ECOLE POLYTECHNIQUE FEDERALE DE LAUSANNE

On the preconditioning of the INTERNODES matrix for applications in contact mechanics

Student:	Yannis Voet	
Supervisors:	Dr. Guillaume Anciaux	Section of Civil Engineering
	Prof. Simone Deparis	Section of Mathematics
Expert:	Prof. Paola Gervasio	

Master thesis

August 13, 2021

Contents

Introduction

Contact mechanics studies the interaction of bodies as they move closer to each other. The nature of this interaction may be very different but typically involves energy exchanges in the form of heat, electric charge or strain for example. This last type of energy exchange will be of special interest to us in this thesis. The theory of elasticity will be used to find the deformed configuration of bodies coming into contact. Common academic problems include contact between a deformable body and a rigid foundation or two deformable bodies. Contrary to single-body elasticity problems, contact problems are intrinsically nonlinear even for the simplest linear constitutive model. The nonlinearity is tied to the unknown contact interface and the resulting inequality constraints. Mathematicians have spent a great deal of effort on those problems and entire books have been devoted to them. We should point out that the existence and uniqueness of contact problems has only been proved in special cases. Yet, the industry needs are such that even without the well-posedness of contact problems, computational scientists have developed a wide range of solution methods. Unsurprisingly, the inequality constraints tightly bound computational contact mechanics with optimization. In addition, finite element discretizations are used to approximate the infinite-dimensional variational problem. Yet again, additional challenges arise in the computations which one does not encounter in a single-body elasticity problem. Indeed, different bodies lead to different domains and the solution to the underlying partial differential equation must be coupled across these different domains. This task is hindered by a priori independent discretizations of the bodies, which lead to nonconforming meshes at the interfaces. The underlying type of nonconformity is inherently geometric, meaning there may exist small gaps or overlaps at the discrete interfaces. The issue has been traditionally addressed using the mortar finite element method. It is based on projection techniques for transferring information across the interface. However, this method is known to be difficult to implement and requires a few ad hoc strategies to meet specific challenges. These include the use of numerical quadrature and the special treatment of cross-points when more than two subdomains meet. The INTERNODES (INTERpolation for NOn-conforming DEcompositionS) method (Deparis et al., 2016) appears as a very promising alternative for solving problems in contact mechanics. It is a flexible interpolation based method which overcomes a great deal of the implementation issues of the mortar method and was shown to be at least as accurate (Gervasio and Quarteroni, 2018). Preliminary work in computational contact mechanics showed that the INTERNODES method could be successfully applied but also revealed several challenges in efficiently solving the sequence of linear systems arising from the method (Günther-Hanssen, 2020).

In this work, we investigate preconditioning techniques for solving these linear systems. We propose an efficient preconditioner which achieves very fast convergence independent of the mesh size. The performance of the method relies on a single matrix factorization. The method performs best when this factorization is computed exactly. Nevertheless, it still performs well for inexact factorizations. The remainder of the thesis is structured as follows. Interpolation being one of the key features of the INTERNODES method, Chapter 1 provides the necessary background. The emphasis is set on a specific type of interpolation called radial basis function interpolation. The section extends the work of Deparis et al. (2014) and discusses implementation aspects. Chapter 2 covers the mathematical modeling of contact problems in its most basic form. A linear elastic constitutive model is assumed for simplicity. From the strong form of the differential problem, the weak form is derived followed by a finite element discretization. The section covers the application of the INTERNODES method to contact problems and further discusses implementation aspects. Chapter 3 gathers some of the properties of the INTERNODES matrix which is an essential step towards preconditioning. Solution techniques for large scale linear systems are discussed in Chapter 4 with an emphasis on iterative methods. Chapter 5 presents preconditioning techniques tailored for the INTERNODES matrix. This chapter represents the bulk of the thesis and ends with numerical experiments illustrating the effectiveness of the developed preconditioners. The most successful preconditioning techniques are discussed in this chapter. However, some other attempts have been listed in Appendix A for the record. Finally, Chapter 6 concludes with an application of the INTERNODES method to a classic problem of contact mechanics.

Chapter 1

Radial basis function interpolation

1.1 Introduction

Interpolation is at the heart of the INTERNODES method and distinguishes it from mortar finite element methods which are based on projection techniques. In the INTERNODES method, interpolation can be either based on Lagrange or radial basis function (RBF) interpolants (Deparis et al., 2016). However, RBF interpolants are preferred for problems with geometric non-conformities. Since this is one of the essential types of non-conformities encountered in contact mechanics problems, these interpolants will be discussed thoroughly in this chapter. Although the discussion is mostly theoretical, the resulting properties have several interesting computational consequences which will be exploited in the design of preconditioners. Part of the discussion in this chapter extends the work of Deparis et al. (2014) who introduced several modifications to the classic RBF interpolation. RBF interpolation is widely used in a variety of applications involving the interpolation of scattered data. Such applications include the numerical solution of partial differential equations and neural networks. The survey paper of Buhmann (2000) discusses both globally and locally supported radial basis functions and provides an overview of their theoretical properties including convergence. In particular, compactly supported radial basis functions have found fertile grounds for very large interpolation problems. They will be at the center of attention in this chapter.

The chapter is organized as follows: in Section 1.2, a short introduction to radial basis function interpolation is given and the modifications introduced by Deparis et al. are discussed. Necessary conditions for the modifications to be well defined were already established by the same authors. In Section 1.3, sufficient conditions are derived, based on which some theoretical properties of the interpolation matrices are proved. Finally, Section 1.4 discusses algorithmic aspects used to ensure the interpolation matrices satisfy the sufficient conditions.

1.2 Radial basis function interpolation

Let $\Xi = \{\boldsymbol{\xi}_m\}_{m=1}^{M}$ be a set of interpolation points where some function evaluations are known. We define the global interpolant $\Pi_f(\mathbf{x})$ of a function $f\colon \mathbb{R}^d \to \mathbb{R}$ as

$$\Pi_f(\mathbf{x}) = \sum_{m=1}^{M} \gamma_m^f \phi(\|\mathbf{x} - \boldsymbol{\xi}_m\|, r)$$

where $\gamma_m^f \in \mathbb{R}$ for $m = 1, \ldots, M$ are coefficients and ϕ is a radial basis function which depends on the distance from the interpolation point $\|\mathbf{x} - \boldsymbol{\xi}_m\|$ and is parameterized by a radius r. Typical choices for radial basis functions are listed in Table 1.1. In the table, $\delta = \frac{\|\mathbf{x}\|}{r}$ is used to denote the relative distance from the center.

Name	ϕ
Thin-plate splines	$\delta^2 \ln \delta$
Inverse multiquadratic	$\frac{1}{r\sqrt{\delta^2+1}}$
Gaussian splines	$\mathrm{e}^{-\delta^2}$
Wendland C^2	$(1-\delta)^4(1+4\delta)$

Table 1.1: Common radial basis functions

Thin-plate splines, inverse multiquadratic and Gaussian radial basis functions are all globally supported whereas the Wendland C^2 radial basis functions are locally supported and we consider only the values of $\delta \in [0, 1]$. This property is very interesting because it leads to sparse interpolation matrices.

Let $\mathbf{f}_\xi \in \mathbb{R}^M$ be the vector containing function evaluations at the interpolation points $\{f(\boldsymbol{\xi}_m)\}_{m=1}^M$ and $\boldsymbol{\gamma}^f \in \mathbb{R}^M$ be the vector of coefficients. The interpolation conditions $\Pi_f(\boldsymbol{\xi}_m) = f(\boldsymbol{\xi}_m)$ for $m = 1, \ldots, M$ lead to the linear system

$$\Phi_{MM}\boldsymbol{\gamma}^f = \mathbf{f}_\xi$$

where $\Phi_{MM} \in \mathbb{R}^{M\times M}$ is such that $(\Phi_{MM})_{ij} = \phi(\|\boldsymbol{\xi}_i - \boldsymbol{\xi}_j\|, r)$. For many radial basis functions of practical interest, the resulting coefficient matrix Φ_{MM} is positive definite and thus the solution to the linear system is unique. This is the case when using for instance Gaussian or inverse multiquadratic radial basis functions. This property is also satisfied for the Wendland C^2 basis functions (Wendland, 1995). However, for thin-plate splines, a low degree polynomial term must be added to the interpolant to ensure the linear system is uniquely solvable (Buhmann, 2000). We will restrict the discussion here to cases where Φ_{MM} is positive definite.

Let $\Lambda = \{\boldsymbol{\zeta}_n\}_{n=1}^N$ be a set of points where the interpolant is to be evaluated. Let $\mathbf{f}_\zeta \in \mathbb{R}^N$ be the vector containing the evaluations $\{\Pi(\boldsymbol{\zeta}_n)\}_{n=1}^N$. Then

$$\mathbf{f}_\zeta = \Phi_{NM}\boldsymbol{\gamma}^f = \Phi_{NM}\Phi_{MM}^{-1}\mathbf{f}_\xi$$

and $\Phi_{NM}\Phi_{MM}^{-1}$ defines an interpolation matrix. Deparis et al. (2014) introduced two modifications:

1. A localized radius was chosen for the radial basis functions. Thus,

$$\begin{aligned}(\Phi_{MM})_{ij} &= \phi(\|\boldsymbol{\xi}_i - \boldsymbol{\xi}_j\|, r_j) \quad i, j = 1, \ldots, M\\ (\Phi_{NM})_{ij} &= \phi(\|\boldsymbol{\zeta}_i - \boldsymbol{\xi}_j\|, r_j) \quad i = 1, \ldots, N \; j = 1, \ldots, M\end{aligned}$$

   Using a localized radius allows to take advantage of a nonuniform distribution of interpolation points. In regions where the density of points is high, the radius can be reduced without affecting much the accuracy and allows to spare storage for the interpolation matrices.

2. A rescaling of the radial basis functions was introduced which enables the exact interpolation of constant functions. Let $\mathbf{1}_\xi \in \mathbb{R}^M$ be a vector containing only ones. We define

$$\begin{aligned}\hat{\mathbf{f}}_\zeta &= \Phi_{NM}\Phi_{MM}^{-1}\mathbf{f}_\xi\\ \mathbf{g}_\zeta &= \Phi_{NM}\Phi_{MM}^{-1}\mathbf{1}_\xi\end{aligned}$$

   and set $f_{\zeta_i} = \frac{\hat{f}_{\zeta_i}}{g_{\zeta_i}} \quad i = 1, 2, \ldots, N$. This rescaling is in fact equivalent to defining a new interpolant $\bar{\Pi}_f(\mathbf{x}) = \frac{\Pi_f(\mathbf{x})}{\Pi_1(\mathbf{x})}$. Thus, if Π was initially a polynomial function, then $\bar{\Pi}$ is a rational function. On the algebraic side, the rescaling is equivalent to defining the interpolation matrix

$$R_{NM} = D_{NN}^{-1}\Phi_{NM}\Phi_{MM}^{-1}$$

   where $D_{NN} = \mathrm{diag}(\mathbf{g}_\zeta)$ is a diagonal matrix formed by the components of the vector $\mathbf{g}_\zeta$. The matrix R_{NM} will be used extensively throughout the next chapters.

The numerical results presented by the authors showed that the rescaling greatly improved the accuracy of the interpolation. However, the localized radius unfortunately destroys the symmetry of the matrix Φ_{MM}. Moreover, it raises two important concerns:

1. The matrix Φ_{MM} must be invertible.

2. The vector $\mathbf{g}_\zeta$ must not contain a zero entry.

Clearly, a necessary condition for the second issue not to occur is that Φ_{NM} does not contain a row of zeros. It is equivalent to ensuring that each evaluation point $\zeta_n \quad n = 1, 2, \ldots, N$ lies in the support of at least one basis function. Our goal is to find sufficient conditions to safely avoid these two issues.

1.3 Properties of the radial basis function matrices

In this section, we will first find an easy sufficient condition for the matrix Φ_{MM} to be invertible. We will then gather some of the properties of the matrix Φ_{MM} and its inverse. Finally, we will show that under the same condition, the second issue is safely avoided without further assumptions. Some of the results of this section are not used in the next chapters. They have nevertheless been included to support ongoing research. The essential results needed for the remaining chapters are summarized at the end of this section. One of the important concepts of this section is the one of (strict) diagonal dominance. A matrix $A \in \mathbb{C}^{n\times n}$ is said to be strictly diagonally dominant by rows if

$$|a_{ii}| > \sum_{\substack{j=1\\j\neq i}}^{n} |a_{ij}| \quad i = 1, 2, \ldots, n$$

It is strictly diagonally dominant by columns if

$$|a_{ii}| > \sum_{\substack{j=1\\j\neq i}}^{n} |a_{ji}| \quad i = 1, 2, \ldots, n$$

Before proceeding any further, let us recall the important Gershgoring theorem (Gershgorin, 1931).

Theorem 1.1: Gershgorin theorem

Let $A \in \mathbb{C}^{n\times n}$ and let λ be an eigenvalue of A. Then,

$$\lambda \in \bigcup_{i=1}^{n} \mathcal{D}_i$$

where $\mathcal{D}_i = \{z \in \mathbb{C} \colon |z - a_{ii}| \leq \sum_{\substack{j=1\\j\neq i}}^{n} |a_{ij}|\}$ is the ith Gershgorin disc.

Proof. Let $(\lambda, \mathbf{x})$ be an eigenpair of A and let i be such that $|x_i| = \max_{j=1,\ldots,n} |x_j| = \|\mathbf{x}\|_\infty$. Obviously, $x_i \neq 0$ because $\mathbf{x} \neq \mathbf{0}$. Then, the ith equation of $A\mathbf{x} = \lambda\mathbf{x}$ reads

$$\lambda x_i = \sum_{j=1}^{n} a_{ij} x_j$$

Hence,

$$(\lambda - a_{ii}) x_i = \sum_{\substack{j=1\\j\neq i}}^{n} a_{ij} x_j$$

Taking the modulus and using the triangle inequality leads to

$$|\lambda - a_{ii}||x_i| \leq \sum_{\substack{j=1\\j\neq i}}^{n} |a_{ij}||x_j| \leq |x_i| \sum_{\substack{j=1\\j\neq i}}^{n} |a_{ij}|$$

Therefore,

$$|\lambda - a_{ii}| \leq \sum_{\substack{j=1\\j\neq i}}^{n} |a_{ij}|$$

This equation is a disc of center a_{ii} and radius $\sum_{\substack{j=1\\j\neq i}}^{n} |a_{ij}|$. Obviously, for a given eigenvalue we do not know what the index i is. Therefore, by considering all $i = 1, 2, \ldots, n$, we are guaranteed that the eigenvalue belongs to at least one of the discs. □

Remark: For real matrices, one can apply the same proof steps to A^T instead of A since the eigenvalues of A and A^T are the same. The region of the complex plane can then be defined as the union of row and column discs.

We are seeking sufficient conditions for the matrix Φ_{MM} to be invertible. In that respect, the following theorem will be useful.

Theorem 1.2: Levy–Desplanques theorem

Let $A \in \mathbb{C}^{n\times n}$ be a strictly diagonally dominant matrix by rows or columns. Then, A is invertible.

Proof. A is invertible if and only if its adjoint A^* is. Thus, without loss of generality, we may restrict ourselves to diagonal dominance by rows. The proof is then a straightforward consequence of Gershgorin's theorem. Indeed, using the strict diagonal dominance property, the definition of the ith Gershgorin disc reduces to

$$\mathcal{D}_i = \{z \in \mathbb{C} \colon |z - a_{ii}| < |a_{ii}|\}$$

Since $0 \notin \mathcal{D}_i$ and $\lambda \in \bigcup_{i=1}^{n} \mathcal{D}_i$, then λ cannot be equal to zero. Therefore, the matrix is invertible. □

Consequently, ensuring that Φ_{MM} is strictly diagonally dominant by rows is sufficient for its invertibility. Additionally, if this property is satisfied, the spectrum of Φ_{MM} is better characterized. The following corollary provides an interesting result.

Corollary 1.1

Let $A \in \mathbb{R}^{n\times n}$ be a strictly diagonally dominant matrix by rows which additionally satisfies $a_{ii} = 1$ for $i = 1, \ldots, n$ and let λ denote an eigenvalues A. Then,

$$\begin{aligned} 0 < \Re(\lambda) < 2 \\ -1 < \Im(\lambda) < 1 \end{aligned}$$

Proof. The Gershgorin discs all reduce to

$$\mathcal{D}_i = \{z \in \mathbb{C} \colon |z - 1| < 1\} \quad i = 1, \ldots, n$$

which are discs of unit radius centered at 1. The result naturally follows. □

Lemma 1.1

Let $A \in \mathbb{R}^{n\times n}$ be a strictly diagonally dominant matrix by rows which additionally satisfies $a_{ii} = 1$ for $i = 1, \ldots, n$ and $a_{ij} \geq 0$ for $i \neq j$. Then, the following inequalities hold:

$$\begin{aligned} 1 \leq \|A\|_\infty < 2 \\ 1 \leq \|A^{-1}\|_\infty \leq \frac{1}{\alpha} \end{aligned}$$

where $\alpha = \min_{i=1,\ldots,n} \left(|a_{ii}| - \sum_{\substack{j=1\\j\neq i}}^{n} |a_{ij}|\right)$

Proof. The first inequalities are a simple consequence of the definition of A. Even though $a_{ij} \geq 0 \quad i,j = 1,\dots,n$, we will still use absolute values in order to identify general results more easily. For any index i, we have

$$1 \leq |a_{ii}| + \sum_{\substack{j=1\\j\neq i}}^{n} |a_{ij}| = \sum_{j=1}^{n} |a_{ij}| \leq \max_{i=1,2,\dots,n} \sum_{j=1}^{n} |a_{ij}| = \|A\|_\infty$$

Moreover, for any index i

$$\sum_{j=1}^{n} |a_{ij}| = |a_{ii}| + \sum_{\substack{j=1\\j\neq i}}^{n} |a_{ij}| < 2|a_{ii}| = 2$$

In particular, it holds for the maximum among all i. Thus $\|A\|_\infty < 2$. For the bounds on the inverse, let $B = A^{-1}$. Then, from $AB = I$, we deduce that

$$1 = |\sum_{k=1}^{n} b_{ik}a_{ki}| \leq \sum_{k=1}^{n} |b_{ik}||a_{ki}| \leq \sum_{k=1}^{n} |b_{ik}| \leq \max_{i=1,\dots,n} \sum_{k=1}^{n} |b_{ik}| = \|A^{-1}\|_\infty$$

Finally, the upper bound is a particular case of a more general result from Varga (1976). We will proceed in two steps. The proof arguments are taken from Nick Higham's website. Let $\mathbf{x} \neq \mathbf{0}$ and $\mathbf{y}$ be such that $\mathbf{y} = A\mathbf{x}$. Let i be such that $|x_i| = \max_{j=1,\dots,n} |x_j| = \|\mathbf{x}\|_\infty$. Then, from the ith equation, we obtain

$$a_{ii}x_i = y_i - \sum_{\substack{j=1\\j\neq i}}^{n} a_{ij}x_j$$

Taking the absolute value leads to

$$|a_{ii}||x_i| = |a_{ii}|\|\mathbf{x}\|_\infty \leq |y_i| + \sum_{\substack{j=1\\j\neq i}}^{n} |a_{ij}||x_j| \leq |y_i| + \|\mathbf{x}\|_\infty \sum_{\substack{j=1\\j\neq i}}^{n} |a_{ij}|$$

From which we deduce

$$\|\mathbf{x}\|_\infty \leq \frac{|y_i|}{|a_{ii}| - \sum_{\substack{j=1\\j\neq i}}^{n} |a_{ij}|} \leq \frac{\|\mathbf{y}\|_\infty}{\alpha}$$

where we defined $\alpha = \min_{i=1,\dots,n} \left(|a_{ii}| - \sum_{\substack{j=1\\j\neq i}}^{n} |a_{ij}|\right)$. Now, let $\mathbf{v}$ be the vector such that

$$\|A^{-1}\|_\infty = \frac{\|A^{-1}\mathbf{v}\|_\infty}{\|\mathbf{v}\|_\infty}$$

Defining $\mathbf{x}$ as $\mathbf{x} = A^{-1}\mathbf{v}$, we may apply the previous result from which we conclude

$$\|A^{-1}\|_\infty = \frac{\|\mathbf{x}\|_\infty}{\|\mathbf{v}\|_\infty} \leq \frac{1}{\alpha}$$

□

This result can be used to bound the condition number in the infinity norm as

$$\kappa_\infty(A) = \|A\|_\infty\|A^{-1}\|_\infty < \frac{2}{\alpha}$$

However, the resulting bound is quite pessimistic. It is interesting to note that the inverse $B = A^{-1}$ still enjoys some weaker form of diagonal dominance. The result is given in the next lemma and allows to better characterize the magnitude of the entries.

Lemma 1.2

Let $A \in \mathbb{R}^{n\times n}$ be a strictly diagonally dominant matrix by rows which additionally satisfies $a_{ii} = 1$ for $i = 1,\dots,n$ and $a_{ij} \geq 0$ for $i \neq j$. Let $B = A^{-1}$, then

$$|b_{ij}| < |b_{jj}| \quad i = 1,2,\dots,n \quad i \neq j$$
$$b_{ii} > \frac{1}{2} \quad i = 1,2,\dots,n$$

Proof. The first result holds for any strictly diagonally dominant matrix by rows whereas the second result is a consequence of the first one and requires the additional assumptions on A. Since $AB = BA = I$, we have

$$\sum_{k=1}^{n} a_{ik} b_{kj} = 0 \quad i \neq j$$

Now, let us denote $\beta_j = \max_{i=1,\ldots,n} |b_{ij}|$ for $j = 1, \ldots, n$. Then, by the triangle inequality, we obtain for $i \neq j$

$$|a_{ii}||b_{ij}| \leq \sum_{\substack{k=1 \\ k\neq i}}^{n} |a_{ik}||b_{kj}| \leq \beta_j \sum_{\substack{k=1 \\ k\neq i}}^{n} |a_{ik}| < \beta_j |a_{ii}|$$

Thus, $|b_{ij}| < \beta_j$ holds for all $i \neq j$. Hence $\beta_j = |b_{jj}|$. For the second result, we use the fact that

$$\sum_{k=1}^{n} a_{ik} b_{ki} = 1 \quad i = 1, 2, \ldots, n$$

Therefore, due to the properties of A, we deduce that

$$|1 - b_{ii}| = |1 - a_{ii} b_{ii}| \leq \sum_{\substack{k=1 \\ k\neq i}}^{n} |a_{ik}||b_{ki}| < |b_{ii}| \sum_{\substack{k=1 \\ k\neq i}}^{n} |a_{ik}| < |b_{ii}||a_{ii}| = |b_{ii}|$$

Thus, we have the interesting relation $|1 - b_{ii}| < |b_{ii}|$ which implies that $b_{ii} > \frac{1}{2}$. □

We are particularly interested in knowing something about $\boldsymbol{\gamma} = \Phi_{MM}^{-1} \mathbf{1}$. Note that if Φ_{MM}^{-1} were also diagonally dominant by rows, then we could conclude immediately that $\gamma_i > 0 \quad i = 1, \ldots, n$. However, as we have just seen, Φ_{MM}^{-1} only has a much weaker form of diagonal dominance. In fact, the final result still holds but one has to proceed differently to prove it.

Theorem 1.3

Let $\boldsymbol{\gamma}$ be the solution of $A\boldsymbol{\gamma} = \mathbf{1}$ where $A \in \mathbb{R}^{n\times n}$ is a strictly diagonally dominant matrix by rows which additionally satisfies $a_{ii} = 1$ for $i = 1, \ldots, n$ and $a_{ij} \geq 0$ for $i \neq j$ and $\mathbf{1}$ is the vector of all ones. Then,

$$0 < \gamma_i \leq 1 \quad i = 1, 2, \ldots, n$$

Proof. We begin by proving a few useful implications. Considering the ith equation of $A\boldsymbol{\gamma} = \mathbf{1}$, we have

$$\gamma_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij} \gamma_j = 1 \quad i = 1, \ldots, n \tag{1.1}$$

Let us investigate different cases:

- Assume $\gamma_i > 1$, then, from equation (1.1)

$$\underbrace{\gamma_i}_{>1} + \underbrace{\sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij} \gamma_j}_{<0} = 1$$

  Thus, there exists a least one index $k \neq i$ such that $\gamma_k < 0$ and $a_{ik} > 0$.

- Assume $\gamma_i < 0$, then we deduce that

$$\underbrace{\gamma_i}_{<0} + \underbrace{\sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij} \gamma_j}_{>1} = 1$$

Thus, there exists $\gamma_k > 1$ for some index $k \neq i$. Indeed, since $a_{ij} \geq 0$, even if all $\gamma_j = 1$ for $j \neq i$, we would have

$$\sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij}\gamma_j = \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} < 1$$

which does not satisfy the equation. Similarly, if all $\gamma_j < 1$ for $j \neq i$, we still have $\sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij}\gamma_j < 1$. Hence, there exists a least one index $k \neq i$ such that $\gamma_k > 1$.

