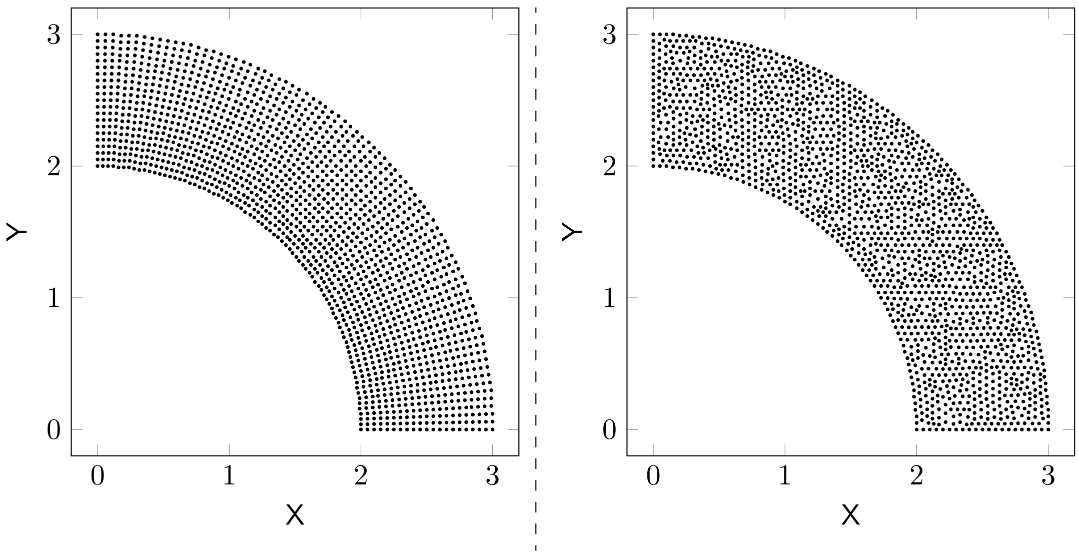


Figure 5.10: 2D Cylinder Node Distribution - Structured consisting of 1680 Nodes (Left) and Free consisting of 1762 Nodes (Right). The structured node distribution is based on constant angle and radius increments while the free node distribution uses a Delaunay triangulation of the domain.

GFD

We present in Figure 5.11 and Figure 5.12 the results obtained with the GFD method for the 2D cylinder and the 2D L-shape problems, respectively. The results are presented for the two node distributions considered with and without use of Voronoi weights. We observe a small error reduction for the L_2 weighted and the L_∞ error norms when Voronoi weights are used. We do not observe this reduction for all node densities. For the 2D cylinder and the 2D L-shape problems, we observe an error reduction of around 2% when Voronoi based weights are used with a regular discretization of the domain. We observe a more significant error reduction for the 2D cylinder problem with a free discretization of the domain. The error reduction is around 17%. For the 2D L-shape problem, we observe an error increase of 3% when Voronoi weights are used with a free discretization of the domain.

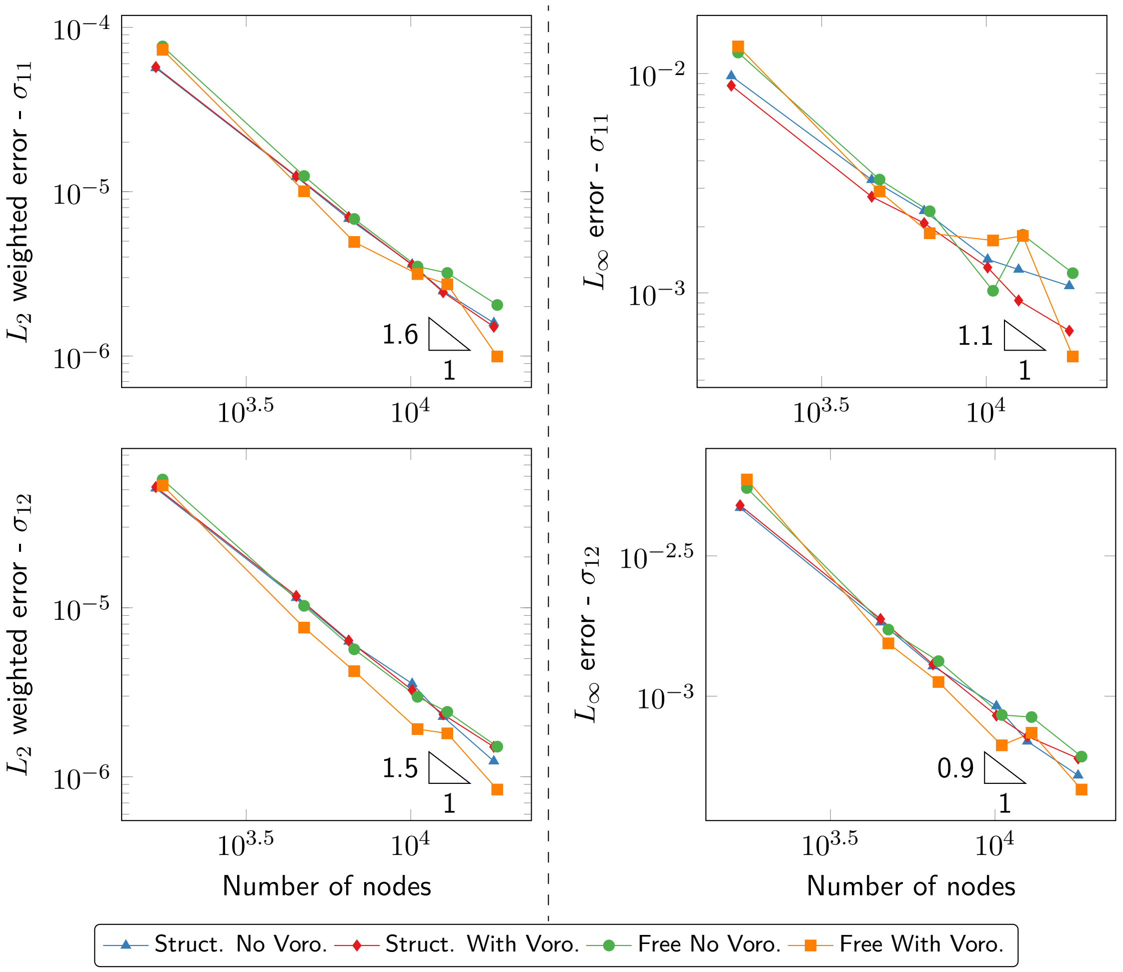


Figure 5.11: Impact of Voronoi based Weights on the Errors for the GFD Method - 2D Cylinder. L_2 (Left) and L_∞ (Right) errors for structured and free node distributions. A reduction in the error is only observed for the free node distribution.

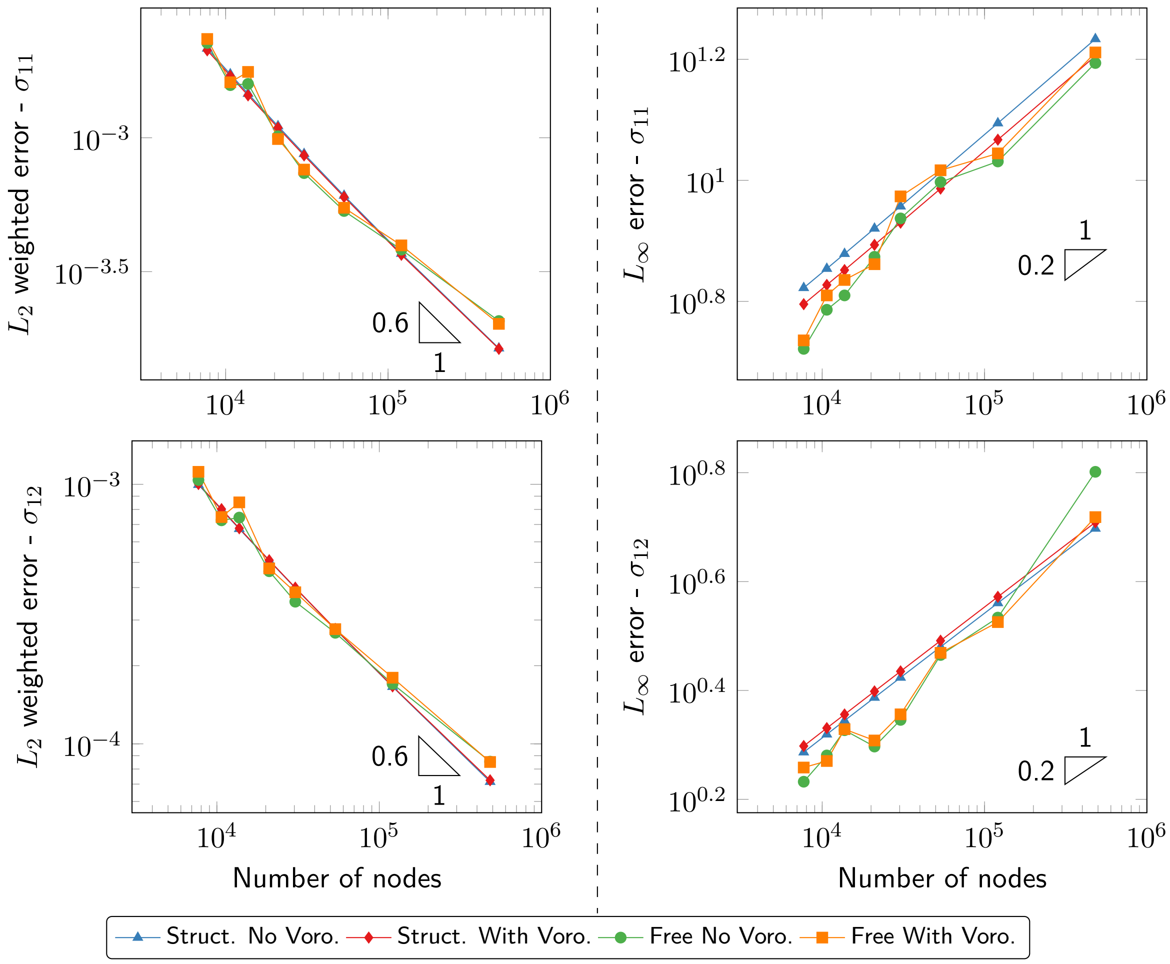


Figure 5.12: Impact of Voronoi based Weights on the Errors for the GFD Method - 2D L-Shape. L_2 (Left) and L_∞ (Right) errors for structured and free node distributions. A slight reduction in the error is observed for the structured node distribution.

DC PSE

We assess the impact of Voronoi based volumes, used as part of the convolution step of the DC PSE method, on the observed error for the two node distributions for the 2D cylinder and the 2D L-shape problems. We present the results in Figure 5.13 and Figure 5.14.

We observed from Figure 5.13 that, for the 2D cylinder problem, the use of Voronoi based volumes leads to a large error increase for the structured node distribution. For the free node distribution, we observe an average reduction of 10%.

We see from Figure 5.14 that the trend for the 2D L-shape problem is the opposite than for the 2D cylinder problem. We observe an error reduction of around 5% when Voronoi based volumes are used with the structured node distribution. We observe a slight error increase (less than 1%) for the free node distribution when Voronoi based volumes are used.

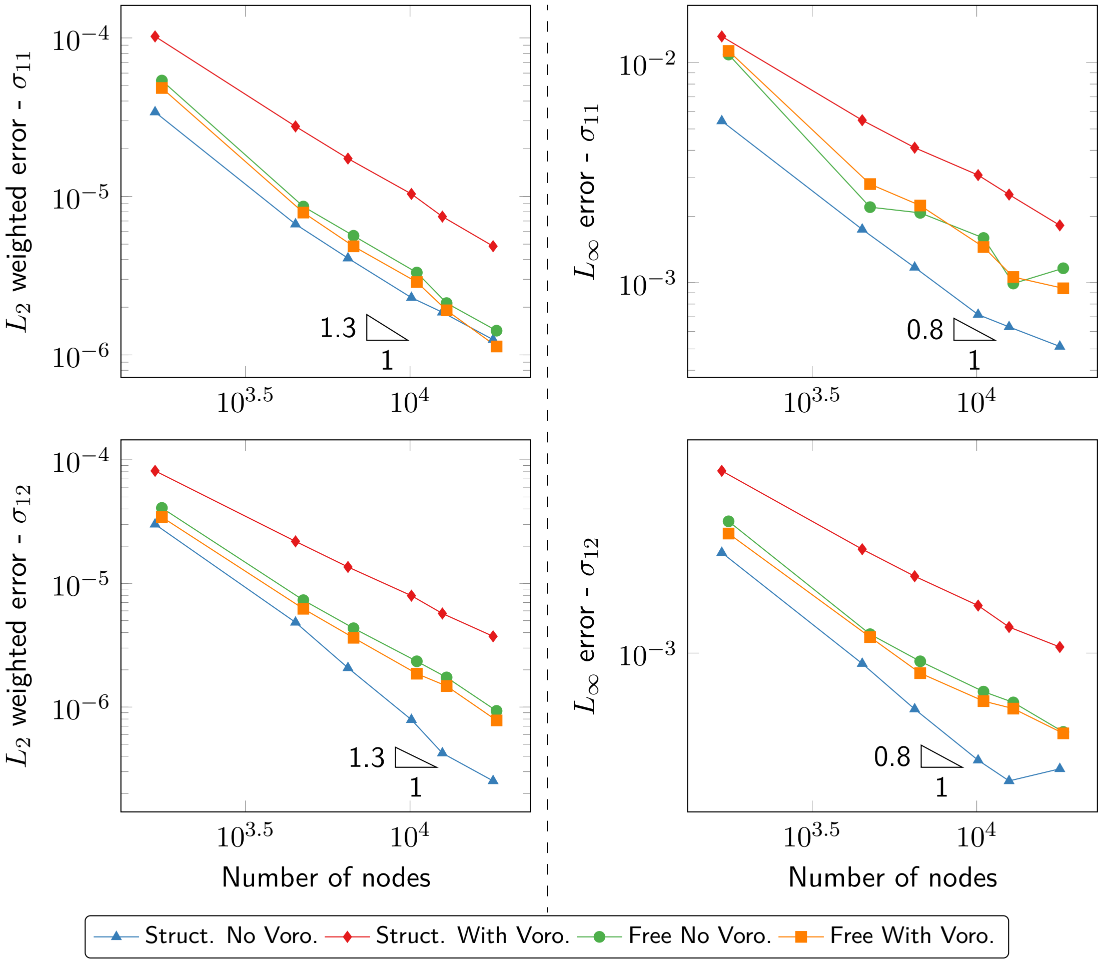


Figure 5.13: Impact of Voronoi Integration on the Errors for the DC PSE Method - 2D Cylinder. L_2 (Left) and L_∞ (Right) errors for structured and free node distributions. A slight reduction in the error is only observed for the free node distribution.

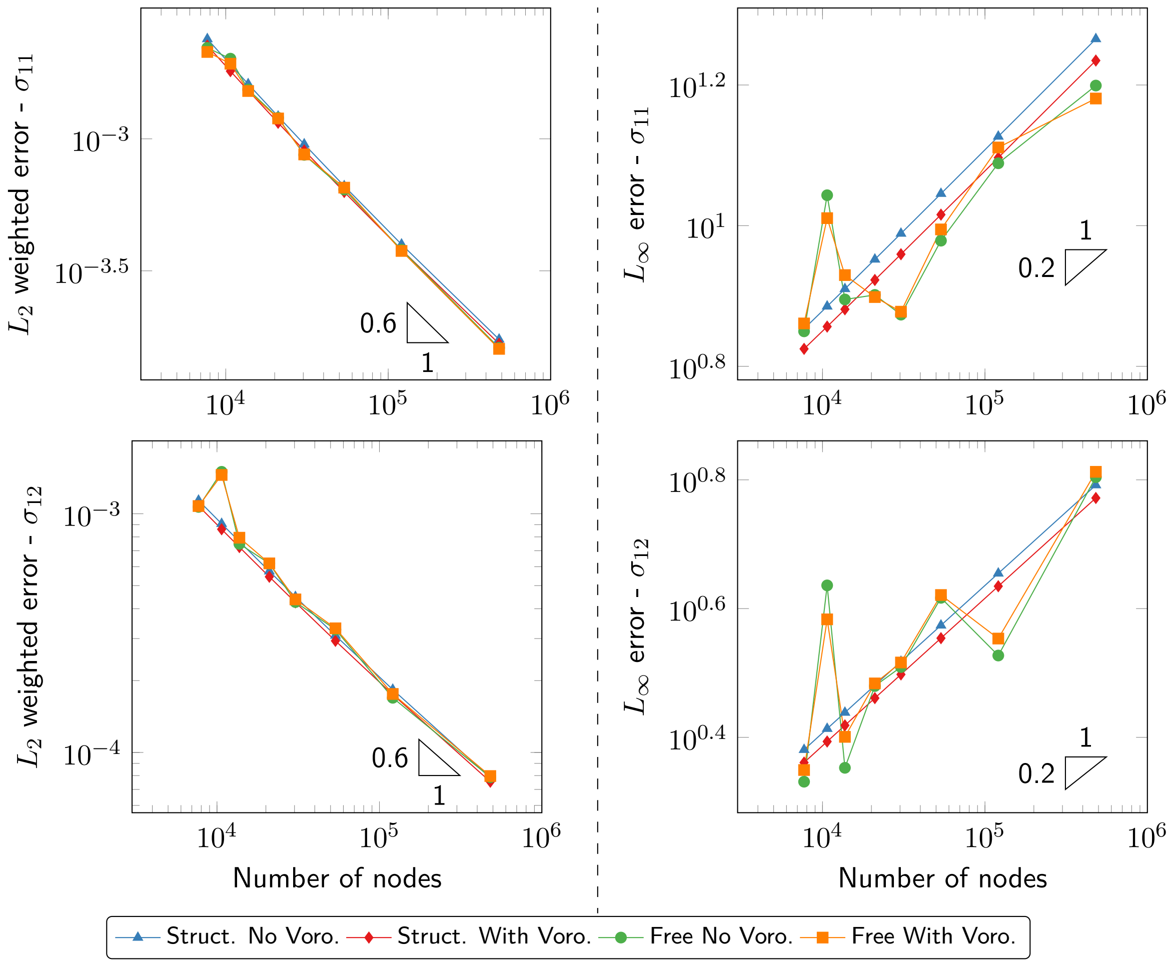


Figure 5.14: Impact of Voronoi based Weights on the Errors for the DC PSE Method - 2D L-Shape. L_2 (Left) and L_∞ (Right) errors for structured and free node distributions. A slight reduction in the error is observed for the structured node distribution.

Discussion

We observe from the results presented from Figure 5.11 to Figure 5.14 that the error reduction achieved by the use of Voronoi based volumes is not guaranteed. The use of such volumes for the 2D cylinder problem leads to an error reduction of up to 17%. However, for the 2D L-shape problem, we observe an error increase of 3% for the free node distribution. From this study, we conclude that the use of the Voronoi based weights shall be used with care as it may lead to a larger error than without such weights.

5.4 Collocation method stabilization

5.4.1 General

In this section we assess if a stabilization of the boundary conditions leads to an error reduction for the GFD and DC PSE methods.

Within the framework of collocation, the boundary conditions are applied at the nodes using the strong form of the partial differential equations. This may lead to ill conditioning of the linear system of equations, as both the boundary conditions and the equilibrium equation cannot be enforced simultaneously at a boundary node. To overcome this issue, a stabilization method, known as the *Finite Increment Calculus* (FIC), was introduced by E. Oñate [70, 71] for structural problems that are solved with the Finite Point Method. The method is presented in Subsection 5.4.2. We use this stabilization approach for the GFD and DC PSE methods and present the results in Subsection 5.4.3.

5.4.2 Stabilized equations

Considering an unknown field f, a partial differential problem is defined by a differential operator $\mathcal{A}$ applied to the interior domain Ω, a field $\overline{f}$ set to the boundary Γ_u, and a differential operator $\mathcal{B}$ applied to the boundary Γ_t (see Figure 5.15).

$$\begin{aligned} \mathcal{A}(\mathbf{f}) &= 0 \quad \text{in} \quad \Omega, \\ \mathbf{f} - \overline{\mathbf{f}} &= 0 \quad \text{on} \quad \Gamma_u, \\ \mathcal{B}(\mathbf{f}) &= 0 \quad \text{on} \quad \Gamma_t. \end{aligned} \tag{5.5}$$

Based on [70] and [72], the stabilized system of equations is:

$$\begin{aligned} \mathcal{A}(\mathbf{f}) - \frac{1}{2}\sum_{j=1}^{m} h_j \frac{\partial \mathcal{A}(\mathbf{f})}{\partial x_j} &= 0 \quad \text{in} \quad \Omega, \\ \mathbf{f} - \overline{\mathbf{f}} &= 0 \quad \text{on} \quad \Gamma_u, \\ \mathcal{B}(\mathbf{f}) - \sum_{j=1}^{m} h_j n_j \mathcal{A}(\mathbf{f}) &= 0 \quad \text{on} \quad \Gamma_t, \end{aligned} \tag{5.6}$$

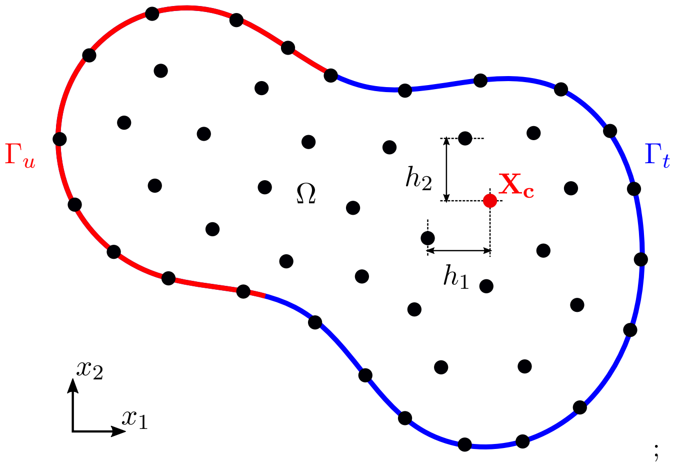


Figure 5.15: 2D domain Ω on which Dirichlet boundary conditions are applied to the boundary Γ_u and Neumann boundary conditions to Γ_t. The characteristic lengths h_1 and h_2 are presented for the collocation node $\mathbf{X_c}$.

where m is the dimension of the domain, h_j is the characteristic length of the domain in the direction j, and n is the unit normal.

For isotropic support weight functions, h_j reduces to h and can be expressed as follows:

$$\begin{aligned} h =& R_c \left(\frac{\pi}{N_c}\right)^{\frac{1}{2}} && \text{for 2D problems,} \\ h =& R_c \left(\frac{4\pi}{3N_c}\right)^{\frac{1}{3}} && \text{for 3D problems,} \end{aligned} \tag{5.7}$$

where R_c and N_c, respectively represent, the radius and the number of support nodes associated with a collocation node $\mathbf{X_c}$.

5.4.3 Results

We apply Equation (5.6) to the boundary nodes where the maximum error is usually observed. Also, the stabilized equation on the boundary does not require the approximation of an additional derivative order. We present the results obtained with the stabilization approach from Figure 5.16 to Figure 5.19 for the GFD and DC PSE methods, for the problems of the 2D cylinder and the problem of the 2D L-shape. L_∞ error results have not been presented in this section as this error highly depends on the proximity of the closest node to the singularity.

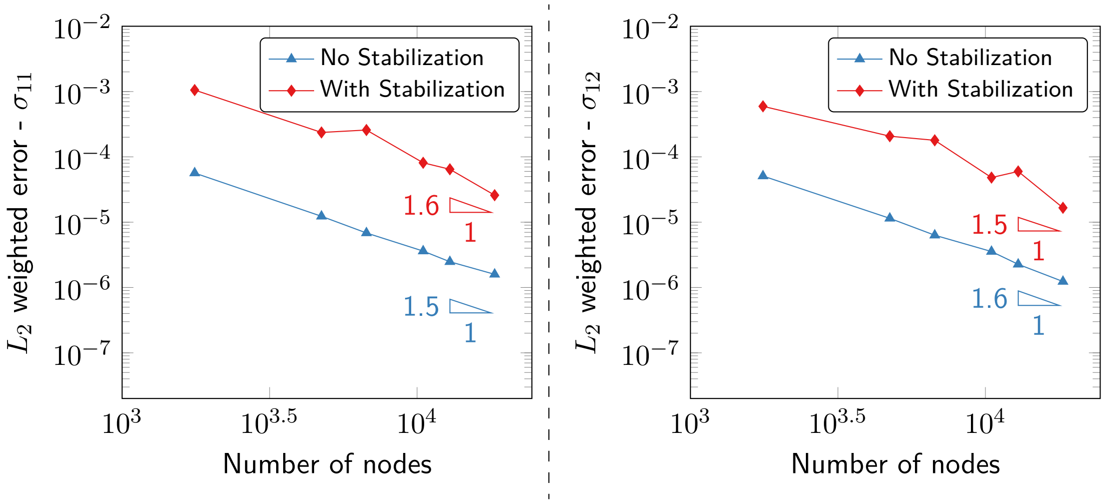


Figure 5.16: Stabilization results comparison - 2D Cylinder - GFD Method. L_2 weighted error for stabilized and non-stabilized PDE for increasing node numbers. A lower error is observed for the non-stabilized PDE.