- Assume $\gamma_i = 0$, then, we note that

$$\sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij}\gamma_j = 1$$

For the exact same reasons as previously, we deduce that there exists a least one index $k \neq i$ such that $\gamma_k > 1$

To summarize, we have the following implications:

- $\gamma_i > 1 \implies \exists \gamma_k < 0 \quad k \neq i$
- $\gamma_i < 0 \implies \exists \gamma_k > 1 \quad k \neq i$
- $\gamma_i = 0 \implies \exists \gamma_k > 1 \quad k \neq i$. And from the first point we deduce that $\exists \gamma_l < 0 \quad l \neq k$

These implications form a circle. In particular, combining the two last implications, we have shown that if there exists an index k such that $\gamma_k \leq 0$, then there exists an index $i \neq k$ such that $\gamma_i > 1$. We will prove that having a $\gamma_i > 1$ is in fact impossible. Therefore, $\gamma_i \in (0, 1]$ for $i = 1, \ldots, n$.

Let us denote $\gamma_i = \max_j \gamma_j$ and $\gamma_k = \min_j \gamma_j$. Hence, we assume the maximum is attained for the index i and the minimum for the index k. Assume $\gamma_i > 1$ such that it can be expressed as $\gamma_i = 1 + d$ with $d > 0$. Then, from equation (1.1)

$$-\sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij}\gamma_j = d$$

Since we have assumed $\gamma_i > 1$, there exists a $\gamma_j < 0$. Thus, $\gamma_k < 0$ since it is the minimum. Now, let us remark that

$$d = -\sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij}\gamma_j \leq -\gamma_k \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} < -\gamma_k$$

Thus, $\gamma_i < 1 - \gamma_k$. Now considering equation k of the linear system $A\boldsymbol{\gamma} = \mathbf{1}$

$$\gamma_k + \sum_{\substack{j=1 \\ j \neq k}}^{n} a_{kj}\gamma_j = 1$$

Thus,

$$1 - \gamma_k = \sum_{\substack{j=1 \\ j \neq k}}^{n} a_{kj}\gamma_j \leq \gamma_i \sum_{\substack{j=1 \\ j \neq k}}^{n} a_{kj} < \gamma_i$$

Thus, $\gamma_i > 1 - \gamma_k$ which is a contradiction. Hence, $\gamma_i > 1$ is impossible. Since assuming the existence of a $\gamma_k \leq 0$ leads to the existence of a $\gamma_i > 1$, having a $\gamma_k \leq 0$ is also impossible. Therefore, $\gamma_i \in (0, 1]$ for $i = 1, \ldots, n$. □

From these results we also deduce the following corollary.

Corollary 1.2

Let $\boldsymbol{\gamma}$ be the solution of $A\boldsymbol{\gamma} = \mathbf{1}$ where $A \in \mathbb{R}^{n\times n}$ is a strictly diagonally dominant matrix by rows which additionally satisfies $a_{ii} = 1$ for $i = 1, \dots, n$ and $a_{ij} \geq 0$ for $i \neq j$ and $\mathbf{1}$ is the vector of all ones. Then,

$$\gamma_i = 1 \iff a_{ij} = 0 \quad \forall j = 1, \dots, n \quad j \neq i$$

Proof. Let $\gamma_i = 1$, then, from the ith equation of the linear system

$$1 + \sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij}\gamma_j = 1$$

$$\sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij}\gamma_j = 0$$

However, $a_{ij} \geq 0$ and from the previous theorem $\gamma_j > 0 \quad j = 1, 2 \dots, n$. Hence, $a_{ij} = 0 \quad \forall j = 1, \dots, n \quad j \neq i$

Now, let $a_{ij} = 0 \quad \forall j = 1, \dots, n \quad j \neq i$, then, straightforwardly $\gamma_i = 1$. ☐

In fact, $\max_{i=1,\dots,n} |\gamma_i|$ can be characterized more precisely. The result is given in the next corollary.

Corollary 1.3

Let $\boldsymbol{\gamma}$ be the solution of $A\boldsymbol{\gamma} = \mathbf{1}$ where $A \in \mathbb{R}^{n\times n}$ is a strictly diagonally dominant matrix by rows which additionally satisfies $a_{ii} = 1$ for $i = 1, \dots, n$ and $a_{ij} \geq 0$ for $i \neq j$ and $\mathbf{1}$ is the vector of all ones. Then,

$$\frac{1}{2} < \|\boldsymbol{\gamma}\|_\infty \leq 1$$

Proof. There are different ways of proving these results. One of the easiest ways is to use sub-multiplicativity. From $A\boldsymbol{\gamma} = \mathbf{1}$, we obtain

$$1 = \|A\boldsymbol{\gamma}\|_\infty \leq \|A\|_\infty \|\boldsymbol{\gamma}\|_\infty$$

From which we deduce the lower bound thanks to Lemma 1.1 as

$$\|\boldsymbol{\gamma}\|_\infty \geq \frac{1}{\|A\|_\infty} > \frac{1}{2}$$

As for the upper bound, we already know from Theorem 1.3 that $\gamma_i \leq 1$ for $i = 1, \dots, n$. ☐

From all these results, we conclude that asking for Φ_{MM} to be strictly diagonally dominant by rows is enough. This single requirement solves both the invertibility and the rescaling issue. However, although $\gamma_i > 0$, it could be nevertheless arbitrarily close to zero. This can be easily verified with a simple example. Let us consider the linear system

$$\begin{pmatrix} 1 & 1-\epsilon \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

with $0 < \epsilon \leq 1$. The unique solution is

$$\begin{pmatrix} \gamma_1 \\ \gamma_2 \end{pmatrix} = \begin{pmatrix} \epsilon \\ 1 \end{pmatrix}$$

which is arbitrarily close to zero by taking the limit $\epsilon \to 0$.

In practice, such cases are unlikely to occur. One may come to the misleading conclusion that they are encountered for very weakly diagonally dominant matrices (matrices for which $\alpha = \min_{i=1,\dots,n} \left(|a_{ii}| - \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}|\right)$ is small). We found out experimentally this was incorrect. The phenomenon seems related to the dispersion of $\alpha_i = |a_{ii}| -$

$\sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}|$ across the rows $i = 1, \ldots, n$. In particular, cases for which α_i is roughly constant behave well. We conjecture that

$$\gamma_i \to 1 \text{ for } \alpha_i \to 1$$
$$\gamma_i \to \frac{1}{2} \text{ for } \alpha_i \to 0$$

These are easily understood. When $\alpha_i \to 1$ for $i = 1, \ldots, n$, the matrix A is very strongly diagonally dominant and close to the identity matrix. Thus, the solution to the linear system $A\boldsymbol{\gamma} = \mathbf{1}$ should not be too different from the vector of all ones. When $\alpha_i \to 0$ for $i = 1, \ldots, n$, A is close to loosing its diagonal dominance. In particular, the row-wise sum of the entries of A is close to 2 for all rows. Thus, all the components of $\boldsymbol{\gamma}$ must roughly be $\frac{1}{2}$.

Let us summarize the main results of this section. Provided the following conditions are satisfied:

1. Φ_{NM} does not contain a row of zeros,
2. Φ_{MM} is strictly diagonally dominant by rows,
3. $(\Phi_{MM})_{ii} = 1$ for $i = 1, \ldots, M$ and $(\Phi_{MM})_{ij} \geq 0$ for $i \neq j$,

then the matrix Φ_{MM} is invertible and $\mathbf{g}_\zeta = \Phi_{NM}\Phi_{MM}^{-1}\mathbf{1}_\xi > 0$ component-wise. In particular, the third condition is satisfied by Wendland C^2 basis functions. The next section discusses the first two conditions.

1.4 Algorithmic aspects

In the first part of this section, we will discuss how to ensure the matrices Φ_{MM} constructed are strictly diagonally dominant by rows. We will consider only the compactly supported Wendland C^2 radial basis functions. In Deparis et al. (2014), different choices of RBF are considered. The rescaled localized Wendland C^2 radial basis functions were the most promising because of their good interpolation accuracy while being only locally supported. Moreover, numerical experiments revealed the condition number grew very slowly with the size of the interpolation problem.

We recall that the Wendland C^2 radial basis functions are defined as

$$\phi(\|\mathbf{x}\|, r) = \Big(1 - \frac{\|\mathbf{x}\|}{r}\Big)^4 \Big(1 + 4\frac{\|\mathbf{x}\|}{r}\Big)$$

The definition suggests $\phi(\|\mathbf{x}\|, r)$ decays extremely fast as $\|\mathbf{x}\|$ grows. This observation is well illustrated in Figure 1.1 where a typical Wendland C^2 basis function for a problem in 2D is plotted.

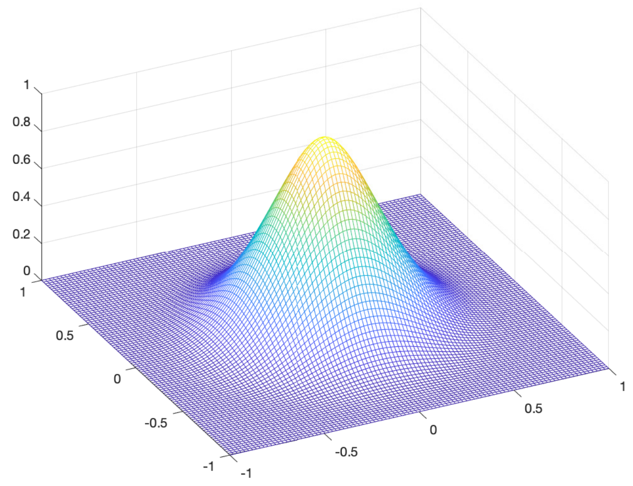

Figure 1.1: A typical Wendland C^2 basis function

We must now investigate how to enforce the two following requirements:

1. Φ_{NM} does not contain a row of zeros.
2. Φ_{MM} is strictly diagonally dominant by rows.

In fact, for computational reasons, we will enforce a stronger condition on Φ_{NM}: any of its nonzero entries must be safely away from zero. This requirement is set to avoid dividing by a very small number (potentially even smaller than machine precision). However, this condition alone is not enough: as we have seen, the entries of $\boldsymbol{\gamma}$ could be arbitrarily close to zero. Thus, we could potentially encounter the situation where the only nonzero in a row of Φ_{NM} multiplies an entry of $\boldsymbol{\gamma}$ which is ϵ away from zero. Fortunately, this situation is unlikely in practice.

We will enforce the two conditions:

$$\|\boldsymbol{\xi}_i - \boldsymbol{\xi}_j\|_2 \geq \gamma r_j \quad i \neq j \tag{1.2}$$

$$\exists j \colon \|\boldsymbol{\zeta}_i - \boldsymbol{\xi}_j\|_2 \leq \Gamma r_j \quad i = 1, \ldots, N \tag{1.3}$$

where $\gamma, \Gamma \in (0,1)$ are scalar parameters satisfying $\gamma < \Gamma$. Condition (1.2) prevents quick loss of diagonal dominance of Φ_{MM}. Indeed, if two distinct points $\boldsymbol{\xi}_i$ and $\boldsymbol{\xi}_j$ are too close to each other (relative to the radius chosen), then $\phi(\|\boldsymbol{\xi}_i - \boldsymbol{\xi}_j\|, r_j) \approx \phi(0, r_j) = 1$ and diagonal dominance will be quickly lost. Condition (1.2) alone is critical and prevents us from choosing a too large radius. On the other hand, condition (1.3) avoids having a single nonzero entry in a row of Φ_{NM} which is too close to zero. Condition (1.3) is equivalent to stating that any point $\boldsymbol{\zeta}_i$ is well within the support of at least one interpolation point $\boldsymbol{\xi}_j$. In other words, it is safely away from the boundary of the support. If this condition is not met, point $\boldsymbol{\zeta}_i$ will be considered isolated and removed from the set of evaluation points. A feasible situation is illustrated in Figure 1.2.

From the figure, the number of nonzeros in the jth column of Φ_{MM} and Φ_{NM} can be deduced. These are the number of points $\boldsymbol{\xi}_i$ for $i = 1, \ldots, M$ and $\boldsymbol{\zeta}_i$ for $i = 1, \ldots, N$ respectively in the support of the $\boldsymbol{\xi}_j$. Both are equal to 3 in Figure 1.2. However, the number of nonzeros in each row of Φ_{MM} and Φ_{NM} can only be deduced once all radiuses have been computed. More specifically, the number of nonzeros in the ith row of Φ_{MM} is the number of supports to which point $\boldsymbol{\xi}_i$ belongs. Similarly, the number of nonzeros in the ith row of Φ_{NM} is the number of supports to which point $\boldsymbol{\zeta}_i$ belongs. If the point $\boldsymbol{\zeta}_i$ does not belong to any support, then the ith row of Φ_{NM} will contain only zeros and the rescaling fails. Such a point will also be considered isolated and removed from the set of evaluation points.

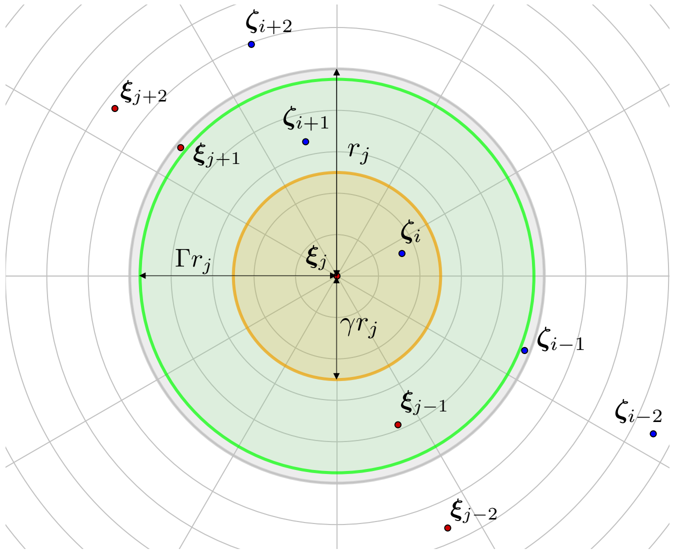

Figure 1.2: A feasible radius r_j satisfying condition (1.2). Point $\boldsymbol{\zeta}_{i-1}$ belongs indeed to the support of $\boldsymbol{\xi}_j$ but $\|\boldsymbol{\zeta}_{i-1} - \boldsymbol{\xi}_j\|_2 > \Gamma r_j$. If $\nexists k \colon \|\boldsymbol{\zeta}_{i-1} - \boldsymbol{\xi}_k\|_2 \leq \Gamma r_k$, then the point will be considered isolated

Thanks to condition (1.2), since ϕ is a decreasing function of $\|\mathbf{x}\|$, we deduce that

$$\phi(\|\boldsymbol{\xi}_i - \boldsymbol{\xi}_j\|, r_j) \leq (1-\gamma)^4(1+4\gamma) \quad i \neq j$$

Let us assume point $\boldsymbol{\xi}_i$ belongs to the support of n different radial basis functions (excluding itself). Then, there are n nonzero off-diagonal elements in the ith row of Φ_{MM}. Consequently,

$$\sum_{\substack{j=1 \\ j\neq i}}^{M} \phi(\|\boldsymbol{\xi}_i - \boldsymbol{\xi}_j\|, r_j) \leq n(1-\gamma)^4(1+4\gamma)$$

In order to enforce strict diagonal dominance by rows, we set the condition

$$n(1-\gamma)^4(1+4\gamma) < \phi(0, r_i) = 1$$

Hence,

$$n < \frac{1}{(1-\gamma)^4(1+4\gamma)}$$

Thus, the interpolation point $\boldsymbol{\xi}_i$ must not belong to too many different supports. Unfortunately, this condition can only be verified once all radiuses have been computed. If the condition is not satisfied, the value of γ is slightly increased and the radiuses are recomputed. The process is repeated until the condition is finally satisfied.

In order to choose a suitable value for γ, we can think of how many nonzeros we should allow in each row of Φ_{MM}. The function $f(\gamma) = \frac{1}{(1-\gamma)^4(1+4\gamma)}$ is plotted in Figure 1.3.

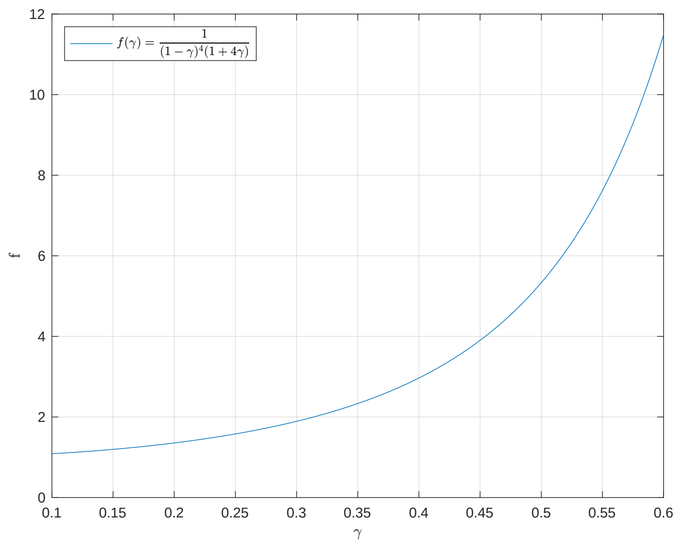

Figure 1.3: Upper bound on n

For feasibility purposes, γ must not be too small. In fact $n \geq 2$ is a minimum requirement for 2D problems. The reason will become clear with the application we are considering. On the other hand, choosing γ too large will much increase the orange area in Figure 1.2 and might eventually also lead to an infeasible problem. In all our experiments, we chose $\gamma = 0.5$ and never encountered any problem. This implies $n \leq 5$ and Φ_{MM} will never contain more than 5 off-diagonal nonzeros per row. Interestingly, this requirement also ensures great sparsity of Φ_{MM}.

Choosing Γ is less critical. Figure 1.4 plots the function $f(\Gamma) = (1-\Gamma)^4(1+4\Gamma)$. It is the one dimensional profile of a Wendland C^2 radial basis function.

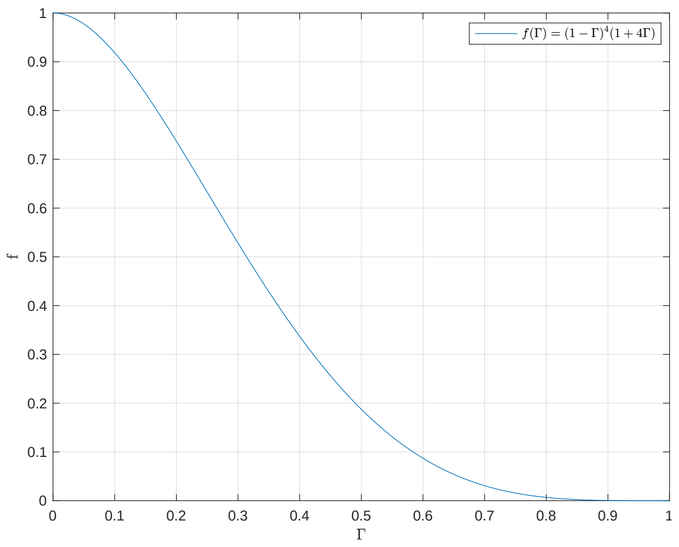

Figure 1.4: Choosing Γ

In this case, any value of Γ for which $f(\Gamma)$ is safely away from machine precision is suitable. In our experiments we chose $\Gamma = 0.95$. Consequently, if $\boldsymbol{\zeta}_i$ is not an isolated node, then there exists an index j such that $\phi(\|\boldsymbol{\zeta}_i - \boldsymbol{\xi}_j\|, r_j) \geq 3 \times 10^{-5}$ thanks to inequality (1.3).

The discussion of the last section shows that the properties which are enforced on the matrices Φ_{NM} and Φ_{MM} are controlled by the radiuses of the radial basis functions only. Thus, the matrices Φ_{NM} and Φ_{MM} which are subsequently assembled satisfy the properties by construction. As we will see in the chapters to come, this requirement leads to several other benefits on the computational side and is partly behind the success in the design of efficient preconditioners. We believe there are yet other advantages but their connection with strict diagonal dominance could not be precisely established.

Chapter 2

Modeling of contact problems

The mathematical description of contact problems is very difficult. Entire books have been dedicated to it. For an in-depth discussion of the functional analysis aspects, we refer to the mathematical literature such as Kikuchi (1988) and Sofonea (2012). For a discussion related to computational and implementation aspects, we refer to Wriggers (2006). The description we provide in this chapter is not complete nor rigorous. Instead, we wish to illustrate the application of the INTERNODES method for the numerical solution of contact problems. The three most popular solution methods in computational contact mechanics are the penalty method, Lagrange multiplier method and augmented Lagrangian method. The method presented in this chapter is a Lagrange multiplier method.

In this chapter, we will first consider the continuous description of a contact problem between two deformable elastic bodies. We will assume a linear elastic constitutive model in quasi-static. The first three sections of this chapter are a formalization of the work of Günther-Hanssen. Section 2.1 begins by stating the strong form of the problem. The weak form is then established in Section 2.2. The finite element discretization is covered in Section 2.3 and is combined with the INTERNODES method. The discretization leads to an algebraic system of equations and the coefficient matrix will be referred to as the INTERNODES matrix. The nonlinear nature of the contact constraints requires in fact solving a sequence of such linear systems. Section 2.4 therefore presents the full contact algorithm and discusses some implementation aspects.

2.1 Strong form

The strong (or differential) form of the problem stems from the balance of momentum and Newton's laws of motion. We must define the mathematical properties of the quantities involved. Let Ω^k, $k = 1, 2$, be an open connected domain of $\mathbb{R}^d$, $d = 2, 3$, with Lipschitz boundary such that $\Omega^1 \cap \Omega^2 = \emptyset$. Let Γ_D^k, Γ_N^k and Γ_C^k form a partition of the boundary $\partial\Omega^k$. The contact interface is common to both bodies. Thus, $\Gamma_C^1 = \Gamma_C^2 = \Gamma_C$. We write $\partial\Omega^k = \overline{\Gamma_D^k} \cup \overline{\Gamma_N^k} \cup \overline{\Gamma_C^k}$ and $\Gamma_D^k \cap \Gamma_N^k = \emptyset$, $\Gamma_D^k \cap \Gamma_C^k = \emptyset$ and $\Gamma_N^k \cap \Gamma_C^k = \emptyset$. An illustration is provided in Figure 2.1. In this chapter, we will make all the necessary assumptions on the regularity of the data for the problem to be well-posed.

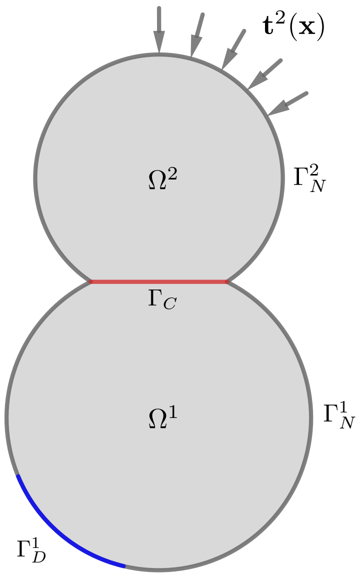

Figure 2.1: Contact problem between two deformable bodies

Our goal is to approximate the displacement field $\mathbf{u}^k$ of each body $k = 1, 2$. We will formulate the equilibrium equations for a given body k. For readability purposes we drop the superscript k from the notations for the time being. The strong form of the problem reads:

Find $\mathbf{u}\colon \Omega \to \mathbb{R}^d$ such that

$$\begin{cases} -\operatorname{div}(\sigma(\mathbf{u})) = \mathbf{f}(\mathbf{x}) & \text{in } \Omega & (2.1)\\ \mathbf{u}(\mathbf{x}) = \mathbf{g}(\mathbf{x}) & \text{on } \Gamma_D & (2.2)\\ \sigma(\mathbf{u})\mathbf{n} = \mathbf{t}(\mathbf{x}) & \text{on } \Gamma_N & (2.3)\\ \sigma(\mathbf{u})\mathbf{n} = \boldsymbol{\lambda}(\mathbf{x}) & \text{on } \Gamma_C & (2.4)\\ \boldsymbol{\lambda}(\mathbf{x}) \cdot \mathbf{n} \le 0 & \text{on } \Gamma_C & (2.5) \end{cases}$$

Equation (2.1) is the differential equation to be solved. $\sigma\colon \mathbb{R}^d \to \mathbb{R}^{d\times d}$ is the Cauchy stress tensor. $\mathbf{f} \in [L^2(\Omega)]^d$ is the body force acting on the solid (for example the gravity).

Equation (2.2) prescribes Dirichlet boundary conditions. $\mathbf{g}$ is a prescribed displacement field on Γ_D. Equation (2.3) prescribes Neumann boundary conditions. $\mathbf{t} \in [L^2(\Gamma_N)]^d$ are surface tractions prescribed on Γ_N and $\mathbf{n}$ is the unit outward normal vector. Equation (2.4) results form the interaction of the two bodies. $\boldsymbol{\lambda}$ is the stress vector at the contact interface and is unknown. Inequality (2.5) is one of the Hertz-Signorini-Moreau conditions which are equivalent to the KKT conditions in optimization. Physically speaking, the stress state must be in compression all along the contact interface. Compression is negative according to the sign convention used in structural mechanics.

These sets of equations must be complemented with another of the Hertz-Signorini-Moreau conditions. It takes the form of a compatibility condition of the displacement fields at the interface. Indeed, in the deformed configuration, the current position on the contact interface must be identical. Thus,

$$\mathbf{u}^1 + \mathbf{r}^1 = \mathbf{u}^2 + \mathbf{r}^2 \text{ on } \Gamma_C \tag{2.6}$$

where $\mathbf{r}^k$ is the position vector in the initial configuration. Equality must be understood in a weak sense. This equation ensures the coupling between the two bodies. Furthermore, $\boldsymbol{\lambda}^1$ and $\boldsymbol{\lambda}^2$ are related through Newton's third law by $\boldsymbol{\lambda}^2 = -\boldsymbol{\lambda}^1$.

We now introduce our constitutive model to relate stresses to strains. In linear elasticity, the stress tensor σ is a linear function of the infinitesimal strain tensor ϵ. This infinitesimal strain tensor is the restriction to first order terms of the more general Green-Lagrange strain tensor used in the theory of elasticity. The constitutive model we will be using may be written as

$$\sigma(\mathbf{u}) = 2\mu\epsilon(\mathbf{u}) + \lambda\,\mathrm{trace}(\epsilon(\mathbf{u}))I \tag{2.7}$$

where $\mu, \lambda \geq 0$ are known as the Lamé constants which depend on the material and I is the identity tensor. In fact, equation (2.7) is a simplification for isotropic materials of the more general Hooke's law. We will restrict the discussion in this chapter to materials which follow this law. All tensor fields appearing in equation (2.7) are second order symmetric tensors which may be identified with matrix fields. The infinitesimal strain tensor is the symmetric part of the gradient of the displacement field

$$\epsilon(\mathbf{u}) = \frac{1}{2}(\nabla\mathbf{u} + \nabla\mathbf{u}^T)$$

Due to the restriction to first order terms, the linear elastic model is well-suited when elastic bodies undergo small (ideally infinitesimal) displacements.

2.2 Weak form

Let us multiply equation (2.1) with a test function $\mathbf{v}$ and integrate over the domain Ω. We obtain for the left-hand side

$$-\int_\Omega \mathrm{div}(\sigma(\mathbf{u})) \cdot \mathbf{v}\ \mathrm{d}\Omega$$

where we omit the dependence on the variable $\mathbf{x}$ for readability. We integrate by parts this term using Green's identities and obtain

$$\begin{aligned}-\int_\Omega \mathrm{div}(\sigma(\mathbf{u})) \cdot \mathbf{v}\ \mathrm{d}\Omega &= \int_\Omega \langle\sigma(\mathbf{u}), \nabla\mathbf{v}\rangle\ \mathrm{d}\Omega - \int_{\partial\Omega} \sigma(\mathbf{u})\mathbf{n}\cdot\mathbf{v}\ \mathrm{d}S \\ &= \int_\Omega \langle\sigma(\mathbf{u}), \nabla\mathbf{v}\rangle\ \mathrm{d}\Omega - \int_{\Gamma_D} \sigma(\mathbf{u})\mathbf{n}\cdot\mathbf{v}\ \mathrm{d}S - \int_{\Gamma_N} \sigma(\mathbf{u})\mathbf{n}\cdot\mathbf{v}\ \mathrm{d}S - \int_{\Gamma_C} \sigma(\mathbf{u})\mathbf{n}\cdot\mathbf{v}\ \mathrm{d}S\end{aligned}$$

The surface tractions are unknown on the boundary Γ_D. Hence, let us choose $\mathbf{v} \in V_0$ where $V_0 = \{\mathbf{v} \in [H^1(\Omega)]^d\colon\ \mathbf{v}|_{\Gamma_D} = \mathbf{0}\}$ where the evaluation on the boundary must be understood in the sense of traces of $[H^1(\Omega)]^d$ functions. Moreover, we have that $\langle\sigma(\mathbf{u}), \nabla\mathbf{v}\rangle = \langle\sigma(\mathbf{u}), \epsilon(\mathbf{v})\rangle$ because the trace inner product of a symmetric and skew-symmetric matrix is zero. Thus, only the symmetric part of $\nabla\mathbf{u}$, that is to say $\epsilon(\mathbf{u})$, remains. Now, after plugging in our constitutive model, using the linearity of the inner product and noting that $\mathrm{trace}(\epsilon(\mathbf{u})) = \mathrm{div}(\mathbf{u})$ and the fact that $\sigma(\mathbf{u})\mathbf{n}$ is known on Γ_N, we obtain

$$-\int_\Omega \mathrm{div}(\sigma(\mathbf{u})) \cdot \mathbf{v}\ \mathrm{d}\Omega = \int_\Omega 2\mu\langle\epsilon(\mathbf{u}), \epsilon(\mathbf{v})\rangle + \lambda\,\mathrm{div}(\mathbf{u})\,\mathrm{div}(\mathbf{v})\ \mathrm{d}\Omega - \int_{\Gamma_N} \mathbf{t}\cdot\mathbf{v}\ \mathrm{d}S - \int_{\Gamma_C} \boldsymbol{\lambda}\cdot\mathbf{v}\ \mathrm{d}S$$

Let us furthermore define the space $V_{\Gamma_D} = \{\mathbf{v} \in [H^1(\Omega)]^d\colon\ \mathbf{v}|_{\Gamma_D} = \mathbf{g}\}$ and $\Lambda = [L^2(\Gamma_C)]^d$.
The weak form now reads:

Find $(\mathbf{u}, \boldsymbol{\lambda}) \in V_{\Gamma_D} \times \Lambda$ such that

$$\int_\Omega 2\mu\langle\epsilon(\mathbf{u}), \epsilon(\mathbf{v})\rangle + \lambda\,\mathrm{div}(\mathbf{u})\,\mathrm{div}(\mathbf{v})\ \mathrm{d}\Omega - \int_{\Gamma_C} \boldsymbol{\lambda}\cdot\mathbf{v}\ \mathrm{d}S = \int_\Omega \mathbf{f}\cdot\mathbf{v}\ \mathrm{d}\Omega + \int_{\Gamma_N} \mathbf{t}\cdot\mathbf{v}\ \mathrm{d}S \quad \forall\mathbf{v} \in V_0 \tag{2.8}$$

Let us define the bilinear forms

$$a(\mathbf{u}, \mathbf{v}) = \int_\Omega 2\mu\langle\epsilon(\mathbf{u}), \epsilon(\mathbf{v})\rangle + \lambda\,\mathrm{div}(\mathbf{u})\,\mathrm{div}(\mathbf{v})\ \mathrm{d}\Omega$$

$$b(\boldsymbol{\lambda}, \mathbf{v}) = -\int_{\Gamma_C} \boldsymbol{\lambda}\cdot\mathbf{v}\ \mathrm{d}S$$

and the linear functional

$$F(\mathbf{v}) = \int_\Omega \mathbf{f}\cdot\mathbf{v}\ \mathrm{d}\Omega + \int_{\Gamma_N} \mathbf{t}\cdot\mathbf{v}\ \mathrm{d}S$$

The problem may be more compactly written as:

Find $(\mathbf{u}, \boldsymbol{\lambda}) \in V_{\Gamma_D} \times \Lambda$ such that

$$a(\mathbf{u}, \mathbf{v}) + b(\boldsymbol{\lambda}, \mathbf{v}) = F(\mathbf{v}) \quad \forall \mathbf{v} \in V_0$$

Now, for the compatibility conditions in equation (2.6), taking the inner product with a test function $\mathbf{w} \in \Lambda$ and integrating over the contact interface leads to

$$\int_{\Gamma_C} \mathbf{u}^1 \cdot \mathbf{w} \, \mathrm{d}S - \int_{\Gamma_C} \mathbf{u}^2 \cdot \mathbf{w} \, \mathrm{d}S = \int_{\Gamma_C} \mathbf{r}^2 \cdot \mathbf{w} \, \mathrm{d}S - \int_{\Gamma_C} \mathbf{r}^1 \cdot \mathbf{w} \, \mathrm{d}S$$

We define the linear functional

$$\tilde{b}(\mathbf{u}, \mathbf{w}) = \int_{\Gamma_C} \mathbf{u} \cdot \mathbf{w} \, \mathrm{d}S$$

and the equation may be compactly written as

$$\tilde{b}(\mathbf{u}^1 - \mathbf{u}^2, \mathbf{w}) = \tilde{b}(\mathbf{r}^2 - \mathbf{r}^1, \mathbf{w})$$

All the equations can now be written in a single system as

$$\begin{aligned} a_1(\mathbf{u}^1, \mathbf{v}^1) + b_1(\boldsymbol{\lambda}^1, \mathbf{v}^1) &= F_1(\mathbf{v}^1) \quad \forall \mathbf{v}^1 \in V_0^1 \\ a_2(\mathbf{u}^2, \mathbf{v}^2) + b_2(\boldsymbol{\lambda}^2, \mathbf{v}^2) &= F_2(\mathbf{v}^2) \quad \forall \mathbf{v}^2 \in V_0^2 \\ \tilde{b}(\mathbf{u}^1 - \mathbf{u}^2, \mathbf{w}) &= \tilde{b}(\mathbf{r}^2 - \mathbf{r}^1, \mathbf{w}) \quad \forall \mathbf{w} \in \Lambda \end{aligned}$$

To account for the Dirichlet boundary conditions, we write $\mathbf{u}^k = \mathbf{u}_0^k + \mathbf{s}^k$ where $\mathbf{u}_0^k \in V_0^k$ and $\mathbf{s}^k|_{\Gamma_D^k} = \mathbf{g}^k$ in the sense of the trace. The last equation does not involve any Dirichlet boundary conditions. Thus, we may directly replace $\mathbf{u}^k$ with $\mathbf{u}_0^k$. The system can now be written as

$$\begin{aligned} a_1(\mathbf{u}_0^1, \mathbf{v}^1) + b_1(\boldsymbol{\lambda}^1, \mathbf{v}^1) &= F_1(\mathbf{v}^1) - a_1(\mathbf{s}^1, \mathbf{v}^1) \quad \forall \mathbf{v}^1 \in V_0^1 \\ a_2(\mathbf{u}_0^2, \mathbf{v}^2) + b_2(\boldsymbol{\lambda}^2, \mathbf{v}^2) &= F_2(\mathbf{v}^2) - a_2(\mathbf{s}^2, \mathbf{v}^2) \quad \forall \mathbf{v}^2 \in V_0^2 \\ \tilde{b}(\mathbf{u}_0^1 - \mathbf{u}_0^2, \mathbf{w}) &= \tilde{b}(\mathbf{r}^2 - \mathbf{r}^1, \mathbf{w}) \quad \forall \mathbf{w} \in \Lambda \end{aligned}$$

These equations form a saddle point type system which is typical from constrained optimization problems. The system is so named because the solution is a saddle point of a Lagrangian function. These equations are the first order conditions for a stationary point of the Lagrangian function. We will now derive their discrete counterpart obtained through the finite element method.

2.3 Finite element approximation

We will approximate the weak form using the Galerkin method and more precisely the finite element method. For this purpose we introduce a triangulation τ_h of the domain $\Omega^1 \cup \Omega^2$ where $\tau_h = \bigcup_{i=1}^N K_i$ with K_i being the finite elements and N the number of elements. The triangulation is typically nonconforming in the vicinity of the discrete contact interface. The nonconformity is essentially geometric. There exists small gaps or overlaps between the finite elements of both bodies. However, the triangulation is conforming within each body.

Let $\mathcal{D}$ be the set of degrees of freedom. They will be associated to displacements. Thus, $\mathcal{D} = \{1, 2, \cdots, dn\}$ where n is the number of nodes. For the treatment of boundary conditions, the set of degrees of freedom is partitioned into a set $\mathcal{D}_D$, and a set $\mathcal{D}_F$. The set $\mathcal{D}_D$ collects all the degrees of freedom associated to Dirichlet boundary conditions and the set $\mathcal{D}_F$ those associated to unknown displacements. All these degrees of freedom are split between the two bodies.

$$\begin{aligned} \mathcal{D}_D &= \mathcal{D}_D^1 \cup \mathcal{D}_D^2 \\ \mathcal{D}_F &= \mathcal{D}_F^1 \cup \mathcal{D}_F^2 \end{aligned}$$

For each body, it will be useful to distinguish interface degrees of freedom from the others. Thus, we further partition $\mathcal{D}_F^k$ into the disjoint subsets $\mathcal{D}_\Omega^k$ and $\mathcal{D}_\Gamma^k$ such that

$$\mathcal{D}_F^k = \mathcal{D}_\Omega^k \cup \mathcal{D}_\Gamma^k \quad k = 1, 2$$

We are seeking an approximation $\mathbf{u}_{0,h}^k$ of $\mathbf{u}_0^k$ with $\mathbf{u}_{0,h}^k \in V_{0,h}^k \subset V_0^k$. Similarly, we look for an approximation $\boldsymbol{\lambda}_h^k \in \Lambda_h^k \subset \Lambda^k$. Note that in the discrete problem, the interface spaces Λ_h^1 and Λ_h^2 must be distinguished. Moreover, we must choose in which of these two spaces the test function $\mathbf{w}_h$ belongs. This choice amounts to deciding which body should be the master. For the discrete problem, we introduce some finite element spaces. Let X_h^r be the space of piecewise polynomials of degree r in each of the d components with $d = 2, 3$ such that

$$X_h^r = \{\mathbf{v}_h \in [C^0(\overline{\Omega})]^d \colon \mathbf{v}_h|_K \in [\mathbb{P}_r]^d \quad K \in \tau_h\}$$

Its subsets are

$$V_{h,0}^k = \{\mathbf{v}_h \in X_h^r \colon v_{h,j}(\mathbf{x_i}) = 0 \; \forall (i,j) \colon d(i-1) + j \in \mathcal{D}_D^k\}$$
$$V_{h,\Gamma_D}^k = \{\mathbf{v}_h \in X_h^r \colon v_{h,j}(\mathbf{x_i}) = g_j^k(\mathbf{x_i}) \; \forall (i,j) \colon d(i-1) + j \in \mathcal{D}_D^k\}$$

Let $\{\boldsymbol{\phi}_i\}_{i=1}^{dn}$ be the Lagrange basis of X_h^r such that

$$\boldsymbol{\phi}_{d(i-1)+1}(\mathbf{x}) = \begin{pmatrix} \phi_i(\mathbf{x}) \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad , \ldots , \quad \boldsymbol{\phi}_{di}(\mathbf{x}) = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ \phi_i(\mathbf{x}) \end{pmatrix}$$

In other words, the same nodal basis functions are used for the finite element approximation of the different components of the displacement field. Similarly, the functions $\{\boldsymbol{\phi}_j\}_{j \in \mathcal{D}_\Gamma^k}$ form a basis for Λ_h^k.

We must now introduce some intergrid transfer operators. This is one of the distinguishing features of the INTERNODES method. We define

$$\Pi_{12} \colon \Lambda_h^2 \to \Lambda_h^1 \quad \Pi_{21} \colon \Lambda_h^1 \to \Lambda_h^2$$

Due to the geometric nonconformities encountered in our problem, they will be based on radial basis function interpolation. The finite element approximation reads:

Find $(\mathbf{u}_{0,h}^k, \boldsymbol{\lambda}_h^k) \in V_{h,0}^k \times \Lambda_h^k$ for $k = 1, 2$ such that

$$a_1(\mathbf{u}_{0,h}^1, \mathbf{v}_h^1) + b_1(\boldsymbol{\lambda}_h^1, \mathbf{v}_h^1) = F_1(\mathbf{v}_h^1) - a_1(\mathbf{s}_h^1, \mathbf{v}_h^1) \quad \forall \mathbf{v}_h^1 \in V_{0,h}^1$$
$$a_2(\mathbf{u}_{0,h}^2, \mathbf{v}_h^2) + b_2(\boldsymbol{\lambda}_h^2, \mathbf{v}_h^2) = F_2(\mathbf{v}_h^2) - a_2(\mathbf{s}_h^2, \mathbf{v}_h^2) \quad \forall \mathbf{v}_h^2 \in V_{0,h}^2$$
$$\tilde{b}(\mathbf{u}_{0,h}^1 - \mathbf{u}_{0,h}^2, \mathbf{w}_h) = \tilde{b}(\mathbf{r}_h^2 - \mathbf{r}_h^1, \mathbf{w}_h) \quad \forall \mathbf{w}_h \in \Lambda_h^1$$

Since any $\mathbf{v}_h^k \quad k = 1, 2$ and $\mathbf{w}_h$ can be expressed as a linear combination of the basis functions, it is enough to enforce equality for all basis functions of the appropriate spaces. Moreover, $\mathbf{u}_{0,h}^k$ and $\boldsymbol{\lambda}_h^k$ can be expressed in their Lagrange basis as

$$\mathbf{u}_{0,h}^k(\mathbf{x}) = \sum_{j \in \mathcal{D}_F^k} u_j^k \boldsymbol{\phi}_j(\mathbf{x})$$
$$\boldsymbol{\lambda}_h^k(\mathbf{x}) = \sum_{j \in \mathcal{D}_\Gamma^k} \lambda_j^k \boldsymbol{\phi}_j(\mathbf{x})$$

Then, we obtain

$$\sum_{j \in \mathcal{D}_F^k} u_j^k a_k(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) + \sum_{j \in \mathcal{D}_\Gamma^k} \lambda_j^k b_k(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) = F_k(\boldsymbol{\phi}_i(\mathbf{x})) - \sum_{j \in \mathcal{D}_D^k} u_j^k a_k(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) \quad \forall i \in \mathcal{D}_F^k$$

However, as we have already noted, the coefficients λ_j^k for $k = 1, 2$ are related. In the discrete setting, this relation may be conveniently expressed using the intergrid transfer operators. Expressing for instance $\boldsymbol{\lambda}_h^1$ in the basis of Λ_h^2, we can write

$$\begin{aligned} \boldsymbol{\lambda}_h^2 = -\Pi_{21}(\boldsymbol{\lambda}_h^1) &= -\sum_{l \in \mathcal{D}_\Gamma^2} \boldsymbol{\lambda}_h^1(\mathbf{x}_l) \boldsymbol{\phi}_l(\mathbf{x}) \\ &= -\sum_{l \in \mathcal{D}_\Gamma^2} \sum_{j \in \mathcal{D}_\Gamma^1} \lambda_j^1 \boldsymbol{\phi}_j(\mathbf{x}_l) \boldsymbol{\phi}_l(\mathbf{x}) \end{aligned}$$

Similarly, due to the compatibility condition in the deformed configuration

$$\tilde{b}(\mathbf{u}^1_{0,h} + \mathbf{r}^1_h, \mathbf{w}_h) = \tilde{b}(\Pi_{12}(\mathbf{u}^2_{0,h} + \mathbf{r}^2_h), \mathbf{w}_h) \quad \forall \mathbf{w}_h \in \Lambda^1_h$$

Collecting all the equations, we obtain

$$\begin{aligned}
&\sum_{j\in\mathcal{D}^1_F} u^1_j a_1(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) &&+ \sum_{j\in\mathcal{D}^1_\Gamma} \lambda^1_j b_1(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) &&= F_1(\boldsymbol{\phi}_i(\mathbf{x})) - \sum_{j\in\mathcal{D}^1_D} u^1_j a_1(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) \quad \forall i \in \mathcal{D}^1_F \\
&\sum_{j\in\mathcal{D}^2_F} u^2_j a_2(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) &&- \sum_{l\in\mathcal{D}^2_\Gamma}\sum_{j\in\mathcal{D}^1_\Gamma} \lambda^1_j \boldsymbol{\phi}_j(\mathbf{x}_l) b_2(\boldsymbol{\phi}_l(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) &&= F_2(\boldsymbol{\phi}_i(\mathbf{x})) - \sum_{j\in\mathcal{D}^2_D} u^2_j a_2(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) \quad \forall i \in \mathcal{D}^2_F \\
&\sum_{j\in\mathcal{D}^1_\Gamma} u^1_j \tilde{b}(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) &&- \sum_{l\in\mathcal{D}^1_\Gamma}\sum_{j\in\mathcal{D}^2_\Gamma} u^2_j \boldsymbol{\phi}_j(\mathbf{x}_l) \tilde{b}(\boldsymbol{\phi}_l(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) &&= \sum_{l\in\mathcal{D}^1_\Gamma}\sum_{j\in\mathcal{D}^2_\Gamma} r^2_j \boldsymbol{\phi}_j(\mathbf{x}_l) \tilde{b}(\boldsymbol{\phi}_l(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) \\
& && &&- \sum_{j\in\mathcal{D}^1_\Gamma} r^1_j \tilde{b}(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) \quad \forall i \in \mathcal{D}^1_\Gamma
\end{aligned}$$

Defining the matrices

$$\begin{aligned}
(K_{\Omega_k,\Omega_k})_{ij} &= a_k(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) && \forall (i,j) \in \mathcal{D}^k_\Omega \times \mathcal{D}^k_\Omega && k = 1,2 \\
(K_{\Gamma_k,\Omega_k})_{ij} &= a_k(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) && \forall (i,j) \in \mathcal{D}^k_\Gamma \times \mathcal{D}^k_\Omega && k = 1,2 \\
(K_{\Omega_k,\Gamma_k})_{ij} &= a_k(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) && \forall (i,j) \in \mathcal{D}^k_\Omega \times \mathcal{D}^k_\Gamma && k = 1,2 \\
(K_{\Gamma_k,\Gamma_k})_{ij} &= a_k(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) && \forall (i,j) \in \mathcal{D}^k_\Gamma \times \mathcal{D}^k_\Gamma && k = 1,2
\end{aligned}$$

Defining the vectors

$$\begin{aligned}
(\mathbf{u}_{\Omega_k})_i &= u^k_i && \forall i \in \mathcal{D}^k_\Omega && k = 1,2 \\
(\mathbf{u}_{\Gamma_k})_i &= u^k_i && \forall i \in \mathcal{D}^k_\Gamma && k = 1,2 \\
(\boldsymbol{\lambda})_i &= \lambda^1_i && \forall i \in \mathcal{D}^1_\Gamma
\end{aligned}$$

And the right-hand side vectors

$$\begin{aligned}
(\mathbf{f}_{\Omega_k})_i &= F_k(\boldsymbol{\phi}_i(\mathbf{x})) - \sum_{j\in\mathcal{D}^k_D} u^k_j a_k(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) && \forall i \in \mathcal{D}^k_\Omega && k = 1,2 \\
(\mathbf{f}_{\Gamma_k})_i &= F_k(\boldsymbol{\phi}_i(\mathbf{x})) - \sum_{j\in\mathcal{D}^k_D} u^k_j a_k(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) && \forall i \in \mathcal{D}^k_\Gamma && k = 1,2
\end{aligned}$$

In the INTERNODES method, we end up with small interface mass matrices

$$(M_k)_{ij} = b_k(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) \quad \forall (i,j) \in \mathcal{D}^k_\Gamma \times \mathcal{D}^k_\Gamma \quad k = 1,2$$

and interpolation matrices

$$\begin{aligned}
(R_{12})_{ij} &= \boldsymbol{\phi}_j(\mathbf{x}_i) \quad \forall (i,j) \in \mathcal{D}^1_\Gamma \times \mathcal{D}^2_\Gamma \\
(R_{21})_{ij} &= \boldsymbol{\phi}_j(\mathbf{x}_i) \quad \forall (i,j) \in \mathcal{D}^2_\Gamma \times \mathcal{D}^1_\Gamma
\end{aligned}$$

One should notice that

$$b_1(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) = -\tilde{b}(\boldsymbol{\phi}_j(\mathbf{x}), \boldsymbol{\phi}_i(\mathbf{x})) \quad \forall (i,j) \in \mathcal{D}^1_\Gamma \times \mathcal{D}^1_\Gamma$$

The algebraic counterpart of the previous equations is written as

$$\begin{pmatrix}
K_{\Omega_1\Omega_1} & K_{\Omega_1\Gamma_1} & 0 & 0 & 0 \\
K_{\Gamma_1\Omega_1} & K_{\Gamma_1\Gamma_1} & 0 & 0 & -M_1 \\
0 & 0 & K_{\Omega_2\Omega_2} & K_{\Omega_2\Gamma_2} & 0 \\
0 & 0 & K_{\Gamma_2\Omega_2} & K_{\Gamma_2\Gamma_2} & M_2 R_{21} \\
0 & M_1 & 0 & -M_1 R_{12} & 0
\end{pmatrix}
\begin{pmatrix}
\mathbf{u}_{\Omega_1} \\ \mathbf{u}_{\Gamma_1} \\ \mathbf{u}_{\Omega_2} \\ \mathbf{u}_{\Gamma_2} \\ \boldsymbol{\lambda}
\end{pmatrix}
=
\begin{pmatrix}
\mathbf{f}_{\Omega_1} \\ \mathbf{f}_{\Gamma_1} \\ \mathbf{f}_{\Omega_2} \\ \mathbf{f}_{\Gamma_2} \\ \mathbf{s}
\end{pmatrix}$$

The vector $\mathbf{s}$ is given by $\mathbf{s} = M_1(R_{12}\mathbf{r}_{\Gamma_2} - \mathbf{r}_{\Gamma_1})$ and R_{12} and R_{21} are the interpolation matrices introduced in the previous chapter. The matrix M_1 can be eliminated from the last block row as it is symmetric positive definite (and therefore invertible). We will therefore solve the linear system $A\mathbf{x} = \mathbf{b}$ with

$$A = \begin{pmatrix} K_{\Omega_1\Omega_1} & K_{\Omega_1\Gamma_1} & 0 & 0 & 0 \\ K_{\Gamma_1\Omega_1} & K_{\Gamma_1\Gamma_1} & 0 & 0 & -M_1 \\ 0 & 0 & K_{\Omega_2\Omega_2} & K_{\Omega_2\Gamma_2} & 0 \\ 0 & 0 & K_{\Gamma_2\Omega_2} & K_{\Gamma_2\Gamma_2} & M_2R_{21} \\ 0 & I & 0 & -R_{12} & 0 \end{pmatrix} \quad \mathbf{x} = \begin{pmatrix} \mathbf{u}_{\Omega_1} \\ \mathbf{u}_{\Gamma_1} \\ \mathbf{u}_{\Omega_2} \\ \mathbf{u}_{\Gamma_2} \\ \boldsymbol{\lambda} \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} \mathbf{f}_{\Omega_1} \\ \mathbf{f}_{\Gamma_1} \\ \mathbf{f}_{\Omega_2} \\ \mathbf{f}_{\Gamma_2} \\ \mathbf{d} \end{pmatrix}$$

with $\mathbf{d} = R_{12}\mathbf{r}_{\Gamma_2} - \mathbf{r}_{\Gamma_1}$.