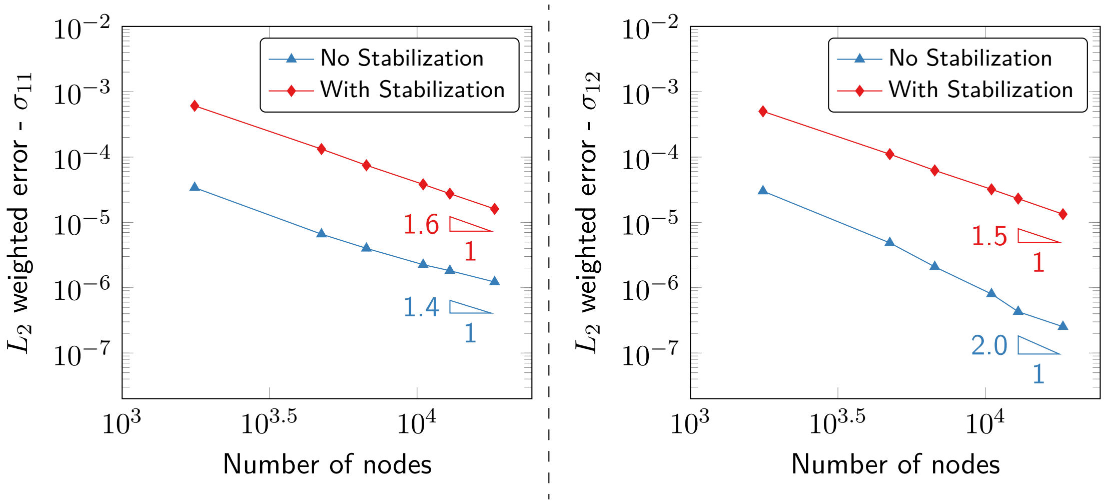


Figure 5.17: Stabilization results comparison - 2D Cylinder - DC PSE Method. L_2 weighted error for stabilized and non-stabilized PDE for increasing node numbers. A lower error is observed for the non-stabilized PDE.

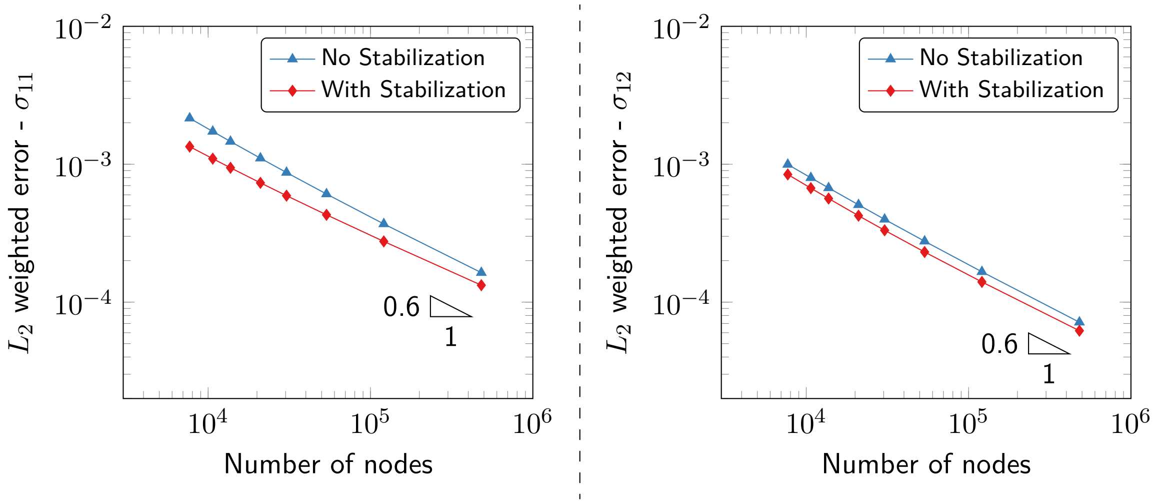


Figure 5.18: Stabilization Results Comparison - 2D L-Shape - GFD Method. L_2 weighted error for stabilized and non-stabilized PDE for increasing node numbers. A lower error is observed for the stabilized PDE.

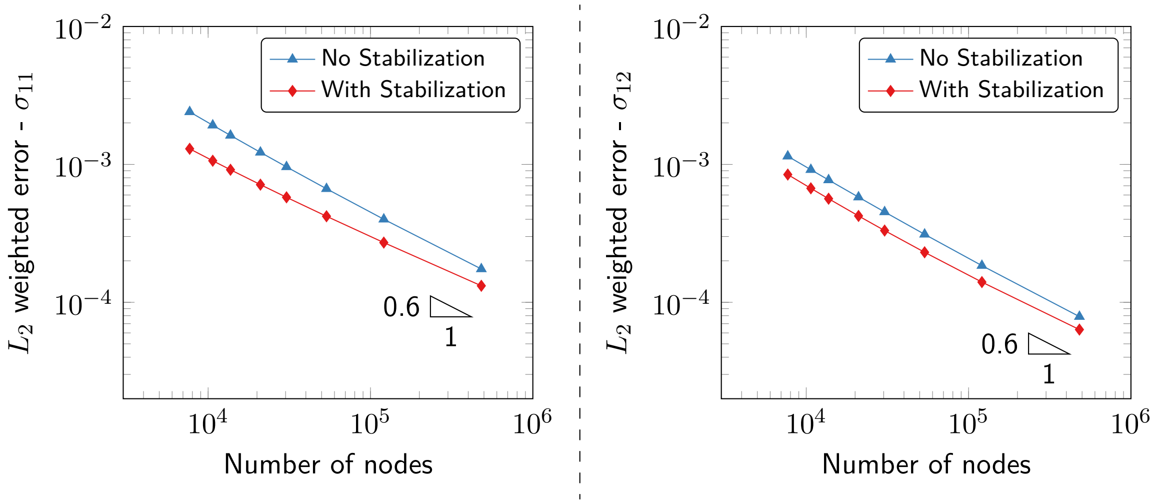


Figure 5.19: Stabilization results comparison - 2D L-Shape - DC PSE Method. L_2 weighted error for stabilized and non-stabilized PDE for increasing node numbers. A lower error is observed for the stabilized PDE.

We observe from the results presented from Figure 5.16 and Figure 5.17 that the stabilization equations lead to an error increase for the 2D cylinder problem. Using the stabilization method increases by a factor 30 the error for the GFD method and by a factor 20 for the DC PSE method.

We can see from Figure 5.18 and Figure 5.19 that the use of the stabilization method leads to a reduction of the error for the L-shape problem. An average error reduction of 25% is observed for the GFD method and of 35% for the DC PSE method.

For the 2D cylinder, the loading is applied via Neumann boundary conditions, which represent the pressure loading. For the L-shape problem on the other hand, the domain is loaded using Dirichlet boundary conditions. We conclude from this study that stabilization of the Neumann boundary conditions does not lead to a reduction of the observed error. Thus, stabilization of Neumann loaded problems does not seem to be an effective solution for the considered methods.

5.5 Conclusion

We noted that the various parameters involved in the GFD and DC PSE methods have a significant impact on the solution, and that they should therefore be carefully chosen. Another main conclusion of this work is that common approaches used in practice to improve collocation methods must be used with caution as they do not always lead to the reduction of the overall error. We observed the following:

1. For the GFD method, the weight function based on the 4th order spline leads to the minimum error for problems with a polynomial solution such as the pressurized cylinder. For singular problems, such as the L-shape in mode I loading, both linear and 4th order spline weight functions lead to a minimum error.

2. For the DC PSE method, the weight function based on the exponential functions leads to the minimum error for both polynomial and singular problems.

3. For the DC PSE methd, for the problems with a polynomial solution, a polynomial correction function basis leads to an error approximately fifteen times lower than with an exponential basis function. For the singular problems, an exponential correction function basis leads to an error approximately 5% lower than with an exponential basis function. A polynomial correction function is recommended for most problems as the solution type is not known a priori.

4. Voronoi diagrams can be used to give additional information to the collocation methods on the spatial arrangement of the nodes over the domain.

    (a) For the GFD method, Voronoi diagrams allow the selection of weights which depend on the node placement in the collocation node support and on the local node density.

    (b) For the DC PSE method, Voronoi diagrams are expected to improve the accuracy of the convolution, but they may also lead to an increased error for some node distributions.

    The use of Voronoi diagrams helps in reducing the error for the 2D cylinder problem with a free node distribution (based on Delaunay triangulation). A reduction of up to 17% is observed for the GFD method and of up to 10% for the DC PSE methods. For a regular node distribution an error increase of 3% is observed for the considered problems when Voronoi diagrams are used. For the L-shape problem, the use of Voronoi diagrams has no significant impact on the error. It can be concluded that the use of Voronoi diagrams does not allow a significant error reduction for the considered node arrangements, and their use is not recommended in the general case.

5. The stabilization method reduces the error for the L-shape problem by respectively 25% and 35% for the GFD and DC PSE methods. However, we observe a large error increase (up to a factor 30) for both methods for the pressurized cylinder problem. This difference is due to the type of boundary conditions imposed. The stabilization method appears more suitable to Dirichlet loaded problems than to Neumann loaded problems.

Chapter 6

Smart cloud adaptivity

6.1 General

Discretization adaptivity has always been of interest in numerical methods. The performance of this approach in the framework of point collocation was shown in many articles for the GFD [34, 73, 74, 75] or the RBF-FD [76, 77, 78] methods for instance.

In contrast to existing work, we present in this chapter a smart cloud adaptive collocation scheme that uses the exact definition of a given geometry based on a CAD file. The scheme can be applied to most domains with the assurance that the exactness of the geometry is not lost as part of the refinement process. We use 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.

We show in this chapter 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. We focus on linear elasticity problems (2D and 3D) using the Generalized Finite Difference method. The method can also be applied to other types of elliptic problems, to the DC PSE method and to most point collocation methods.

We present in Section 6.2 two error indicators that we used to identify the zones where the error is the greatest. Then, we show in Section 6.3 how error indicators are used to refine locally the domain and improve the convergence rate of the solution. Finally we apply the proposed method to practical 2D and 3D problems in Section 6.4.

6.2 Error indicators

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

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

We use the term "error indicator" rather than the term "error estimator". The proposed 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. We describe the considered indicators in the subsections below.

6.2.1 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 [79] and extended by Bordas and Duflot [80, 81] and Rodenas et al. [82] 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.

We compute 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 collocation node.

More specifically, considering a domain Ω, we compare the von Mises stress obtained at a collocation node $\mathbf{X_c}$ (denoted $\sigma^c_{\mathsf{vM}}\left(\mathbf{X_c}\right)$) to a moving least square approximation of the von Mises stress field at the collocation node $\mathbf{X_c}$ (denoted $\sigma^s_{\mathsf{vM}}\left(\mathbf{X_c}\right)$). We determine the moving least square approximation based on the von Mises stress $\sigma^c_{\mathsf{vM}}\left(\mathbf{X_{pi}}\right)$ computed at the support nodes $\mathbf{X_{pi}}$ of the collocation node $\mathbf{X_c}$ (see Figure 6.1).

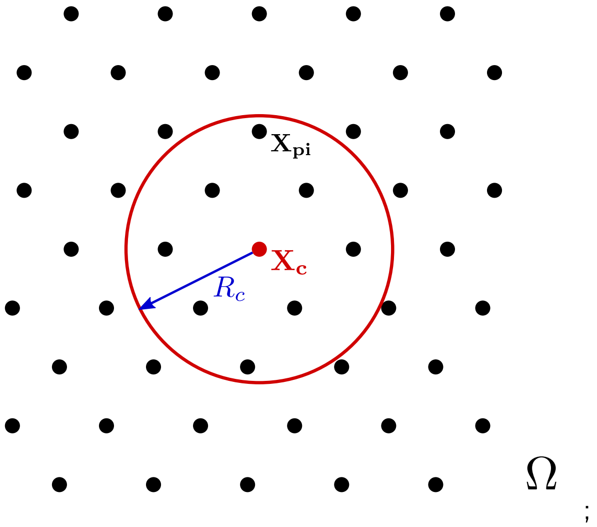


Figure 6.1: Discretization of a portion of a domain Ω and identification of the nodes $\mathbf{X_{pi}}$, located within a support of radius R_c of a collocation node $\mathbf{X_c}$, involved in the computation of the smooth von Mises field $\sigma^s_{\mathsf{vM}}\left(\mathbf{X_c}\right)$.

We used 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, to compute the moving least square approximation. 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:

$$\mathbf{p}\left(\mathbf{X}, \mathbf{X_c}\right) = \begin{bmatrix} 1 \\ (x - x_c) \\ (y - y_c) \\ (x - x_c)^2 \\ (x - x_c)(y - y_c) \\ (y - y_c)^2 \end{bmatrix}. \tag{6.1}$$

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

$$\sigma^s_{\mathsf{vM}}\left(\mathbf{X}, \mathbf{X_c}\right) = \mathbf{p}\left(\mathbf{X}, \mathbf{X_c}\right)^T \mathbf{a}\left(\mathbf{X_c}\right). \tag{6.2}$$

We compute the coefficients $\mathbf{a}\left(\mathbf{X_c}\right)$ that 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 $B\left(\mathbf{X_c}\right)$ presented in Equation (6.3). We weight the error 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.

$$B\left(\mathbf{X_c}\right) = \sum_{i=1}^{m} w\Big(\mathbf{X_{pi}}, \mathbf{X_c}\Big) \left(\mathbf{p}\left(\mathbf{X_{pi}}, \mathbf{X_c}\right)^T \mathbf{a}\left(\mathbf{X_c}\right) - \sigma^c\left(\mathbf{X_{pi}}\right)\right)^2. \tag{6.3}$$

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

$$\frac{\partial B\left(\mathbf{X_c}\right)}{\partial \mathbf{a}\left(\mathbf{X_c}\right)} = 0. \tag{6.4}$$

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

$$\mathbf{A}\left(\mathbf{X_c}\right)\mathbf{a}\left(\mathbf{X_c}\right) = \mathbf{E}\left(\mathbf{X_c}\right)\mathbf{f}\left(\mathbf{X_c}\right). \tag{6.5}$$

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

and $\mathbf{f}\left(\mathbf{X_c}\right)$ are:

$$\mathbf{A}\left(\mathbf{X_c}\right)=\begin{bmatrix} m_{11} & m_{12} & \dots & m_{1q} \\ m_{21} & m_{22} & \dots & m_{2q} \\ \vdots & & & \vdots \\ m_{q1} & m_{q2} & \dots & m_{qq} \end{bmatrix} \in \mathbb{R}^{q\times q}, \tag{6.6}$$

$$\mathbf{E}\left(\mathbf{X_c}\right)=\begin{bmatrix} w(\mathbf{X_{p1}},\mathbf{X_c})\mathbf{p}\left(\mathbf{X_{p1}},\mathbf{X_c}\right)_1 & \dots & w(\mathbf{X_{pm}},\mathbf{X_c})\mathbf{p}\left(\mathbf{X_{pm}},\mathbf{X_c}\right)_1 \\ w(\mathbf{X_{p1}},\mathbf{X_c})\mathbf{p}\left(\mathbf{X_{p1}},\mathbf{X_c}\right)_2 & \dots & w(\mathbf{X_{pm}},\mathbf{X_c})\mathbf{p}\left(\mathbf{X_{pm}},\mathbf{X_c}\right)_2 \\ \vdots & & \vdots \\ w(\mathbf{X_{p1}},\mathbf{X_c})\mathbf{p}\left(\mathbf{X_{p1}},\mathbf{X_c}\right)_q & \dots & w(\mathbf{X_{pm}},\mathbf{X_c})\mathbf{p}\left(\mathbf{X_{pm}},\mathbf{X_c}\right)_q \end{bmatrix} \in \mathbb{R}^{q\times m}, \tag{6.7}$$

$$\mathbf{f}\left(\mathbf{X_c}\right)=\begin{bmatrix} \sigma^c\left(\mathbf{X_{p1}}\right) & \sigma^c\left(\mathbf{X_{p2}}\right) & \dots & \sigma^c\left(\mathbf{X_{pm}}\right) \end{bmatrix}^T, \tag{6.8}$$

where

$$m_{ij}=\sum_{k=1}^{m} w\Big(\mathbf{X_{pk}},\mathbf{X_c}\Big)\mathbf{p}\left(\mathbf{X_{pk}},\mathbf{X_c}\right)_i\mathbf{p}\left(\mathbf{X_{pk}},\mathbf{X_c}\right)_j. \tag{6.9}$$

$\mathbf{p}\left(\mathbf{X_{pk}},\mathbf{X_c}\right)_i$ refers to the i^{th} component of the vector $\mathbf{p}\left(\mathbf{X_{pk}},\mathbf{X_c}\right)$.

The solution of Equation (6.5) at $\mathbf{X_c}$ allows the computation of $\sigma^s\left(\mathbf{X_c}\right)$ and of an error indicator $e\left(\mathbf{X_c}\right)$ as follows:

$$e\left(\mathbf{X_c}\right)=\left|\sigma^c\left(\mathbf{X_c}\right)-\sigma^s\left(\mathbf{X_c}\right)\right|. \tag{6.10}$$

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

6.2.2 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 reference [83, 84]. In this thesis, we select the corners of the Voronoi cell associated to a considered collocation node $\mathbf{X_c}$ to compute the residual of the PDE.

We calculate the residual error at a collocation node $\mathbf{X_c}$ as the average of the residual of the PDE at each Voronoi corner points $\mathbf{X_{vi}}$ of the Voronoi cell associated with $\mathbf{X_c}$ (see Figure 6.2).

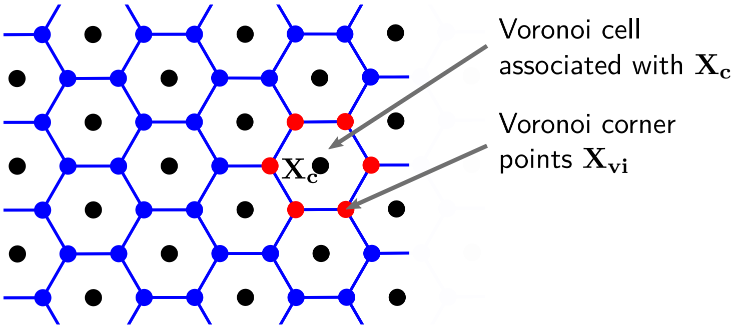

Voronoi cell associated with $\mathbf{X_c}$
Voronoi corner points $\mathbf{X_{vi}}$
$\mathbf{X_c}$

Figure 6.2: 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 with $\mathbf{X_c}$.

We compute the residual at each Voronoi center $\mathbf{X_{vi}}$ using the solution at the neighboring collocation nodes based on 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 $e\left(\mathbf{X_c}\right)$ as follows:

$$e\left(\mathbf{X_c}\right) = \frac{1}{q}\sum_{i=1}^{q}\left|\boldsymbol{\nabla}\cdot\boldsymbol{\sigma}\left(\mathbf{X_{vi}}\right)+\mathbf{b}\left(\mathbf{X_{vi}}\right)\right|. \tag{6.11}$$

6.2.3 Parameter Variation and Indicators Comparison

We compare in this subsection the error obtained from the indicators presented in Subsections 6.2.1 and 6.2.2. 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 use two 2D problems for benchmarking purposes: the infinite plate with an elliptical hole and the infinite body with a cylindrical hole. The models, boundary conditions and exact solutions for these problems are presented in Section 1.4.

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

140,000 nodes for the body with a cylindrical hole.

The exact error for the regular nodes arrangements presented in Section 3.2 (i.e. square lattice and triangular lattice for 2D problems) are presented in Figure 6.3 for the problem of a plate with an elliptical hole and in Figure 6.4 for the problem of a body with a cylindrical 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 Figure 1.4.2 (c))). 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 Figure 1.4.2 (b)).

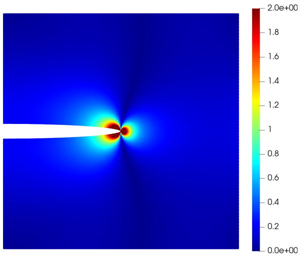
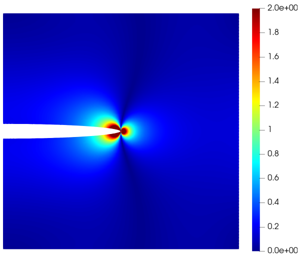
	Square lattice	**Triangular lattice**
Exact Error σ_{vM}		

Figure 6.3: Exact von Mises stress error for the problems of an infinite 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.

We compare in Figure 6.5 and Figure 6.6 the error patterns for various parameters of the ZZ-type indicator and for the residual-type error indicator, respectively, for the two benchmark problems. We compare the results obtained for an "unweighted" indicator (i.e. $w(\mathbf{X}, \mathbf{X_c}) = 1$ in Equation (6.3)) and for a "weighted" indicator. We selected a 4th order spline for the radial basis

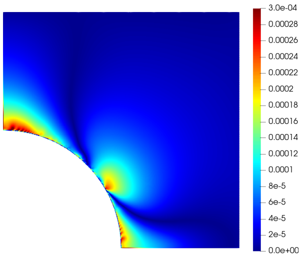
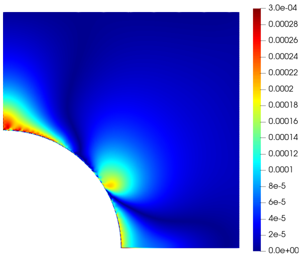
	Square lattice	**Triangular lattice**
Exact Error σ_{vM}		

Figure 6.4: Exact von Mises stress error for the problems of an infinite body with a cylindrical 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.

function w in Equation (6.3). The equation of the spline is presented in Equation (6.12).

$$w(s) = \begin{cases} 1 - 6s^2 + 8s^3 - 3s^4 & \text{if } s \leq 1 \\ 0 & \text{if } s > 1. \end{cases} \tag{6.12}$$

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:

$$w(\mathbf{X}, \mathbf{X_c}) = w(s) \quad \text{with} \quad s = \frac{\|\mathbf{X} - \mathbf{X_c}\|_2}{R_c}. \tag{6.13}$$

In Figures 6.5 and 6.6 we also compare the results obtained from a "indirect" computation of the indicator from the von Mises stress field to results obtained from an "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 $\sigma^e_{vM}(\mathbf{X_{vi}})$ at each support nodes $\mathbf{X_{vi}}$ of $\mathbf{X_c}$ to compute $\sigma^s_{vM}(\mathbf{X_c})$. The figures show the result for both the unweighted and weighted configurations, and both the direct and indirect computation methods. We show the results for each of these configurations 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. We set the amplitude of the scale 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 observe lines where the computed error indicator is lower. This phenomenon is the most significant for the square lattice discretization techniques, for the case of a ZZ-type error indicator computed using an "indirect" computation of σ^s_{vM}. Such lines are not observed for the residual-type error indicator computed for the domain discretized using a triangular lattice. We also see that the presence of weights in the computation of the ZZ-type error indicator has little impact on the results. The "direct" computation of σ^s_{vM} leads to better results than an "indirect" computation of σ^s_{vM} in the sense that the error indicator appears less affected by the discretization strategy.

The trend of the results obtained for the problem of a body with a cylindrical 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. We notice however that the ZZ-type error indicator allows for the identification of a zone, close to the cavity, where the error is greater. This zone is not identified with the residual-type error indicator. The exact error presented in Figure 6.4 shows that this zone corresponds to a zone where the error is significant.

The trend of the results presented in figures 6.5 and 6.6 is in favor of the residual-type indicator because this indicator appears to be the least affected by the discretization of the geometry. Among the different settings of the proposed ZZ-type indicator, the weighted - direct computation of σ^s_{vM}, combined with a discretization of the domain based on a triangular lattice, is the configuration that appears to be the least affected by the discretization of the geometry. We further investigated the impact of other parameters of the method for this configuration of the indicator.

The computation of the ZZ-type error indicator necessitates the selection of a stencil size or radius since we consider the distance criterion for the selection of the stencil nodes. The results presented in Figure 6.5 and Figure 6.6 are computed based on the same stencil as the stencil considered for the solution of the collocation problem. We present in Figure 6.7 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 selected domains discretized based on a triangular lattice and a direct computation of σ_{vM}^{s} considering a weighted moving least square approximation.

The results presented in Figure 6.7 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 body with a cylindrical hole. We observe that the intensity of some zones where the error is low 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 Figure 6.6).

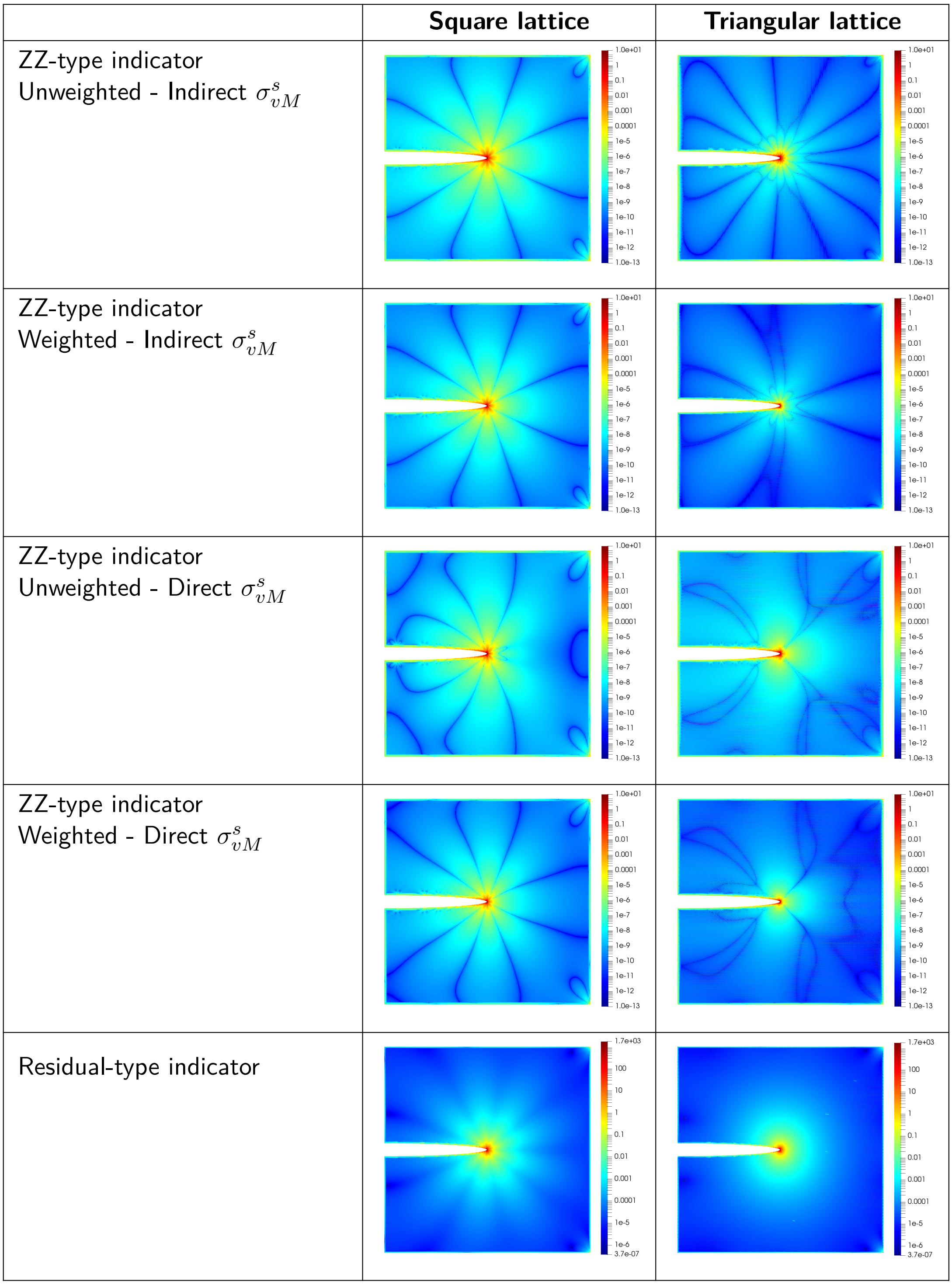


Figure 6.5: Comparison of the error pattern for ZZ-type error indicators computed with various parameters and for a residual-type error indicator for the problem of a plate with an elliptical hole. The results are shown for square and triangular lattice discretizations of the interior of the domain.