2.4 Implementation aspects

One should notice immediately that the sign of the Lagrange multipliers has not been enforced anywhere. These inequality constraints are one of the sources of nonlinearity in computational contact mechanics. The nonlinearity of contact constraints leads to an iterative procedure which requires solving a sequence of linear systems $A^{(k)}\mathbf{x}^{(k)} = \mathbf{b}^{(k)}$. Upon initialization, a potential contact interface is identified based on an intersection radius R. A sketch of the algorithm is presented in Figure 2.2 and a pseudo-code in Algorithm 2.1. The algorithm starts with the initialization of an empty list. Nodes belonging to the boundary of both bodies are then visited one after the other. For each node, a circle (for 2D problems) or a sphere (for 3D problems) of radius R is drawn around this node. The node is added to the list if any node from the other body is contained in the circle. Once the boundary of both bodies has been entirely covered, the subset $\mathcal{C}_0$ is created. For the contact algorithm to converge, the final and all intermediate contact interfaces must be contained in it. Hence, $\mathcal{C}^{(k)} \subseteq \mathcal{C}_0 \subseteq \partial\Omega$. In order to meet this requirement, the intersection radius must be chosen large enough. If the properties of the contact problem are unknown, setting $R = \infty$ such that $\mathcal{C}_0 = \partial\Omega$ is the safest choice. It must be emphasized that in general $\mathcal{C}^{(0)} \neq \mathcal{C}_0$ although it might happen in exceptional circumstances. The subset $\mathcal{C}^{(0)}$ is a correction of $\mathcal{C}_0$ based on radial basis function interpolation requirements. Namely, diagonal dominance of the interpolation matrices is enforced at this stage. Moreover, zero rows in the matrix Φ_{NM} enable to detect isolated nodes which are discarded from the potential contact interface. Once $\mathcal{C}^{(0)}$ has been identified, the linear system can be solved.

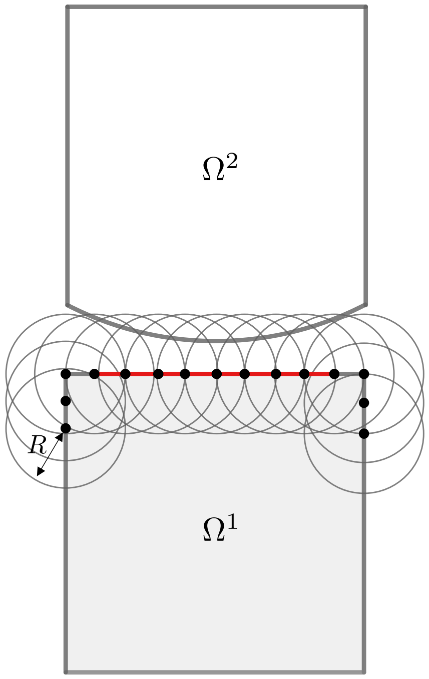

Figure 2.2: Definition of $\mathcal{C}_0$

Algorithm 2.1 Definition of $\mathcal{C}_0$

```
1: **Input**: Intersection radius $R$, set of boundary nodes $\mathcal{N}_1, \mathcal{N}_2$ with $\mathcal{N}_1 \cup \mathcal{N}_2 = \mathcal{N}$ and $\mathcal{N}_1 \cap \mathcal{N}_2 = \emptyset$
2: **Output**: Initial potential contact interface $\mathcal{C}_0$
3: Initialize $\mathcal{N}_c = \emptyset$
4: for k=1,2 do
5:     for $i \in \mathcal{N}_k$ do
6:         if $\exists j \in \mathcal{N} \setminus \mathcal{N}_k : \|\mathbf{x}_i - \mathbf{x}_j\|_2 \leq R$ then
7:             $\mathcal{N}_c \leftarrow \mathcal{N}_c \cup \{i\}$
8:         end if
9:     end for
10: end for
11: Initialize and return $\mathcal{C}_0$ constructed from $\mathcal{N}_c$
```

Referring to the terminology of optimization, this contact interface forms an active set which involves nodes belonging to the boundary of both bodies. The linear system enforces contact on all of the potential contact interface. Once the solution has been computed, the Hertz-Signorini-Moreau conditions must be verified. Due to the discretized geometry, they are not verified exactly but up to a certain tolerance for the interpenetration of the bodies. This tolerance typically depends on the mesh size. If the conditions are not verified, the contact interface must be updated by adding or removing nodes depending on the condition which is violated. A priori, both conditions could be violated but at different locations on the contact interface. However, the algorithm is designed such that nodes can only be added or removed at each iteration but not both in order to avoid getting stuck in endless loops. Consequently, the size of the linear system is changing at each iteration. A sketch of the algorithm is provided in Algorithm 2.2.

Algorithm 2.2 Contact algorithm

```
1: Input: Intersection radius $R$
2: Output: Solution of the contact problem
3: Initialize interface based on intersection radius $R$.    ▷ If $R$ is large enough, all the nodes on the boundary will form the initial interface.
4: Update interface through RBF interpolation by detecting isolated nodes.
5: while Not converged do
6:     Solve the linear system
7:     if Projected Lagrange multipliers are all negative then    ▷ All of the interface is in compression
8:         if No penetration is detected then
9:             Break    ▷ Successfully converged
10:        else
11:            Update the interface by adding nodes of interpenetrating bodies
12:        end if
13:    else
14:        Update the interface by removing nodes in tension
15:    end if
16: end while
```

Chapter 3

Properties of the INTERNODES matrix

3.1 Introduction

As we have seen, the continuous problem already has a saddle point structure which we recover on the discrete level. It seems therefore natural to preserve the structure from which the linear system stems. It leads to considering a 2×2 block structure commonly known as a saddle point system in the literature. Thus, we will most frequently work with the submatrices and subvectors of this linear system defined as follows:

$$K = \begin{pmatrix} K_{\Omega_1\Omega_1} & K_{\Omega_1\Gamma_1} & & \\ K_{\Gamma_1\Omega_1} & K_{\Gamma_1\Gamma_1} & & \\ 0 & 0 & K_{\Omega_2\Omega_2} & K_{\Omega_2\Gamma_2} \\ 0 & 0 & K_{\Gamma_2\Omega_2} & K_{\Gamma_2\Gamma_2} \end{pmatrix} \quad \mathbf{u} = \begin{pmatrix} \mathbf{u}_{\Omega_1} \\ \mathbf{u}_{\Gamma_1} \\ \mathbf{u}_{\Omega_2} \\ \mathbf{u}_{\Gamma_2} \end{pmatrix} \quad \mathbf{f} = \begin{pmatrix} \mathbf{f}_{\Omega_1} \\ \mathbf{f}_{\Gamma_1} \\ \mathbf{f}_{\Omega_2} \\ \mathbf{f}_{\Gamma_2} \end{pmatrix}$$

$$B = \begin{pmatrix} 0 \\ -M_1 \\ 0 \\ M_2 R_{21} \end{pmatrix} \quad \tilde{B} = \begin{pmatrix} 0 & I & 0 & -R_{12} \end{pmatrix}$$

Thus, we consider the linear system

$$\begin{pmatrix} K & B \\ \tilde{B} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{d} \end{pmatrix} \tag{3.1}$$

That is $A\mathbf{x} = \mathbf{b}$ with

$$A = \begin{pmatrix} K & B \\ \tilde{B} & 0 \end{pmatrix} \quad \mathbf{x} = \begin{pmatrix} \mathbf{u} \\ \boldsymbol{\lambda} \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} \mathbf{f} \\ \mathbf{d} \end{pmatrix}$$

The stiffness matrix K represents the bulk of the matrix. Based on the previous notations, we denote $n = |\mathcal{D}_F|$ the number of degrees of freedom in displacement and $m = |\mathcal{D}_\Gamma^1|$ the number of degrees of freedom on the interface of body 1. It represents the number of constraints linking the degrees of freedom of the interface. Thus, $K \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $\tilde{B} \in \mathbb{R}^{m \times n}$ with $n >> m$ and the vectors $\mathbf{u}, \mathbf{f} \in \mathbb{R}^n$ and $\boldsymbol{\lambda}, \mathbf{d} \in \mathbb{R}^m$.

It is very important to bare in mind that contrary to single-body elasticity problems, the stiffness matrix K is only positive semidefinite in general. There might exist non-trivial vectors in its kernel. Let us therefore denote $s = \dim \ker(K)$, the nullity of K. In the context of mechanics, such vectors are in fact combinations of translations and rotations and are called rigid modes. The displacement fields associated to these vectors do not generate any deformation of the body. Consequently, the kernel of the stiffness matrix is a small dimensional subspace with $s = 3$ in 2D (2 translations, 1 rotation) and $s = 6$ in 3D (3 translations, 3 rotations) at most (Bochev and Lehoucq, 2011). More precisely, the stiffness matrix is singular if these rigid modes are not blocked simultaneously for both bodies. A particular case is when Dirichlet boundary conditions are not prescribed on one of the bodies. More generally, it is important to note that if Dirichlet boundary conditions are allowed to be prescribed along separate directions (as is customary in structural mechanics), both blocks of the stiffness matrix could be singular simultaneously. When investigating computational methods, the structure of K will be entirely ignored as it depends on the numbering of the nodes in the mesh. We must only bear in mind that it is symmetric positive semidefinite.

The matrices B and $\tilde{B}$ are coupling matrices linking the degrees of freedom on the interface. The iterative nature of the contact algorithm means these matrices will change from one iteration to the next. Clearly the matrices B and $\tilde{B}$ are very sparse. However, they contain small dense submatrices. It is easy to verify both matrices have full rank m. The matrix $\tilde{B}$ contains the identity matrix in one of its blocks whereas the matrix B contains the interface mass matrix of body 1 which, from finite element theory, is known to be symmetric positive definite (SPD).

The interface mass matrices and interpolation matrices which form part of the matrices B and $\tilde{B}$ have a special structure which we will take advantage of. It has been shown that the Galerkin mass matrix for vector valued PDEs has a Kronecker product structure (Voet, 2020). Such a structure arises when different components of a vector field are approximated in functional spaces sharing the same basis functions. This situation is frequently encountered in applications. In particular, it was shown that when the unknowns are ordered such that all nodal degrees of freedom are grouped together, then the mass matrix is expressed as $M = (\bar{M} \otimes I_d)$. The corresponding displacement vector is given by $\mathbf{u}^T = (u_{1,x}, u_{1,y}, u_{1,z}, \ldots, u_{n,x}, u_{n,y}, u_{n,z})^T$ for $d = 3$. The size of $\bar{M}$ is only equal to the number of nodes and I_d is the identity matrix of size d. The scalar case is recovered when $d = 1$. This expression leads to major computational savings both in terms of storage and floating point operations. Indeed, one only needs to store and assemble the smaller matrix $\bar{M}$ instead of the full matrix M. Moreover, the structure can be exploited when solving linear systems or computing matrix-vector products. It turns out the interpolation matrices have a similar Kronecker product structure. Again, this can be understood from the fact that all components of the displacement field are interpolated using the same radial basis functions. Hence, we can write $R_{ij} = (\bar{R}_{ij} \otimes I_d)$ for $i, j = 1, 2$ and $i \neq j$. This structure will be exploited on various occasions throughout the project.

Let us now return to the linear system with the INTERNODES matrix. The linear system in equation (3.1) is a nonsymmetric saddle point system. Such systems arise in an impressively large collection of scientific disciplines including fluid dynamics, optimization and contact mechanics to name just a few. Due to the numerous underlying applications, they have been extensively studied by the scientific community and there exists abundant literature on the topic. We will later rely heavily on it for the design of preconditioners.

For the time being, we must verify the well-posedness of the problem. Our eventual goal being application driven, we will focus on verifying the well-posedness of the discrete finite-dimensional problem. From a linear algebra perspective, it translates to verifying the invertibility of the INTERNODES matrix in equation (3.1).

3.2 Invertibility conditions

The invertibility conditions are given in the following theorem (Cao, 2008).

Theorem 3.1

The nonsymmetric saddle point matrix

$$A = \begin{pmatrix} K & B \\ \tilde{B} & 0 \end{pmatrix}$$

is invertible if and only if the following conditions are satisfied:

1. $\operatorname{rank}(B) = \operatorname{rank}(\tilde{B}) = m$
2. $V_{B_2}^T K V_{\tilde{B}_2}$ is invertible, where V_{B_2} and $V_{\tilde{B}_2}$ are bases for the kernels of B^T and $\tilde{B}$ respectively

Proof. The first condition is a straightforward necessary condition. We proceed by contradiction. Assume A is invertible and $\operatorname{rank}(B) < m$, then there exists a vector $\mathbf{p} \in \ker(B)$ with $\mathbf{p} \neq \mathbf{0}$. Setting $\mathbf{x}^T = (\mathbf{0}^T, \mathbf{p}^T)$, we have $A\mathbf{x} = \mathbf{0}$ with $\mathbf{x} \neq \mathbf{0}$ and A is singular which is a contradiction. A similar argument for A^T instead of A shows that $\operatorname{rank}(\tilde{B}) < m$ is impossible.

The proof for the second condition is much inspired from Cao (2008). Let

$$B^T = U_B[\Sigma_B \; 0]V_B^T$$
$$\tilde{B} = U_{\tilde{B}}[\Sigma_{\tilde{B}} \; 0]V_{\tilde{B}}^T$$

be the singular value decomposition of B^T and $\tilde{B}$ respectively. $U_B, U_{\tilde{B}} \in \mathbb{R}^{m\times m}$ and $V_B, V_{\tilde{B}} \in \mathbb{R}^{n\times n}$ are orthogonal matrices and $\Sigma_B, \Sigma_{\tilde{B}} \in \mathbb{R}^{m\times m}$ are diagonal matrices containing the singular values. The matrices V_B and $V_{\tilde{B}}$ are further partitioned as

$$V_B = [V_{B_1} V_{B_2}] \text{ and } V_{\tilde{B}} = [V_{\tilde{B}_1} V_{\tilde{B}_2}]$$

and note that V_{B_2} and $V_{\tilde{B}_2}$ are bases for the kernels of B^T and $\tilde{B}$ respectively. We now form the orthogonal matrices S_1 and S_2 such that

$$S_1 = \begin{pmatrix} V_B^T & 0 \\ 0 & U_{\tilde{B}}^T \end{pmatrix} \quad S_2 = \begin{pmatrix} V_{\tilde{B}} & 0 \\ 0 & U_B \end{pmatrix}$$

and consider the matrix $S_1 A S_2$. Since S_1 and S_2 are orthogonal matrices (and therefore invertible), $S_1 A S_2$ is invertible if and only if A is. Explicitly, we have

$$\begin{aligned} S_1 A S_2 &= \begin{pmatrix} V_B^T & 0 \\ 0 & U_{\tilde{B}}^T \end{pmatrix} \begin{pmatrix} K & B \\ \tilde{B} & 0 \end{pmatrix} \begin{pmatrix} V_{\tilde{B}} & 0 \\ 0 & U_B \end{pmatrix} \\ &= \begin{pmatrix} V_B^T K V_{\tilde{B}} & V_B^T B U_B \\ U_{\tilde{B}}^T \tilde{B} V_{\tilde{B}} & 0 \end{pmatrix} \\ &= \begin{pmatrix} V_{B_1}^T K V_{\tilde{B}_1} & V_{B_1}^T K V_{\tilde{B}_2} & \Sigma_B \\ V_{B_2}^T K V_{\tilde{B}_1} & V_{B_2}^T K V_{\tilde{B}_2} & 0 \\ \Sigma_{\tilde{B}} & 0 & 0 \end{pmatrix} \end{aligned}$$

After permuting the first and last block rows, we obtain

$$\begin{pmatrix} \Sigma_{\tilde{B}} & 0 & 0 \\ V_{B_2}^T K V_{\tilde{B}_1} & V_{B_2}^T K V_{\tilde{B}_2} & 0 \\ V_{B_1}^T K V_{\tilde{B}_1} & V_{B_1}^T K V_{\tilde{B}_2} & \Sigma_B \end{pmatrix}$$

The determinant of a block triangular matrix being the product of the determinants of its diagonal blocks, the matrix is invertible if and only if $V_{B_2}^T K V_{\tilde{B}_2}$ is invertible. $\square$

These conditions must be verified by the INTERNODES matrix. The first condition is trivially satisfied and the second condition implies that

$$\ker(K) \cap \ker(\tilde{B}) = \emptyset \tag{3.2}$$

$$\ker(K) \cap \ker(B^T) = \emptyset \tag{3.3}$$

where we made use of the symmetry of K. We will show that these two conditions are satisfied under some assumptions on the boundary conditions.

As we have already noted, the kernel of the stiffness matrix is a small dimensional subspace which only contains linear combinations of translations and rotations. On the other hand, by the rank-nullity theorem, $\dim\ker(\tilde{B}) = \dim\ker(B^T) = n - m$ is extremely large. Thus, we would like to show that $\ker(K) \not\subseteq \ker(\tilde{B})$ and $\ker(K) \not\subseteq \ker(B^T)$. Unfortunately, the necessary boundary conditions to be prescribed are a priori unknown. Thus, we will initially assume $\Gamma_D^1 = \Gamma_D^2 = \emptyset$ and understand why the resulting linear system is singular. Based on this knowledge, we will deduce the necessary Dirichlet boundary conditions to be prescribed. To simplify the discussion, we will further assume a two-dimensional problem.

The assumptions we have made imply that the coupled body can be freely translated or rotated. These so-called rigid modes live in the kernel of K. The displacement vector $\mathbf{u}_h$ belongs to the subspace

$$\{\mathbf{u} = \mathbf{a} + R\mathbf{r} \colon \mathbf{a} \in \mathbb{R}^2\}$$

where $R \in \mathbb{R}^{2\times 2}$ is a rotation matrix and $\mathbf{r}$ is the coordinate vector. We will assume rotations around the origin for simplicity. When we think in terms of vectors of coefficients, $\mathbf{u}_{\Gamma_k}$ either belongs to $\mathcal{U}_T^k$ or $\mathcal{U}_R^k$ which are spaces of translations and rotations respectively. The space $\mathcal{U}_T^k$ is defined as

$$\mathcal{U}_T^k = \operatorname{span}\{\mathbf{1}_k \otimes \mathbf{e}_1, \mathbf{1}_k \otimes \mathbf{e}_2\} \quad k = 1, 2$$

where $\mathbf{e}_i$ is the ith unit vector of the canonical basis of $\mathbb{R}^2$, $\mathbf{1}_k \in \mathbb{R}^{n_k}$ is the vector of all ones and n_k is the number of nodes on the contact interface of body k. Clearly, $\dim \mathcal{U}_T^k = 2$. In the case of rotations, $\mathcal{U}_R^k$ is defined as

$$\begin{aligned} \mathcal{U}_R^k &= \operatorname{span}\{\big(I_{n_k} \otimes R\big)\big((\mathbf{r}_x^k \otimes \mathbf{e}_1) + (\mathbf{r}_y^k \otimes \mathbf{e}_2)\big)\} \\ &= \operatorname{span}\{(\mathbf{r}_x^k \otimes R\mathbf{e}_1) + (\mathbf{r}_y^k \otimes R\mathbf{e}_2)\} \quad k = 1, 2 \end{aligned}$$

and clearly $\dim \mathcal{U}_R^k = 1$. We first consider a case of translation where all components of $\mathbf{u}$ for displacements along x are equal to u and all components for displacements along y are equal to v. In particular, the interface displacements can be expressed as $\mathbf{u}_{\Gamma_k} = u(\mathbf{1}_k \otimes \mathbf{e}_1) + v(\mathbf{1}_k \otimes \mathbf{e}_2)$.

Let us now compute $\tilde{B}\mathbf{u}$ explicitly. We have

$$\begin{aligned}
\tilde{B}\mathbf{u} &= \mathbf{u}_{\Gamma_1} - R_{12}\mathbf{u}_{\Gamma_2} \\
&= u(\mathbf{1}_1 \otimes \mathbf{e}_1) + v(\mathbf{1}_1 \otimes \mathbf{e}_2) - (\bar{R}_{12} \otimes I_2)\big(u(\mathbf{1}_2 \otimes \mathbf{e}_1) + v(\mathbf{1}_2 \otimes \mathbf{e}_2)\big) \\
&= u(\mathbf{1}_1 \otimes \mathbf{e}_1) + v(\mathbf{1}_1 \otimes \mathbf{e}_2) - \big(u(\bar{R}_{12}\mathbf{1}_2 \otimes \mathbf{e}_1) + v(\bar{R}_{12}\mathbf{1}_2 \otimes \mathbf{e}_2)\big)
\end{aligned}$$

Now recall that the rescaled radial basis functions are designed to interpolate exactly a constant function. This was one of the features introduced by Deparis et al. (2014). The algebraic meaning is that $c\bar{R}_{ij}\mathbf{1}_j = c\mathbf{1}_i \quad \forall c \in \mathbb{R}$. Hence,

$$\tilde{B}\mathbf{u} = u(\mathbf{1}_1 \otimes \mathbf{e}_1) + v(\mathbf{1}_1 \otimes \mathbf{e}_2) - \big(u(\mathbf{1}_1 \otimes \mathbf{e}_1) + v(\mathbf{1}_1 \otimes \mathbf{e}_2)\big) = 0$$

Thus, we have found a vector $\mathbf{u} \in \{\ker(K) \cap \ker(\tilde{B})\}$ and the INTERNODES matrix is singular. Consequently, Dirichlet boundary conditions must be prescribed such that at least some of the rigid body motions are removed from the kernel of K. Applying Dirichlet boundary conditions will change the size of K and shrink the dimension of $\mathcal{U}_T^k$ and $\mathcal{U}_R^k$. However, the remaining rigid modes have the same structure.

In engineering applications, it is customary to apply Dirichlet boundary conditions on separate components of the displacements. For instance, prescribing u_x on some portion of the boundary while not prescribing the other components. The analysis is much more elaborate in this case and special care is required. For two-dimensional problems, these boundary conditions fix the direction vector of the translation such that $\mathbf{u}_{\Gamma_k} \in \mathcal{U}_T^k = \operatorname{span}\{u_k(\mathbf{1}_k \otimes \mathbf{e}_1) + v_k(\mathbf{1}_k \otimes \mathbf{e}_2)\}$ for fixed constants u_k and v_k. Thus, $\dim \mathcal{U}_T^k = 1$. In such an event, the dimension of the kernel of K is indeed reduced as we cannot choose the direction of translation anymore. If we assume such boundary conditions are prescribed on both bodies, we obtain

$$\begin{aligned}
\tilde{B}\mathbf{u} &= u_1(\mathbf{1}_1 \otimes \mathbf{e}_1) + v_1(\mathbf{1}_1 \otimes \mathbf{e}_2) - \big(u_2(\mathbf{1}_1 \otimes \mathbf{e}_1) + v_2(\mathbf{1}_1 \otimes \mathbf{e}_2)\big) \\
&= (u_1 - u_2)(\mathbf{1}_1 \otimes \mathbf{e}_1) + (v_1 - v_2)(\mathbf{1}_1 \otimes \mathbf{e}_2)
\end{aligned}$$

form which we deduce the necessary conditions

$$\begin{aligned}
u_1 &\neq u_2 \\
v_1 &\neq v_2
\end{aligned}$$

It means that the constant vector-valued functions

$$\mathbf{u}_h^1 = \begin{pmatrix} u_1 \\ v_1 \end{pmatrix} \text{ and } \mathbf{u}_h^2 = \begin{pmatrix} u_2 \\ v_2 \end{pmatrix}$$

point in different directions. If Dirichlet boundary conditions are prescribed for all components of the displacement field of one of the bodies, then $\dim \mathcal{U}_T^1 = 0$ or $\dim \mathcal{U}_T^2 = 0$. Let us assume $\dim \mathcal{U}_T^1 = 0$ and $\dim \mathcal{U}_T^2 = 2$. Then, $\mathbf{u}_{\Gamma_1} = \mathbf{0}$. Moreover,

$$\tilde{B}\mathbf{u} = -\big(u_2(\mathbf{1}_1 \otimes \mathbf{e}_1) + v_2(\mathbf{1}_1 \otimes \mathbf{e}_2)\big)$$

which is nonzero. We obviously assume u_2 and v_2 are not simultaneously zero since we are interested in non-trivial vectors. All other combinations are treated similarly and lead to a nonzero vector.

Let us now deal with rotations. We assume

$$\mathbf{u}_{\Gamma_k} \in \mathcal{U}_R^k = \operatorname{span}\{(\mathbf{r}_x^k \otimes R\mathbf{e}_1) + (\mathbf{r}_y^k \otimes R\mathbf{e}_2)\} \quad k = 1, 2$$

and we again compute $\tilde{B}\mathbf{u}$. We obtain

$$\begin{aligned}
\tilde{B}\mathbf{u} &= (\mathbf{r}_x^1 \otimes R\mathbf{e}_1) + (\mathbf{r}_y^1 \otimes R\mathbf{e}_2) - \big((\bar{R}_{12}\mathbf{r}_x^2 \otimes R\mathbf{e}_1) + (\bar{R}_{12}\mathbf{r}_y^2 \otimes R\mathbf{e}_2)\big) \\
&= (\mathbf{r}_x^1 - \bar{R}_{12}\mathbf{r}_x^2) \otimes R\mathbf{e}_1 + (\mathbf{r}_y^1 - \bar{R}_{12}\mathbf{r}_y^2) \otimes R\mathbf{e}_2
\end{aligned}$$

In general, both terms are nonzero because the interpolation is not perfect. However, we encounter troubles in the conforming case since $\bar{R}_{12} = I$, $\mathbf{r}_x^1 = \mathbf{r}_x^2$ and $\mathbf{r}_y^1 = \mathbf{r}_y^2$. Rotations will always be eliminated provided Γ_D^k is a set of nonzero measure.

Let us now proceed with the same verification for the matrix B^T. We obtain for translations

$$\begin{aligned}B^T\mathbf{u} &= -M_1\mathbf{u}_{\Gamma_1} + R_{21}^T M_2 \mathbf{u}_{\Gamma_2}\\ &= -u_1(\bar{M}_1\mathbf{1}_1 \otimes \mathbf{e}_1) - v_1(\bar{M}_1\mathbf{1}_1 \otimes \mathbf{e}_2) + u_2(\bar{R}_{21}^T\bar{M}_2\mathbf{1}_2 \otimes \mathbf{e}_1) + v_2(\bar{R}_{21}^T\bar{M}_2\mathbf{1}_2 \otimes \mathbf{e}_2)\\ &= \bar{R}_{21}^T\bar{M}_2\mathbf{1}_2 \otimes (u_2\mathbf{e}_1 + v_2\mathbf{e}_2) - \bar{M}_1\mathbf{1}_1 \otimes (u_1\mathbf{e}_1 + v_1\mathbf{e}_2)\end{aligned}$$

The only potential issue here is linked to the matrix $\bar{R}_{21}^T$. If $m_2 > m_1$, then $\dim\ker(\bar{R}_{21}^T) > 0$ and we must ensure $\bar{R}_{21}^T\bar{M}_2\mathbf{1}_2$ does not vanish in the event that $u_1 = v_1 = 0$. It is difficult to say anything in this case given that the vector $\bar{M}_2\mathbf{1}_2$ and the matrix $\bar{R}_{21}^T$ are completely unrelated. It seems highly unlikely that $\bar{M}_2\mathbf{1}_2 \in \ker(\bar{R}_{21}^T)$. Yet, in the conforming case, $\bar{R}_{21}^T = I$ and $\bar{M}_1 = \bar{M}_2 = \bar{M}$ and $\mathbf{1}_1 = \mathbf{1}_2 = \mathbf{1}$ such that the right-hand side becomes

$$B^T\mathbf{u} = \bar{M}\mathbf{1} \otimes \big((u_2 - u_1)\mathbf{e}_1 + (v_2 - v_1)\mathbf{e}_2\big)$$

Thus, to be on the safe side, the same conditions on u_1, u_2 and v_1, v_2 are needed.

Finally, for rotations

$$B^T\mathbf{u} = (\bar{R}_{21}^T\bar{M}_2\mathbf{r}_x^2 - \bar{M}_1\mathbf{r}_x^1) \otimes R\mathbf{e}_1 + (\bar{R}_{21}^T\bar{M}_2\mathbf{r}_y^2 - \bar{M}_1\mathbf{r}_y^1) \otimes R\mathbf{e}_2$$

None of these quantities are likely to be zero in general. However, the conforming case is again problematic. Since we have enumerated all cases, let us summarize:

- If the meshes are conforming and $\Gamma_D^1 = \Gamma_D^2 = \emptyset$, then there exists a vector $\mathbf{u} \in \{\ker(K) \cap \ker(\tilde{B})\}$ and $\mathbf{u} \in \{\ker(K) \cap \ker(B^T)\}$. This is not surprising since in the conforming case the kernels of $\tilde{B}$ and B^T are the same.
- If the meshes are nonconforming and $\Gamma_D^1 = \Gamma_D^2 = \emptyset$, then there exists a vector $\mathbf{u} \in \{\ker(K) \cap \ker(\tilde{B})\}$.
- If $\dim\ker(K_1) > 0$ and $\dim\ker(K_2) > 0$, then $\ker(K) \cap \ker(\tilde{B}) = \emptyset$ and $\ker(K) \cap \ker(B^T) = \emptyset$ provided the rigid body motions $\mathbf{u}_h^1$ and $\mathbf{u}_h^2$ are not collinear.
- If $\dim\ker(K_1) = 0$ or $\dim\ker(K_2) = 0$, then $\ker(K) \cap \ker(\tilde{B}) = \emptyset$ and $\ker(K) \cap \ker(B^T) = \emptyset$ except maybe in some unlikely circumstances.

These results can be generalized to 3D problems.

3.3 Properties of the INTERNODES matrix and its blocks

In this section, we prove and discuss some of the properties of the matrices involved. This information is useful when investigating the stability of a linear system and may also help in the design of preconditioners. Not all results will be used in upcoming chapters but they are nevertheless mentioned as a potential backbone for future work.

Before discussing the properties of the matrices, we begin by recalling some important results from finite element theory. They can be found in Pearson and Wathen (2012), Elman et al. (2014) and Ern (2002) for instance.

Theorem 3.2

Let $\Omega \subset \mathbb{R}^2$ and $M \in \mathbb{R}^{n\times n}$ be the Galerkin mass matrix resulting from $\mathbb{P}_m$ or $\mathbb{Q}_m$ finite element or spectral element discretizations with $m \geq 1$. Then, the following bounds hold:

$$ch^2 \leq \frac{\mathbf{v}^T M \mathbf{v}}{\|\mathbf{v}\|_2^2} \leq Ch^2 \quad \forall \mathbf{v} \in \mathbb{R}^n$$

where c and C are positive constants independent of the mesh size but dependent on m. The equivalent result for $\Omega \subset \mathbb{R}^3$ is

$$ch^3 \leq \frac{\mathbf{v}^T M \mathbf{v}}{\|\mathbf{v}\|_2^2} \leq Ch^3 \quad \forall \mathbf{v} \in \mathbb{R}^n$$

These results provide the dependency on the mesh size of the smallest and largest eigenvalues of the mass matrix. The next theorem provides equivalent results for the stiffness matrix.

Theorem 3.3

Let $\Omega \subset \mathbb{R}^2$ and $K \in \mathbb{R}^{n\times n}$ be the Galerkin stiffness matrix resulting from $\mathbb{P}_m$ or $\mathbb{Q}_m$ finite element or spectral element discretizations with $m \geq 1$. Then, the following bounds hold:

$$ch^2 \leq \frac{\mathbf{v}^T K \mathbf{v}}{\|\mathbf{v}\|_2^2} \leq C \quad \forall \mathbf{v} \in \mathbb{R}^n$$

where c and C are positive constants independent of the mesh size but dependent on m. The equivalent result for $\Omega \subset \mathbb{R}^3$ is

$$ch^3 \leq \frac{\mathbf{v}^T K \mathbf{v}}{\|\mathbf{v}\|_2^2} \leq Ch \quad \forall \mathbf{v} \in \mathbb{R}^n$$

Although the proof in Elman et al. (2014) is for the scalar Poisson equation, the results are much more general and hold for second order equations under suitable assumptions. General proofs are provided in Ern (2002) for the mass matrix (pages 342-344) and for the stiffness matrix (pages 344-346).

For stability and preconditioning purposes, it is valuable to have information on the spectral condition number of the INTERNODES matrix defined as $\kappa(A) = \|A\|_2 \|A^{-1}\|_2$. We will focus on finding a lower and upper bound on the spectral norm of the matrix. The result is given in the next theorem.

Theorem 3.4

Let $\lambda_1(K)$ denote the largest eigenvalue of the stiffness matrix. Then, the spectral norm of the INTERNODES matrix satisfies the following inequalities

$$\lambda_1(K) \leq \|A\|_2 \leq \sqrt{(\lambda_1(K) + \|B\|_2)^2 + \|\tilde{B}\|_2^2}$$

Proof. By definition, $\sigma_1 = \|A\|_2 = \max_{\|\mathbf{x}\|_2=1} \|A\mathbf{x}\|_2$. The INTERNODES matrix $A \in \mathbb{R}^{(n+m)\times(n+m)}$ is expressed as

$$A = \begin{pmatrix} K & B \\ \tilde{B} & 0 \end{pmatrix} \text{ with } K \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{n\times m} \text{ and } \tilde{B} \in \mathbb{R}^{m\times n}$$

For the lower bound,

$$\|A\|_2^2 = \max_{\|\mathbf{x}\|_2=1} \|A\mathbf{x}\|_2^2 \geq \|A\mathbf{y}\|_2^2 \text{ with } \|\mathbf{y}\|_2 = 1$$

Choose $\mathbf{y}^T = (\mathbf{u}^T, \mathbf{0}^T)$ with $\|\mathbf{u}\|_2 = 1$ such that $\|K\mathbf{u}\|_2^2$ is maximized. Then,

$$\|A\mathbf{y}\|_2^2 = \|K\mathbf{u}\|_2^2 + \|\tilde{B}\mathbf{u}\|_2^2 \geq \|K\mathbf{u}\|_2^2 = \lambda_1^2(K)$$

For the upper bound, let us write $\mathbf{x}^T = (\mathbf{u}^T, \mathbf{p}^T)$ with $\mathbf{u} \in \mathbb{R}^n$ and $\mathbf{p} \in \mathbb{R}^m$. Then,

$$\begin{aligned}
\max_{\|\mathbf{x}\|_2=1} \|A\mathbf{x}\|_2^2 &= \max_{\|\mathbf{x}\|_2=1} (\|K\mathbf{u} + B\mathbf{p}\|_2^2 + \|\tilde{B}\mathbf{u}\|_2^2) \\
&\leq \max_{\|\mathbf{x}\|_2=1} ((\|K\mathbf{u}\|_2 + \|B\mathbf{p}\|_2)^2 + \|\tilde{B}\mathbf{u}\|_2^2) \\
&\leq (\max_{\|\mathbf{u}\|_2=1} \|K\mathbf{u}\|_2 + \max_{\|\mathbf{p}\|_2=1} \|B\mathbf{p}\|_2)^2 + \max_{\|\mathbf{u}\|_2=1} \|\tilde{B}\mathbf{u}\|_2^2 \\
&= (\|K\|_2 + \|B\|_2)^2 + \|\tilde{B}\|_2^2 \\
&= (\lambda_1(K) + \|B\|_2)^2 + \|\tilde{B}\|_2^2
\end{aligned}$$

Taking the square root completes the proof. □

Unfortunately, not much is known about $\|A^{-1}\|_2$. Thus, we do not have a bound on the condition number itself. However, the bounds on the spectral norm are already insightful results. Indeed, for 2D problems, the largest eigenvalue of the stiffness matrix is bounded by a constant independent of the mesh size according to Theorem 3.3. Since this quantity appears both in the lower and upper bound, it indicates that the spectral norm of the INTERNODES matrix can only grow if the spectral norm of at least one of the blocks B or $\tilde{B}$ grows. It is therefore important to characterize the singular values of the blocks B and $\tilde{B}$. The results are summarized in the next few lemmas.

Lemma 3.1

Let $\lambda_1(M_1)$ and $\lambda_1(M_2)$ denote the largest eigenvalue of M_1 and M_2 respectively and $\lambda_m(M_1)$ the smallest eigenvalue of M_1. Then, the singular values of the matrix B satisfy the inequalities

$$\lambda_m(M_1) \leq \sigma_i(B) \leq \sqrt{\lambda_1^2(M_1) + \lambda_1^2(M_2)\|R_{21}\|_2^2} \quad i = 1, 2, \ldots, m$$

Proof. We begin by recalling the definition of B as

$$B = \begin{pmatrix} 0 \\ -M_1 \\ 0 \\ M_2R_{21} \end{pmatrix}$$

and note that for a vector $\mathbf{p} \in \mathbb{R}^m$, $\|B\mathbf{p}\|_2^2 = \|M_1\mathbf{p}\|_2^2 + \|M_2R_{21}\mathbf{p}\|_2^2$. For the lower bound,

$$\sigma_m^2(B) = \min_{\|\mathbf{p}\|_2=1} (\|M_1\mathbf{p}\|_2^2 + \|M_2R_{21}\mathbf{p}\|_2^2)$$

Depending on the value of m_1 and m_2, R_{21} could potentially have a non-trivial kernel. Therefore, the only safe assessment is to state that $\|M_2R_{21}\mathbf{p}\|_2^2 \geq 0 \quad \forall \mathbf{p} \in \mathbb{R}^m$ which leads to

$$\sigma_m^2(B) \geq \min_{\|\mathbf{p}\|_2=1} \|M_1\mathbf{p}\|_2^2 = \lambda_m^2(M_1)$$

Furthermore, for the upper bound,

$$\begin{aligned}\sigma_1^2(B) = \max_{\|\mathbf{p}\|_2=1} \|B\mathbf{p}\|_2^2 &\leq \max_{\|\mathbf{p}\|_2=1} \|M_1\mathbf{p}\|_2^2 + \max_{\|\mathbf{p}\|_2=1} \|M_2R_{21}\mathbf{p}\|_2^2 \\ &\leq \|M_1\|_2^2 + \|M_2\|_2^2\|R_{21}\|_2^2 \\ &= \lambda_1^2(M_1) + \lambda_1^2(M_2)\|R_{21}\|_2^2\end{aligned}$$

Taking the square root, we obtain the desired result. □

Lemma 3.2

The singular values of the matrix $\tilde{B}$ satisfy the inequalities

$$1 \leq \sigma_i(\tilde{B}) \leq \sqrt{1 + \|R_{12}\|_2^2} \quad i = 1, 2, \ldots, m$$

Proof. Recalling the definition of $\tilde{B}$ as

$$\tilde{B} = \begin{pmatrix} 0 & I & 0 & -R_{12} \end{pmatrix}$$

The singular values of $\tilde{B}$ are the square root of the eigenvalues of $\tilde{B}\tilde{B}^T$. We obtain straightforwardly

$$\begin{aligned}\sigma_m^2(\tilde{B}) &= \min_{\|\mathbf{p}\|_2=1} \|\tilde{B}^T\mathbf{p}\|_2^2 = \min_{\|\mathbf{p}\|_2=1} (1 + \|R_{12}^T\mathbf{p}\|_2^2) \geq 1 \\ \sigma_1^2(\tilde{B}) &= \max_{\|\mathbf{p}\|_2=1} \|\tilde{B}^T\mathbf{p}\|_2^2 = \max_{\|\mathbf{p}\|_2=1} (1 + \|R_{12}^T\mathbf{p}\|_2^2) = 1 + \|R_{12}\|_2^2\end{aligned}$$

Once again, we conclude by taking the square root. □

Some important consequences can be drawn from these results. We are most interested in knowing under which circumstances the upper bounds on the singular values of B and $\tilde{B}$ are growing. In such an event, $\|B\|_2$ would most likely grow as well (whereas $\|\tilde{B}\|_2$ would certainly grow) and hence could lead to a growth of $\|A\|_2$ according to Theorem 3.4. For the matrix B, the situation seems under control. From Theorem 3.2, the largest eigenvalues of the interface mass matrices depend on the mesh size. In fact, they depend only linearly on the mesh size for 2D problems because the integral is taken over the interface. Thus, any growth of $\|R_{21}\|_2$ with mesh refinement would be damped by a factor which is decreasing with the mesh size. The only possibility for the upper bound to grow is that $\|R_{21}\|_2$ depends on the inverse of the mesh size to some power $q \geq 2$, which is unlikely.

The situation is unfortunately far more troublesome for the matrix $\tilde{B}$. This time, nothing is damping a potential growth of $\|R_{12}\|_2$. Thus, if this quantity is growing, so will $\|\tilde{B}\|_2$.

Therefore, if any growth in $\|A\|_2$ is experienced, it is most likely coming from the matrix $\tilde{B}$. More specifically, it is entirely linked to the interpolation.

A corollary of these results can be easily derived in the context of a conforming mesh.

Corollary 3.1

In the context of a conforming mesh, the singular values of B and $\tilde{B}$ satisfy the inequalities

$$\begin{aligned} \sqrt{2}\lambda_m(M) \leq \sigma_i(B) \leq \sqrt{2}\lambda_1(M) \qquad & i = 1, 2, \ldots, m \\ \sigma_i(\tilde{B}) = \sqrt{2} \qquad & i = 1, 2, \ldots, m \end{aligned}$$

Proof. When the mesh is conforming the matrices B and $\tilde{B}$ simplify as $M_1 = M_2 = M$ and $R_{12} = R_{21} = I$. Hence,

$$B = \begin{pmatrix} 0 \\ -M \\ 0 \\ M \end{pmatrix} \text{ and } \tilde{B} = \begin{pmatrix} 0 & I & 0 & -I \end{pmatrix}$$

Thus, $\forall \mathbf{p} \in \mathbb{R}^m$ with $\|\mathbf{p}\|_2 = 1$, $\|B\mathbf{p}\|_2^2 = 2\|M\mathbf{p}\|_2^2$. Hence

$$\begin{aligned} \sigma_m^2(B) &= 2\lambda_m^2(M) \\ \sigma_1^2(B) &= 2\lambda_1^2(M) \end{aligned}$$

Moreover, $\forall \mathbf{p} \in \mathbb{R}^m$ with $\|\mathbf{p}\|_2 = 1$, $\|\tilde{B}^T\mathbf{p}\|_2^2 = 2\|\mathbf{p}\|_2^2 = 2$. Therefore, in particular $\sigma_i^2(\tilde{B}) = 2 \quad i = 1, 2, \ldots, m$. □

Consequently, the next result states that for a conforming mesh the spectral norm of A cannot grow.

Corollary 3.2

In the context of a conforming mesh, the spectral norm of the INTERNODES matrix satisfies the following inequalities

$$\lambda_1(K) \leq \|A\|_2 \leq \sqrt{(\lambda_1(K) + \sqrt{2}\lambda_1(M))^2 + 2}$$

Proof. Direct consequence of Corollary 3.1 □

3.4 Stability of the linear system

One of the most important parameters when solving a contact problem with the INTERNODES method is the choice of radius for the radial basis functions. As, we have seen, this choice is much constrained by the requirement for strict diagonal dominance of the interpolation matrices. In this section, we will investigate the stability of the method with respect to the choice of radius. We will consider two linear systems obtained with two different sets of radiuses $\mathcal{R}_1$ and $\mathcal{R}_2$. Let R_{12} and R_{21} denote the interpolation matrices computed from a set of radiuses $\mathcal{R}_1$ and $\hat{R}_{12}$ and $\hat{R}_{21}$ be the same matrices computed from a set of radiuses $\mathcal{R}_2$. For these different set of radiuses, we are solving the linear systems

$$\begin{aligned} A\mathbf{x} &= \mathbf{b} \\ \hat{A}\hat{\mathbf{x}} &= \hat{\mathbf{b}} \end{aligned}$$

Ideally, we would like to quantify $\|\hat{\mathbf{x}} - \mathbf{x}\|_2$. The first step is to get an estimate of $\|\delta\mathbf{b}\|_2 = \|\hat{\mathbf{b}} - \mathbf{b}\|_2$ and $\|\delta A\|_2 = \|\hat{A} - A\|_2$. These quantities can be easily bounded. Recalling the definitions of the matrices and vectors involved

$$A = \begin{pmatrix} K & B \\ \tilde{B} & 0 \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} \mathbf{f} \\ \mathbf{d} \end{pmatrix}$$

with $\mathbf{d} = R_{12}\mathbf{r}_{\Gamma_2} - \mathbf{r}_{\Gamma_1}$. The blocks and their perturbations are given by

$$B = \begin{pmatrix} 0 \\ -M_1 \\ 0 \\ M_2R_{21} \end{pmatrix} \quad \delta B = \begin{pmatrix} 0 \\ 0 \\ 0 \\ M_2\delta R_{21} \end{pmatrix} \quad \tilde{B} = \begin{pmatrix} 0 & I & 0 & -R_{12} \end{pmatrix} \quad \delta\tilde{B} = \begin{pmatrix} 0 & 0 & 0 & -\delta R_{12} \end{pmatrix}$$

The perturbations on the coefficient matrix and right-hand side are given by

$$\delta A = \begin{pmatrix} 0 & \delta B \\ \delta\tilde{B} & 0 \end{pmatrix} \quad \delta\mathbf{b} = \begin{pmatrix} 0 \\ \delta R_{12}\mathbf{r}_{\Gamma_2} \end{pmatrix}$$

The next lemma bounds the magnitude of the perturbations.

Lemma 3.3

Let $\delta R_{12} = \hat{R}_{12} - R_{12}$, $\delta R_{21} = \hat{R}_{21} - R_{21}$ and let $\mathbf{r}_{\Gamma_2}$ be the vector of coordinates of the nodes on the interface of body 2. Then, the following results hold:

$$\begin{aligned} \|\delta\mathbf{b}\|_2 &= \|\delta R_{12}\mathbf{r}_{\Gamma_2}\|_2 \\ \|\delta A\|_2 &\leq \sqrt{\|M_2\delta R_{21}\|_2^2 + \|\delta R_{12}\|_2^2} \end{aligned}$$

Proof. The result on $\|\delta\mathbf{b}\|_2$ is immediate. For $\|\delta A\|_2$, we consider $\mathbf{x}^T = (\mathbf{u}^T, \mathbf{p}^T)$ and obtain

$$\begin{aligned} \|\delta A\|_2^2 = \max_{\|\mathbf{x}\|_2=1} \|\delta A\mathbf{x}\|_2^2 &= \max_{\|\mathbf{x}\|_2=1} (\|\delta B\mathbf{u}\|_2^2 + \|\delta\tilde{B}\mathbf{p}\|_2^2) \\ &\leq \max_{\|\mathbf{u}\|_2=1} \|\delta B\mathbf{u}\|_2^2 + \max_{\|\mathbf{p}\|_2=1} \|\delta\tilde{B}\mathbf{p}\|_2^2 \\ &= \|\delta B\|_2^2 + \|\delta\tilde{B}\|_2^2 \\ &= \|M_2\delta R_{21}\|_2^2 + \|\delta R_{12}\|_2^2 \end{aligned}$$

Taking the square root concludes the proof. □

The first term in the upper bound on $\|\delta A\|_2$ is not expected to contribute much since $\|M_2\delta R_{21}\|_2^2 \leq \|M_2\|_2^2\|\delta R_{21}\|_2^2 = \lambda_1^2(M_2)\|\delta R_{21}\|_2^2$ and $\lambda_1(M_2)$ depends on the mesh size. However, the second term in more worrisome: numerical experiments indicated that $\|\delta R_{12}\|_2$ and $\|\delta R_{21}\|_2$ could become quite large if no constraints were tied to the matrices Φ_{11} and Φ_{22} and the bound on $\|\delta A\|_2$ was found to be very tight. Unfortunately, bounding $\|\hat{\mathbf{x}} - \mathbf{x}\|_2$ requires assumptions on the magnitude of the perturbation $\|\delta A\|_2$ which cannot be verified at this stage.

It must be recognized that all the results from this section do not depend on the properties of the interpolation matrices. In the next section, we will illustrate the discussion with numerical experiments and we will verify the tightness of the bounds.