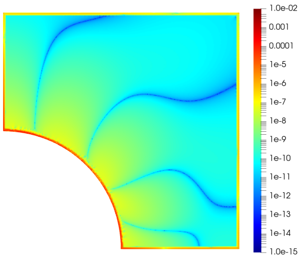
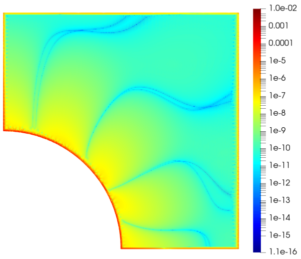
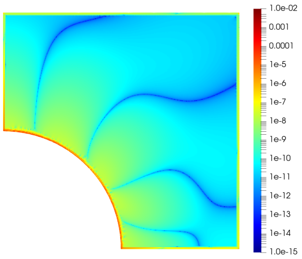
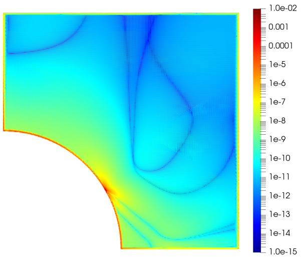
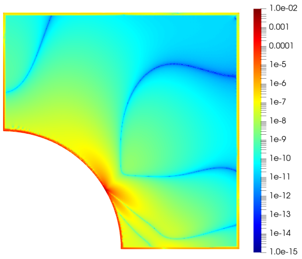
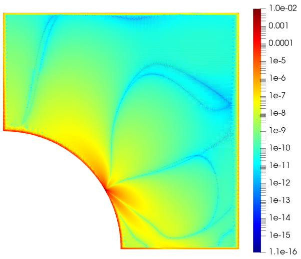
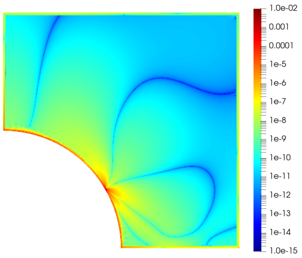
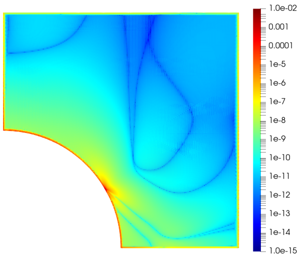
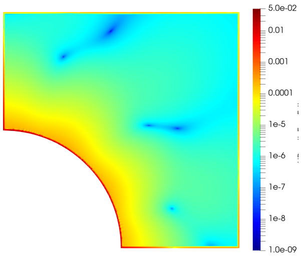
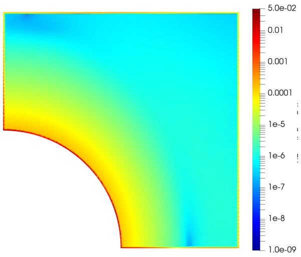
	Square lattice	**Triangular lattice**
ZZ-type indicator Unweighted - Indirect σ^s_{vM}		
ZZ-type indicator Weighted - Indirect σ^s_{vM}		
ZZ-type indicator Unweighted - Direct σ^s_{vM}		
ZZ-type indicator Weighted - Direct σ^s_{vM}		
Residual-type indicator		

Figure 6.6: Comparison of the error pattern for ZZ-type error indicators computed with various parameters and for a residual-type error indicator for the problem of a body with a cylindrical hole. The results are shown for square and triangular lattice discretizations of the interior of the domain.

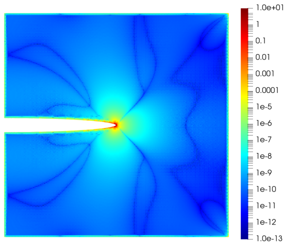
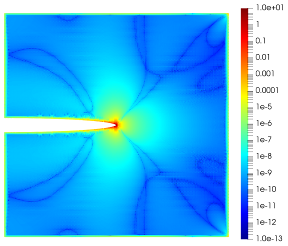
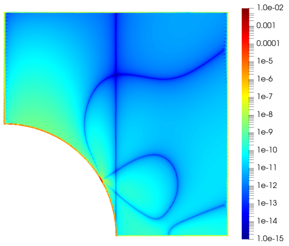
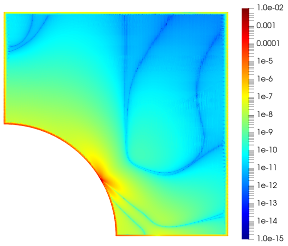
	Stencil Scaling Factor = 0.8	**Stencil Scaling Factor = 1.5**
Plate with an elliptical hole ZZ-type indicator Weighted - Direct σ_{vM}^{s}		
Body with a cylindrical hole ZZ-type indicator Weighted - Direct σ_{vM}^{s}		

Figure 6.7: 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.

6.2.4 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 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 Figures 6.3 and 6.4.

The weighted - direct computation of σ_{vM}^{s} 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 the computational cost of 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 Figure 6.8. 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 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 then, we solved the linear system using a LU factorization based one thread and one process. We observe from Figure 6.8 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 body with a cylindrical 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 body with a cylindrical 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. We

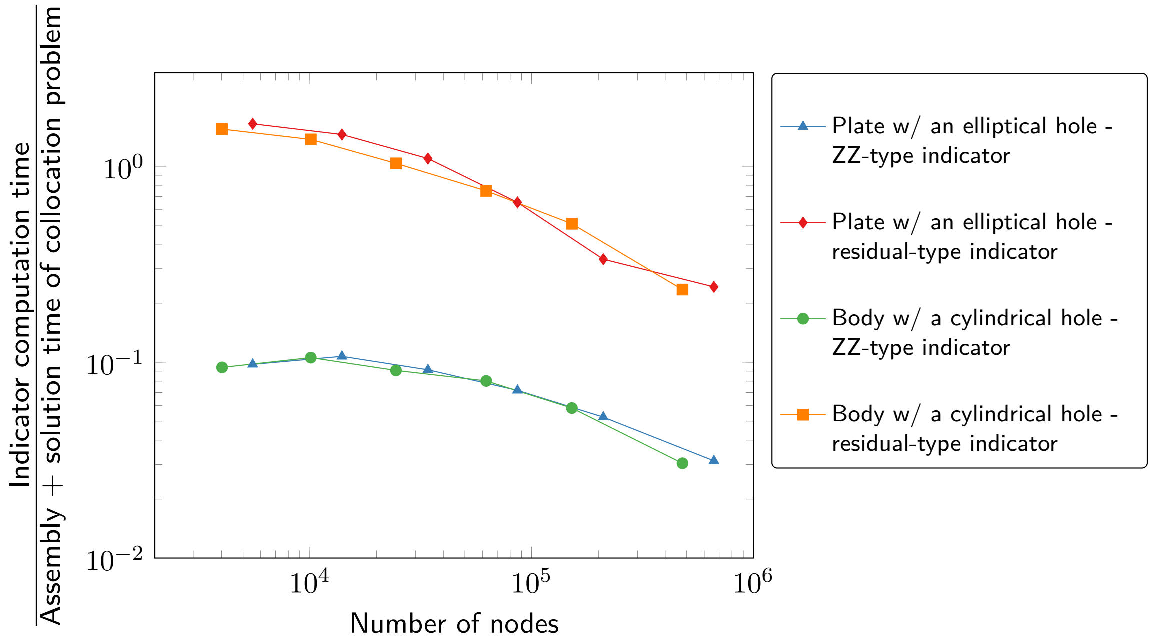


Figure 6.8: Comparison of the ratio of the indicator computation time to the time needed to assemble and solve the collocation problem. Both problems lead to similar results. The computation time of the residual-type error indicator is approximately 10 times the computation time of the ZZ-type error 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.

present the results in Figure 6.9 and Figure 6.10. 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 the reference solution. We also computed the L_2 weighted error norm of the residual-type indicator computed using Equation (6.14). We selected this norm since a reference solution is not computed as part of the indicator.

$$L_2W(e) = \frac{\sqrt{\sum_{k=1}^{n} \left(e\left(\mathbf{X_c}\right)\right)^2}}{n}. \tag{6.14}$$

We observe from the results presented in Figure 6.9 and in Figure 6.10 a convergence of both error indicators for both problems. The convergence rate is higher for the problem of a body with a cylindrical 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 body with a cylindrical hole. This result is expected since the solution is smooter 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.

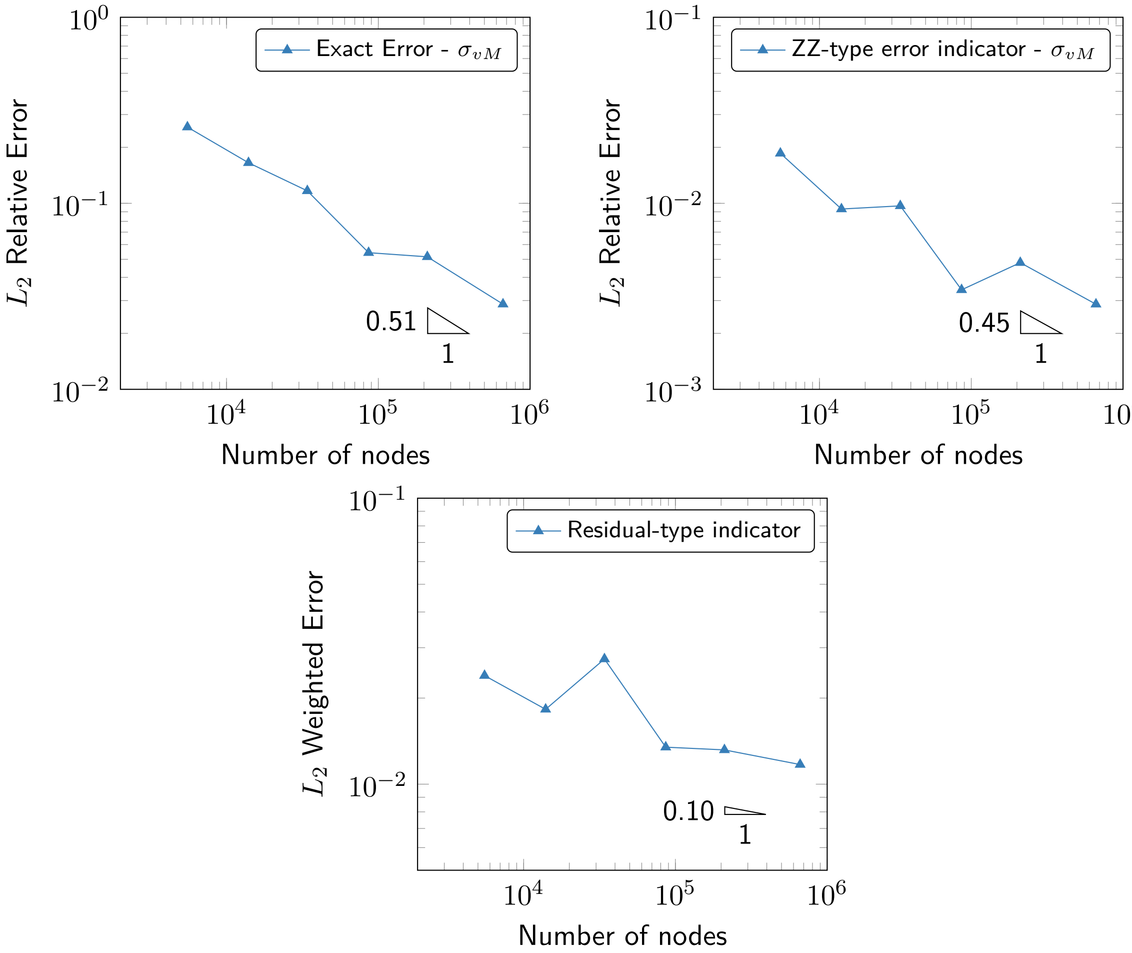


Figure 6.9: Plate with an elliptical hole - Comparison of the exact and estimated error. The results are presented in terms of the L_2 relative error norm for the exact error on the von Mises stress and for the ZZ-type indicator of the error on the von Mises stress. The results are presented in terms of the L_2 weighted error norm for the residual-type indicator.

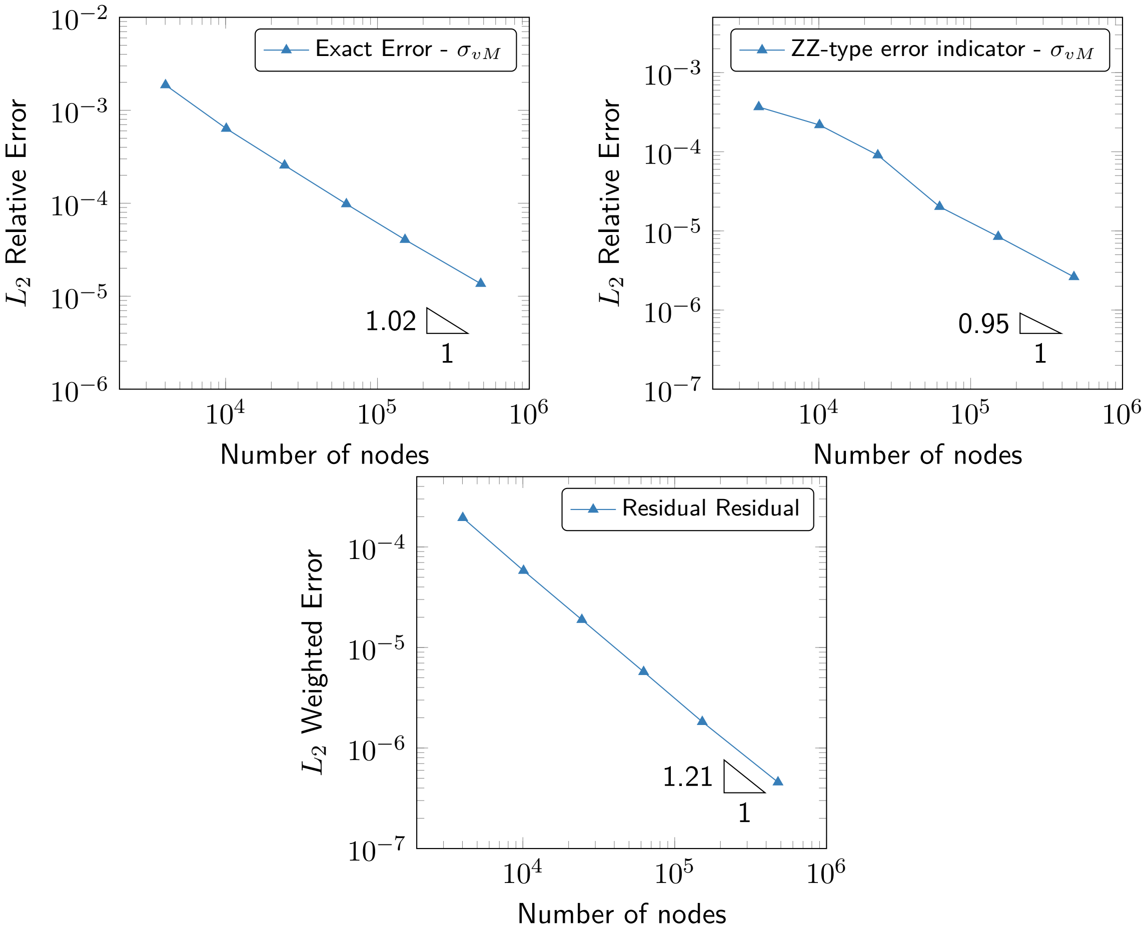


Figure 6.10: Body with a cylindrical hole - Comparison of the exact and indicative error. The results are presented in terms of the L_2 relative error norm for the exact error on the von Mises stress and for the ZZ-type indicator of the error on the von Mises stress. The results are presented in terms of the L_2 weighted error norm for the residual-type indicator.

Based on the results presented in this subsection, we select 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 prefer this indicator rather than the residual-type error indicator because of its reduced computational expense and because of its simplicity.

6.3 Discretization refinement

We describe in this section a scheme to refine a point cloud based on the results obtained from an *a posteriori* error indicator. 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 [85], Davydov [76], Gavete [74] or Slak and Kosec [86] within the framework of point collocation. h-adaptivity has also been considered in literature based on geometric indicators [87, 88]. 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 [89, 90]. Jancic et al. showed the benefits of p-refinement for a Poisson problem with a strong source within the domain [91] in the framework of the radial basis function-generated finite difference (RBF-FD) method.

The scheme presented in this section 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 follow are presented in the form of a pseudo code in Figure 6.11.

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.

```
$h$-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
```

Figure 6.11: h-adaptivity algorithm considered for the presented adaptive refinement method.

6.3.1 Identification of the refinement areas

The error indicators presented in Section 6.2 are computed at the collocation nodes. Therefore, we decide 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 present the computed error indicator, sorted from the lowest error to the highest error in Figure 6.12 for the considered benchmark problems. We added vertical bars to the figure to visualize the percentage of nodes in the different error zones. The presented error indicators are ZZ-type error indicators, computed based on the parameters presented in Section 6.2.

We observe three distinct zones from the results presented in Figure 6.12. A limited number of

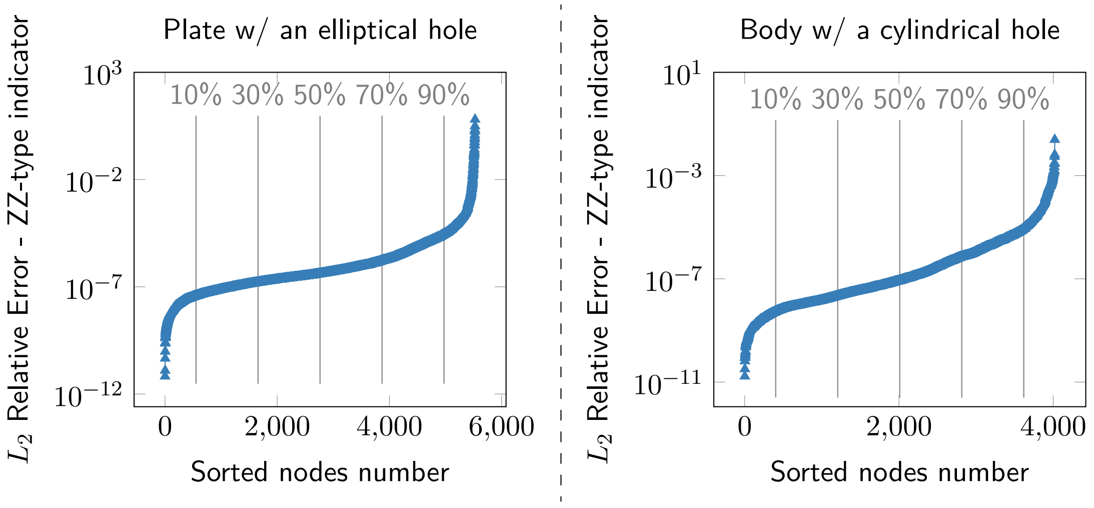


Figure 6.12: Distribution of the error in terms of ZZ-type error indicator for the problem of a plate with an elliptical hole (left) and for the problem of a body with a cylindrical hole (right). Three distinct zones are observed on these graphs. Less than 10% of the nodes have a very low error (nearly 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 body with a cylindrical hole). Less than 10% of the nodes have an error much larger than the other nodes.

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 an between 10^{-8} and 10^{-5} for the body with a cylindrical 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 body with a cylindrical hole, respectively. The zones of low and high error represent each approximately 5% of the total number of nodes. Based on the these observations, we decide 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 errors are selected for local refinement. The impact of the selected threshold on the convergence rate is analyzed in Subsection 6.3.4. The selected approach has similarities with the Dörfler marking strategy [92] used in the framework of adaptive finite element method [93].

In order to obtain as smooth a refinement pattern as possible, we also select, for local refinement, all the stencil nodes associated with the selected nodes of highest error. The results presented in Figure 6.13 show the benefits of this approach for the problems of a plate with an elliptical hole and for the body with a cylindrical hole. We present 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.

6.3.2 Placement of new nodes

We presented in Subsection 6.3.1 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 Section 4.4. Those elements are used as part of the refinement process to facilitate the placement of new nodes far from existing nodes. The approach for 2D and 3D problems is different. We present both approaches below.

In 2D, the new nodes are added in the center of all the elements, which have not already been refined, connected to the boundary nodes marked for local refinement. The new nodes are then projected onto the boundary of the domain using the exact geometry from the CAD file. The refined surface elements are split into two new surface elements. The element normal vectors are then computed.

In 3D, the new nodes are added in the center of the edges of all the elements, which have not

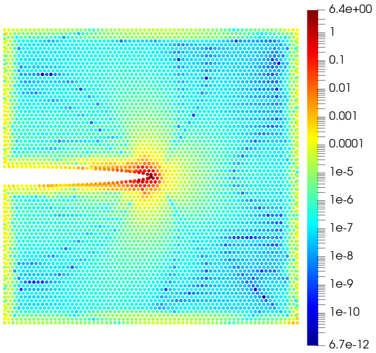
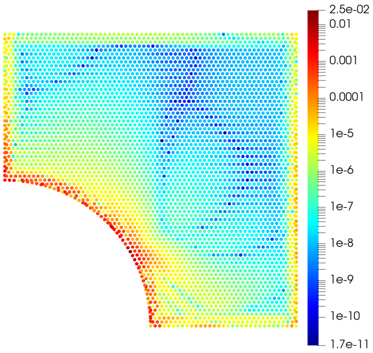
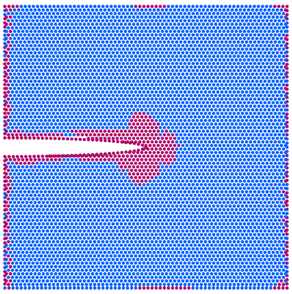
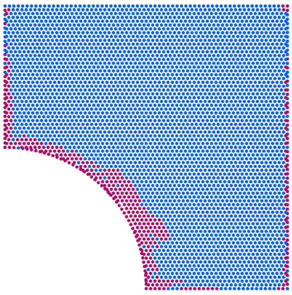
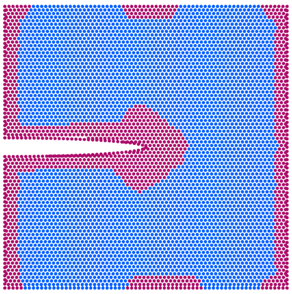
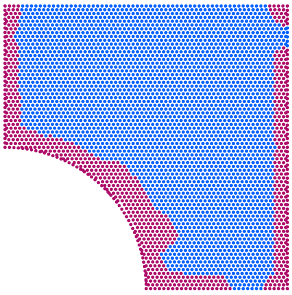
	Plate with an elliptical hole	**Body with a cylindrical hole**
ZZ-type indicator Weighted - Direct σ^s_{vM}		
Marked collocation nodes (in red) based on a threshold criteria corresponding to 10% of the nodes of highest error		
Marked collocation nodes (in red) based on a threshold criteria corresponding to 10% of the nodes of highest error and the associated stencil nodes		

Figure 6.13: 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 body with a cylindrical hole.

already been refined, connected to the boundary nodes marked for local refinement. The new nodes are then projected onto the boundary faces or edges of the domain. The refined surface elements are split into four new surface elements. The element normal vectors are computed.

Figure 6.14 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 Figure 6.15. 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 surfaces, marked ① and ② respectively. A projection on a gray surface would lead to a node outside of the domain.

To facilitate the projection process, the references 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 Vornoi 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 Figure 6.16.

Voronoi diagrams are computed at all the collocation nodes marked for local refinement. The

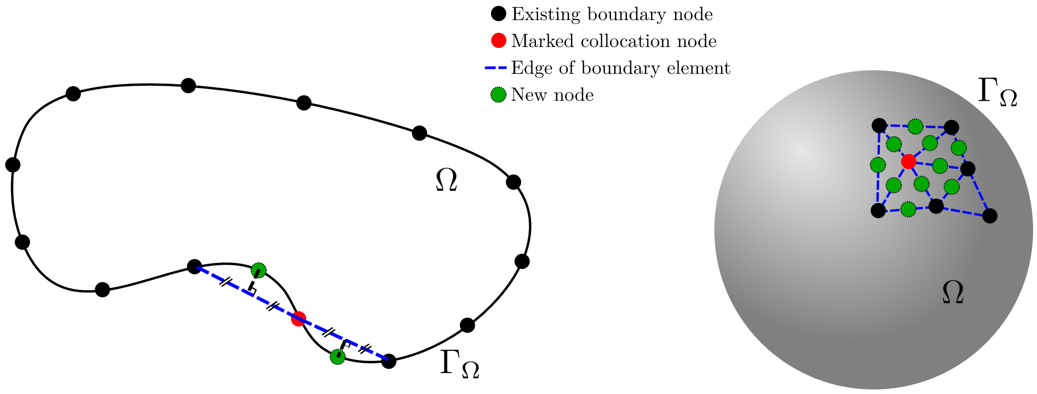


Figure 6.14: Computation of the position of new boundary nodes. Collocation nodes marked for local refinement are shown in red color. New boundary nodes, shown in green color bounded by dotted lines, are added in the middle of the edges of the elements connected to the collocation node and projected onto the surface or edge of the domain. The obtained nodes are part of the refined discretization.