3.5 Numerical experiments

For a given geometry, we have experimented with different sets of radiuses for the radial basis functions. We have constructed interpolation matrices which did not satisfy the sufficient conditions of Chapter 1. The purpose is to illustrate the influence of the choice of radius and verify the tightness of the bounds from the previous section. The experimentations will be carried out on a contact problem between a curved and a flat boundary. This problem is referred to in the literature as a Hertzian contact problem. We will encounter it again in future chapters. The experiments will be repeated for uniform and localized radiuses. We will solve the same problem for different mesh sizes. However, the radiuses of support of the radial basis functions have been chosen independently of the mesh size. The supports of the radial basis functions are shown in Figure 3.1 for a mesh size $h = 0.01$. Note that in order to preserve the diagonal dominance of the interpolation matrices, the radiuses must depend on the mesh size. This condition is violated in the experiments and diagonal dominance is certainly lost on fine meshes. Nevertheless, we did not encounter any issues related to the invertibility of the interpolation matrices nor related to the rescaling. Suitable boundary conditions were chosen such that the problem was solvable.

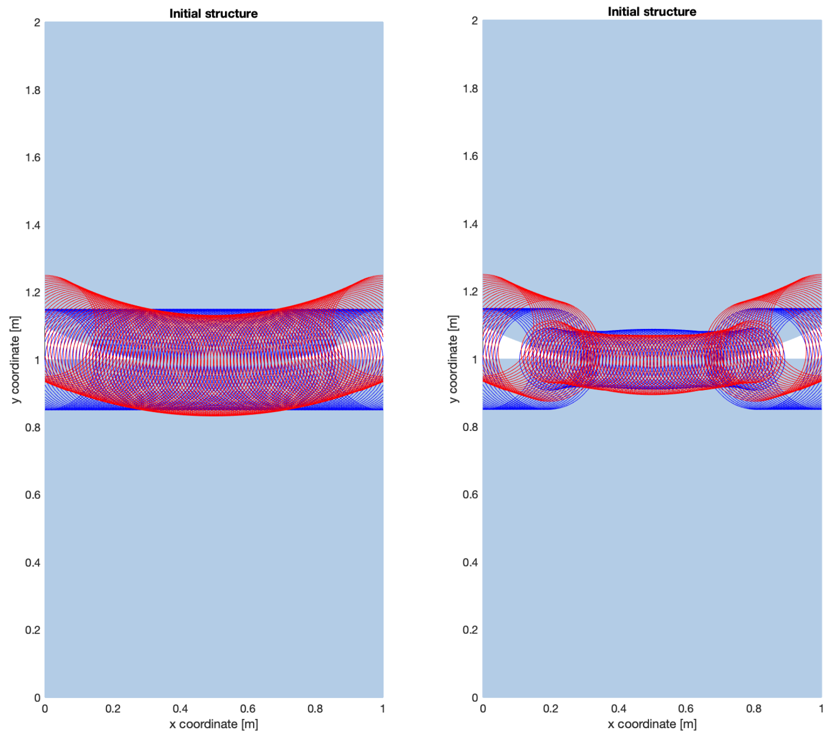

Figure 3.1: Support of the radial basis functions for a uniform radius (left figure) and a localized radius (right figure)

In Figure 3.2 and 3.3, the spectral norm of the INTERNODES matrix is reported for uniform and localized radiuses respectively. The bounds reported are those of Theorem 3.4. The sharpness of the bounds is quite stunning. In fact, when using a uniform radius, the lower bound was always attained. We immediately notice a sharp increase of the spectral norm on fine meshes when using a localized radius. This increase is necessarily linked to an increase of the spectral norm of either B or $\tilde{B}$. However, according to our previous discussion, an increase of the spectral norm of $\tilde{B}$ is more likely. The next figures provide a verification.

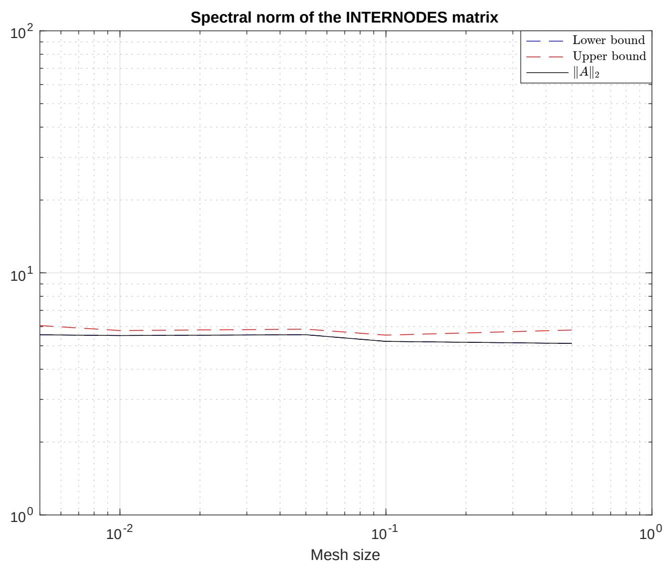

Figure 3.2: Spectral norm of the INTERNODES matrix with uniform radiuses

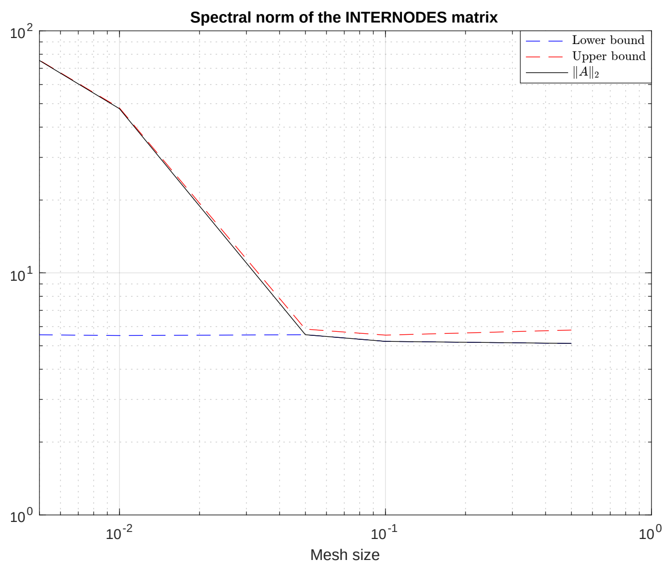

Figure 3.3: Spectral norm of the INTERNODES matrix with localized radiuses

The smallest and largest singular value of B are shown in Figure 3.4 and 3.5 for uniform and localized radiuses respectively. The bounds reported are those from Lemma 3.1. The smallest singular value seems to decay as the smallest eigenvalue of the interface mass matrix of body 1. This decay is linear in the mesh size as predicted by the theory. The largest singular value also behaves nicely for the case with uniform radiuses. However, it increases slightly for localized radiuses. The non-monotone pattern in this case seems to indicate its growth is being damped. It is most likely thanks to the eigenvalues of the interface mass matrices.

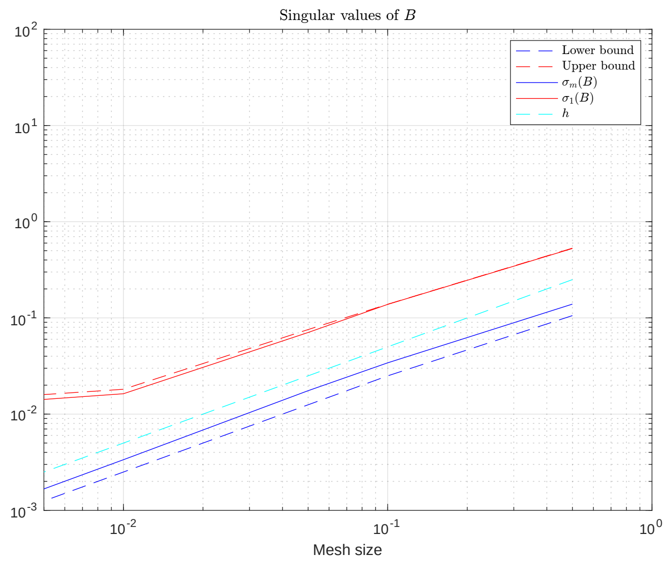

Figure 3.4: Singular values of B with uniform radiuses

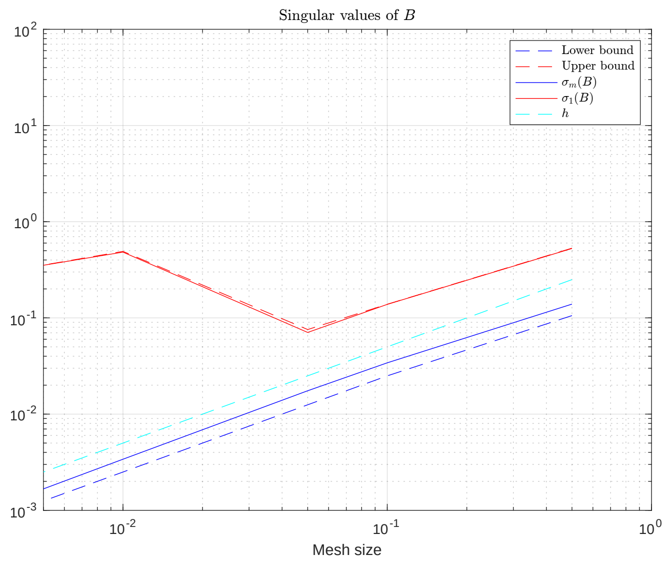

Figure 3.5: Singular values of B with localized radiuses

We now finally check the smallest and largest singular value of $\tilde{B}$. They have been reported in Figure 3.6 and 3.7 for uniform and localized radiuses respectively along with the bounds from Lemma 3.2. In this case, the interface mass matrices are not here to prevent a growth of the singular values. The smallest singular value of $\tilde{B}$ is not reported for fine meshes because of convergences issues with the solver. However, it seems to get increasingly close to 1. Although the largest singular value still behaves nicely for uniform radiuses, it increases tremendously for localized radiuses.

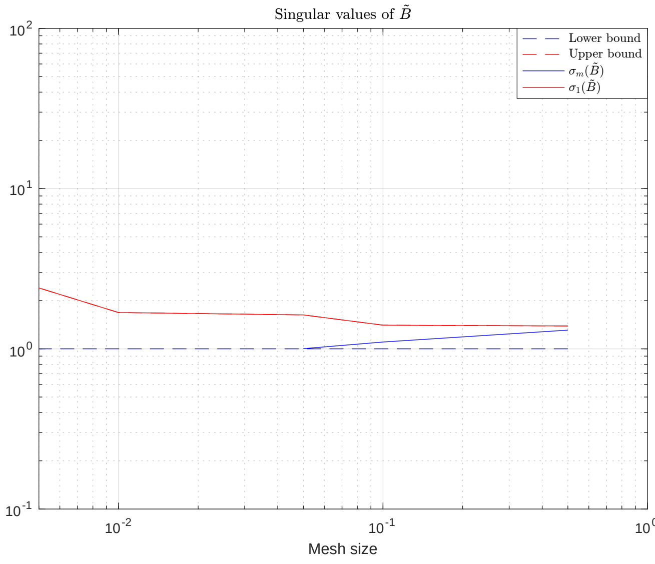

Figure 3.6: Singular values of $\tilde{B}$ with uniform radiuses

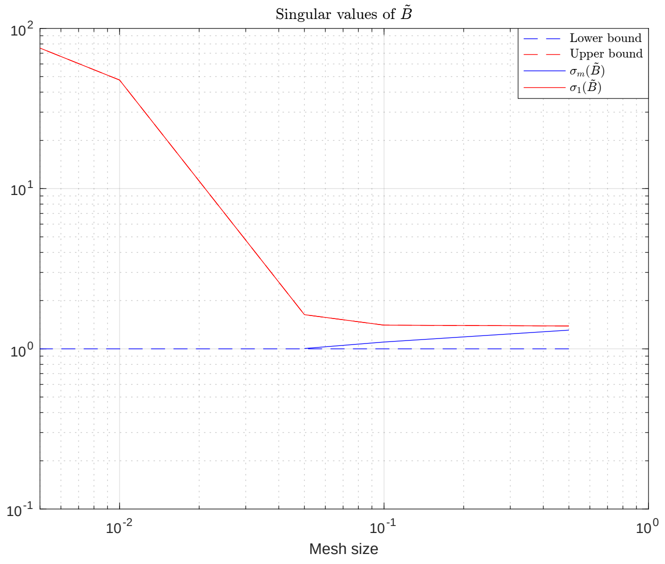

Figure 3.7: Singular values of $\tilde{B}$ with localized radiuses

We have also computed the smallest singular value of the INTERNODES matrix for these same experiments although we do not have any theoretical bounds to compare it with. The results are shown in Figure 3.8 and 3.9 for uniform and localized radiuses respectively. The smallest eigenvalue of the stiffness matrix, known to decay as $O(h^2)$ for 2D problems from Theorem 3.3, is also shown for comparison. From these results, it seems that the smallest singular value of the INTERNODES matrix is not too different from the smallest eigenvalue of the stiffness matrix provided the interpolation is accurate enough. However, it may decrease much faster for low quality interpolation. Therefore, the smallest singular value of the INTERNODES matrix must be some complicated function of the eigenvalues of K and singular values of B and $\tilde{B}$. Finding tight lower and upper bounds on this quantity is still ongoing research.

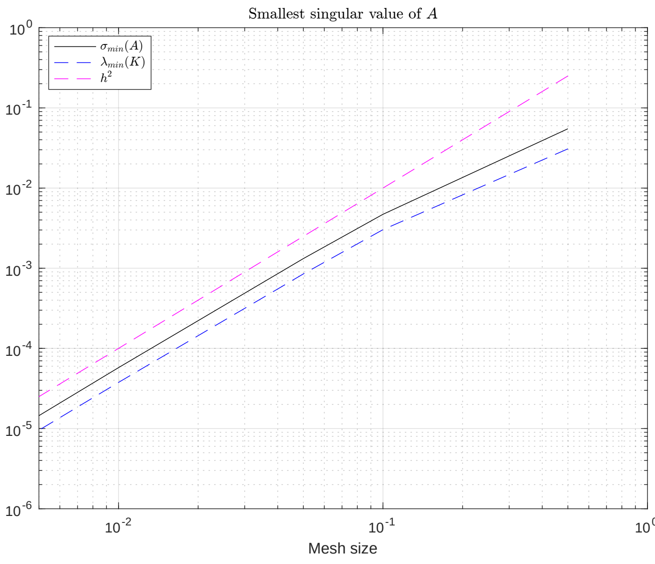

Figure 3.8: Smallest singular value of the INTERNODES matrix with uniform radiuses

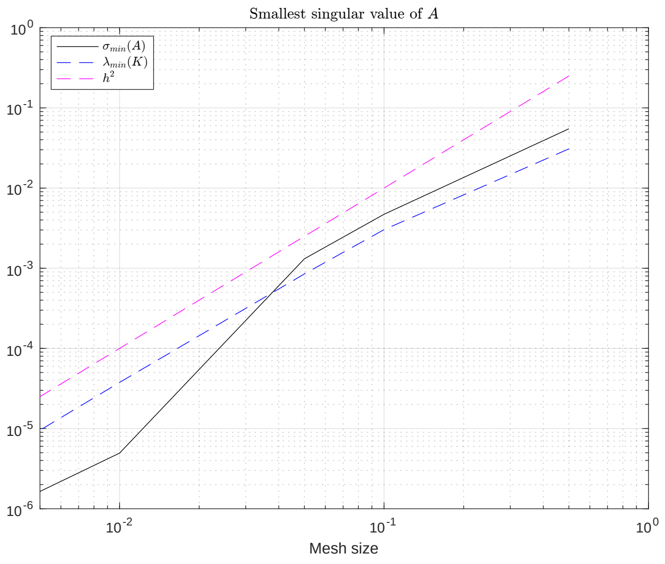

Figure 3.9: Smallest singular value of the INTERNODES matrix with localized radiuses

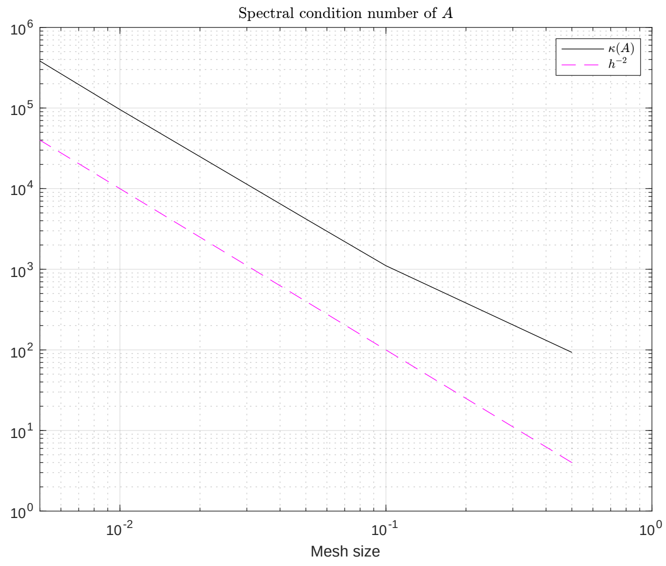

Figure 3.10: Spectral condition number of the INTERNODES matrix with uniform radiuses

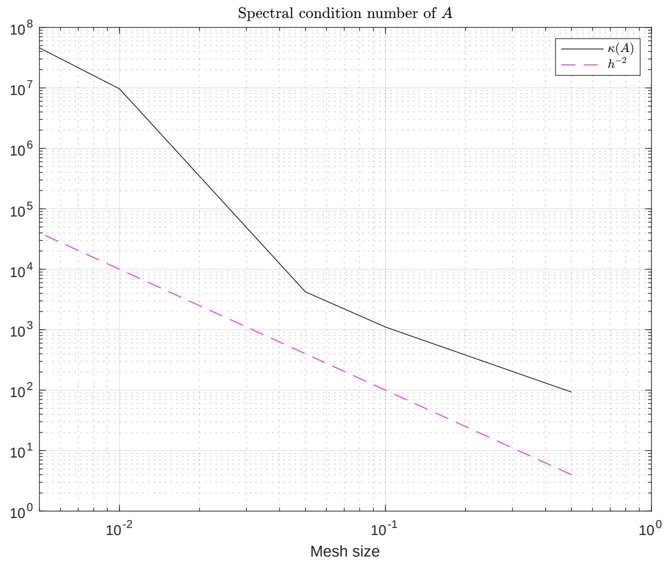

Figure 3.11: Spectral condition number of the INTERNODES matrix with localized radiuses

Finally, we conclude these experiments with a verification of the spectral condition number of the INTERNODES matrix. The results are reported in Figure 3.10 and 3.11 for uniform and localized radiuses respectively. For uniform radiuses, the results are in good agreement with those from Günther-Hanssen. A typical $O(h^{-2})$ growth of the condition number is recovered. However, the situation is much worse for low quality interpolation. For the smallest mesh size, the condition number for localized radiuses has gained two orders of magnitude with respect to uniform radiuses. From our previous results, it seems both $\|A\|_2$ and $\|A^{-1}\|_2$ are contributing to this increase.

Based on this finding, one could reasonably question whether it is worthwhile keeping the interface mass matrix of body 1 in the matrix $\tilde{B}$ as it originally appeared in the coefficient matrix. Our experiments indeed indicated it had a favorable effect on the singular values of $\tilde{B}$ and also on the spectral norm of the INTERNODES matrix. However, it surprisingly did not improve the condition number of the matrix because the modification seemed to have increased the spectral norm of the inverse instead.

The stability results were not tested for this problem. However, they were verified for a different geometry. The tests confirmed the tightness of the bounds and the potential instability of the method in case of large perturbations to the interpolation matrices. It is unclear whether the diagonal dominance of the interpolation matrices improves the stability of the method. However, the requirement surely restricts the choice of radius for the radial basis functions which might indirectly bound δR_{12} and δR_{21}. Unfortunately, the precise connection between stability and diagonal dominance could not be established and is left as future work.

Now that we know many of the properties of the blocks of the INTERNODES matrix, we can investigate how to solve large scale linear systems with this matrix.

Chapter 4

Numerical solution techniques

This chapter aims at providing the background for the discussion of the next chapter on preconditioners. We will begin by motivating the choice for the solution method to the linear system. Naturally, the emphasis will be made on iterative methods. However, direct methods will be also briefly discussed as they are used to solve numerous smaller linear systems.

4.1 Motivation

There exists a great number of ways of solving a linear system including both direct and iterative methods. The choice, of course, depends on the matrix at hand, its properties and the size of the problem. Obviously, iterative methods only make sense if they converge relatively quickly. When applied to an $n \times n$ dense matrix, an iterative method based on matrix-vector multiplications is expected to cost $O(n^2)$ operations per iteration. It is therefore only worthwhile considering when it can converge after say k iterations with $k < n$. The reason being most direct methods based on matrix factorizations will cost $O(n^3)$ or slightly less.

In the context of matrices arising from finite difference or finite element discretizations of partial differential equations (PDEs), the matrices are typically sparse and therefore iterative solvers will only cost $O(kn)$ per iteration with $k < n$. However, convergence rates of iterative solvers depend on the properties of the matrix and the method itself. Unfortunately, convergence may be extremely slow and these methods are then much less attractive. This problem can be circumvented by effective preconditioning techniques as we will see in the next chapter.

Direct solvers, on the other hand, do not have this issue. Their essential cost component is usually linked to computing a matrix factorization. Solving the linear system is subsequently an easier task.

Apart from the cost related to floating point operations is the cost of storage. Unfortunately, when matrix factorizations are computed on sparse matrices, the sparsity pattern of the factors is generally destroyed. This is particularly problematic for large sparse matrices because the factors generated from their decomposition may not fit in memory. However, it has been shown that this phenomenon, called fill-in, could be greatly reduced by reordering the rows and columns of the matrix. Unfortunately, computing the optimal reordering turns out to be NP-complete (Yannakakis, 1981). Obviously, it would not make any sense to solve such a problem when initially our wish is just to compute the solution of a linear system. For this reason, a number of heuristic methods have been developed to find good but not optimal reorderings. Some examples for heuristic strategies are Cuthill-McKee and reverse Cuthill-McKee. Others are theoretically motivated such as Minimum Degree ordering or its cheaper counterpart called Approximate Minimum Degree ordering. A very popular one for which there are in fact optimality results is the Nested Dissection method. The last method is particularly effective for matrices with an underlying graph structure inherited from a finite difference grid or a finite element mesh. Overall, sparse direct methods based on reordering strategies have matured to a point that they nowadays represent the method of choice for solving large sparse linear systems, at least for discretizations of two dimensional problems. These strategies will not be discussed in this report and we instead refer to Davis (2006) for all the theoretical and implementation details.

In the previous chapter, we saw that the linear system we are trying to solve is indeed sparse to a great extent.

However, our problem is not just solving a linear system, it is solving the entire contact problem. Let us recall that the blocks B and $\tilde{B}$ of our linear system are changing between each iteration of the contact algorithm. One may come up with different ideas. For example, we could reassemble the entire system at each iteration. However, such an operation may be quite costly. An even worse idea would be to change only the lines and columns of the matrix which have changed from one iteration to the next. Updating the entries of a sparse matrix has a prohibitive cost due to the shifting of indices in the vectors storing the position of nonzero entries.

Apart from updating the matrix is the issue related to solving the linear system. The use of a direct solver would require recomputing a factorization at each iteration of the contact algorithm. Based on our previous discussion, these operations are expected to be very costly. Finally, a decisive observation is that the matrices B and $\tilde{B}$ contain a few dense blocks. For very large problems, such blocks simply cannot be stored. Therefore, forming the INTERNODES matrix itself becomes infeasible, left alone its factorization. It is precisely this last point which leads us to consider iterative methods. Iterative methods allow us to never assemble the INTERNODES matrix, nor the matrices B and $\tilde{B}$. Instead, only the multiplication between the matrices B and $\tilde{B}$ and a given vector must be coded up. Hence, we define the abstract linear operators $\mathcal{B}\colon \mathbb{R}^m \to \mathbb{R}^n$ and $\tilde{\mathcal{B}}\colon \mathbb{R}^n \to \mathbb{R}^m$. The algorithm for these operators will be described in Section 4.4.

We now still have to decide which iterative method to use. Although there are many iterative methods available, the choice is here restricted to methods for general nonsymmetric matrices. Popular choices include the generalized minimal residual method (GMRES) (Saad and Schultz, 1986), the full orthogonalization method (FOM) (Saad, 1981) or the biconjugate gradient stabilized method (BiCGSTAB) (Van der Vorst, 1992). These methods all belong to the class of Krylov subspace methods which we briefly describe next.

4.2 Krylov subspace methods

For any matrix $A \in \mathbb{C}^{n\times n}$ and vector $\mathbf{r} \in \mathbb{C}^n$, Krylov subspaces are defined as

$$\mathcal{K}_k(A, \mathbf{r}) = \operatorname{span}\{\mathbf{r}, A\mathbf{r}, \ldots, A^{k-1}\mathbf{r}\} \quad k \geq 1$$

Note that any vector $\mathbf{v} \in \mathcal{K}_k(A, \mathbf{r})$ can be written as $\mathbf{v} = q(A)\mathbf{r}$ with $q \in \mathbb{P}_{k-1}$. Let us now consider a general linear system $A\mathbf{x} = \mathbf{b}$ with $A \in \mathbb{C}^{n\times n}$ and $\mathbf{b} \in \mathbb{C}^n$.

Given an initial guess $\mathbf{x}_0$ and defining the initial residual $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$, Krylov subspace methods look for an approximate solution $\mathbf{x}_k$ such that $\mathbf{x}_k - \mathbf{x}_0 \in \mathcal{K}_k(A, \mathbf{r}_0)$. Thus,

$$\mathbf{x}_k = \mathbf{x}_0 + q(A)\mathbf{r}_0$$

Consequently,

$$\begin{aligned}\mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k &= \mathbf{b} - A\mathbf{x}_0 - Aq(A)\mathbf{r}_0 \\ &= (I - Aq(A))\mathbf{r}_0 \\ &= p(A)\mathbf{r}_0\end{aligned}$$

with $p \in \mathbb{P}_k$ and $p(0) = 1$.

The criterion based on which the approximate solution $\mathbf{x}_k$ is computed and some orthogonality relations between subspaces distinguish the various methods. For example, in GMRES, $\mathbf{x}_k$ is computed such that is minimizes $\|\mathbf{r}_k\|_2$ which explains the name generalized minimal residual method. In FOM, the orthogonality relation $\mathbf{r}_k \perp \mathcal{K}_k(A, \mathbf{r}_0)$ is enforced whereas in BiCGSTAB, two sequences of vectors satisfying biorthogonality and biconjugacy relations respectively are computed. There exists many other methods apart from the three listed above. Before choosing a method, let us recall a few facts.

First of all, since our discussion will eventually turn to preconditioners, the choice of method is not a primal concern. It is frequently acknowledged in the literature that the quality of the preconditioner is more important than the choice of the Krylov subspace method (Rozložník, 2018). Our choice was primarily guided by popularity and numerical experimentation.

The convergence properties of GMRES are among the most studied and various convergence bounds have been found. Moreover, in a preliminary study, Günther-Hanssen solved linear systems involving the INTERNODES matrix using two different Krylov subspace methods. Namely, GMRES and the conjugate gradient squared (CGS) method. BiCGSTAB is an improved version of CGS which aims at stabilizing its erratic convergence. The results of Günther-Hanssen indicated poorer convergence (and occasional failure) of CGS. We therefore decided to carry on with GMRES and did not experiment with BiCGSTAB. However, the improved convergence properties of GMRES have a price. Since this method was used throughout the project, the next section provides a more detailed description of it.

4.3 GMRES

The standard GMRES algorithm is presented in Algorithm 4.1 and is taken from the lecture notes of Kressner (2020). The preconditioned version of GMRES will be presented in the next chapter. For the exposition, we consider the most general case where $A \in \mathbb{C}^{n\times n}$, but obviously in our application $A \in \mathbb{R}^{n\times n}$.

Algorithm 4.1 GMRES

```
1: **Input**: Matrix $A \in \mathbb{C}^{n\times n}$, right-hand side $\mathbf{b} \in \mathbb{C}^n$, starting vector $\mathbf{x}_0$, $k \in \mathbb{N}$
2: **Output**: Approximate solution $\mathbf{x}_k$ to $A\mathbf{x} = \mathbf{b}$
3: Set $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$
4: Set $\beta_0 = \|\mathbf{r}_0\|_2$, $\mathbf{u}_1 = \mathbf{r}_0/\beta_0$, $U_1 = \mathbf{u}_1$
5: for $j = 1, 2, \ldots, k$ do
6:     Compute $\mathbf{w}_j = A\mathbf{u}_j$
7:     Compute $\mathbf{h}_j = U_j^* \mathbf{w}_j$
8:     Compute $\tilde{\mathbf{u}}_{j+1} = \mathbf{w}_j - U_j \mathbf{h}_j$
9:     Set $h_{j+1,j} = \|\tilde{\mathbf{u}}_{j+1}\|_2$
10:    Set $\mathbf{u}_{j+1} = \tilde{\mathbf{u}}_{j+1}/h_{j+1,j}$
11:    Set $U_{j+1} = (U_j, \mathbf{u}_{j+1})$
12:    Find $\mathbf{y}_j = \text{argmin}_{\mathbf{y}\in\mathbb{R}^j} \|\beta_0\mathbf{e}_1 - \tilde{H}_j\mathbf{y}\|_2$
13:    Set $\beta_j = \|\beta_0\mathbf{e}_1 - \tilde{H}_j\mathbf{y}_j\|_2$
14: end for
15: Return $\mathbf{x}_k = \mathbf{x}_0 + U_k\mathbf{y}_k$
```

The algorithm consists in building an orthonormal basis U_k for the Krylov subspace

$$\mathcal{K}_k(A, \mathbf{r}_0) = \text{span}\{\mathbf{r}_0, A\mathbf{r}_0, \ldots, A^{k-1}\mathbf{r}_0\}$$

Without any breakdown, the method increases the dimension of the Krylov subspace by one at each iteration. This is done by projecting the last vector of the basis in the orthogonal complement of $\mathcal{K}_k(A, \mathbf{r}_0)$, normalizing it and adding it to the basis. In order to compute $\mathbf{x}_k$, the approximate solution at step k, a linear least squares problem involving the matrix $\tilde{H}_k$ must be solved. This approximate solution is the vector which minimizes the 2-norm of the residual. In practice, the iterations of GMRES are stopped when the norm of this residual vector is small enough. There are two standard stopping criteria used in GMRES:

1. A criterion based on the relative residual:
$$\frac{\|\mathbf{r}_k\|_2}{\|\mathbf{r}_0\|_2} \leq tol$$

2. A criterion based on the absolute residual:
$$\|\mathbf{r}_k\|_2 \leq tol$$

where *tol* is some prescribed tolerance. The l^2 norm is the natural norm in GMRES. However, any norm could be used in the stopping criterion. In fact, it might be worthwhile using the l^∞ norm because it does not depend on the size of the vector. For example, assuming each component of $\mathbf{r}_k \in \mathbb{R}^n$ is equal to $c << 1$, then $\|\mathbf{r}_k\|_2 = \sqrt{n}|c|$. For $n = 10^6$ it might be quite pessimistic. Whereas $\|\mathbf{r}_k\|_\infty = |c|$ is independent of the size of the problem.

Another important remark is linked to the cost of GMRES. The essential cost component of GMRES is concentrated in lines 6, 7 and 8. Line 6 computes a matrix-vector multiplication which is expected to cost $O(cn)$ with $c < n$ for a sparse matrix. Lines 7 and 8 orthogonalize the resulting vector against the Krylov basis computed up to this stage. Both lines cost $O(jn)$ each. Therefore, the cost of these operations grows with the dimension of the Krylov basis. Eventually, depending on how large c is, these operation might end up being more expensive than the matrix-vector multiplication. The storage cost also increases at each iteration because of the growth of the Krylov basis. Occasional reorthogonalization is also needed to prevent loss of orthogonality of the vectors of the Krylov basis.

The increasing computational burden can be cut down thanks to restarting. It simply consists in computing the approximate solution $\mathbf{x}_k$ for a prescribed k and restarting GMRES using as initial guess the previously computed approximation. However, restarting GMRES may severely slow down convergence (Wathen, 2015). Thus, GMRES is very efficient only if it is guaranteed to convergence after a small number of iterations. Preconditioning techniques are designed to meet this requirement. Before proceeding any further, we must recall a well known convergence bound.

Theorem 4.1

Assume $A \in \mathbb{C}^{n\times n}$ is a diagonalizable matrix such that $A = XDX^{-1}$ is a spectral decomposition for A. Let $\mathbf{r}_k$ be the residual obtained after applying k iterations of GMRES with starting vector $\mathbf{x}_0$. Then,

$$\frac{\|\mathbf{r}_k\|_2}{\|\mathbf{r}_0\|_2} \leq \kappa(X) \min_{\substack{p\in\mathbb{P}_k\\ p(0)=1}} \max_{\lambda\in\Lambda(A)} |p(\lambda)|$$

Proof. We have already noted that $\mathbf{r}_k = p(A)\mathbf{r}_0$ with $p \in \mathbb{P}_k$ and $p(0) = 1$ holds for any Krylov subspace method. GMRES minimizes the norm of this residual over all such polynomials. Thus,

$$\|\mathbf{r}_k\|_2 = \min_{\substack{p\in\mathbb{P}_k\\ p(0)=1}} \|p(A)\mathbf{r}_0\|_2 = \min_{\substack{p\in\mathbb{P}_k\\ p(0)=1}} \|Xp(D)X^{-1}\mathbf{r}_0\|_2$$

Bounding its norm using submultiplicativity, we obtain straightforwardly

$$\begin{aligned}\|\mathbf{r}_k\|_2 &\leq \min_{\substack{p\in\mathbb{P}_k\\ p(0)=1}} \|X\|_2\|X^{-1}\|_2\|p(D)\|_2\|\mathbf{r}_0\|_2 \\ &= \kappa(X) \min_{\substack{p\in\mathbb{P}_k\\ p(0)=1}} \max_{\lambda\in\Lambda(A)} |p(\lambda)|\|\mathbf{r}_0\|_2\end{aligned}$$

□

Note that if A is normal and its eigenvectors are normalized such that X forms an orthonormal basis, then $\kappa(X) = 1$. In this case, the convergence of GMRES is entirely controlled by its spectrum. In particular, in the Hermitian case, the convergence bound of MINRES (minimal residual method) is recovered. GMRES was designed as a generalization of MINRES for general non-Hermitian matrices. Theorem 4.1 is a clear motivation for preconditioning. The polynomial term is small if at least one of the following conditions is met:

1. The polynomial degree is large.
2. The eigenvalues are tightly clustered at a few specific locations away from the origin.

The first condition is not interesting as it would require running GMRES for too many iterations. The second condition can be achieved by a suitable transformation of the linear system. It is precisely one of the main purposes of preconditioning and we will return to it in the next chapter.

However, in the nonnormal case, the situation is far more troublesome. In general, not much is known about $\kappa(X)$ and the spectrum no longer controls the convergence of GMRES entirely. In fact, it may have no influence at all. It has been shown that any non-increasing curve represents the residual norm as a function of the iteration number for some problem. The eigenvalues for that problem are arbitrary (Greenbaum et al., 1996). Thus, even a very tight clustering of the eigenvalues would not improve the convergence. Fortunately, for matrices arising in real life applications the situation is less dramatic and we can expect the eigenvalues to have an influence. There exists other convergence bounds for GMRES which can be found in Greenbaum (1997) for instance. However, none of them are descriptive in the sense that they depend on unknown quantities or quantities which cannot be bounded. General descriptive convergence bounds for GMRES are still a research topic.

We end this section by noting that there exists variants of the implementation of GMRES. Algorithm 4.1 uses the Arnoldi method which is a modified Gram-Schmidt process. Alternatively, Householder transformations have also been used.

4.4 Fast matrix-vector multiplication

Applying GMRES to solve our linear system requires computing matrix-vector multiplications with the INTERNODES matrix

$$A = \begin{pmatrix} K & B \\ \tilde{B} & 0 \end{pmatrix}$$

Denoting $\mathbf{y} = A\mathbf{x}$ we must compute

$$\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix} = \begin{pmatrix} K\mathbf{x}_1 + B\mathbf{x}_2 \\ \tilde{B}\mathbf{x}_1 \end{pmatrix}$$

Computing $K\mathbf{x}_1$ can be done using sparse matrix-vector multiplication. However, computing $B\mathbf{x}_2$ and $\tilde{B}\mathbf{x}_1$ deserves some attention. Among the blocks of the matrices B and $\tilde{B}$ are small interface mass matrices and interpolation matrices and we will naturally take advantage of the Kronecker product structure. We recall that they can be expressed as

$$M_1 = (\bar{M}_1 \otimes I_d) \qquad R_{12} = (\bar{R}_{12} \otimes I_d)$$
$$M_2 = (\bar{M}_2 \otimes I_d) \qquad R_{21} = (\bar{R}_{21} \otimes I_d)$$

Taking advantage of this structure allows to never store the full matrices. Denoting n_1 and n_2 the number of nodes on the interface of body 1 and 2 respectively, then $\bar{M}_1 \in \mathbb{R}^{n_1 \times n_1}$, $\bar{M}_2 \in \mathbb{R}^{n_2 \times n_2}$. As for the interpolation matrices, $\bar{R}_{12} \in \mathbb{R}^{n_1 \times n_2}$ and $\bar{R}_{21} \in \mathbb{R}^{n_2 \times n_1}$.

4.4.1 Matrix-vector multiplication with B

Given $\mathbf{u} \in \mathbb{R}^m$, we must compute

$$B\mathbf{u} = \begin{pmatrix} 0 \\ -M_1\mathbf{u} \\ 0 \\ M_2 R_{21}\mathbf{u} \end{pmatrix}$$

In the following, we will often use the identity

$$\operatorname{vec}(AB^T) = (B \otimes I_m)\operatorname{vec}(A)$$

for matrices $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{p \times n}$ and where vec denotes the vectorization operator. Now, using the Kronecker product structure of the matrices and denoting $\mathbf{u} = \operatorname{vec}(U)$ with $U \in \mathbb{R}^{d \times n_1}$, we compute the products:

1. $M_1\mathbf{u} = (\bar{M}_1 \otimes I_d)\operatorname{vec}(U) = \operatorname{vec}(U\bar{M}_1^T) = \operatorname{vec}(U\bar{M}_1)$.
2. $M_2R_{21}\mathbf{u} = (\bar{M}_2 \otimes I_d)(\bar{R}_{21} \otimes I_d)\operatorname{vec}(U) = (\bar{M}_2\bar{R}_{21} \otimes I_d)\operatorname{vec}(U) = \operatorname{vec}(U\bar{R}_{21}^T\bar{M}_2)$

4.4.2 Matrix-vector multiplication with $\tilde{B}$

In this case, given $\mathbf{u} \in \mathbb{R}^n$, we compute

$$\tilde{B}\mathbf{u} = \begin{pmatrix} 0 & I & 0 & -R_{12} \end{pmatrix} \begin{pmatrix} \mathbf{u}_{\Omega_1} \\ \mathbf{u}_{\Gamma_1} \\ \mathbf{u}_{\Omega_2} \\ \mathbf{u}_{\Gamma_2} \end{pmatrix} = \mathbf{u}_{\Gamma_1} - R_{12}\mathbf{u}_{\Gamma_2}$$

The vectors $\mathbf{u}_{\Gamma_1}$ and $\mathbf{u}_{\Gamma_2}$ can be reshaped into the matrices $U_1 \in \mathbb{R}^{d\times n_1}$ and $U_2 \in \mathbb{R}^{d\times n_2}$ such that $\text{vec}(U_1) = \mathbf{u}_{\Gamma_1}$ and $\text{vec}(U_2) = \mathbf{u}_{\Gamma_2}$. Thus,

$$\tilde{B}\mathbf{u} = \text{vec}(U_1) - (\bar{R}_{12} \otimes I_d)\,\text{vec}(U_2) = \text{vec}(U_1 - U_2\bar{R}_{12}^T)$$

4.4.3 Algorithms

The linear operators avoid explicitly computing the interpolation matrices expressed as

$$\begin{aligned} R_{12} &= (\bar{R}_{12} \otimes I_d) = (\bar{D}_{11}^{-1}\bar{\Phi}_{12}\bar{\Phi}_{22}^{-1} \otimes I_d) \\ R_{21} &= (\bar{R}_{21} \otimes I_d) = (\bar{D}_{22}^{-1}\bar{\Phi}_{21}\bar{\Phi}_{11}^{-1} \otimes I_d) \end{aligned}$$

These matrices still have some sparsity for $d \geq 2$ thanks to the Kronecker product with the identity matrix. However, for extremely large problems, the explicit computation of the interpolation matrices would still lead to storage issues. Matrix-vector multiplies with R_{ij} will be most efficiently performed by pre-computing and storing the sparse LU factors of Φ_{jj}. Obviously, the diagonal matrix D_{ii} is never formed. Only the vector containing the rescaling factors $\mathbf{g}_i$ is needed. This vector is also pre-computed and stored.

In the following algorithms, we recall that $\mathcal{D}_\Gamma^1$ and $\mathcal{D}_\Gamma^2$ are the sets of degrees of freedom for the interface of body 1 and 2 respectively. Algorithms 4.2 and 4.3 refer to the linear operators $\mathcal{B}$, and $\tilde{\mathcal{B}}$ which implement the multiplications $B\mathbf{u}$ and $\tilde{B}\mathbf{u}$ respectively. Algorithm 4.4 implements the matrix-vector multiplication with $\bar{R}_{ij}$.

Algorithm 4.2 Linear operator $\mathcal{B}$

```
1: Input: Interface mass matrices $\bar{M}_1$, $\bar{M}_2$, interpolation matrix $\bar{R}_{21}$, sets $\mathcal{D}_\Gamma^1$, $\mathcal{D}_\Gamma^2$, vector $\mathbf{u}$
2: Output: Vector $\mathbf{y} = \mathcal{B}(\mathbf{u})$
3: Reshape $\mathbf{u}$ into a matrix $U$
4: Compute $Z_1 = -U\bar{M}_1$
5: Compute $Z_2 = U\bar{R}_{21}^T\bar{M}_2$
6: Initialize $\mathbf{y}$
7: Set $\mathbf{y}(\mathcal{D}_\Gamma^1) = \text{vec}(Z_1)$
8: Set $\mathbf{y}(\mathcal{D}_\Gamma^2) = \text{vec}(Z_2)$
9: Return $\mathbf{y}$
```

Algorithm 4.3 Linear operator $\tilde{\mathcal{B}}$

```
1: Input: Interpolation matrix $\bar{R}_{12}$, vector $\mathbf{u}$
2: Output: Vector $\mathbf{y} = \tilde{\mathcal{B}}(\mathbf{u})$
3: Extract $\mathbf{u}_{\Gamma_1}$ and reshape it as $U_1$
4: Extract $\mathbf{u}_{\Gamma_2}$ and reshape it as $U_2$
5: Compute $Z = U_1 - U_2\bar{R}_{12}^T$
6: Return $\mathbf{y} = \text{vec}(Z)$
```

Algorithm 4.4 Linear operator $\mathcal{R}_{ij}$

```
1: Input: $LU$ factors such that $\bar{\Phi}_{jj} = LU$, interpolation matrix $\bar{\Phi}_{ij}$, vector $\mathbf{g}_i$ of rescaling factors.
2: Output: Vector $\mathbf{y} = \mathcal{R}_{ij}(\mathbf{u})$
3: Compute $\mathbf{u} \leftarrow U^{-1}L^{-1}\mathbf{u}$
4: Compute $\mathbf{u} \leftarrow \bar{\Phi}_{ij}\mathbf{u}$
5: Compute element-wise division $\mathbf{u} \leftarrow \mathbf{u}/\mathbf{g}_i$
6: Return $\mathbf{u}$
```

Chapter 5

Preconditioning techniques

Experimentally, the iterative condition number of the INTERNODES matrix does not behave very differently from the one for the standard Poisson problem. The usual h^{-2} growth of the iterative condition number with the mesh size h was experienced even for very different applications (Deparis et al., 2016, Günther-Hanssen, 2020). This property depends on the differential operator and can be expected from second order unbounded operators. Although the iterative condition number alone cannot fully describe the behavior of iterative methods in the nonnormal case, we can still expect the number of iterations to increase with the condition number. As we have discussed in the previous chapter, such a feature is particularly undesirable when using GMRES because of its growing cost per iteration. Preconditioning techniques are used to keep the iteration count small despite the growing size of the problem. Such techniques are absolutely essential for large scale matrix computations. They will be at the center of attention in this chapter.

The remainder of the chapter is organized as follows: Section 5.1 provides a general introduction to preconditioning. Section 5.2 follows up with the preconditioned version of GMRES. In Section 5.3 a preconditioner is proposed and theoretical properties related to the eigenvalues of the preconditioned matrix are proved. The quality of the preconditioner depends on a parameter which must be chosen suitably. Sections 5.4 and 5.5 gather theoretical results for the tuning of the preconditioner. The solution to linear systems with this preconditioner is not trivial and therefore an algorithm is described in Section 5.6. All of our solution methods heavily rely on sparse approximate inverse techniques as building blocks for larger preconditioners. A brief introduction is included in Section 5.7. Finally, Section 5.9 concludes this chapter with numerical experiments. The quality of the preconditioner is tested on contact problems of increasingly large size.

5.1 Introduction

Preconditioning is a very active research area in mathematics. Although preconditioning techniques are mostly recognized in the context of linear systems, they may also be applied for other purposes such as the computation of eigenvalues. In this project, however, they will indeed be used for the solution of linear systems. An intuitive idea is the following: given a linear system $A\mathbf{x} = \mathbf{b}$ with $A \in \mathbb{C}^{n\times n}$ nonsingular and $\mathbf{b} \in \mathbb{C}^n$ which is hard to solve. The idea is to instead solve a transformed linear system $\hat{A}\hat{\mathbf{x}} = \hat{\mathbf{b}}$ which has much nicer properties in some sense and is easier to solve. The transformation may involve a change of variables from $\mathbf{x}$ to $\hat{\mathbf{x}}$.

To date, there does not exist a universal preconditioner nor a universal strategy for building efficient preconditioners. This has led to an impressive collection of methods. The choice greatly depends on whether one is ”given a problem” or ”given a matrix” (Wathen, 2015). In the ”given a problem” setting, one may advantageously exploit the origin of the linear system to design preconditioners. This is most relevant in the context of discretization of partial differential equations (PDEs) where the properties of the underlying differential operators from which the matrices stem may be used in the design process. Operator preconditioning is a direct application of these ideas. See for instance Hiptmair (2006) for a brief introduction.

On the other hand, when one is ”given a matrix”, the origin of the problem is unknown and therefore cannot be used. More general techniques such as incomplete factorizations and sparse inverses must then be applied.

These techniques can be applied in principle to any matrix but have found fertile grounds in particular in the context of optimization. This observation has led authors to classify preconditioners on the basis of applications. Nevertheless, the distinction is not always clear since some preconditioners may be successfully applied to very different problems. We refer to the excellent survey papers of Wathen (2015) and Pearson and Pestana (2020) for a general introduction to preconditioning and to Benzi (2002) for a focus on algebraic preconditioners. The lack of universal solution procedures and the prominent role intuition plays in the design of preconditioners have led several authors to characterize it as an art rather than a science. However, one should be confident in thinking that the more knowledge we have, the more of a science it gets.

Preconditioning linear systems with non-Hermititan (or more generally nonnormal) matrices is even more challenging. Indeed, as we have already discussed, the spectrum alone does not condition entirely the performance of Krylov subspace methods. To date, there does not exist any descriptive convergence bound for nonsymmetric Krylov solvers. Therefore, preconditioning strategies for nonnormal matrices are usually only heuristic. However, there exist rigorous results based on field of values analysis. Unfortunately, such an analysis is strongly problem dependent and may get quite technical. It was carried out for instance by Benzi and Olshanskii (2011) for the linearized Navier-Stokes equations.

There exists various forms of preconditioning commonly classified in three categories.

1. Left preconditioning simply consists in pre-multiplying the linear system with M^{-1} such that $M^{-1} \approx A^{-1}$ in some sense. M is called the preconditioning matrix (or preconditioner). We obtain

$$\begin{aligned} M^{-1}A\mathbf{x} &= M^{-1}\mathbf{b} \\ \hat{A}\mathbf{x} &= \hat{\mathbf{b}} \end{aligned}$$

with $\hat{A} = M^{-1}A$ and $\hat{\mathbf{b}} = M^{-1}\mathbf{b}$. The residual becomes

$$\hat{\mathbf{r}} = \hat{\mathbf{b}} - \hat{A}\mathbf{x} = M^{-1}(\mathbf{b} - A\mathbf{x}) = M^{-1}\mathbf{r}$$

Thus, left preconditioning also implies a transformation of the residual vector. This transformed residual vector is called the preconditioned residual.

2. Right preconditioning, instead, inserts the preconditioner between the coefficient matrix and the solution vector. Therefore, it implies a change of variables as

$$\begin{aligned} AM^{-1}M\mathbf{x} &= \mathbf{b} \\ \hat{A}\hat{\mathbf{x}} &= \mathbf{b} \end{aligned}$$

with $\hat{A} = AM^{-1}$ and $\hat{\mathbf{x}} = M\mathbf{x}$. The residual becomes

$$\hat{\mathbf{r}} = \mathbf{b} - \hat{A}\hat{\mathbf{x}} = \mathbf{b} - A\mathbf{x} = \mathbf{r}$$

In this case, the preconditioned residual and the true (unpreconditioned) residual are the same.

3. Split preconditioning (also called two-sided preconditioning) combines the ideas of left and right preconditioning. The transformation is

$$\begin{aligned} M_1^{-1}AM_2^{-1}M_2\mathbf{x} &= M_1^{-1}\mathbf{b} \\ \hat{A}\hat{\mathbf{x}} &= \hat{\mathbf{b}} \end{aligned}$$

The preconditioned residual is expressed as $\hat{\mathbf{r}} = M_1^{-1}\mathbf{r}$.