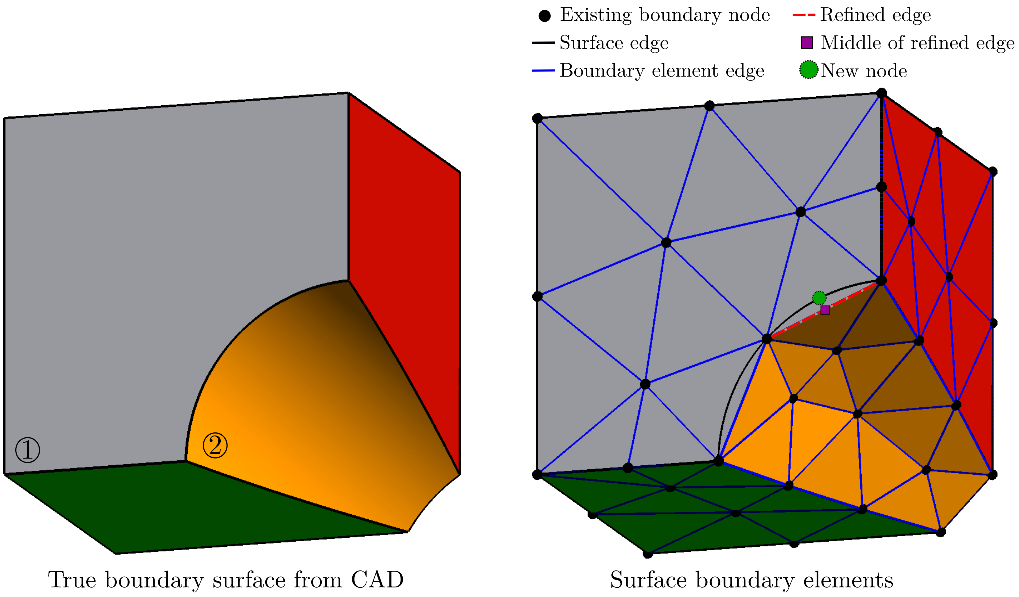
existing nodes of the discretization and the new boundary nodes are considered in the computation of the Voronoi diagrams. Only the Voronoi cells around the marked collocation nodes are of interest. All the corners of the selected cells are added to a list of new candidate nodes. The duplicated nodes 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 from the initial discretization. This minimum distance is denoted h_{Loc}. The minimum allowed inter-node distance, denoted h_{Min}, is the ratio of h_{Loc} to a factor α:

$$h_{\mathsf{Min}} = \frac{h_{\mathsf{Loc}}}{\alpha}. \tag{6.15}$$

We select $\alpha = 3$ in this thesis.



True boundary surface from CAD

Surface boundary elements

Figure 6.15: 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.

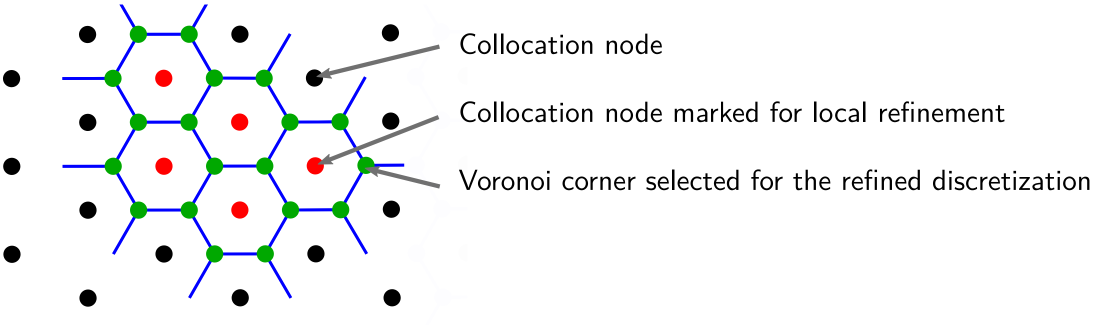

Collocation node
Collocation node marked for local refinement
Voronoi corner selected for the refined discretization

Figure 6.16: Computed Voronoi diagram for a set of collocation nodes - Use for new node selection.

6.3.3 Generation of the updated collocation model

We presented in Section 6.3.2 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 3.3. The boundary nodes of the smart cloud have references to their parent edge or surface and also references to the boundary conditions associated with these entities. New boundary nodes can be located at the intersection of multiple surfaces. In this case, we apply first non-zero Neumann boundary conditions, then Dirichlet boundary conditions and, finally, homogeneous Neumann boundary conditions.

6.3.4 Node selection threshold sensitivity

We assess in this section the impact of the node selection threshold on the convergence of the error using smart cloud adaptivity. We consider node selection threshold values between 0.02 and 0.2 based on the trend of the results presented in Figure 6.12. We present the results in Figure 6.17 and Figure 6.18 for the two benchmark problems considered in this chapter (i.e. the infinite plate with an elliptical hole and the infinite body with a cylindrical hole). We compare the results in terms of "exact" L_2 relative error norm and in terms of "indicative" L_2 relative error norm. We also compare the results obtained from the adaptive refinement technique presented in this chapter 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 body with a cylindrical 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 those obtained using a global refinement strategy. We selected a threshold of 0.05 for the problems solved in the next sections of the chapter. This value is a good compromise between rapid convergence of the solution and non monotonic error reduction.

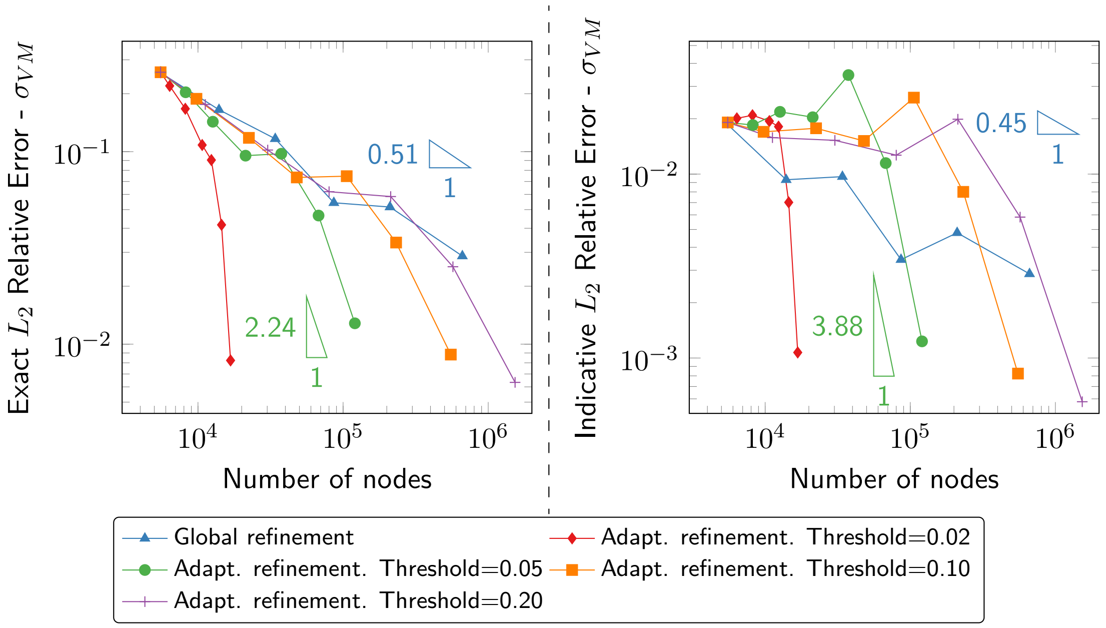


Figure 6.17: Exact and indicative L_2 relative error norm for the problem of a plate with an elliptical hole. Results from adaptive refinement based on a ZZ-type error indicator for adaptive threshold ratio between 0.02 and 0.20 are compared results obtained from a global refinement of the domain discretization.

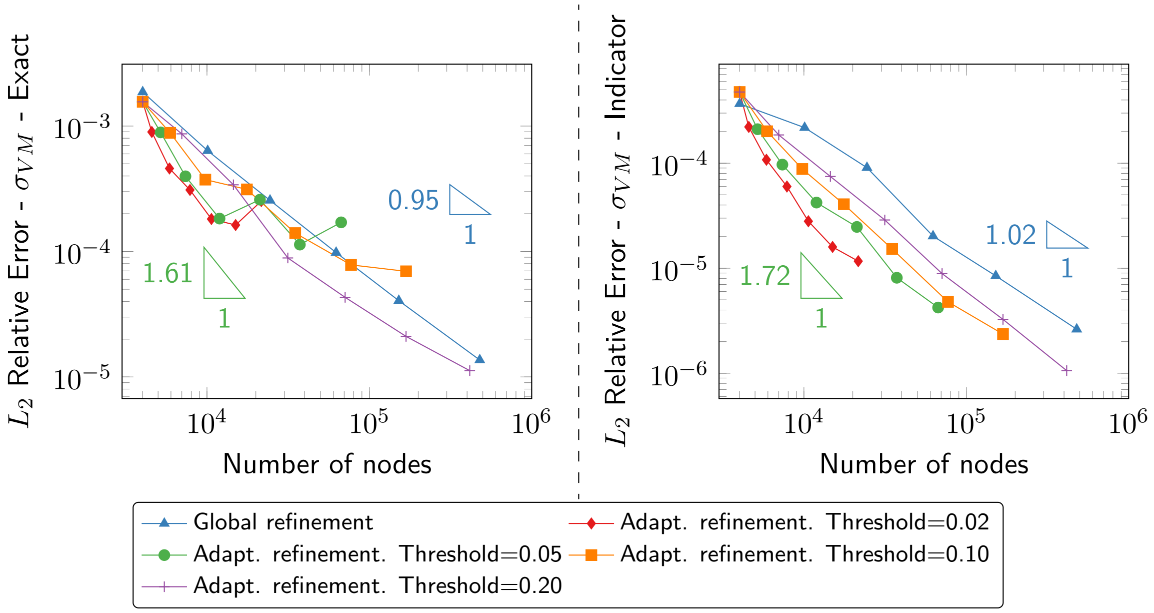


Figure 6.18: Exact and indicative L_2 relative error norm for the problem of a body with a cylindrical hole. Results from adaptive refinement based on a ZZ-type error indicator for adaptive threshold ratio between 0.02 and 0.20 are compared results obtained from a global refinement of the domain discretization.

6.3.5 Node relaxation

The addition of new nodes in the domain leads to regions of relatively high node density close to regions of relatively low node density. Node relaxation can be used to smooth-out the 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 use the relaxation method of the library Medusa for this purpose [94].

Rather than applying node relaxation to the entire domain, we prefer 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 Figure 6.19.

We show in Figure 6.20 and Figure 6.21 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. 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 thesis.

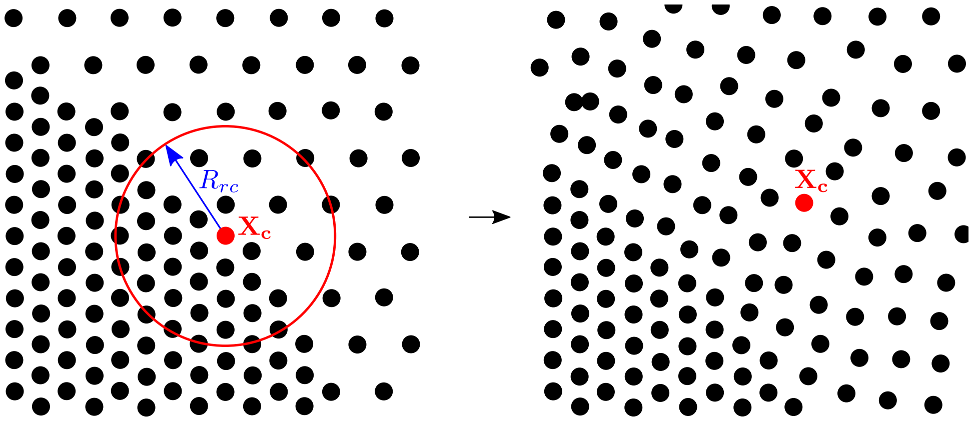


Figure 6.19: Point cloud before (left) and after (right) node relaxation. Node relaxation is performed locally around collocation nodes such as X_c over the domain of radius R_{rc}. Relaxation is performed when between 20% and 80% of the nodes within R_{rc} are located in “fine” discretization regions.

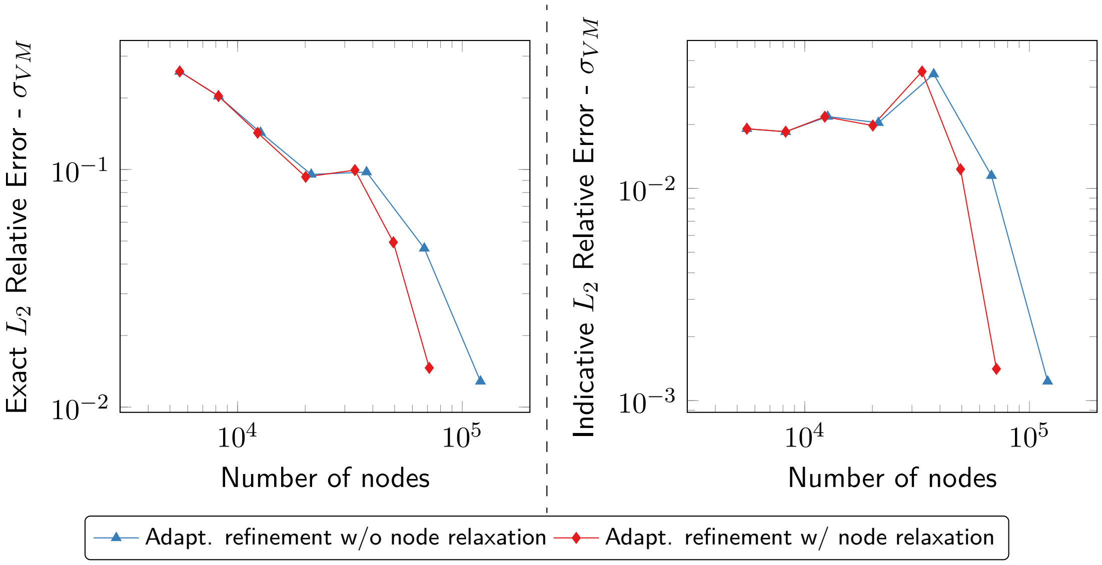


Figure 6.20: Comparison of the error with and without node relaxation for the problem of a plate with an elliptical hole. The exact and indicative L_2 relative error norms are shown in the left and right subfigures, respectively. We select a node selection threshold ratio of 0.05. The error obtained with node relaxation is lower than the one obtained without node relaxation. The trend of the results is similar for both cases.

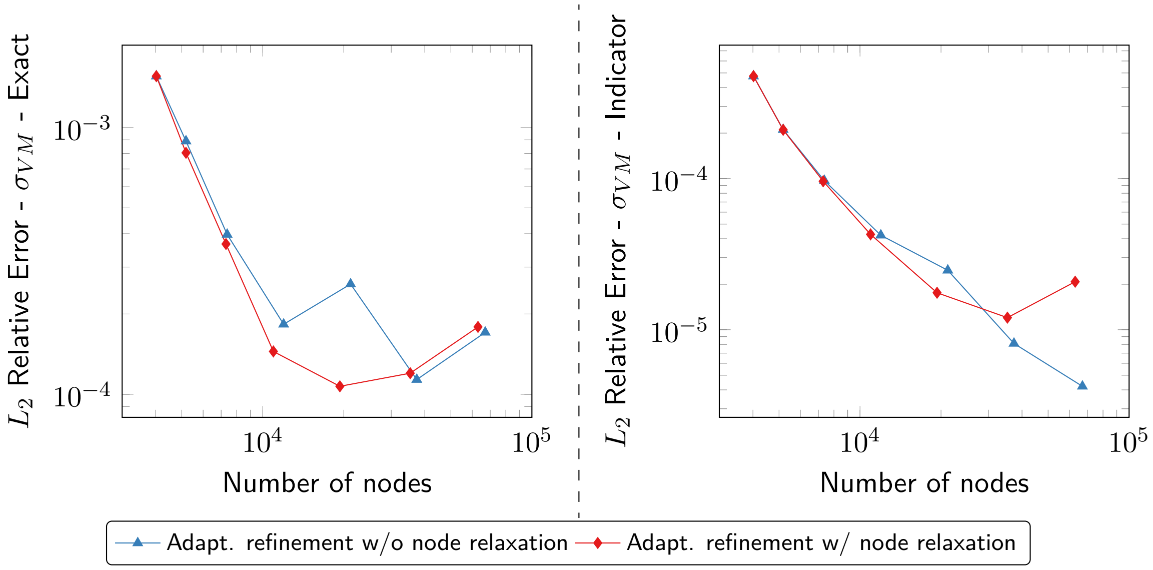


Figure 6.21: Comparison of the error with and without node relaxation for the problem of a body with a cylindrical hole. The exact and indicative L_2 relative error norms are shown in the left and right subfigures, respectively. We select a node selection threshold ratio of 0.05. The error obtained with 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.

6.4 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).

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, based on the approach introduced in Chapter 3. 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 adaptive refinement iterations. We used the ZZ-type error indicator, based on the parameters presented in Section 6.2, 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 6.3.

6.4.1 Gear coupled to a shaft

We present in Figure 6.22 the geometry of a gear, coupled to a shaft by a key that we considered (same problem as the one presented in Figure 4.24). 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 [67] for this purpose. We show the finite element reference solution to this problem in Figure 6.23. We observe three areas of stress concentration: the top left corner of the groove and both roots of the loaded tooth.

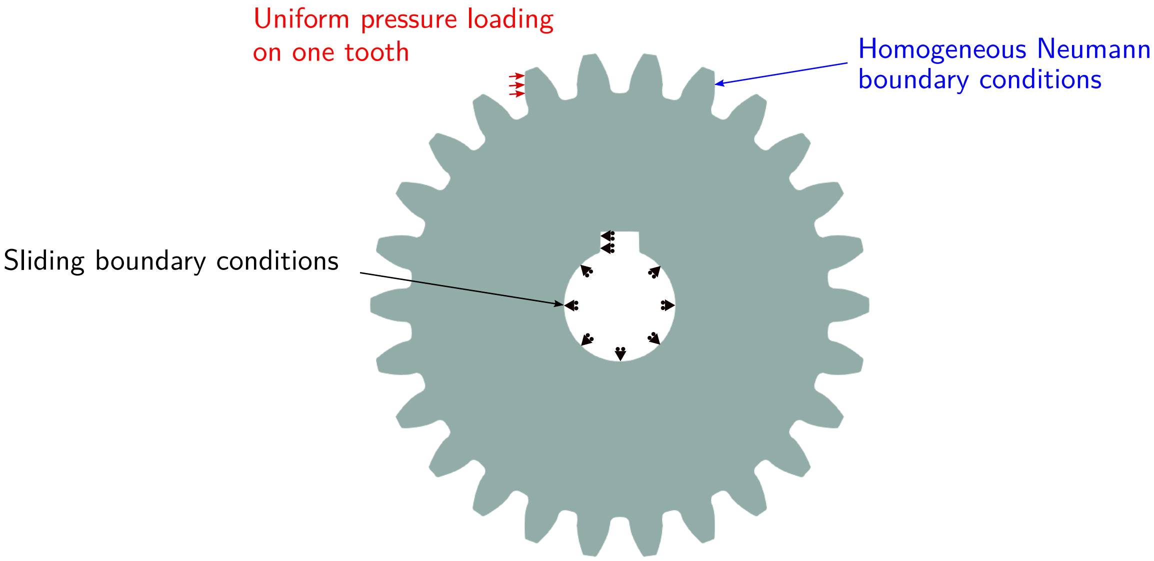
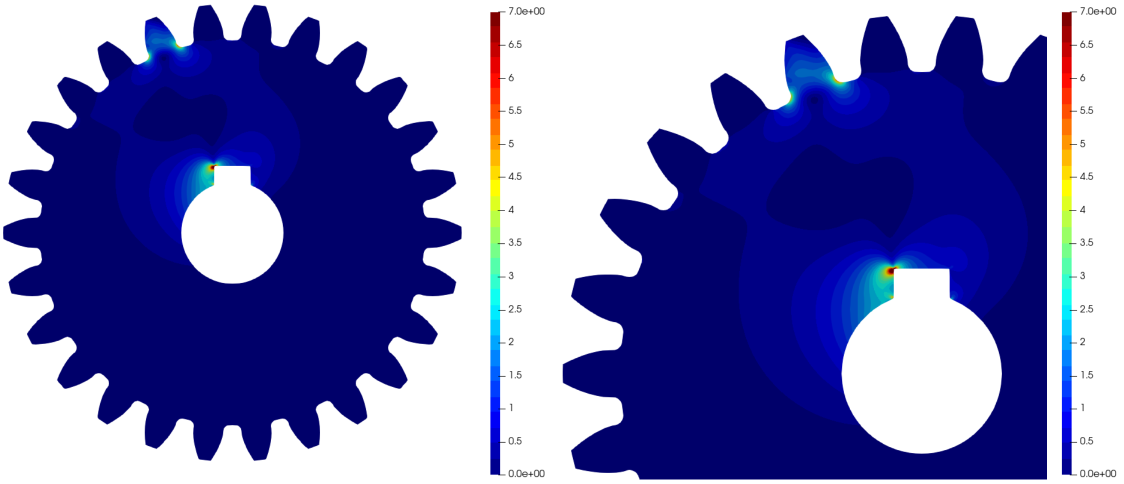


Figure 6.22: Boundary conditions applied to model a gear coupled to a shaft by a key.



(a) (b)

Figure 6.23: 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 show in Figure 6.24 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. We present the results 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.

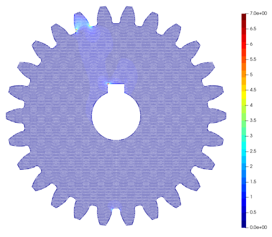
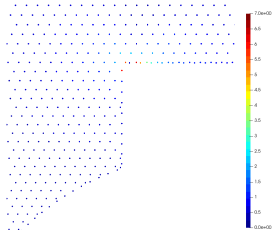
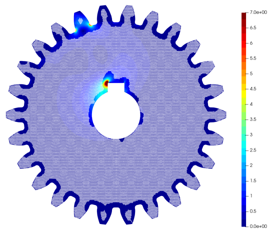
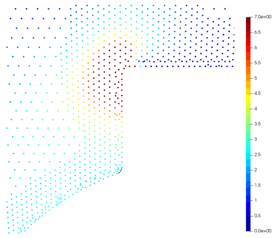
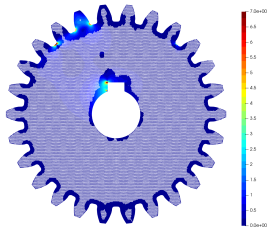
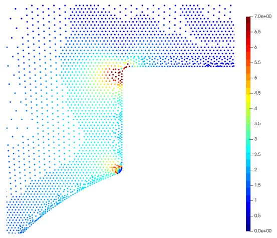
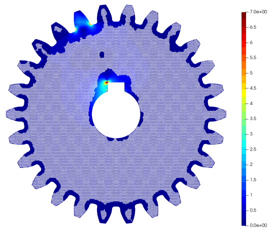
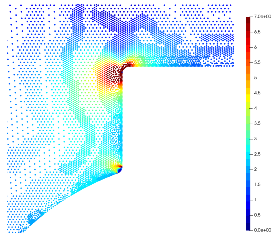
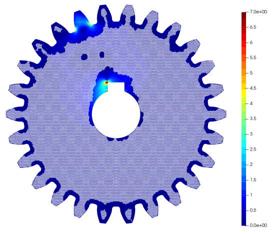
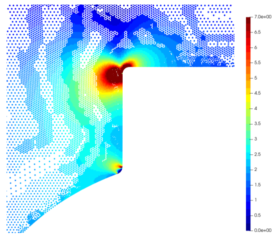
	Overall view	**Groove detail view**
Initial discretization 46,868 nodes		
Refinement iteration #1 59,544 nodes		
Refinement iteration #2 78,004 nodes		
Refinement iteration #3 104,527 nodes		
Refinement iteration #4 143,763 nodes		

Figure 6.24: 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.

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 Figure 6.23. We compare both solutions in greater detail in Figure 6.25. 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.

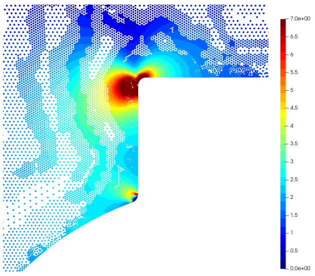
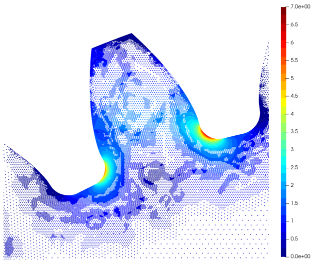
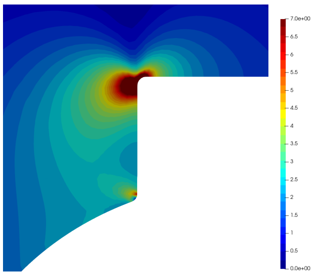
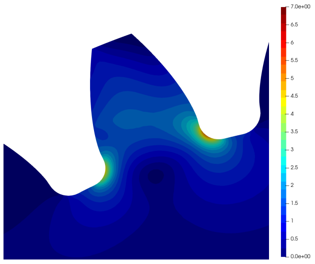
	Groove	**Loaded tooth**
Smart cloud collocation solution ▶ Refinement iteration #4 ▶ 143,763 nodes ▶ Approx. inter-node distance: 0.018 in dense regions 0.29 in coarse regions		
FEA solution ▶ 132,665 nodes ▶ Approx. inter-node distance: 0.15		

Figure 6.25: 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.

We present in Figure 6.26 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%.

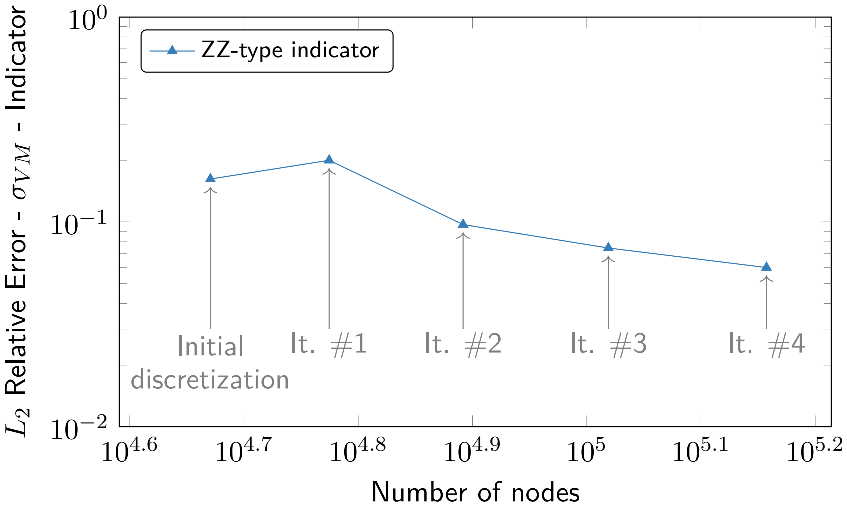


Figure 6.26: Evolution of the error indicator for the four iterations of adaptive refinement for the problem of the gear coupled to a shaft.

6.4.2 Closed cylinder subject to pressure

We present in Figure 6.27 the geometry of a closed cylinder subject to uniform pressure loading on the top of its closed end and the associated boundary conditions. The bottom of the closed cylinder is fixed in the direction normal to the closed end of the cylinder. We consider only a quarter of the cylinder to reduce the computational cost. We apply 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 Figure 6.28. 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. We show the position of the box in Figure 6.29. 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. We show the results for the stress range 0-37 for comparison purposes.

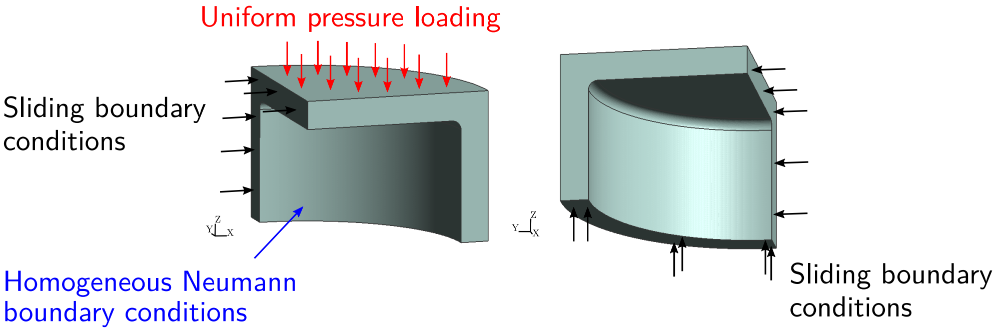

Uniform pressure loading
Sliding boundary conditions
Homogeneous Neumann boundary conditions
Sliding boundary conditions
Z
Y
X
Z
Y
X

Figure 6.27: 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.

We observe from Figure 6.28 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 Figure 6.30 the results obtained from the refined point cloud to a reference finite element solution obtained using code_aster [67]. We used a 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 solution. 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. Also, for the smart cloud collocation scheme, the strong form of the partial differential equation is solved. These aspects could explain the different stress concentration obtained from both models. We also observe that the stress field in the fillet is smoother for the results obtained with the smart cloud collocation scheme than for the results obtained with the finite element method. Both methods lead to similar results in terms of von Mises stress on the top side of the closed cylinder.

We present in Figure 6.31 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%. We observe a very slight increase of the error indicator between the first and second refinement steps.

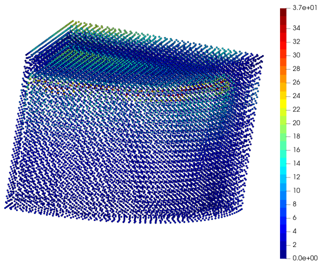
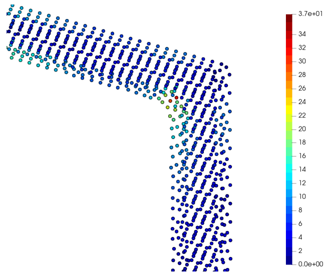
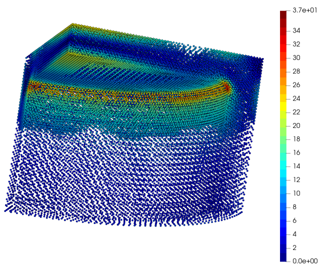
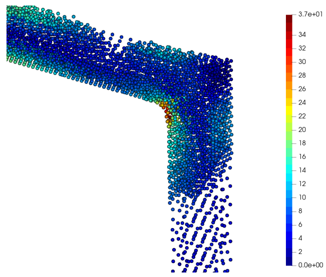
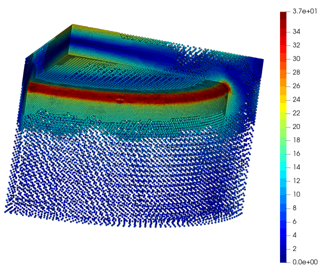
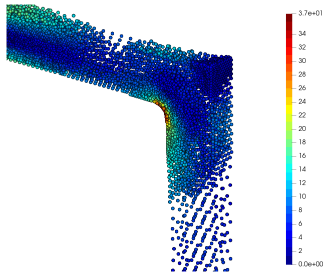
	Bottom view	**Slice**
Initial discretization 14,827 nodes	 3.7e+01 34 32 30 28 26 24 22 20 18 16 14 12 10 8 6 4 2 0.0e+00	 3.7e+01 34 32 30 28 26 24 22 20 18 16 14 12 10 8 6 4 2 0.0e+00
Refinement iteration #1 39,517 nodes	 3.7e+01 34 32 30 28 26 24 22 20 18 16 14 12 10 8 6 4 2 0.0e+00	 3.7e+01 34 32 30 28 26 24 22 20 18 16 14 12 10 8 6 4 2 0.0e+00
Refinement iteration #2 104,314 nodes	 3.7e+01 34 32 30 28 26 24 22 20 18 16 14 12 10 8 6 4 2 0.0e+00	 3.7e+01 34 32 30 28 26 24 22 20 18 16 14 12 10 8 6 4 2 0.0e+00

Figure 6.28: 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.

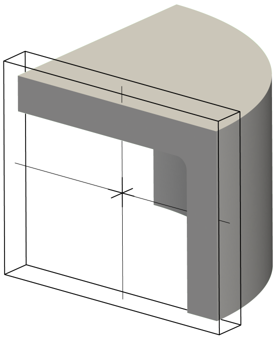


Figure 6.29: Selection of a thick “slice” of the domain to allow a closer analysis of the discretization and of the results.

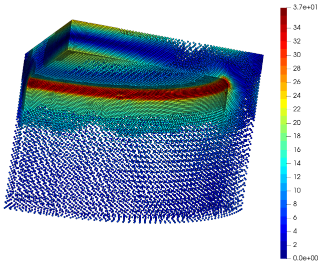
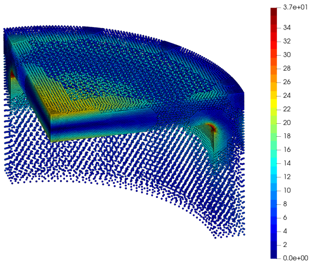
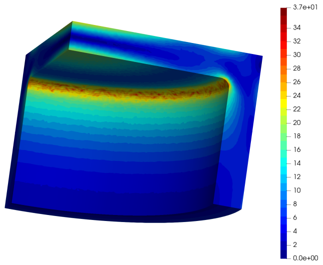
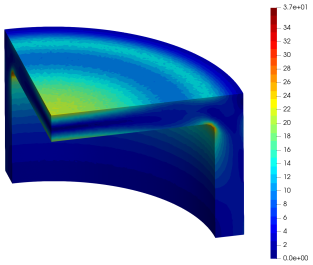
	Bottom view	**Top view**
Smart cloud collocation solution ▶ Refinement iteration #2 ▶ 104,314 nodes ▶ Approx. inter-node distance: 0.0077 in dense regions 0.03 in coarse regions		
FEA solution ▶ 166,193 nodes ▶ Approx. inter-node distance: 0.011		

Figure 6.30: Comparison, in terms of von Mises stress, of results obtained with the smart cloud method after two adaptive refinement iterations to results from the reference finite element solution.

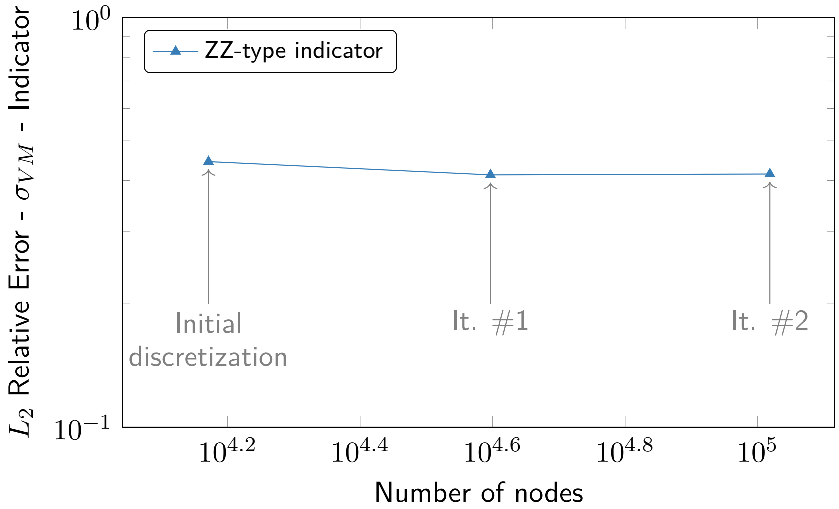


Figure 6.31: Evolution of the error indicator for the two iterations of adaptive refinement for the problem of the closed cylinder subject to pressure.

6.5 Conclusion

Discretization 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 chapter: 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 chapter is effective and leads to a faster convergence of the solution to the exact solution. The scheme 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 smart cloud adaptivity. These aspects are particularly true for domains with significant curvature, re-entrant corners or singularities.

Chapter 7

Solution of collocation linear problems

7.1 General

Solving efficiently large linear problems is a challenge in computational science. As the costs of computational resources keep decreasing, companies and research groups are more and more attracted to solving complex problems. People are not satisfied anymore with rough approximations and coarse models. The geometries need to represent exactly the reality and all the possible scenarios need to be simulated.

Methods to efficiently solve large systems have been developed since decades. The methods make use of massively parallel architectures such as Central Processing Units (CPUs) or Graphics Processing Unit (GPUs) and rely on a wide range of frameworks (MPI, OpenMP, CUDA, OpenCL ...). Depending on the considered problem (size, sparsity, condition number ...), various solution techniques can be considered. The most widely used are:

- direct methods;
- iterative methods;
- mixed methods.

We focus in this chapter on problems solved using the GFD point collocation method on CPUs and more specifically on Algebraic Multigrid (AMG) methods. We start by briefly describing the basis of the linear systems solution techniques in Section 7.2. Then, we present the bases of multigrid methods in Section 7.3. Finally, we perform in Section 7.4 a sensitivity analysis with the aim of assessing the influence of main parameters of an AMG preconditioner on the solution of a problem solved using the GFD method. We do not provide great details about the theory behind the presented methods and about each parameter of the selected AMG preconditioner. Details can be found in the listed references.

7.2 Bases of direct and iterative methods

We describe the bases of direct and iterative methods for a linear system:

$$\mathbf{Ku} = \mathbf{f}. \tag{7.1}$$

For linear elastic problems, $\mathbf{K}$ is called the "stiffness" matrix, $\mathbf{u}$ is the unknown displacement field vector and $\mathbf{f}$ is the force vector.

For most point collocation methods, a row of the linear system corresponds to an equation associated with a collocation node (an interior node or a boundary node). The matrix $\mathbf{K}$ is usually a square matrix. The number of unknown values equals the number of rows of the matrix. The system is said to be underdetermined if the number of rows of the matrix $\mathbf{K}$ is lower than the number of unknowns. Such a system has either no solution or an infinity. If the number of rows is greater than the number of unknowns, the system is said to be overdetermined. We will focus here on the case where $\mathbf{K}$ is a square matrix. Such a system is the most common type. Overdetermined systems can be considered in the framework of collocation. For instance, in addition to the equations solved at the collocation nodes, one can decide to also solve the Partial Differential Equation between specific nodes to reduce the error in selected regions.

7.2.1 Direct methods

The principle of direct methods is to use row and/or column permutations to transform the linear system into a system where all the unknown of the system can be calculated one after the other. The computed solution is exact. The most commonly used algorithm is the LU decomposition. The matrix $\mathbf{K}$, or the product of the matrix $\mathbf{K}$ with other matrices, is decomposed into the product of two triangular matrices $\mathbf{LU}$. The matrix $\mathbf{L}$ is a “Lower” triangular matrix. The matrix $\mathbf{U}$ is an “Upper” triangular matrix.

Considering a LU decomposition with partial pivoting, the linear problem (7.1) can be transformed by the computation of an appropriate permutation matrix $\mathbf{P}$ such that:

$$\mathbf{PKu} = \mathbf{LUu} = \mathbf{Pf}. \tag{7.2}$$

Considering that $\mathbf{L}$ and $\mathbf{U}$ are lower and upper triangular matrices, respectively, the system can easily be solved in two steps:

$$\begin{aligned} \mathbf{Ly} &= \mathbf{Pf} \qquad &\Rightarrow \qquad \mathbf{y} &= \mathbf{L^{-1}Pf} \\ \mathbf{Uu} &= \mathbf{y} \qquad &\Rightarrow \qquad \mathbf{u} &= \mathbf{U^{-1}y}. \end{aligned} \tag{7.3}$$

$\mathbf{LU}$ decomposition can be done efficiently on sequential and distributed memory systems using some libraries like MUMPS [39, 40] or SuperLU [95, 96, 97].

7.2.2 Classic iterative methods

Iterative methods rely on repeated approximations of the solution vector $\mathbf{u}$ to converge to a vector which solves the linear system with an error lower than a defined threshold. The approximation is updated at each iteration based on the error of the previous step. A vector $\mathbf{u^0}$ is selected to initialize the iteration loop. The null vector is typically used.

The most simple algorithms are the Jacobi, the Gauss-Seidel and the ILU methods. The Jacobi and the Gauss-Seidel methods use a decomposition of the matrix $\mathbf{K}$ into the sum of a diagonal matrix $\mathbf{D}$, a lower triangular matrix $\mathbf{L}$ and a upper triangular matrix $\mathbf{U}$. With the ILU method

(Incomplete LU factorization), the matrix $\mathbf{K}$ is decomposed into $\tilde{\mathbf{D}}$, $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{U}}$ such that: $\mathbf{K} - \left(\tilde{\mathbf{D}} + \tilde{\mathbf{L}}\right) \tilde{\mathbf{D}}^{-\mathbf{1}} \left(\tilde{\mathbf{D}} + \tilde{\mathbf{U}}\right) = \mathbf{R}$. $\mathbf{R}$ is a residual matrix. For a level of fill k, the matrices $\tilde{\mathbf{D}}$, $\tilde{\mathbf{L}}$ and $\tilde{\mathbf{U}}$ have sparsity patterns of the matrix $\mathbf{K}^{k+1}$.

The solution $\mathbf{u^{i+1}}$ is computed based on the vector $\mathbf{u^{i}}$ using the equation:

$$\begin{aligned} & u^{i+1} = u^i + \mathbf{A}^{-\mathbf{1}} \left(f - \mathbf{K}u^i\right) \\ & \text{where} \\ & \quad \mathbf{A} = \mathbf{D} && \text{(Jacobi method)} \\ & \quad \mathbf{A} = \mathbf{D} + \mathbf{L} && \text{(Gauss-Seidel method)} \\ & \quad \mathbf{A} = \left(\tilde{\mathbf{D}} + \tilde{\mathbf{L}}\right) \tilde{\mathbf{D}}^{-\mathbf{1}} \left(\tilde{\mathbf{D}} + \tilde{\mathbf{U}}\right) && \text{(ILU method).} \end{aligned} \tag{7.4}$$

The classical iterative methods are usually slow to converge compared to other methods. The ILU method is robust but requires important memory resources.

7.2.3 Krylov methods

Krylov subspace methods are more widely used as they lead to much faster convergences. These methods rely on the Cayley-Hamilton theorem. This theorem states that if a matrix $\mathbf{K}$ belongs to $\mathbb{R}^{n\times n}$, then a set of scalars α_i can be found so that:

$$\sum_{i=0}^{n} \alpha_i \mathbf{K}^i = 0_{n\times n}. \tag{7.5}$$

Multiplying this equation by $\mathbf{K}^{-1}/\alpha_0$, we obtain:

$$\mathbf{K}^{-1} = -\frac{1}{\alpha_0} \sum_{i=0}^{n-1} \alpha_{i+1} \mathbf{K}^i. \tag{7.6}$$

The Krylov methods make use of Krylov subspaces $\mathbb{K}_m = \text{span}\left(\mathbf{b}, \mathbf{Kb}, \mathbf{K}^2\mathbf{b}, ..., \mathbf{K}^m\mathbf{b}\right)$ of dimension m to approximate the solution based on successive projections [98, 99, 100]. The vector $\mathbf{b}$ is often selected equal to the force vector $\mathbf{f}$. With such a vector, the subspace $\mathbb{K}_m$ reproduces the exact solution of the linear system if m equals the dimension of the linear system n. However, an approximation of the solution can be obtained with Krylov subspaces of much lower sizes. The GMRES method [101] and the BiConjugate Gradient method [102] are popular Krylov methods to solve sparse linear systems with non-symmetric matrices.

7.2.4 Preconditioners

Preconditioners are used to enhance the convergence rate and the robustness of iterative methods. Preconditioners transform the linear system into a system easier to solve. The system presented in Equation (7.1) is transformed by a preconditioner $\mathbf{M}$ and becomes:

$$\mathbf{M^{-1}Ku} = \mathbf{M^{-1}f}. \tag{7.7}$$

The condition number of the matrix resulting from the product $\mathbf{M^{-1}K}$ should be as close as possible to unity for an optimum convergence. Such a system is said to be left preconditioned.

The Jacobi preconditioner is the most simple form of preconditioner. $\mathbf{M}$ is chosen equal to the diagonal of the matrix $\mathbf{K}$. $\mathbf{M^{-1}}$ can easily be computed. Such a preconditioner is efficient for matrices $\mathbf{K}$ which are greatly diagonal dominated. Similarly, the Gauss-Seidel and the ILU methods can be used as preconditioners. $\mathbf{M}$ is set equal to the matrices $\mathbf{A}$ presented in Equation (7.4).

7.3 Multigrid methods

The multigrid methods constitute a family of methods which have been initiated by Fedorenko in 1962 [103] and further developed/assessed in the 70s by Brandt [104] and Hackbusch [105]. These methods have been extensively studied because of their good performances. They present the advantage to often scale linearly with the number of degrees of freedom. Many variations of the method have been proposed. The method can be used as a solver or preconditioner.

Figure 7.1 shows the concept of the multigrid methods. The solution domain Ω_0 is reduced successively to a domain of a smaller size (here Ω_2) where the solution of the problem can easily be computed using a direct solver or an incomplete LU factorization. “Restriction” and “Interpolation” operators are computed to transform successively the problem from one “grid” or level to another. Smoothers are applied before every restriction and after every interpolation to "smooth" the high frequency errors.

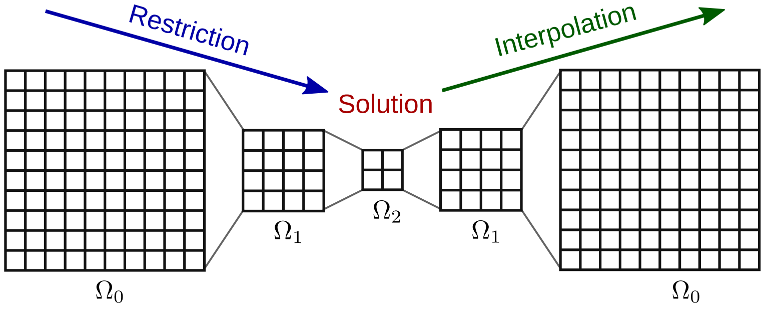


Figure 7.1: Concept of multigrid methods.

Figure 7.1 shows the typical "V" cycle of multigrid methods. The "V" corresponds to successive restriction steps followed by interpolation steps up to the initial size of the problem. Other types of cycles such as the "W" or the "F" cycles are also used. These alternative cycles use multiple successions of restriction and interpolation steps per multigrid iteration.

Considering an operator $\mathbf{A}$, the following key operations are performed in a "setup" phase for the levels "k" comprising between 0 and $M-1$ (M being the coarsest level selected).

- selection of the coarse points or variables;
- computation of the interpolation operators, noted $\mathbf{I}^k$;
- computation of the restriction operators, noted $\mathbf{R}^k$ ($\mathbf{R}^k$ is often selected as $\left(\mathbf{I}^k\right)^T$);
- computation of the grid operators $\mathbf{A}^{k+1} = \mathbf{R}^k\mathbf{A}^k\mathbf{P}^k$;
- selection of a smoother $\mathbf{S}^k$ used to reduce high frequency errors.

Once the "setup" phase is completed, the "solution" phase begins. One or multiple cycles can be performed using the following algorithm based on the one presented by Yang in Reference [106].

```
Algorithm MG_VCycle $\left(\mathbf{A}^k, \mathbf{R}^k, \mathbf{P}^k, \mathbf{S}^k, \mathbf{u}^k, \mathbf{f}^k, \mu_1, \mu_2\right)$

if $k = M$ then
    Solve the linear problem with a direct solver:
        $\mathbf{A}^M \mathbf{u}^M = \mathbf{f}^M$.
else
    Apply $\mu_1$ time the smoother $\mathbf{S}^k$ to the linear problem:
        $\mathbf{A}^k \mathbf{u}^k = \mathbf{f}^k$.
    Perform coarse grid correction (restriction):
        $r^k = \mathbf{f}^k - \mathbf{A}^k \mathbf{u}^k$.
        $r^{k+1} = \mathbf{R}^k r^k$.
    Repeat the algorithm up to the level M:
        MG_VCycle $\left(\mathbf{A}^{k+1}, \mathbf{R}^{k+1}, \mathbf{P}^{k+1}, \mathbf{S}^{k+1}, \mathbf{u}^{k+1}, \mathbf{f}^{k+1}, \mu_1, \mu_2\right)$.
    Interpolate the results from the coarse grid to the finer grid:
        $e^k = \mathbf{P}^k e^{k+1}$.
    Correct the solution based on the computed error:
        $u^k = u^k e^k$.
    Apply $\mu_2$ time the smoother $\mathbf{S}^k$ to the linear problem:
        $\mathbf{A}^k \mathbf{u}^k = \mathbf{f}^k$.
```

Figure 7.2: Multigrid V-cycle algorithm.

Multigrid methods can be split into two categories:

- the Geometric Multigrid Methods;
- the Algebraic Multigrid Methods.

The difference between the two methods reside in the construction of a coarse grid. Geometric multigrid methods use the discretization of the domain to construct successive coarser grids and to compute the restriction and interpolation operators. Algebraic Multigrid (AMG) methods only use the operator to determine the coarse grid levels and the associated restriction and interpolation operators.

AMG methods are of particular interest in the framework of point collocation methods since these methods rely on a strong form approximation of the partial differential equation and since no strong connection exists between the nodes. Comprehensive overviews of algebraic multigrid Methods are presented by Stuben [107], Hameyer [108] or Yang [106]. We apply a selected AMG preconditioner to a 3D problem solved using the GFD point collocation method in Section 7.4 and analyze the sensitivity of the preconditioner to key parameters.

7.4 AMG preconditioner sensitivity analysis

Algebraic multigrid methods have been implemented in a number of packages. The most popular ones are:

- GAMG which is part of the PETSc project [41, 42];
- ML which is part of the Trilinos project [109];
- BoomerAMG which is part of the *hypre* project [110].

Most AMG methods have been developed for symmetric matrices. Although these methods can also be applied to non-symmetric matrices the smoothed aggregation method used as part of the GAMG and ML packages is more based on the assumption of symmetry than the method implemented in the BoomerAMG package [111]. Therefore, we selected the package BoomerAMG from *hypre* and analyzed in a sensitivity study the influence of a selection of parameters on the solution time and on the number of iterations to achieve a selected tolerance. We do not aim at describing in detail the theory and role of each parameter of the preconditioner in this section. We focus on the impact of the parameters on the solution time and on the number of iterations of a selected iterative solver for a 3D collocation problem. We provide references to research works where the parameters are introduced, presented or compared.

We selected the model of a 3D sphere under internal pressure, presented in Figure 4.1 for the purpose of this sensitivity study. We present the base parameters analyzed in Figure 7.1. We chose the parameters based on recommendations given in the *hypre* manual and based on a preliminary parameter variation study.

Table 7.1: Base parameters considered for the *hypre* BoomerAMG preconditioner in the sensitivity study.

Preconditioner parameters	
max number of AMG levels	25
coarsening type	Falgout-CLJP
number of aggressive coarsening levels	2
aggressive coarsening threshold	0.95
interpolation on aggressive levels	multipass interpolation
number of cycles	20
cycle type	V
number of sweeps	3 down, 3 up
Krylov solver method	
solver type	GMRES
number of restarts	30
solver relative tolerance	1e-10
Machine used	
processor	Intel Xeon E5-1650 (3.2 GHz)
memory	32 GB
number of processes used	4

We compare first in Figure 7.3 the computation time and the number of iteration obtained from different types of solver:

- a direct solver;
- an iterative solver (GMRES) with a Jacobi preconditioner;
- an iterative solver (GMRES) with the BoomerAMG solver using the base parameters presented in Table 7.1.

We did not compare results obtained with the ILU preconditioner because the computation time and the number of iterations depends heavily on the selected number of fill levels. The results that we would have obtained with such a preconditioner would lay between the direct solver and the Jacobi preconditioner. The proximity to each solver would depend on the selected number of fill levels. The problem with the highest node density could not be solved with the direct solver due to memory limitations. Therefore no results are presented in Figure 7.3 for this case.

We observe from Figure 7.3 that, for the considered problem and number of degrees of freedom, the solution time with a direct solver is much larger than with an iterative solver. For small problems, direct solvers can be faster. However, as the size of the problem increases, iterative solvers show a much better performance. The memory required for a solution with a direct solver is also considerably larger than with an iterative one because the triangular matrices of the LU decomposition are dense. We can see from these results that the solution time does not increase regularly as the number of DOF increases for the Jacobi preconditioner. We observe the same trend on the number of iterations. The solution time increases regularly as the number of DOF increases when the multigrid preconditioner is used. The number of iterations with a direct solver is always equal to one since the problem is solved in only one step once the LU decomposition is performed. The number of iterations is not constant when the Jacobi preconditioner is used. The use of the multigrid preconditioner leads to a nearly constant number of iterations. The number of iterations only slightly increases for the largest problem considered.

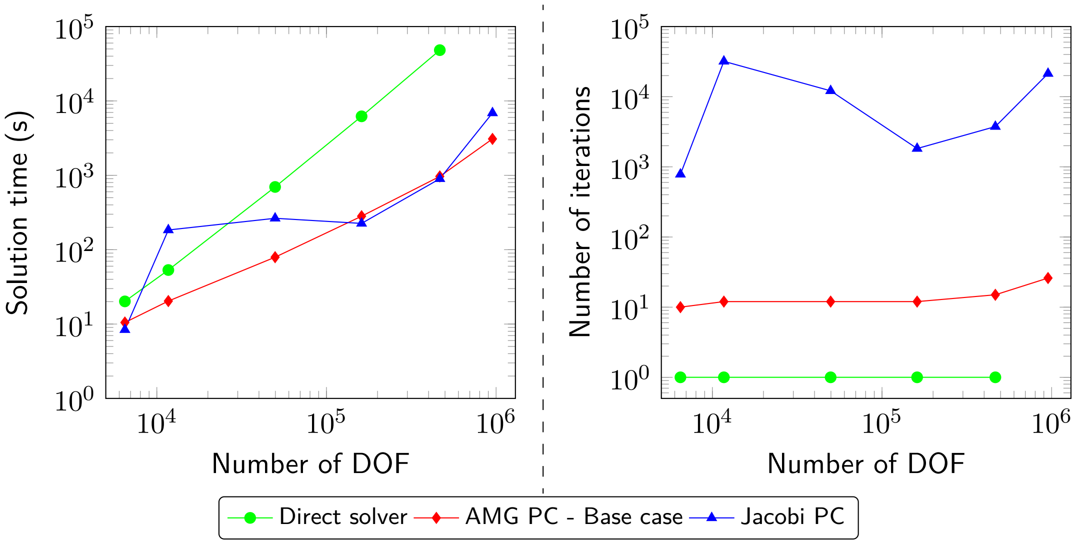


Figure 7.3: Comparison in terms of solution time and number of iterations of direct and iterative methods with and without multigrid preconditioner.

Figure 7.4 presents the results of a sensitivity analysis on the number of AMG cycles between each iteration of the Krylov solver. We vary the number of iterations between one (1) and fifty (50). The problem with the largest number of DOF did not converge within 500 iterations when one and two AMG cycles were selected. Therefore, we do not present results in Figure 7.4 for these cases.

We can see from the results presented in Figure 7.4 that a low number of AMG iterations (one or two) leads, for the considered problem, to a faster solution but also to a less stable number of iterations. The base case configuration (i.e. 20 iterations) leads to a relatively stable solution with a solution time close to the ones for five or ten iterations. The use of 50 iterations leads to a significantly higher solution time compared to the other number of iterations considered.

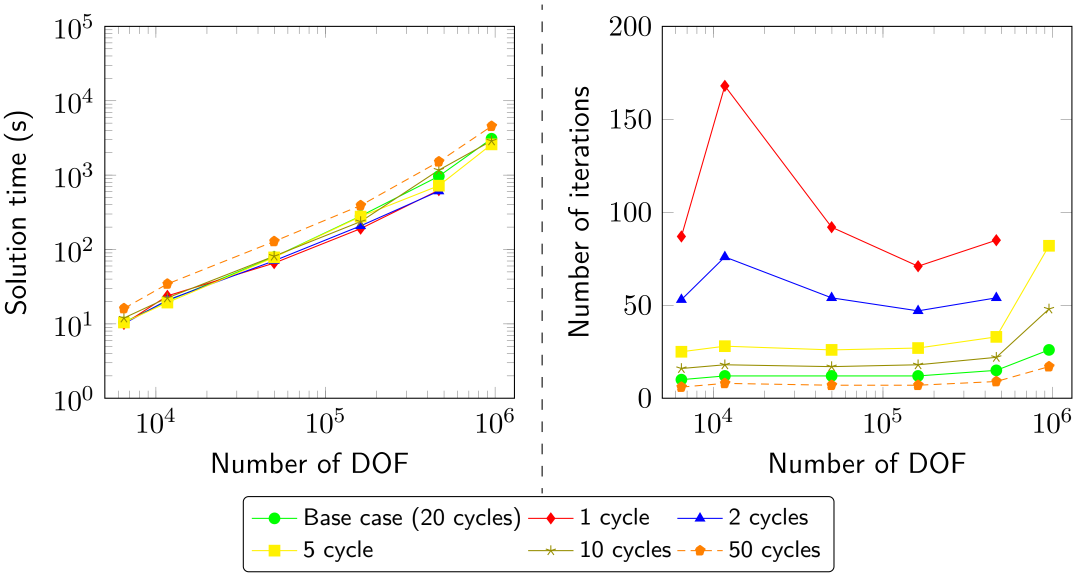


Figure 7.4: Number of cycles sensitivity - Comparison in terms of solution time and number of iterations for number of cycles ranging between 1 and 50. Results for the highest node density are not presented for 1 and 2 cycles because the iterative solver did not converge.

We present in Figure 7.5 the results of a sensitivity study on the number of relaxation sweeps considered before and after each level of the AMG cycle. We can see from these results that the number of sweeps has little impact on the solution time but allows reducing the number of iterations for the considered problem.

Figures 7.6 to 7.8 present results of a sensitivity analysis on the coarsening strategy. For AMG methods, the coarsening strategy is particularly important as it only depends on the connections of the matrix.

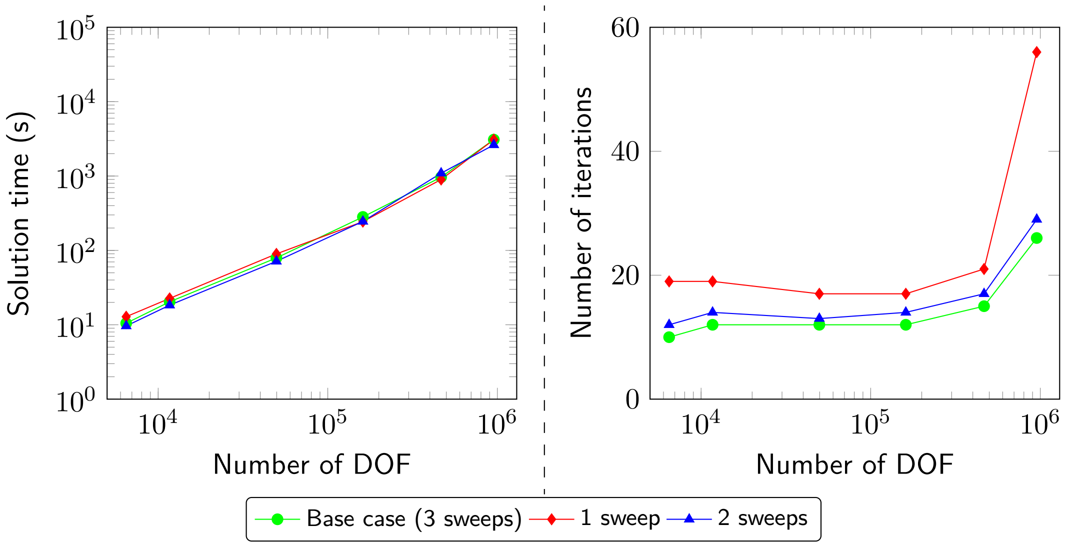


Figure 7.5: Number of sweeps sensitivity - Comparison in terms of solution time and number of iterations for number of sweeps ranging between 1 and 3.

Figure 7.6 shows the impact of the coarsening strategy on the solution time and on the number of iterations. We compare the Falgout coarsening strategy [110, 112, 113, 114] to the HMIS and PMIS coarsening strategies [115] with and without aggressive coarsening [107]. We shall note that the HMIS coarsening strategy is the default strategy of *hypre*. We observe that the base case configuration leads to the lowest solution time and the lowest number of iterations for the considered discretizations. The PMIS coarsening strategy with two levels of aggressive coarsening is the method that leads to the longest solution time and the greatest number of iterations.

We present in Figure 7.7 the impact of the number of aggressive coarsening levels on the solution for the base case strategy (i.e. Falgout coarsening). We see that the base case configuration (i.e. 2 levels of aggressive coarsening) leads to the lowest computation time for most discretizations. The Falgout coarsening strategy without aggressive coarsening leads the lowest number of iterations for most discretizations but to a non monotonic increase of the solution time and a non monotonic increase of the number of iterations compared to configurations of the Falgout coarsening strategy with aggressive coarsening. The results show that an increasing number of aggressive coarsening levels leads to an increasing number of iterations. The impact on the solution time is less clear. Two levels of aggressive coarsening appears to be a good choice for the considered problem and node densities as it leads to a relatively low solution time for most node densities.

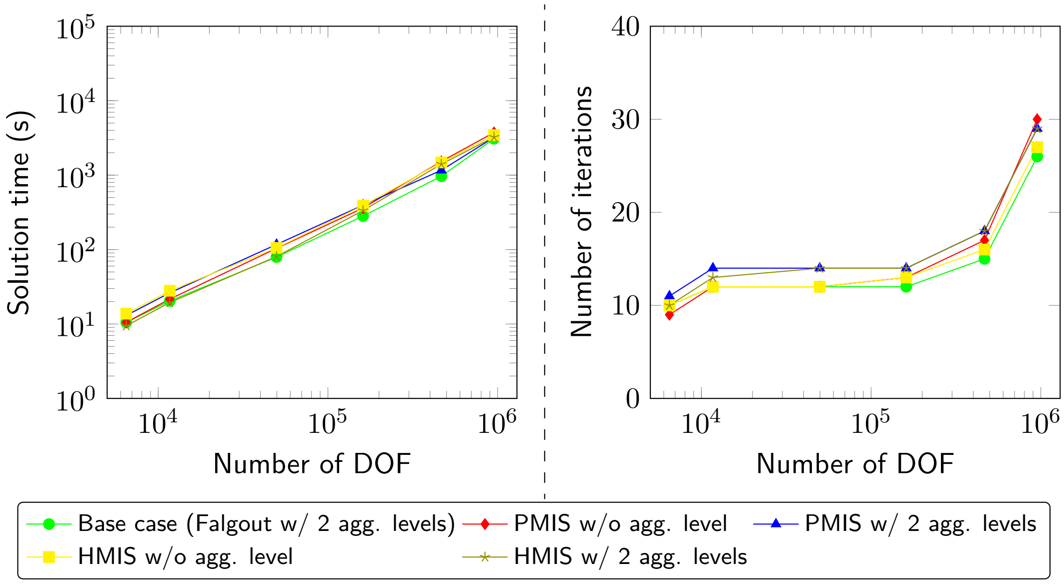


Figure 7.6: Coarsening strategy sensitivity - Comparison in terms of solution time and number of iterations for various coarsening strategies w/ and w/o aggressive coarsening levels.

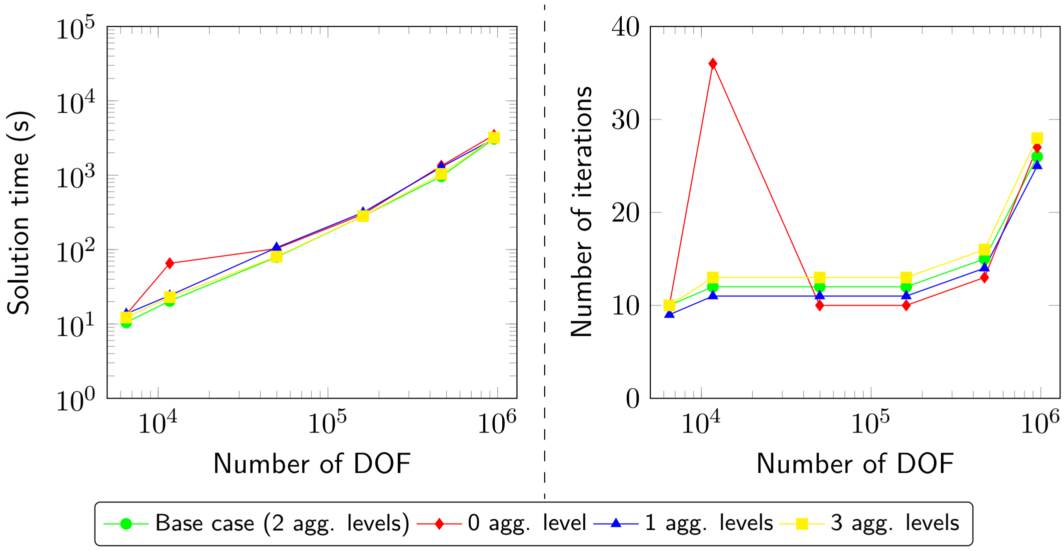


Figure 7.7: Number of aggressive coarsening levels sensitivity - Comparison in terms of solution time and number of iterations for aggressive coarsening levels ranging between 0 and 3. Two levels of aggressive coarsening (base case) lead to the lowest solution time even if the number of iterations of the iterative solver is not the lowest.

The results presented in Figure 7.7 are all using an aggressive coarsening threshold of 0.95. We analyze in Figure 7.8 the impact of this threshold on the solution time and number of iterations for the base case coarsening strategy (i.e. Falgout coarsening) with two levels of aggressive coarsening. We observe that the lowest threshold considered (i.e. 0.50) leads to the lowest number of iterations and solution time for the problems with the highest node densities. However, convergence of the iterative solver was not reached for the second node density considered. We observe that an increasing threshold value leads to an increasing number of iterations. The impact of the threshold on the solution time is less obvious. For some node densities, the solution time increases with the threshold value. A threshold of 0.95 leads to low solution time except for the highest node density for which lower threshold values lead to lower solution times.

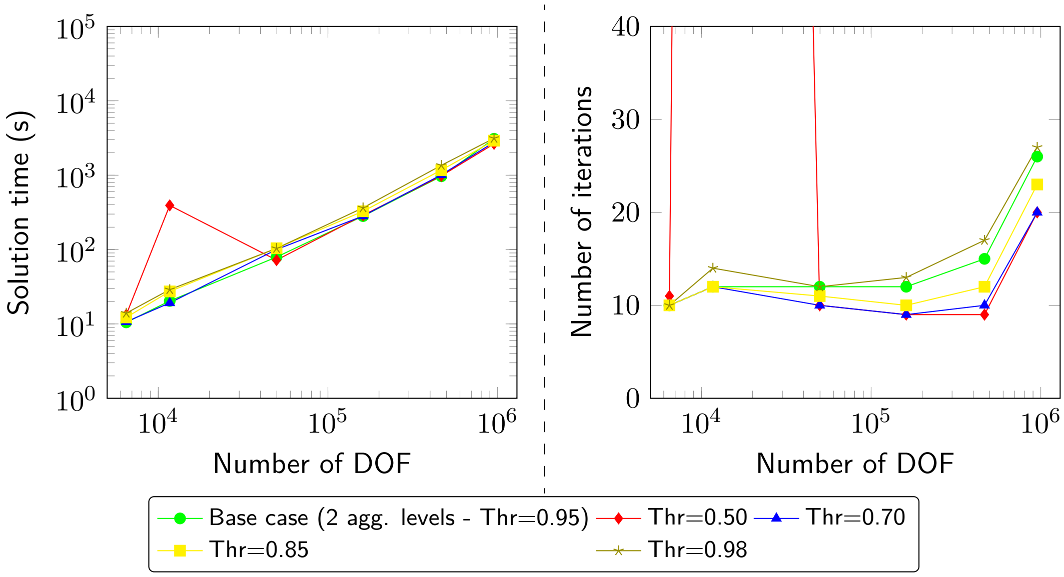


Figure 7.8: Aggressive coarsening threshold sensitivity - Comparison in terms of solution time and number of iterations for various aggressive coarsening thresholds ranging between 0.50 and 0.98. The iterative solver did not converge within 500 iterations for the second node density when a threshold of 0.50 is selected. Therefore this data-point should be discarded.

The results of the sensitivity study show that the base case parameters lead to low solution time for most of the considered densities compared to other configurations of the preconditioner. Preliminary results show however that these parameters are not optimum for other problems from the field of linear elasticity. To complete the study, a similar sensitivity study should be performed for problems loaded by different boundary conditions and having other types of solutions. Problems having a singular solution could be considered for instance.

7.5 Conclusion

Most of the problems solved in this thesis were solved using either direct solvers (for most 2D problems of relatively small size) or with the GMRES iterative solver. We used ILU preconditioners with one to three levels of fill depending on the complexity of the problem. We also used the GAMG preconditioners and the BoomerAMG preconditioners for large 3D problems to reduce the computation time and the memory usage compared to ILU preconditioned linear systems.

We showed in this chapter the bases of the solution of linear problems and focused especially on algebraic multigrid preconditioners. AMG preconditioning of a linear system, in a powerful technique to speed-up the convergence of a linear system while minimizing the memory utilization, however, many different methods exist and the methods rely on a large number of parameters. The configuration of AMG preconditioners is an aspect that has been studied during this thesis but that should be further analyzed, in the context of linear elastic problems solved using point collocation methods. The aim would be to identify a robust set of parameters that can be applied to most problems or to give practical guidelines for the selection of the parameters for any given problem from the field of linear elasticity. To ensure convergence of the solution with minimum effort a ILU preconditioner is a good choice. The flexibility that such a preconditioner has on the number fill levels is an advantage over AMG preconditioners at the expense of memory usage.

Chapter 8

Conclusions and perspectives

8.1 Conclusions

The aim of this thesis was to develop a unified workflow that allows the solution of complex problems directly from a CAD model using point collocation methods. The aim was also to propose methods, applicable to most problems without any a priori knowledge of the solution, to improve the results with minimum computational effort. We focused both on the GFD and DC PSE methods in Chapters 2, 4 and 5 because the results/methods presented in these chapters depend on the considered method. The results/methods presented in the other chapters of the thesis were derived using the the GFD method but can be applied with little to no modification to other point collocation methods.

We believe that the objectives of the thesis are met. We showed in Chapter 3 how CAD files could be used to generate smart clouds used as input for point collocation simulations. We studied the impact of the number of stencil nodes at different locations of the domain in Chapter 4 and introduced a unified node selection algorithm, based on a generalization of the visibility criterion, that can be applied to any problem. We studied in Chapter 5 the impact of base parameters of the GFD and DC PSE methods on the solution of singular and non-singular problems from the field of linear elasticity. We also proposed methods, based on the use of Voronoi diagrams, to improve the solution of collocation problems based on the GFD and DC PSE methods and studied a stabilization method aiming at reducing the error close to the boundary of the domain. We introduced in Chapter 6 two error indicators that allow the identification of the zones of the point cloud where the error is the greatest. Based on such indications we developed a discretization adaptivity scheme, based on the exact CAD geometry and on smart clouds, that allows a faster convergence of the results toward the exact solution. Finally, we presented in Chapter 7 different types of solver that can be selected for the solution of the collocation linear systems. We assessed in particular the impact of parameters of an algebraic multigrid preconditioner that led to the most promising results in terms of solution time and number of iterations.

The main results/conclusions obtained from the above chapters are listed below.

- The three identified variations of the DC PSE methods lead to similar results. The DCPSE1 method leads often to the lowest error. The DCPSE2 method is more computationally efficient.

- Among the compared computation methods, the GFD and DC PSE methods are the methods leading to the lowest error for the 2D cylinder problem (polynomial solution). For the L-shape problem (singular solution), the FEA method leads to the lowest error. For practical 3D problems, the GFD methods leads to similar results as those obtained using finite element simulations.

- Point clouds can be generated directly from CAD files and transformed into "smart clouds" by the addition of additional node information. Such information relates to the base CAD geometry, to the boundary conditions or information for solution enhancement methods such as the generalized visibility criterion.

- The proposed discretization method from CAD geometries leads to results similar to those obtained from discretizations based on Delaunay triangulations of the domains. The method only relies on the selection of a threshold parameter which we use to assess if interior nodes are close to boundary nodes. A threshold of 0.3 leads to robust results.

- Increasing the stencil size of the boundary nodes leads to a significant error reduction with a relatively low increase of the fill of the linear system.

- The visibility criterion can be generalized to all types of problems. This approach shall be used in conjunction with a threshold angle to maximize its applicability to all problems. A threshold angle of 5.0° can be used as a starting point for most problems.
- Voronoi diagrams can be used to give additional information to the collocation methods about the local area associated with each collocation node. However, the proposed methods do not lead to a reduction of the error for all of the considered problems.
- Stabilization of the PDE considered at the boundary conditions does not lead to a reduction of the error for all of the considered problems.
- The proposed ZZ-type and the residual-type error indicators allow both an identification of the areas of the domain where the error is the greatest and can be used as basis for discretization adaptivity. The ZZ-indicator is computationally more efficient. Both indicators do not give a robust estimation of the error.
- Smart cloud h-adaptivity based on the exact CAD geometry allows speeding up convergence of the results toward the exact solution.
- The different types of solver lead to significant differences in terms of computation time and memory usage although we did not study precisely this second aspect.
- Algebraic multigrid preconditioners allow a rapid convergence of the linear problem to the selected solution tolerance. However, these preconditioners rely on a large number of parameters and a unique set of parameters does not apply well to all the problems.
- ILU preconditioners are flexible on the number fill levels which directly impacts the duration of the preconditioning step and the memory requirement. They lead to a robust convergence of the linear system to the solution.

8.2 Perspectives

Point collocation methods are promising thanks to their flexibility to the placement of the nodes involved in the discretization of the domain. This aspect is a key strength in the context of smart cloud adaptivity. We presented in this thesis a unified workflow that allows users of these methods to efficiently solve engineering problems despite the many parameters on which point collocation methods are based. In the continuity of this thesis, we propose research directions that could be investigated to further facilitate the efficient solution of new problems and to improve one step further the quality of the solution.

- The discretization method from CAD presented in this thesis uses a regular discretization of the domain. An anisotropic discretization algorithm based on identified principal directions of a given CAD geometry is an aspect that could be studied. Such discretization would lead to a reduction of the computation cost without any loss of accuracy of the solution. It should be developed together with a robust anisotropic stencil selection that could be based on the work from Kennett et al. [55].

- The smart clouds generated from CAD are coupled to the CAD geometry. In case of modification of dimensions of the base CAD geometry, the generated smart cloud could be adapted to the modified geometry. This would avoid the generation of a new discretization and save computation time. The solution of the base problem could also be used to initialize the solution of the modified problem and speed-up the convergence of the iterative solver.

- Intrinsic or extrinsic enrichment methods are widely used in the context of the Element Free Galerkin method [59, 116, 117, 118, 119] for singular problems. However, little work was done in the framework of point collocation methods such as the Generalized Finite Difference method. The use of enriched weight functions and enriched stencils near singularities could be considered in order to improve the results obtained in regions of rapid field change near singularities.

- In this thesis, we based our smart cloud adaptive scheme on the refinement of the regions where the error is the greatest. In addition to this, a criterion could be used to remove nodes in the zones where the node density is deemed too high based on a local assessment of the quality of the solution.

- The restriction and interpolation algorithms of algebraic multigrid preconditioners mostly rely on the linear system matrix and are independent from the geometry. Such an aspect is both an advantage and a drawback of this class of method. AMG methods can be readily applied to any linear problem at the expense of a large number of tuning parameters. To improve the suitability of multigrid methods to most collocation problems, the restriction and interpolation algorithms should be based on the actual geometry and PDE.

In this thesis we generated smart clouds based on CAD geometries. Solving directly given point clouds and adapting them to simulations is another workflow. With the advent of imaging approaches, in particular LiDAR and photogrammetry, being able to solve problems based on this alternative workflow 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. [120] and references therein [121, 122];
- [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 [123, 124];
- [Smart clouds to CAD] Simplify smart point clouds into CAD primitives using classification machine learning approaches [125, 126, 127, 128, 129];
- [Properties identification from photogrammetry] Infer parameter values based on texture and color from photogrammetric images [130, 131, 132];
- [Solution process] Develop preconditioners and parallelization schemes for smart point clouds [41, 110, 109];
- [Integrated simulation pipeline] Develop open source pipelines based on libraries such as

Open CASCADE Technology [46], VTK [133], CGAL [134], Medusa [94], Voro++ [135];

- [Multi-scale model reduction approaches] Defeature complex point clouds in order to accelerate simulations using multi-scale, domain decomposition and model order reduction approaches.

Appendices

Appendix A

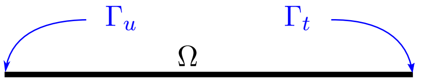
GFD and DC PSE methods comparison for a 1D problem

The GFD and DC PSE methods have many similarities. We present in this appendix a side by side comparison of the methods for a simple 1D problem. We selected the case of a second order PDE for illustration purposes. Considering a differential operator $\mathcal{A}$ and a field $f : \mathbb{R} \to \mathbb{R}$, the following PDE can be written over the domain Ω:

$$\mathcal{A}(f) = 0 \quad \text{in} \quad \Omega. \tag{A.1}$$

The field f shall also verify the conditions imposed on the Dirichlet and Neumann boundaries, which are denoted by Γ_u and Γ_t, respectively. The field values f are set to $\overline{f}$ on Γ_u. On Γ_t, the field shall verify a lower order PDE defined by an operator $\mathcal{B}$.

$$\begin{aligned} f - \overline{f} &= 0 \quad \text{on} \quad \Gamma_u, \\ \mathcal{B}(f) &= 0 \quad \text{on} \quad \Gamma_t. \end{aligned} \tag{A.2}$$



;

Figure A.1: 1D Domain Ω with Γ_u and Γ_t boundaries.

In order to solve this problem by collocation, we need to transform it into a linear system of the form $\mathbf{AF} = \mathbf{B}$, where $\mathbf{A}$ is the problem matrix, $\mathbf{F}$ is a vector containing the field values at each node of the domain, and $\mathbf{B}$ is a vector containing various constraints of the problem. The field derivatives at the collocation centers need to be approximated as a function of the values at the nodes. Collocation is typically performed at the nodes but can also be performed in other locations.

Principle Both the GFD and the DC PSE methods are based on a Taylor's series expansion of the unknown field around the collocation nodes.

GFD	**DC PSE**
The field derivatives at the collocation node are determined so that the field values at the support nodes can be reproduced using a Taylor's series expansion. The field derivatives are approximated simultaneously at each node of the domain.	A convolution function is selected so that the approximated derivative in the Taylor's series expansion only depends on the field values at the support nodes. All the other unknown derivatives are canceled by the selected function. Different convolution functions are used to approximate the derivatives of various orders at a collocation node.

The steps associated with each method are presented below for the case of a second order PDE. In the sections below, the nodes are labeled $\mathbf{X_i}$ and the 1D coordinates associated with the node are written x_i.

Step 1: Taylor's series approximation

For both methods, the first step consists in writing an approximation of the Taylor's series expansion up to the desired order. The approximation order shall be of at the least the highest derivative order of the differential operator $\mathcal{A}$. In 1D, the Taylors's series expansion at a point $\mathbf{X_{pi}}$ in the vicinity of $\mathbf{X_c}$ is:

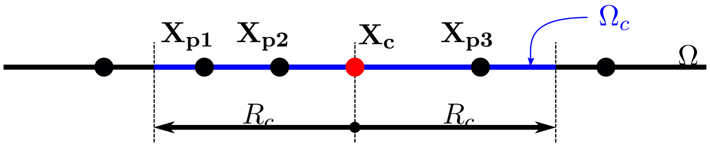
$$f(\mathbf{X_{pi}}) = f(\mathbf{X_c}) + \sum_{i=1}^{+\infty} \frac{(x_{pi} - x_c)^i}{i!} \frac{d^i f(\mathbf{X_c})}{dx^i}. \tag{A.3}$$

Denoting the second order approximation of this expansion by $f_h(\mathbf{X_{pi}})$, Equation (A.3) becomes:

$$f_h(\mathbf{X_{pi}}) = f(\mathbf{X_c}) + (x_{pi} - x_c)\frac{df(\mathbf{X_c})}{dx} + \frac{(x_{pi} - x_c)^2}{2!} \frac{d^2 f(\mathbf{X_c})}{dx^2}. \tag{A.4}$$

Step 2: Support node selection

The nodes in the vicinity of the collocation nodes are used to approximate the desired derivatives. These nodes are selected within a distance or radius R_c from the collocation nodes $\mathbf{X_c}$.



;

Figure A.2: 1D support Ω_c of a collocation node $\mathbf{X_c}$. The radius of the support is R_c. The nodes $\mathbf{X_{p1}}$, $\mathbf{X_{p2}}$ and $\mathbf{X_{p3}}$ are in the support of $\mathbf{X_c}$.

Step 3: Derivatives approximation

GFD The Taylor's series expansion presented in Equation (A.4) can be written for each node of the collocation node support. For the example presented in Figure A.2, three nodes are present in the support of the collocation node. The following system is obtained:

$$\begin{cases} f_h(\mathbf{X_{p1}}) = f(\mathbf{X_c}) + (x_{p1} - x_c)\frac{df(\mathbf{X_c})}{dx} + \frac{(x_{p1}-x_c)^2}{2!}\frac{d^2f(\mathbf{X_c})}{dx^2} \\ f_h(\mathbf{X_{p2}}) = f(\mathbf{X_c}) + (x_{p2} - x_c)\frac{df(\mathbf{X_c})}{dx} + \frac{(x_{p2}-x_c)^2}{2!}\frac{d^2f(\mathbf{X_c})}{dx^2} \\ f_h(\mathbf{X_{p3}}) = f(\mathbf{X_c}) + (x_{p3} - x_c)\frac{df(\mathbf{X_c})}{dx} + \frac{(x_{p3}-x_c)^2}{2!}\frac{d^2f(\mathbf{X_c})}{dx^2}. \end{cases} \tag{A.5}$$

If the number of support nodes is larger than the approximated derivative order (two for the selected example), the system is overdetermined. In that case, the derivatives that best reproduce the field values are determined using the moving least square method. The associated minimization problem is written in the form of a functional B. A weight function w is used to balance the contribution of the support nodes as a function of their distance to the collocation node.

$$B(\mathbf{X_c}) = \sum_{i=1}^{3} w(\mathbf{X_{pi}}-\mathbf{X_c})\Big[f(\mathbf{X_c})-f(\mathbf{X_{pi}})+(x_{pi}-x_c)\frac{df(\mathbf{X_c})}{dx}+\frac{(x_{pi}-x_c)^2}{2!}\frac{d^2f(\mathbf{X_c})}{dx^2}\Big]^2. \tag{A.6}$$

The derivatives $\mathbf{Df}(\mathbf{X}) = \left[\frac{df(\mathbf{X})}{dx}, \frac{d^2f(\mathbf{X})}{dx^2}\right]^T$, that best approximate the known field values based on the Taylor's series expansion, minimize B when:

$$\left.\frac{dB(\mathbf{X})}{d\mathbf{Df}(\mathbf{X})}\right|_{\mathbf{X}=\mathbf{X_c}} = 0, \tag{A.7}$$

$$\Leftrightarrow \begin{cases} \sum_{i=1}^{3} w(\mathbf{X_{pi}} - \mathbf{X_c})(x_{pi} - x_c) \\ \quad \Big[f(\mathbf{X_c}) - f(\mathbf{X_{pi}}) + (x_{pi} - x_c)\frac{df(\mathbf{X_c})}{dx} + \frac{(x_{pi}-x_c)^2}{2!}\frac{d^2f(\mathbf{X_c})}{dx^2}\Big] = 0 \\ \sum_{i=1}^{3} w(\mathbf{X_{pi}} - \mathbf{X_c})\frac{(x_{pi}-x_c)^2}{2!} \\ \quad \Big[f(\mathbf{X_c}) - f(\mathbf{X_{pi}}) + (x_{pi} - x_c)\frac{df(\mathbf{X_c})}{dx} + \frac{(x_{pi}-x_c)^2}{2!}\frac{d^2f(\mathbf{X_c})}{dx^2}\Big] = 0. \end{cases} \tag{A.8}$$

This system can be rearranged in a matrix form as follows:

$$\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} \frac{df(\mathbf{X_c})}{dx} \\ \frac{d^2f(\mathbf{X_c})}{dx^2} \end{bmatrix} = \begin{bmatrix} -m_{01} & m_{01,1} & m_{01,2} & m_{01,3} \\ -m_{02} & m_{02,1} & m_{02,2} & m_{02,3} \end{bmatrix} \begin{bmatrix} f(\mathbf{X_c}) \\ f(\mathbf{X_{p1}}) \\ f(\mathbf{X_{p2}}) \\ f(\mathbf{X_{p3}}) \end{bmatrix}, \tag{A.9}$$

where the moments $m_{ij,k}$ and m_{ij} and the matrix $\mathbf{P}(\mathbf{X_c}) \in \mathbb{R}^{3\times 3}$ correspond to:

$$\begin{aligned} m_{ij,k} &= w(\mathbf{X_{pk}} - \mathbf{X_c}) P_{(i+1)k}(\mathbf{X_c}) P_{(j+1)k}(\mathbf{X_c}), \\ m_{ij} &= \sum_{k=1}^{3} m_{ij,k}, \\ \mathbf{P}(\mathbf{X_c}) &= \begin{bmatrix} 1 & 1 & 1 \\ (x_{p1} - x_c) & (x_{p2} - x_c) & (x_{p3} - x_c) \\ \frac{(x_{p1}-x_c)^2}{2!} & \frac{(x_{p2}-x_c)^2}{2!} & \frac{(x_{p3}-x_c)^2}{2!} \end{bmatrix}. \end{aligned} \tag{A.10}$$

Equation (A.9) can be represented in the form $\mathbf{A}(\mathbf{X_c})\mathbf{Df}(\mathbf{X_c}) = \mathbf{E}(\mathbf{X_c})\mathbf{F}(\mathbf{X_c})$.

DC PSE

The Taylor's series expansion presented in Equation (A.4) can be convoluted by a function η over the support Ω_c of the collocation node X_c:

$$\begin{aligned} \int_{\Omega_c} f_h(\mathbf{X_p})\eta(\mathbf{X_p} - \mathbf{X_c})d\mathbf{X_p} = &\int_{\Omega_c} f(\mathbf{X_c})\eta(X_p - \mathbf{X_c})d\mathbf{X_p} \\ &+ \int_{\Omega_c} \frac{df(\mathbf{X_c})}{dx}(x_p - x_c)\eta(\mathbf{X_p} - \mathbf{X_c})d\mathbf{X_p} \\ &+ \int_{\Omega_c} \frac{d^2 f(\mathbf{X_c})}{dx^2}\frac{(x_p - x_c)^2}{2!}\eta(\mathbf{X_p} - \mathbf{X_c})d\mathbf{X_p}. \end{aligned} \tag{A.11}$$

The integral can be approximated by a sum, assuming that the nodes are regularly distributed over the support and that the field f is sufficiently smooth.

$$\sum_{i=1}^{3} f_h(\mathbf{X_{pi}})\eta(\mathbf{X_{pi}} - \mathbf{X_c}) = f(\mathbf{X_c})M_0(\mathbf{X_c}) + \frac{df(\mathbf{X_c})}{dx}M_1(\mathbf{X_c}) + \frac{df^2(\mathbf{X_c})}{dx^2}M_2(\mathbf{X_c}), \tag{A.12}$$

where the moments M_j are:

$$M_j(\mathbf{X_c}) = \sum_{i=1}^{3} \frac{(x_{pi} - x_c)^j}{j!}\eta(\mathbf{X_{pi}} - \mathbf{X_c}). \tag{A.13}$$

The convolution function is chosen so that all the moments in Equation (A.12) are null except the one multiplying the approximated derivative of order n_x, which is denoted by $D^{n_x} f(\mathbf{X_c})$. This moment is set to unity.

$$\begin{cases} D^{n_x} f(\mathbf{X_c}) = \sum_{i=1}^{3} f_h(\mathbf{X_{pi}})\eta(\mathbf{X_{pi}} - \mathbf{X_c}) \\ \text{with} \quad M_{n_x}(\mathbf{X_c}) = 1 \\ \qquad\quad M_i(\mathbf{X_c}) = 0 \quad \text{if } i \neq n_x. \end{cases} \tag{A.14}$$

In order to satisfy this moment condition, the convolution function is chosen as the product of two functions: a correction function K and a weight function w. The correction function is typically the product of a coefficient vector $\mathbf{a}$ and a polynomial basis $\mathbf{P}$. For the 1D second order approximation, the polynomial basis $\mathbf{P} = [1, x, x^2]^T$ can be selected. The correction function can then be written as follows:

$$\eta(\mathbf{X_p} - \mathbf{X_c}) = \mathbf{P}(\mathbf{X_p} - \mathbf{X_c})^T \mathbf{a}\ w(\mathbf{X_p} - \mathbf{X_c}). \tag{A.15}$$

The coefficients of the vector $\mathbf{a}$ are determined in order to satisfy the moment condition set in Equation (A.14). For instance, the moment condition associated with the second order derivative approximation is:

$$\begin{cases} M_0(\mathbf{X_c}) = 0 \quad \Leftrightarrow \quad \sum_{i=1}^{3} \mathbf{P}(\mathbf{X_{pi}} - \mathbf{X_c})^T \mathbf{a} w(\mathbf{X_{pi}} - \mathbf{X_c}) = 0 \\ M_1(\mathbf{X_c}) = 0 \quad \Leftrightarrow \quad \sum_{i=1}^{3} (x_{pi} - x_c) \mathbf{P}(\mathbf{X_{pi}} - \mathbf{X_c})^T \mathbf{a} w(\mathbf{X_{pi}} - \mathbf{X_c}) = 0 \\ M_2(\mathbf{X_c}) = 1 \quad \Leftrightarrow \quad \sum_{i=1}^{3} \frac{(x_{pi} - x_c)^2}{2!} \mathbf{P}(\mathbf{X_{pi}} - \mathbf{X_c})^T \mathbf{a} w(\mathbf{X_{pi}} - \mathbf{X_c}) = 1. \end{cases} \tag{A.16}$$

The system of equations can be put in a matrix form as follows:

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \tag{A.17}$$

Considering the vector $\mathbf{Q}(\mathbf{X_c}, \mathbf{X_p}) = [1, (x_p - x_c), \frac{(x_p-x_c)^2}{2!}]^T$, the correction function basis $\mathbf{P}$ and the weight function w, the coefficients of the matrix $\mathbf{A} \in \mathbb{R}^{3\times3}$ can be written as:

$$A_{i,j}(\mathbf{X_c}) = \sum_{i=1}^{3} Q_i(\mathbf{X_c}, \mathbf{X_{pi}}) P_j(\mathbf{X_{pi}} - \mathbf{X_c}) w(\mathbf{X_{pi}} - \mathbf{X_c}). \tag{A.18}$$

Step 4: Solution of the collocation linear systems

GFD

The system presented in Equation (A.9) is solved to obtain the derivatives $\mathbf{Df}(\mathbf{X_c})$ as a function of the field values $\mathbf{F}(\mathbf{X_c})$.

DC PSE

The system presented in Equation (A.17) is solved to obtain the coefficients of the correction function η. Once these coefficients are obtained, the convolution function presented in Equation (A.15) can be calculated and the derivative $D^{n_x} f(\mathbf{X_c})$ presented in Equation (A.14) approximated.

The solution of Equation (A.17) needs to be performed for all the moment conditions associated with the approximated derivatives in the differential operators $\mathcal{A}$ and $\mathcal{B}$.

Step 5: Assembly of the linear problem

The steps 1 to 4 allowed the approximation of the derivatives at the collocation nodes as a function of the field values at theses nodes and at their support nodes. Sets of coefficients $\mathbf{C_{n_x}}(\mathbf{X_c})$ are obtained for each derivatives so that $\mathbf{D^{n_x}} f(\mathbf{X_c}) = \mathbf{C_{n_x}}(\mathbf{X_c})\ \mathbf{F}(\mathbf{X_c})$. Finally, based on the differential operator and on the boundary conditions, the problem matrices $\mathbf{A}$ and $\mathbf{B}$ are assembled.

References

[1] C. Runge. In: *Z. Math. u. Physik* 50 (1908), p. 255.

[2] R.H. Macneal. "An asymmetrical finite difference network". eng. In: *Quarterly of Applied Mathematics* 11.3 (1953), pp. 295–310. ISSN: 0033569X.

[3] G.E. Forsythe and W.R. Wasow. *Finite Difference Methods for Partial DifferentialEquations*. Wiley, 1960. ISBN: 1258669250.

[4] P.S. Jensen. "Finite difference techniques for variable grids". In: *Computers & Structures* 2.1-2 (Feb. 1972), pp. 17–29. DOI: 10.1016/0045-7949(72)90020-x. URL: https://doi.org/10.1016/0045-7949(72)90020-x.

[5] N. Perrone and R. Kao. "A general finite difference method for arbitrary meshes". In: *Computers & Structures* 5.1 (Apr. 1975), pp. 45–57. DOI: 10.1016/0045-7949(75)90018-8. URL: https://doi.org/10.1016/0045-7949(75)90018-8.

[6] T. Liszka and J. Orkisz. "The finite difference method at arbitrary irregular grids and its application in applied mechanics". In: *Computers & Structures* 11.1-2 (Feb. 1980), pp. 83–95. DOI: 10.1016/0045-7949(80)90149-2. URL: https://doi.org/10.1016/0045-7949(80)90149-2.

[7] J. Orkisz. "Finite difference method (Part III)". In: *Handbook of Computational Solid Mechanics*. Springer-Verlag, 1998, pp. 335–432.

[8] D. Shepard. "A two-dimensional interpolation function for irregularly-spaced data". In: *Proceedings of the 1968 23rd ACM national conference*. ACM Press, 1968. DOI: 10.1145/800186.810616. URL: https://doi.org/10.1145/800186.810616.

[9] P. Lancaster and K. Salkauskas. “Surfaces generated by moving least squares methods”. In: *Mathematics of Computation* 37.155 (Sept. 1981), pp. 141–141. DOI: 10.1090/s0025-5718-1981-0616367-1. URL: https://doi.org/10.1090/s0025-5718-1981-0616367-1.

[10] E. Oñate et al. “A Finite Point Method in Computational Mechanics. Application to convective transport. and fluid flow.” In: *International Journal for Numerical Methods in Engineering* 39.22 (Nov. 1996), pp. 3839–3866. DOI: 10.1002/(sici)1097-0207(19961130)39:22<3839::aid-nme27>3.0.co;2-r. URL: https://doi.org/10.1002/(sici)1097-0207(19961130)39:22%3C3839::aid-nme27%3E3.0.co;2-r.

[11] E.J. Kansa. “Multiquadrics—A scattered data approximation scheme with applications to computational fluid-dynamics—I surface approximations and partial derivative estimates”. In: *Computers & Mathematics with Applications* 19.8-9 (1990), pp. 127–145. DOI: 10.1016/0898-1221(90)90270-t. URL: https://doi.org/10.1016/0898-1221(90)90270-t.

[12] T.A. Driscoll and B. Fornberg. “Interpolation in the limit of increasingly flat radial basis functions”. In: *Computers & Mathematics with Applications* 43.3-5 (Feb. 2002), pp. 413–422. DOI: 10.1016/s0898-1221(01)00295-4. URL: https://doi.org/10.1016/s0898-1221(01)00295-4.

[13] C. Shu, H. Ding, and K.S Yeo. “Local radial basis function-based differential quadrature method and its application to solve two-dimensional incompressible Navier–Stokes equations”. In: *Computer Methods in Applied Mechanics and Engineering* 192.7-8 (Feb. 2003), pp. 941–954. DOI: 10.1016/s0045-7825(02)00618-7. URL: https://doi.org/10.1016/s0045-7825(02)00618-7.

[14] B. Fornberg and E. Lehto. “Stabilization of RBF-generated finite difference methods for convective PDEs”. In: *Journal of Computational Physics* 230.6 (Mar. 2011), pp. 2270–2285. DOI: 10.1016/j.jcp.2010.12.014. URL: https://doi.org/10.1016/j.jcp.2010.12.014.

[15] O. Davydov and D. Thi Oanh. "On the optimal shape parameter for Gaussian radial basis function finite difference approximation of the Poisson equation". In: *Computers & Mathematics with Applications* 62.5 (Sept. 2011), pp. 2143–2161. DOI: 10.1016/j.camwa.2011.06.037. URL: https://doi.org/10.1016/j.camwa.2011.06.037.

[16] J.J. Monaghan. "Smoothed Particle Hydrodynamics". In: *Annual Review of Astronomy and Astrophysics* 30.1 (Sept. 1992), pp. 543–574. DOI: 10.1146/annurev.aa.30.090192.002551. URL: https://doi.org/10.1146/annurev.aa.30.090192.002551.

[17] W. K. Liu, S. Jun, and Y. F. Zhang. "Reproducing kernel particle methods". In: *International Journal for Numerical Methods in Fluids* 20.8-9 (Apr. 1995), pp. 1081–1106. DOI: 10.1002/fld.1650200824. URL: https://doi.org/10.1002/fld.1650200824.

[18] B. Schrader, S. Reboux, and I.F. Sbalzarini. "Discretization correction of general integral PSE Operators for particle methods". In: *Journal of Computational Physics* 229.11 (June 2010), pp. 4159–4182. DOI: 10.1016/j.jcp.2010.02.004. URL: https://doi.org/10.1016/j.jcp.2010.02.004.

[19] T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs. "Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement". In: *Computer Methods in Applied Mechanics and Engineering* 194.39-41 (Oct. 2005), pp. 4135–4195. DOI: 10.1016/j.cma.2004.10.008. URL: https://doi.org/10.1016/j.cma.2004.10.008.

[20] R.N. Simpson et al. "A two-dimensional Isogeometric Boundary Element Method for elastostatic analysis". In: *Computer Methods in Applied Mechanics and Engineering* 209-212 (Feb. 2012), pp. 87–100. DOI: 10.1016/j.cma.2011.08.008. URL: https://doi.org/10.1016/j.cma.2011.08.008.

[21] H. Lian, R.N. Simpson, and S. Bordas. "Stress analysis without meshing: isogeometric boundary-element method". In: *Proceedings of the Institution of Civil Engineers - Engineering and Computational Mechanics* 166.2 (June 2013), pp. 88–99. DOI: 10.1680/eacm.11.00024. URL: https://doi.org/10.1680/eacm.11.00024.

[22] R.N. Simpson et al. “Acoustic isogeometric boundary element analysis”. In: *Computer Methods in Applied Mechanics and Engineering* 269 (2014), pp. 265–290. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2013.10.026. URL: https://www.sciencedirect.com/science/article/pii/S0045782513002788.

[23] X. Peng et al. “Isogeometric boundary element methods for three dimensional static fracture and fatigue crack growth”. In: *Computer Methods in Applied Mechanics and Engineering* 316 (Apr. 2017), pp. 151–185. DOI: 10.1016/j.cma.2016.05.038. URL: https://doi.org/10.1016/j.cma.2016.05.038.

[24] H. Lian, P. Kerfriden, and S. Bordas. “Shape optimization directly from CAD: An isogeometric boundary element approach using T-splines”. In: *Computer Methods in Applied Mechanics and Engineering* 317 (Apr. 2017), pp. 1–41. DOI: 10.1016/j.cma.2016.11.012. URL: https://doi.org/10.1016/j.cma.2016.11.012.

[25] F. Auricchio et al. “Isogeometric collocation methods”. In: *Mathematical Models and Methods in Applied Sciences* 20.11 (Nov. 2010), pp. 2075–2107. DOI: 10.1142/s0218202510004878. URL: https://doi.org/10.1142/s0218202510004878.

[26] T. Jacquemin et al. “Taylor-Series Expansion Based Numerical Methods: A Primer, Performance Benchmarking and New Approaches for Problems with Non-smooth Solutions”. In: *Archives of Computational Methods in Engineering* (Aug. 2019). DOI: 10.1007/s11831-019-09357-5. URL: https://doi.org/10.1007/s11831-019-09357-5.

[27] Thibault Jacquemin and Stéphane P. A. Bordas. “A unified algorithm for the selection of collocation stencils for convex, concave and singular problems”. In: *International Journal for Numerical Methods in Engineering* (Apr. 2021). DOI: 10.1002/nme.6703. URL: https://doi.org/10.1002/nme.6703.

[28] Phillip Gould. *Introduction to linear elasticity*. New York: Springer, 2013. ISBN: 978-1-4614-4833-4.

[29] Xin-Lin Gao. “A general solution of an infinite elastic plate with an elliptic hole under biaxial loading”. In: *International Journal of Pressure Vessels and Piping* 67.1 (June 1996), pp. 95–

104. DOI: 10.1016/0308-0161(94)00173-1. URL: https://doi.org/10.1016/0308-0161(94)00173-1.

[30] I. Babuška and Manil Suri. "The *h-p* version of the finite element method with quasiuniform meshes". In: *ESAIM: Mathematical Modelling and Numerical Analysis* 21.2 (1987), pp. 199–238. DOI: 10.1051/m2an/1987210201991. URL: https://doi.org/10.1051/m2an/1987210201991.

[31] Mark Ainsworth and Bill Senior. "Aspects of an adaptive hp-finite element method: Adaptive strategy, conforming approximation and efficient solvers". In: *Computer Methods in Applied Mechanics and Engineering* 150.1-4 (Dec. 1997), pp. 65–87. DOI: 10.1016/s0045-7825(97)00101-1. URL: https://doi.org/10.1016/s0045-7825(97)00101-1.

[32] T. Belytschko, Y.Y. Lu, and L. Gu. "Element-free Galerkin methods". In: *International Journal for Numerical Methods in Engineering* 37.2 (Jan. 1994), pp. 229–256. DOI: 10.1002/nme.1620370205. URL: https://doi.org/10.1002/nme.1620370205.

[33] J.J. Benito, F. Ureña, and L. Gavete. "Influence of several factors in the generalized finite difference method". In: *Applied Mathematical Modelling* 25.12 (Dec. 2001), pp. 1039–1053. DOI: 10.1016/s0307-904x(01)00029-4. URL: https://doi.org/10.1016/s0307-904x(01)00029-4.

[34] Sławomir Milewski. "Meshless Finite Difference Method with Higher Order Approximation—Applications in Mechanics". In: *Archives of Computational Methods in Engineering* 19.1 (Feb. 2012), pp. 1–49. DOI: 10.1007/s11831-012-9068-y. URL: https://doi.org/10.1007/s11831-012-9068-y.

[35] P. Degond and S. Mas-Gallic. "The Weighted Particle Method for Convection-Diffusion Equations. Part 1: The Case of an Isotropic Viscosity". In: *Mathematics of Computation* 53.188 (Oct. 1989), p. 485. DOI: 10.2307/2008716. URL: https://doi.org/10.2307/2008716.

[36] J.D. Eldredge, A. Leonard, and T. Colonius. “A General Deterministic Treatment of Derivatives in Particle Methods”. In: *Journal of Computational Physics* 180.2 (Aug. 2002), pp. 686–709. DOI: 10.1006/jcph.2002.7112. URL: https://doi.org/10.1006/jcph.2002.7112.

[37] B. Schrader, S. Reboux, and I.F. Sbalzarini. “Choosing the Best Kernel: Performance Models for Diffusion Operators in Particle Methods”. In: *SIAM Journal on Scientific Computing* 34.3 (Jan. 2012), A1607–A1634. DOI: 10.1137/110835815. URL: https://doi.org/10.1137/110835815.

[38] Dassault Systemes. “Abaqus 2017”. In: (2017). URL: https://www.3ds.com/products-services/simulia/products/abaqus.

[39] P.R. Amestoy et al. “A fully asynchronous multifrontal solver using distributed dynamic scheduling”. In: *SIAM Journal on Matrix Analysis and Applications* 23.1 (2001), pp. 15–41.

[40] P.R. Amestoy et al. “Hybrid scheduling for the parallel solution of linear systems”. In: *Parallel Computing* 32.2 (2006), pp. 136–156.

[41] S. Balay et al. *PETSc Users Manual*. Tech. rep. ANL-95/11 - Revision 3.9. Argonne National Laboratory, 2018.

[42] S. Balay et al. “Efficient Management of Parallelism in Object Oriented Numerical Software Libraries”. In: *Modern Software Tools in Scientific Computing*. Ed. by E. Arge, A.M. Bruaset, and H.P. Langtangen. Birkhäuser Press, 1997, pp. 163–202.

[43] A. Dimitrov, H. Andrä, and E. Schnack. “Efficient computation of order and mode of corner singularities in 3D-elasticity”. In: *International Journal for Numerical Methods in Engineering* 52.8 (Nov. 2001), pp. 805–827. DOI: 10.1002/nme.230. URL: https://doi.org/10.1002/nme.230.

[44] W. Rachowicz, D. Pardo, and L. Demkowicz. “Fully automatic hp-adaptivity in three dimensions”. In: *Computer Methods in Applied Mechanics and Engineering* 195.37-40 (July 2006), pp. 4816–4842. DOI: 10.1016/j.cma.2005.08.022. URL: https://doi.org/10.1016/j.cma.2005.08.022.

[45] N. Zander et al. "The multi-level hp-method for three-dimensional problems: Dynamically changing high-order mesh refinement with arbitrary hanging nodes". In: *Computer Methods in Applied Mechanics and Engineering* 310 (Oct. 2016), pp. 252–277. DOI: 10.1016/j.cma.2016.07.007. URL: https://doi.org/10.1016/j.cma.2016.07.007.

[46] *OpenCASCADE: Open CASCADE Technology, 3D modeling & numerical simulation*. https://dev.opencascade.org/.

[47] Christophe Geuzaine and Jean-François Remacle. "Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities". In: *International Journal for Numerical Methods in Engineering* 79.11 (May 2009), pp. 1309–1331. DOI: 10.1002/nme.2579. URL: https://doi.org/10.1002/nme.2579.

[48] Rainald Löhner and Eugenio Oñate. "An advancing front point generation technique". In: *Communications in Numerical Methods in Engineering* 14.12 (Dec. 1998), pp. 1097–1108. DOI: 10.1002/(sici)1099-0887(199812)14:12<1097::aid-cnm183>3.0.co;2-7. URL: https://doi.org/10.1002/(sici)1099-0887(199812)14:12%3C1097::aid-cnm183%3E3.0.co;2-7.

[49] M. Shimrat. "Algorithm 112: Position of point relative to polygon". In: *Communications of the ACM* 5.8 (Aug. 1962), p. 434. DOI: 10.1145/368637.368653. URL: https://doi.org/10.1145/368637.368653.

[50] W.G. Chinn and N.E. Steenrod. *First Concepts of Topology*. The Mathematical Association of America, June 1966. DOI: 10.5948/upo9780883859339. URL: https://doi.org/10.5948/upo9780883859339.

[51] Joseph O'Rourke. *Computational Geometry in C Second Edition (Cambridge Tracts in Theoretical Computer Science (Paperback))*. Cambridge University Press, Oct. 1998. ISBN: 0521649765. URL: https://www.xarg.org/ref/a/0521649765/.

[52] Tomas Möller and Ben Trumbore. "Fast, Minimum Storage Ray-Triangle Intersection". In: *Journal of Graphics Tools* 2.1 (Jan. 1997), pp. 21–28. DOI: 10.1080/10867651.1997.10487468. URL: https://doi.org/10.1080/10867651.1997.10487468.

[53] Pierre Alliez, Stéphane Tayeb, and Camille Wormser. "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#PkgAABBTree.

[54] W. Schönauer. "Generation of Difference and Error Formulae of Arbitrary Consistency Order on an Unstructured Grid". In: *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik* 78.S3 (1998), pp. 1061–1062. DOI: 10.1002/zamm.19980781599. URL: https://doi.org/10.1002/zamm.19980781599.

[55] D.J. Kennett et al. "Semi-meshless stencil selection for anisotropic point distributions". In: *International Journal of Computational Fluid Dynamics* 26.9-10 (Oct. 2012), pp. 463–487. DOI: 10.1080/10618562.2012.744450. URL: https://doi.org/10.1080/10618562.2012.744450.

[56] Benjamin Seibold. "Minimal positive stencils in meshfree finite difference methods for the Poisson equation". In: *Computer Methods in Applied Mechanics and Engineering* 198.3-4 (Dec. 2008), pp. 592–601. DOI: 10.1016/j.cma.2008.09.001. URL: https://doi.org/10.1016/j.cma.2008.09.001.

[57] Oleg Davydov, Dang Thi Oanh, and Ngo Manh Tuong. "Octant-Based Stencil Selection for Meshless Finite Difference Methods in 3D". In: *Vietnam Journal of Mathematics* 48.1 (Aug. 2019), pp. 93–106. DOI: 10.1007/s10013-019-00364-4. URL: https://doi.org/10.1007/s10013-019-00364-4.

[58] M. Duflot. "Application des méthodes sans maillage en mécanique de la rupture". PhD thesis. 2004.

[59] V.P. Nguyen et al. "Meshless methods: A review and computer implementation aspects". In: *Mathematics and Computers in Simulation* 79.3 (Dec. 2008), pp. 763–813. DOI: 10.1016/j.matcom.2008.01.003. URL: https://doi.org/10.1016/j.matcom.2008.01.003.

[60] Ameneh Taleei and Mehdi Dehghan. “An efficient meshfree point collocation moving least squares method to solve the interface problems with nonhomogeneous jump conditions”. In: *Numerical Methods for Partial Differential Equations* 31.4 (Oct. 2014), pp. 1031–1053. DOI: 10.1002/num.21935. URL: https://doi.org/10.1002/num.21935.

[61] Sang-Ho Lee and Young-Cheol Yoon. “Meshfree point collocation method for elasticity and crack problems”. In: *International Journal for Numerical Methods in Engineering* 61.1 (Aug. 2004), pp. 22–48. DOI: 10.1002/nme.1053. URL: https://doi.org/10.1002/nme.1053.

[62] Sang-Ho Lee, Kyeong-Hwan Kim, and Young-Cheol Yoon. “Particle difference method for dynamic crack propagation”. In: *International Journal of Impact Engineering* 87 (Jan. 2016), pp. 132–145. DOI: 10.1016/j.ijimpeng.2015.06.001. URL: https://doi.org/10.1016/j.ijimpeng.2015.06.001.

[63] D. Organ et al. “Continuous meshless approximations for nonconvex bodies by diffraction and transparency”. In: *Computational Mechanics* 18.3 (July 1996), pp. 225–235. DOI: 10.1007/bf00369940. URL: https://doi.org/10.1007/bf00369940.

[64] Xiaoying Zhuang, Charles Augarde, and Stéphane Bordas. “Accurate fracture modelling using meshless methods, the visibility criterion and level sets: Formulation and 2D modelling”. In: *International Journal for Numerical Methods in Engineering* 86.2 (Jan. 2011), pp. 249–268. DOI: 10.1002/nme.3063. URL: https://doi.org/10.1002/nme.3063.

[65] Z.R. Li, C.W. Lim, and L.H. He. “Stress concentration around a nano-scale spherical cavity in elastic media: effect of surface stress”. In: *European Journal of Mechanics - A/Solids* 25.2 (Mar. 2006), pp. 260–270. DOI: 10.1016/j.euromechsol.2005.09.005. URL: https://doi.org/10.1016/j.euromechsol.2005.09.005.

[66] R. Sharma et al. “An improved stress recovery technique for low-order 3D finite elements”. In: *International Journal for Numerical Methods in Engineering* 114.1 (Jan. 2018), pp. 88–103. DOI: 10.1002/nme.5734. URL: https://doi.org/10.1002/nme.5734.

[67] Electricité de France. *Finite element $code\_aster$, Analysis of structures and thermomechanics for studies and research*. Open source on www.code-aster.org.

[68] N. Sukumar. "Voronoi cell finite difference method for the diffusion operator on arbitrary unstructured grids". In: *International Journal for Numerical Methods in Engineering* 57.1 (2003), pp. 1–34. DOI: 10.1002/nme.664. URL: https://doi.org/10.1002/nme.664.

[69] J.X. Zhou et al. "A subdomain collocation method based on Voronoi domain partition and reproducing kernel approximation". In: *Computer Methods in Applied Mechanics and Engineering* 196.13-16 (Mar. 2007), pp. 1958–1967. DOI: 10.1016/j.cma.2006.10.011. URL: https://doi.org/10.1016/j.cma.2006.10.011.

[70] E. Oñate. "Derivation of stabilized equations for numerical solution of advective-diffusive transport and fluid flow problems". In: *Computer Methods in Applied Mechanics and Engineering* 151.1-2 (Jan. 1998), pp. 233–265. DOI: 10.1016/s0045-7825(97)00119-9. URL: https://doi.org/10.1016/s0045-7825(97)00119-9.

[71] E. Oñate, F. Perazzo, and J. Miquel. "Advances in the stabilized finite point method for structural mechanics". In: München, Germany, Sept. 1999.

[72] E. Oñate, F. Perazzo, and J. Miquel. "A finite point method for elasticity problems". In: *Computers & Structures* 79.22-25 (Sept. 2001), pp. 2151–2163. DOI: 10.1016/s0045-7949(01)00067-0. URL: https://doi.org/10.1016/s0045-7949(01)00067-0.

[73] J. J. Benito et al. "A posteriorierror estimator and indicator in generalized finite differences. Application to improve the approximated solution of elliptic PDEs". In: *International Journal of Computer Mathematics* 85.3-4 (Apr. 2008), pp. 359–370. DOI: 10.1080/00207160601167052. URL: https://doi.org/10.1080/00207160601167052.

[74] Luis Gavete et al. "An Approach to Refinement of Irregular Clouds of Points Using Generalized Finite Differences". In: *Mathematical Problems in Engineering* 2015 (2015), pp. 1–9. DOI: 10.1155/2015/283757. URL: https://doi.org/10.1155/2015/283757.

[75] Pratik Suchde and Jörg Kuhnert. "A meshfree generalized finite difference method for surface PDEs". In: *Computers & Mathematics with Applications* 78.8 (2019), pp. 2789–2805. ISSN: 0898-1221. DOI: https://doi.org/10.1016/j.camwa.2019.04.030. URL: http://www.sciencedirect.com/science/article/pii/S0898122119302469.

[76] O. Davydov and D. Thi Oanh. "Adaptive meshless centres and RBF stencils for Poisson equation". In: *Journal of Computational Physics* 230.2 (Jan. 2011), pp. 287–304. DOI: 10.1016/j.jcp.2010.09.005. URL: https://doi.org/10.1016/j.jcp.2010.09.005.

[77] Dang Thi Oanh, Oleg Davydov, and Hoang Xuan Phu. "Adaptive RBF-FD method for elliptic problems with point singularities in 2D". In: *Applied Mathematics and Computation* 313 (2017), pp. 474–497. ISSN: 0096-3003. DOI: https://doi.org/10.1016/j.amc.2017.06.006. URL: https://www.sciencedirect.com/science/article/pii/S0096300317304186.

[78] Jure Slak. "Adaptive RBF-FD method". PhD thesis. Univerza v Ljubljani, Fakulteta za matematiko in fiziko, 2020.

[79] O. C. Zienkiewicz and J. Z. Zhu. "A simple error estimator and adaptive procedure for practical engineerng analysis". In: *International Journal for Numerical Methods in Engineering* 24.2 (Feb. 1987), pp. 337–357. DOI: 10.1002/nme.1620240206. URL: https://doi.org/10.1002/nme.1620240206.

[80] S. Bordas and M. Duflot. "Derivative recovery and a posteriori error estimate for extended finite elements". In: *Computer Methods in Applied Mechanics and Engineering* 196.35 (2007), pp. 3381–3399. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2007.03.011. URL: https://www.sciencedirect.com/science/article/pii/S0045782507001417.

[81] M. Duflot and S. Bordas. "A posteriorierror estimation for extended finite elements by an extended global recovery". In: *International Journal for Numerical Methods in Engineering* 76.8 (Nov. 2008), pp. 1123–1138. DOI: 10.1002/nme.2332. URL: https://doi.org/10.1002/nme.2332.

[82] Juan J Ródenas et al. "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 (ECCOMAS 2008).* CINME. 2008.

[83] T. Driscoll and A. Heryudono. "Adaptive residual subsampling methods for radial basis function interpolation and collocation problems". In: *Comput. Math. Appl.* 53 (2007), pp. 927–939.

[84] Jaeyoun Oh. "Adaptive Meshfree Methods for Partial Differential Equations". PhD thesis. University of Southern Mississippi, Aug. 2018.

[85] J.J. Benito et al. "An h-adaptive method in the generalized finite differences". In: *Computer Methods in Applied Mechanics and Engineering* 192.5-6 (Jan. 2003), pp. 735–759. DOI: 10.1016/s0045-7825(02)00594-7. URL: https://doi.org/10.1016/s0045-7825(02)00594-7.

[86] Jure Slak and Gregor Kosec. "Adaptive radial basis function–generated finite differences method for contact problems". In: *International Journal for Numerical Methods in Engineering* 119.7 (Apr. 2019), pp. 661–686. DOI: 10.1002/nme.6067. URL: https://doi.org/10.1002/nme.6067.

[87] Pratik Suchde et al. "A flux conserving meshfree method for conservation laws". In: *International Journal for Numerical Methods in Engineering* 112.3 (2017), pp. 238–256. ISSN: 1097-0207. DOI: 10.1002/nme.5511. URL: http://dx.doi.org/10.1002/nme.5511.

[88] Pratik Suchde and Jörg Kuhnert. "A fully Lagrangian meshfree framework for PDEs on evolving surfaces". In: *Journal of Computational Physics* 395 (2019), pp. 38–59. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2019.06.031. URL: https://www.sciencedirect.com/science/article/pii/S0021999119304395.

[89] T.J. Liszka, C.A.M. Duarte, and W.W. Tworzydlo. "hp-Meshless cloud method". In: *Computer Methods in Applied Mechanics and Engineering* 139.1-4 (Dec. 1996), pp. 263–288. DOI: 10.1016/s0045-7825(96)01086-9. URL: https://doi.org/10.1016/s0045-7825(96)01086-9.

[90] C. Armando Duarte and J. Tinsley Oden. "H-p clouds—anh-p meshless method". In: *Numerical Methods for Partial Differential Equations* 12.6 (Nov. 1996), pp. 673–705. DOI: 10.1002/(sici)1098-2426(199611)12:6<673::aid-num3>3.0.co;2-p.

URL: https://doi.org/10.1002/(sici)1098-2426(199611)12:6%3C673::aid-num3%3E3.0.co;2-p.

[91] Mitja Jancic, Jure Slak, and Gregor Kosec. "p-refined RBF-FD solution of a Poisson problem". In: *2021 6th International Conference on Smart and Sustainable Technologies (SpliTech)* (2021), pp. 01–06.

[92] Willy Dorfler. "A Convergent Adaptive Algorithm for Poisson's Equation". In: *SIAM Journal on Numerical Analysis* 33.3 (1996), pp. 1106–1124. ISSN: 00361429. URL: http://www.jstor.org/stable/2158497.

[93] R. Bulle et al. "Hierarchical a posteriori error estimation of Bank-Weiser type in the FEniCS Project". In: *ArXiv* abs/2102.04360 (2021).

[94] Medusa: Coordinate Free Mehless Method implementation. *Ghost nodes (theory) — Medusa: Coordinate Free Mehless Method implementation*, 2019.

[95] James W. Demmel et al. "A supernodal approach to sparse partial pivoting". In: *SIAM J. Matrix Analysis and Applications* 20.3 (1999), pp. 720–755.

[96] James W. Demmel, John R. Gilbert, and Xiaoye S. Li. "An Asynchronous Parallel Supernodal Algorithm for Sparse Gaussian Elimination". In: *SIAM J. Matrix Analysis and Applications* 20.4 (1999), pp. 915–952.

[97] Xiaoye S. Li and James W. Demmel. "SuperLU_DIST: A Scalable Distributed-Memory Sparse Direct Solver for Unsymmetric Linear Systems". In: *ACM Trans. Mathematical Software* 29.2 (June 2003), pp. 110–140.

[98] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Jan. 2003. DOI: 10.1137/1.9780898718003. URL: https://doi.org/10.1137/1.9780898718003.

[99] Ilse C. F. Ipsen and Carl D. Meyer. "The Idea Behind Krylov Methods". In: *The American Mathematical Monthly* 105.10 (Dec. 1998), pp. 889–899. DOI: 10.1080/00029890.1998.12004985. URL: https://doi.org/10.1080/00029890.1998.12004985.

[100] Jörg Liesen and Zdenek Strakos. *Krylov Subspace Methods*. Oxford University Press, Oct. 2012. DOI: 10.1093/acprof:oso/9780199655410.001.0001. URL: https://doi.org/10.1093/acprof:oso/9780199655410.001.0001.

[101] Youcef Saad and Martin H. Schultz. "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems". In: *SIAM Journal on Scientific and Statistical Computing* 7.3 (July 1986), pp. 856–869. DOI: 10.1137/0907058. URL: https://doi.org/10.1137/0907058.

[102] R. Fletcher. "Conjugate gradient methods for indefinite systems". In: *Lecture Notes in Mathematics*. Springer Berlin Heidelberg, 1976, pp. 73–89. DOI: 10.1007/bfb0080116. URL: https://doi.org/10.1007/bfb0080116.

[103] R.P. Fedorenko. "A relaxation method for solving elliptic difference equations". In: *USSR Computational Mathematics and Mathematical Physics* 1.4 (Jan. 1962), pp. 1092–1096. DOI: 10.1016/0041-5553(62)90031-9. URL: https://doi.org/10.1016/0041-5553(62)90031-9.

[104] Achi Brandt. "Multi-level adaptive technique (MLAT) for fast numerical solution to boundary value problems". In: *Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics*. Springer Berlin Heidelberg, 1973, pp. 82–89. DOI: 10.1007/bfb0118663. URL: https://doi.org/10.1007/bfb0118663.

[105] Wolfgang Hackbusch. "Ein iteratives Verfahren zur schnellen Auflösung elliptischer Randwertprobleme". In: *Report 76-12* (Nov. 1976).

[106] Ulrike Meier Yang. "Parallel Algebraic Multigrid Methods — High Performance Preconditioners". In: *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, 2006, pp. 209–236. DOI: 10.1007/3-540-31619-1_6. URL: https://doi.org/10.1007/3-540-31619-1_6.

[107] K. Stüben. *Algebraic Multigrid (AMG): An Introduction with Applications*. GMD-Report. GMD-Forschungszentrum Informationstechnik, 1999. URL: https://books.google.fr/books?id=rzLiGwAACAAJ.

[108] K. Hameyer et al. "Algebraic multigrid for complex symmetric systems". In: *IEEE Transactions on Magnetics* 36.4 (July 2000), pp. 1535–1538. DOI: 10.1109/20.877730. URL: https://doi.org/10.1109/20.877730.

[109] The Trilinos Project Team. *The Trilinos Project Website*.

[110] Robert D. Falgout and Ulrike Meier Yang. "hypre: A Library of High Performance Preconditioners". In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2002, pp. 632–641. DOI: 10.1007/3-540-47789-6_66. URL: https://doi.org/10.1007/3-540-47789-6_66.

[111] M. Adams. PETSc user meeting. June 2016.

[112] Michael Luby. "A Simple Parallel Algorithm for the Maximal Independent Set Problem". In: *SIAM Journal on Computing* 15.4 (Nov. 1986), pp. 1036–1053. DOI: 10.1137/0215074. URL: https://doi.org/10.1137/0215074.

[113] Mark T. Jones and Paul E. Plassmann. "A Parallel Graph Coloring Heuristic". In: *SIAM J. Sci. Comput.* 14 (1993), pp. 654–669.

[114] Andrew J Cleary et al. "Coarse-grid selection for parallel algebraic multigrid". In: *International Symposium on Solving Irregularly Structured Problems in Parallel*. Springer. 1998, pp. 104–115.

[115] Hans De Sterck, Ulrike Meier Yang, and Jeffrey J. Heys. "Reducing Complexity in Parallel Algebraic Multigrid Preconditioners". In: *SIAM Journal on Matrix Analysis and Applications* 27.4 (Jan. 2006), pp. 1019–1039. DOI: 10.1137/040615729. URL: https://doi.org/10.1137/040615729.

[116] M. FLEMING et al. "ENRICHED ELEMENT-FREE GALERKIN METHODS FOR CRACK TIP FIELDS". In: *International Journal for Numerical Methods in Engineering* 40.8 (Apr. 1997), pp. 1483–1504. DOI: 10.1002/(sici)1097-0207(19970430)40:8<1483::aid-nme123>3.0.co;2-6. URL: https://doi.org/10.1002/(sici)1097-0207(19970430)40:8%3C1483::aid-nme123%3E3.0.co;2-6.

[117] Marc Duflot and Hung Nguyen-Dang. "A meshless method with enriched weight functions for fatigue crack growth". In: *International Journal for Numerical Methods in Engineering* 59.14 (Mar. 2004), pp. 1945–1961. DOI: 10.1002/nme.948. URL: https://doi.org/10.1002/nme.948.

[118] S. Bordas, G. Zi, and T. Rabczuk. "Three-dimensional non-linear fracture mechanics by enriched meshfree methods without asymptotic enrichment". In: *IUTAM Symposium on Discretization Methods for Evolving Discontinuities*. Springer Netherlands, 2007, pp. 21–36.

[119] M.H. Afshar, J. Amani, and M. Naisipour. "A node enrichment adaptive refinement in Discrete Least Squares Meshless method for solution of elasticity problems". In: *Engineering Analysis with Boundary Elements* 36.3 (Mar. 2012), pp. 385–393. DOI: 10.1016/j.enganabound.2011.08.012. URL: https://doi.org/10.1016/j.enganabound.2011.08.012.

[120] N. Rahimi et al. "CAD Model Simplification Error Estimation for Electrostatics Problems". In: *SIAM Journal on Scientific Computing* 40.1 (Jan. 2018), B196–B227. DOI: 10.1137/16m1078641. URL: https://doi.org/10.1137/16m1078641.

[121] Atul Thakur, Ashis Gopal Banerjee, and Satyandra K. Gupta. "A survey of CAD model simplification techniques for physics-based simulation applications". In: *Computer-Aided Design* 41.2 (Feb. 2009), pp. 65–80. DOI: 10.1016/j.cad.2008.11.009. URL: https://doi.org/10.1016/j.cad.2008.11.009.

[122] Florence Danglade, Jean-Philippe Pernot, and Philippe Véron. "On the use of Machine Learning to Defeature CAD Models for Simulation". In: *Computer-Aided Design and Applications* 11.3 (Dec. 2013), pp. 358–368. DOI: 10.1080/16864360.2013.863510. URL: https://doi.org/10.1080/16864360.2013.863510.

[123] S. Prudhomme and J.T. Oden. "On goal-oriented error estimation for elliptic problems: application to the control of pointwise errors". In: *Computer Methods in Applied Mechanics and Engineering* 176.1 (1999), pp. 313–331. ISSN: 0045-7825. DOI: https://doi.org/

10.1016/S0045-7825(98)00343-0. URL: https://www.sciencedirect.com/science/article/pii/S0045782598003430.

[124] Ludovic Chamoin and Frédéric Legoll. "Goal-oriented error estimation and adaptivity in MsFEM computations". In: *Computational Mechanics* 67.4 (Mar. 2021), pp. 1201–1228. DOI: 10.1007/s00466-021-01990-x. URL: https://doi.org/10.1007/s00466-021-01990-x.

[125] F Aghighi, OM Ebadati, and H Aghighi. "Classification of LiDAR Points Cloud Using Markov Random Field and Machine Learning Techniques". In: *Iranian Journal of Remote Sensing & GIS* 9.2 (2018), pp. 41–60.

[126] Farzaneh Aghighi, Hossein Aghighi, and Omid Mahdi Ebadati. "Conditional Random Fields for Airborne Lidar Point Cloud Classification in Urban Area". In: *Engineering Journal of Geospatial Information Technology* 7.4 (2020), pp. 139–156.

[127] Lingxiao Li et al. "Supervised Fitting of Geometric Primitives to 3D Point Clouds". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2019). DOI: 10.1109/cvpr.2019.00276. URL: http://dx.doi.org/10.1109/CVPR.2019.00276.

[128] Baptiste Angles. "Geometric modeling with primitives". PhD thesis. Université Paul Sabatier -, Toulouse III, 2019.

[129] Tsahi Saporta and Andrei Sharf. "Unsupervised recursive deep fitting of 3D primitives to points". In: *Computers & Graphics* (2021). ISSN: 0097-8493. DOI: https://doi.org/10.1016/j.cag.2021.10.020. URL: https://www.sciencedirect.com/science/article/pii/S0097849321002314.

[130] Q. Zhan, Yubin Liang, and Yinghui Xiao. "Color-based segmentation of point clouds". In: *ISPRS Laser Scanning Workshop* 38 (July 2009).

[131] Mohammad Awrangjeb, Chunsun Zhang, and Clive S. Fraser. "Automatic extraction of building roofs using LIDAR data and multispectral imagery". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 83 (Sept. 2013), pp. 1–18. DOI: 10.1016/j.

isprsjprs.2013.05.006. URL: https://doi.org/10.1016/j.isprsjprs.2013.05.006.

[132] Yahya Alshawabkeh. "Linear feature extraction from point cloud using color information". In: *Heritage Science* 8.1 (Mar. 2020). DOI: 10.1186/s40494-020-00371-6. URL: https://doi.org/10.1186/s40494-020-00371-6.

[133] Will Schroeder. *The visualization toolkit : an object-oriented approach to 3D graphics*. Clifton Park, N.Y: Kitware, 2006. ISBN: 978-1-930934-19-1.

[134] 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.

[135] C.H. Rycroft. "VORO++: A three-dimensional Voronoi cell library in C++". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 19.4 (Dec. 2009), p. 041111. DOI: 10.1063/1.3215722. URL: https://doi.org/10.1063/1.3215722.