If is important to note that the left and right preconditioned matrices are similar: $M^{-1}A = SAM^{-1}S^{-1}$ with $S = M^{-1}$. Thus, they share the same spectrum. It is also true for split preconditioning provided $M = M_1M_2$. Therefore, the form of preconditioning is not important for the spectrum of the preconditioned matrix. However, other aspects are also relevant. For symmetric matrices, for example, split preconditioning is usually preferred because it leads to a symmetric preconditioned matrix provided $M_2 = M_1^T$. There exists also other forms of preconditioning such as polynomial preconditioning. See for example Liu et al. (2015) and their references.

5.2 Preconditioned GMRES

In Algorithm 5.1, the left preconditioned version of GMRES is presented. In this version, the preconditioned residual is used. This residual may not be a good measure for the convergence to the solution of the linear system. It is sometimes recommended to compute the unpreconditioned residual $\mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k$. However, this requires the explicit computation of $\mathbf{x}_k$ at each iteration and an additional matrix-vector multiplication which in turn increases the computational cost.

Right preconditioning instead allows to work directly with the unpreconditioned residual. For this reason, it is usually preferred in applications. The right preconditioned version of GMRES is presented in Algorithm 5.2. It is quite similar to the left preconditioned version. The main difference is that instead of solving a linear system with the preconditioner for the initial residual, it must be solved when computing the approximate solution. It corresponds to back-transforming the solution to the original variable. Other implementation details can be found in Saad (2003).

A consequence of preconditioning, both left and right, is that a linear system involving the preconditioning matrix must be solved at each iteration in line 6. Obviously, solving these linear systems must be much easier than solving the original system. In fact, it is acknowledged that the cost of solving a linear system with the preconditioner should balance the cost of computing a matrix-vector multiplication with the coefficient matrix (Wathen, 2015).

Algorithm 5.1 Left preconditioned GMRES

```
 1: Input: Matrix $A \in \mathbb{C}^{n\times n}$, Preconditioning matrix $M \in \mathbb{C}^{n\times n}$, right-hand side $\mathbf{b} \in \mathbb{C}^n$, starting vector $\mathbf{x}_0$,
    $k \in \mathbb{N}$
 2: Output: Approximate solution $\mathbf{x}_k$ to $A\mathbf{x} = \mathbf{b}$
 3: Set $\mathbf{r}_0 = M^{-1}(\mathbf{b} - A\mathbf{x}_0)$
 4: Set $\beta_0 = \|\mathbf{r}_0\|_2$, $\mathbf{u}_1 = \mathbf{r}_0/\beta_0$, $U_1 = \mathbf{u}_1$
 5: for $j = 1, 2, \dots, k$ do
 6:     Compute $\mathbf{w}_j = M^{-1}A\mathbf{u}_j$
 7:     Compute $\mathbf{h}_j = U_j^*\mathbf{w}_j$
 8:     Compute $\tilde{\mathbf{u}}_{j+1} = \mathbf{w}_j - U_j\mathbf{h}_j$
 9:     Set $h_{j+1,j} = \|\tilde{\mathbf{u}}_{j+1}\|_2$
10:     Set $\mathbf{u}_{j+1} = \tilde{\mathbf{u}}_{j+1}/h_{j+1,j}$
11:     Set $U_{j+1} = (U_j, \mathbf{u}_{j+1})$
12:     Find $\mathbf{y}_j = \operatorname{argmin}_{\mathbf{y}\in\mathbb{R}^j} \|\beta_0\mathbf{e}_1 - \tilde{H}_j\mathbf{y}\|_2$
13:     Set $\beta_j = \|\beta_0\mathbf{e}_1 - \tilde{H}_j\mathbf{y}_j\|_2$
14: end for
15: Return $\mathbf{x}_k = \mathbf{x}_0 + U_k\mathbf{y}_k$
```

Algorithm 5.2 Right preconditioned GMRES

```
 1: Input: Matrix $A \in \mathbb{C}^{n\times n}$, Preconditioning matrix $M \in \mathbb{C}^{n\times n}$, right-hand side $\mathbf{b} \in \mathbb{C}^n$, starting vector $\mathbf{x}_0$,
    $k \in \mathbb{N}$
 2: Output: Approximate solution $\mathbf{x}_k$ to $A\mathbf{x} = \mathbf{b}$
 3: Set $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$
 4: Set $\beta_0 = \|\mathbf{r}_0\|_2$, $\mathbf{u}_1 = \mathbf{r}_0/\beta_0$, $U_1 = \mathbf{u}_1$
 5: for $j = 1, 2, \dots, k$ do
 6:     Compute $\mathbf{w}_j = AM^{-1}\mathbf{u}_j$
 7:     Compute $\mathbf{h}_j = U_j^*\mathbf{w}_j$
 8:     Compute $\tilde{\mathbf{u}}_{j+1} = \mathbf{w}_j - U_j\mathbf{h}_j$
 9:     Set $h_{j+1,j} = \|\tilde{\mathbf{u}}_{j+1}\|_2$
10:     Set $\mathbf{u}_{j+1} = \tilde{\mathbf{u}}_{j+1}/h_{j+1,j}$
11:     Set $U_{j+1} = (U_j, \mathbf{u}_{j+1})$
12:     Find $\mathbf{y}_j = \operatorname{argmin}_{\mathbf{y}\in\mathbb{R}^j} \|\beta_0\mathbf{e}_1 - \tilde{H}_j\mathbf{y}\|_2$
13:     Set $\beta_j = \|\beta_0\mathbf{e}_1 - \tilde{H}_j\mathbf{y}_j\|_2$
14: end for
15: Return $\mathbf{x}_k = \mathbf{x}_0 + M^{-1}U_k\mathbf{y}_k$
```

5.3 A preconditioner for the INTERNODES matrix

The first issue we are facing is linked to the very different physical nature of the unknowns in the solution vector $\mathbf{x}$ which combines the displacement vector $\mathbf{u}$ and the Lagrange multipliers $\boldsymbol{\lambda}$. These quantities are typically expressed in meters and Newtons per meter respectively. Numerically speaking, they are different by several orders of magnitude. This difference carries over to the different blocks of the INTERNODES matrix. The entries of the stiffness matrix are extremely large in comparison to those of B and $\tilde{B}$. This difference is responsible for much of the ill-conditioning of the matrix. We must therefore proceed with a rescaling. In effect, it will lead to a solution vector $\mathbf{x}$ with all entries having the same units. A simple and yet very efficient rescaling is given by

$$\begin{pmatrix} \zeta^{-1}K & B \\ \tilde{B} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \zeta^{-1}\boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \zeta^{-1}\mathbf{f} \\ \mathbf{d} \end{pmatrix}$$

It simply consists in dividing the entire first block row by a scaling parameter $\zeta > 0$ and performing a change of variables. We redefine $K \leftarrow \zeta^{-1}K$, $\boldsymbol{\lambda} \leftarrow \zeta^{-1}\boldsymbol{\lambda}$ and $\mathbf{f} \leftarrow \zeta^{-1}\mathbf{f}$. This change of variables is only possible because $\boldsymbol{\lambda}$ does not appear on the second block row. The scaling parameter ζ must be chosen depending on the material. The most prominent material parameter and which generates much of the ill-conditioning is the elastic modulus. Thus, we recommend setting $\zeta = E$ in case of a homogeneous material. In case of a nonhomogeneous material, we recommend setting $\zeta = \max_{\mathbf{x}\in\Omega} E(\mathbf{x})$.

The rescaling is in fact already a form of preconditioning. More specifically, it is a form of split preconditioning.

$$\underbrace{\begin{pmatrix} \zeta^{-1}I & 0 \\ 0 & I \end{pmatrix}}_{M_1^{-1}} \begin{pmatrix} K & B \\ \tilde{B} & 0 \end{pmatrix} \underbrace{\begin{pmatrix} I & 0 \\ 0 & \zeta I \end{pmatrix}}_{M_2^{-1}} \underbrace{\begin{pmatrix} I & 0 \\ 0 & \zeta^{-1}I \end{pmatrix}}_{M_2} \begin{pmatrix} \mathbf{u} \\ \boldsymbol{\lambda} \end{pmatrix} = \underbrace{\begin{pmatrix} \zeta^{-1}I & 0 \\ 0 & I \end{pmatrix}}_{M_1^{-1}} \begin{pmatrix} \mathbf{f} \\ \mathbf{d} \end{pmatrix}$$

More compactly written as

$$M_1^{-1}AM_2^{-1}M_2\mathbf{x} = M_1^{-1}\mathbf{b}$$
$$\hat{A}\hat{\mathbf{x}} = \hat{\mathbf{b}}$$

with $\hat{A} = M_1^{-1}AM_2^{-1}$ the preconditioned matrix, $\hat{\mathbf{x}} = M_2\mathbf{x}$ and $\hat{\mathbf{b}} = M_1^{-1}\mathbf{b}$.

However, we will consider it more as some preliminary step. In the following, we will refer to the INTERNODES matrix as being the rescaled matrix. Thus, we set $A = \hat{A}$. This rescaling has consequences for the residual vector. Indeed, the residual norm for the rescaled system behaves almost like a shift of the unpreconditioned residual norm. We have

$$\|\mathbf{r}\|_2^2 = \underbrace{\|\mathbf{f} - (K\mathbf{u} + B\boldsymbol{\lambda})\|_2^2}_{\|\mathbf{r}_1\|_2^2} + \underbrace{\|\mathbf{d} - \tilde{B}\mathbf{u}\|_2^2}_{\|\mathbf{r}_2\|_2^2}$$

$$\|\mathbf{r}_r\|_2^2 = \|M_1^{-1}\mathbf{r}\|_2^2 = \underbrace{\frac{1}{\zeta^2}\|\mathbf{f} - (K\mathbf{u} + B\boldsymbol{\lambda})\|_2^2}_{\|\mathbf{r}_{1,r}\|_2^2} + \underbrace{\|\mathbf{d} - \tilde{B}\mathbf{u}\|_2^2}_{\|\mathbf{r}_{2,r}\|_2^2}$$

with $\|\mathbf{r}_1\|_2 >> \|\mathbf{r}_2\|_2$ but $\|\mathbf{r}_{1,r}\|_2 \approx \|\mathbf{r}_{2,r}\|_2$. The components of the residual vector for the rescaled system have the same units. It can be understood as a form normalization designed to wipe out the effect of material parameters. It allows us to conveniently choose a tolerance in the stopping criterion of GMRES independent of material parameters.

The abundant literature on saddle point systems provides us with several possibilities for preconditioning the INTERNODES matrix. Let us recall the properties of the different blocks:

- $K \in \mathbb{R}^{n\times n}$ is symmetric positive semidefinite.
- $B \in \mathbb{R}^{n\times m}$ and $\tilde{B} \in \mathbb{R}^{m\times n}$ with $\mathrm{rank}(B) = \mathrm{rank}(\tilde{B}) = m$.

Special attention must be paid to the properties of the different blocks of the system. In our case, we are seeking a preconditioner for a saddle point system with a singular $(1,1)$ block. The preconditioner we propose is a slight modification of an augmented Lagrangian preconditioner proposed by Liu (2013) and based on earlier results from Cao (2008). As the name suggests, they are widely used in the context of optimization. Liu proposed a preconditioner of the form

$$M = \begin{pmatrix} K + BW^{-1}\tilde{B} & kB \\ 0 & -W \end{pmatrix}$$

where $k \in \mathbb{R}$ is a scalar parameter and $W \in \mathbb{R}^{m\times m}$ is a symmetric positive definite weight matrix. However, it is sufficient to only require it to be invertible. In his paper, Liu proved theoretical properties related to the clustering of the eigenvalues of the preconditioned system. We will present his derivations here. However, we shall consider a slightly more general preconditioner. It consists in adding scalar parameters $\alpha, \beta, \gamma \in \mathbb{R}^*$. The reason will become clear later. We therefore consider

$$M = \begin{pmatrix} K + \alpha BW^{-1}\tilde{B} & -\beta\gamma^{-1}B \\ 0 & -\gamma^{-1}W \end{pmatrix}$$

We will subsequently discuss suitable choices for the parameters α, β and γ as well as for the matrix W. The preconditioned system reads

$$M^{-1}A\mathbf{x} = M^{-1}\mathbf{b}$$

We are attempting to identify the eigenvalues of the preconditioned matrix $M^{-1}A$. Hence, let us consider the eigenvalue problem

$$M^{-1}A\mathbf{v} = \lambda\mathbf{v}$$

Thus, the eigenvalues of the preconditioned matrix are the eigenvalues of the generalized eigenvalue problem

$$A\mathbf{v} = \lambda M\mathbf{v}$$

More explicitly,

$$\begin{pmatrix} K & B \\ \tilde{B} & 0 \end{pmatrix}\begin{pmatrix} \mathbf{u} \\ \mathbf{p} \end{pmatrix} = \lambda \begin{pmatrix} K + \alpha BW^{-1}\tilde{B} & -\beta\gamma^{-1}B \\ 0 & -\gamma^{-1}W \end{pmatrix}\begin{pmatrix} \mathbf{u} \\ \mathbf{p} \end{pmatrix}$$

Hence

$$((\lambda-1)K + \alpha\lambda BW^{-1}\tilde{B})\mathbf{u} - (\lambda\beta\gamma^{-1}+1)B\mathbf{p} = \mathbf{0} \qquad (5.1)$$

$$\tilde{B}\mathbf{u} + \lambda\gamma^{-1}W\mathbf{p} = \mathbf{0} \qquad (5.2)$$

The second equation yields $\mathbf{p} = -\gamma\lambda^{-1}W^{-1}\tilde{B}\mathbf{u}$. (Obviously, $\lambda \neq 0$, otherwise the preconditioned matrix would be singular). After substituting back in the first equation and regrouping the terms we obtain

$$\lambda(\lambda-1)K\mathbf{u} + (\alpha\lambda^2 + \beta\lambda + \gamma)BW^{-1}\tilde{B}\mathbf{u} = \mathbf{0} \qquad (5.3)$$

Three different cases must be analyzed.

- If $\mathbf{u} \in \ker(\tilde{B})$, then equation (5.3) reduces to

$$\lambda(\lambda-1)K\mathbf{u} = \mathbf{0}$$

  Since $K\mathbf{u} \neq \mathbf{0}$, then $\lambda = 1$ is an eigenvalue of multiplicity $\dim\ker(\tilde{B}) = n - m$.

- If $\mathbf{u} \in \ker(K)$, equation (5.3) reduces to

$$(\alpha\lambda^2 + \beta\lambda + \gamma)BW^{-1}\tilde{B}\mathbf{u} = \mathbf{0}$$

  Since $BW^{-1}\tilde{B}\mathbf{u} \neq \mathbf{0}$, then

$$\alpha\lambda^2 + \beta\lambda + \gamma = 0$$

  which yields

$$\lambda_{1,2} = \frac{-\beta \pm \sqrt{\beta^2 - 4\alpha\gamma}}{2\alpha}$$

  These eigenvalues have multiplicity s each. We must still identify $n+m-(n-m+2s) = 2(m-s)$ eigenvalues. Note that if we set $\beta = -2\alpha$ and $\gamma = \alpha$, then we again obtain $\lambda_{1,2} = 1$ which is very favorable, as we already have a cluster of eigenvalues located at 1.

- If $\mathbf{u} \notin \ker(K)$ and $\mathbf{u} \notin \ker(\tilde{B})$, then both terms of equation (5.3) must be considered. After rearranging the terms, we obtain

  $$K\mathbf{u} = \frac{\alpha\lambda^2 + \beta\lambda + \gamma}{\lambda(1-\lambda)} BW^{-1}\tilde{B}\mathbf{u} = \mu BW^{-1}\tilde{B}\mathbf{u}$$

  having denoted $\mu = \frac{\alpha\lambda^2+\beta\lambda+\gamma}{\lambda(1-\lambda)}$. This is again a generalized eigenvalue problem. We can express λ as a function of μ since

  $$\bar{\lambda}_{1,2} = \frac{\mu - \beta \pm \sqrt{(\beta-\mu)^2 - 4\gamma(\alpha+\mu)}}{2(\alpha+\mu)} \tag{5.4}$$

  In fact, since we have already set $\beta = -2\alpha$ and $\gamma = \alpha$, the expression simplifies considerably as $\Delta = (\beta - \mu)^2 - 4\gamma(\alpha + \mu) = \mu^2$. Therefore, the remaining eigenvalues are

  $$\bar{\lambda}_1 = 1$$
  $$\bar{\lambda}_2 = \frac{\alpha}{\alpha + \mu}$$

  Clearly, $\lim_{\alpha\to\infty} \bar{\lambda}_2 = 1$. However, in practice, $|\alpha| >> |\mu|$ will already lead to an increasingly good eigenvalue clustering. Since the generalized eigenvalues μ are in general complex, $\bar{\lambda}_2$ will have a very small imaginary part.

In summary, provided α is chosen suitably, the eigenvalues of the preconditioned matrix are

- $\lambda = 1$ of multiplicity $n - m$
- $\lambda = 1$ of multiplicity $2s$
- $\lambda = 1$ of multiplicity $(m - s)$
- $\lambda \approx 1$ of multiplicity $(m - s)$

The only problem is identifying what μ may be such that α can be chosen suitably. We will discuss this problem in the next few sections.

5.4 Sign of the generalized eigenvalues μ

There are in fact only $2(m - s)$ non-trivial eigenvalues μ associated to eigenvectors belonging to the subspace $\mathcal{U} = \mathbb{C}^n \setminus \{\ker(K) \cup \ker(B^T) \cup \ker(\tilde{B})\}$. All the others are either zero or infinity and are associated to eigenvalues $\lambda = 1$.

The first issue we must resolve is related to the sign of the real and/or imaginary parts of μ. This information is valuable to avoid some unfortunate situations. For example, assuming μ is real and α is chosen such that $|\alpha| \approx |\mu|$ but $\operatorname{sign}(\alpha) = -\operatorname{sign}(\mu)$, then, computing $\alpha + \mu$ would lead to cancellation and this quantity could get dangerously close to zero. Since it appears at the denominator in equation (5.4), the eigenvalues of the preconditioned matrix could increase tremendously. Numerical experiments confirmed such an event could have disastrous consequences for the preconditioner. We could safely avoid this situation by knowing the sign of μ.

Let us consider the generalized eigenvalue problem

$$K\mathbf{u} = \mu C\mathbf{u} \quad \mathbf{u} \in \mathcal{U}$$

with $C = BW^{-1}\tilde{B}$. The choice of W is important and we will choose $W = M_1$. The reason will become clear in the analysis. When not specified, this choice will be assumed in all the remainder of this work. We define the generalized Rayleigh quotient

$$q(\mathbf{u}) = \frac{\mathbf{u}^*K\mathbf{u}}{\mathbf{u}^*C\mathbf{u}} \quad \mathbf{u} \in \mathcal{U}$$

and consider the field of values

$$\mathcal{F} = \{q(\mathbf{u})\colon \mathbf{u} \in \mathcal{U}\}$$

Clearly, the numerator of $q(\mathbf{u})$ is positive for all $\mathbf{u} \in \mathcal{U}$. Only the denominator must be analyzed. With the particular choice for W, the matrix C is expressed as

$$C = BM_1^{-1}\tilde{B} = -U_1U_2^T$$

where we have defined

$$U_1 = -BM_1^{-1} = \begin{pmatrix} 0 \\ I_m \\ 0 \\ Q_1 \end{pmatrix} \qquad U_2 = \tilde{B}^T = \begin{pmatrix} 0 \\ I_m \\ 0 \\ Q_2 \end{pmatrix}$$

with the matrices $Q_1 = -M_2R_{21}M_1^{-1}$ and $Q_2 = -R_{12}^T$. The explicit expression of C is

$$C = -\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & I & 0 & Q_2^T \\ 0 & 0 & 0 & 0 \\ 0 & Q_1 & 0 & Q_1Q_2^T \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & -I & 0 & R_{12} \\ 0 & 0 & 0 & 0 \\ 0 & M_2R_{21}M_1^{-1} & 0 & -M_2R_{21}M_1^{-1}R_{12} \end{pmatrix}$$

Moreover,

$$\mathbf{u}^*C\mathbf{u} = -(\mathbf{u}_{\Gamma_1} + Q_1^T\mathbf{u}_{\Gamma_2})^*(\mathbf{u}_{\Gamma_1} + Q_2^T\mathbf{u}_{\Gamma_2}) = -\mathbf{y}_1^*\mathbf{y}_2$$

with

$$\begin{aligned} \mathbf{y}_1 &= \mathbf{u}_{\Gamma_1} + Q_1^T\mathbf{u}_{\Gamma_2} \\ \mathbf{y}_2 &= \mathbf{u}_{\Gamma_1} + Q_2^T\mathbf{u}_{\Gamma_2} \end{aligned}$$

Thus, $\mathbf{u}^*C\mathbf{u}$ is simply the inner product of two vectors. Assuming the vector $\mathbf{u}$ has real components, $\mathbf{u}^TC\mathbf{u}$ would be positive only if the vectors $\mathbf{y}_1$ and $\mathbf{y}_2$ would be pointing in two very different directions. This can only happen if the matrices Q_1 and Q_2 are very different in some sense. This situation seems highly unlikely provided the interpolation is accurate enough. Therefore, in the general case, we can expect the eigenvalues μ to have negative real part. In fact, an important result may be stated for the conforming case.

Lemma 5.1

In the context of a conforming mesh, the matrix C obtained after setting $W = M$ is symmetric negative semidefinite and the eigenvalues μ of interest are all strictly negative.

Proof. In the conforming case, $M_1 = M_2 = M$ and $R_{12} = R_{21} = I$. Then, C is obviously symmetric and $\mathbf{y}_1 = \mathbf{y}_2 = \mathbf{y} = \mathbf{u}_{\Gamma_1} - \mathbf{u}_{\Gamma_2}$. Thus, $\mathbf{u}^*C\mathbf{u} = -\|\mathbf{y}\|_2^2 < 0 \quad \forall \mathbf{u} \in \mathcal{U}$. Consequently, $\mathcal{F} \subset \mathbb{R}^-$ and in particular all generalized eigenvalues of interest are real negative numbers. □

5.5 Bounds on the generalized eigenvalues μ

In this section, we seek bounds on the eigenvalues μ of the generalized eigenvalue problem

$$K\mathbf{u} = \mu C\mathbf{u} \quad \mathbf{u} \in \mathcal{U}$$

The matrix C has rank m, and we will assume the following ordering for its singular values

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m > \sigma_{m+1} = \cdots \sigma_n = 0$$

Denoting $r = \text{rank}(K) = n - s$, we will assume the following ordering for the eigenvalues of K

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r > \lambda_{r+1} = \cdots = \lambda_n = 0$$

The next theorem provides interesting bounds.

Theorem 5.1

Let μ be an eigenvalue of the generalized eigenvalue problem

$$K\mathbf{u} = \mu C\mathbf{u} \quad \mathbf{u} \in \mathcal{U}$$

then, it satisfies the following inequalities:

$$\frac{\lambda_r}{\sigma_1} \leq |\mu| \leq \frac{\lambda_1}{\sigma_m}$$

Proof. We define the following subspaces

$$\mathcal{V} = \mathbb{C}^n \setminus \ker(K)$$
$$\mathcal{W} = \mathbb{C}^n \setminus \{\ker(B^T) \cup \ker(\tilde{B})\}$$

Let us note that $\mathcal{U} \subseteq \mathcal{V}$ and $\mathcal{U} \subseteq \mathcal{W}$ and recall that $\mathbf{u} \in \mathcal{U}$. Without loss of generality, we assume all eigenvectors have unit norm. From the eigenvalue problem, we have

$$\|K\mathbf{u}\|_2 = |\mu| \|C\mathbf{u}\|_2$$

We bound the left and right-hand sides

$$\lambda_1 = \max_{\substack{\mathbf{v}\in\mathbb{C}^n \\ \|\mathbf{v}\|_2=1}} \|K\mathbf{v}\|_2 \geq \max_{\substack{\mathbf{v}\in\mathcal{U} \\ \|\mathbf{v}\|_2=1}} \|K\mathbf{v}\|_2 \geq \|K\mathbf{u}\|_2 = |\mu|\|C\mathbf{u}\|_2 \geq \min_{\substack{\mathbf{v}\in\mathcal{U} \\ \|\mathbf{v}\|_2=1}} |\mu|\|C\mathbf{v}\|_2 \geq \min_{\substack{\mathbf{v}\in\mathcal{W} \\ \|\mathbf{v}\|_2=1}} |\mu|\|C\mathbf{v}\|_2 = \sigma_m|\mu|$$

From the left and right extremes we obtain the upper bound

$$|\mu| \leq \frac{\lambda_1}{\sigma_m}$$

Similarly for the lower bound

$$\lambda_r = \min_{\substack{\mathbf{v}\in\mathcal{V} \\ \|\mathbf{v}\|_2=1}} \|K\mathbf{v}\|_2 \leq \min_{\substack{\mathbf{v}\in\mathcal{U} \\ \|\mathbf{v}\|_2=1}} \|K\mathbf{v}\|_2 \leq \|K\mathbf{u}\|_2 = |\mu|\|C\mathbf{u}\|_2 \leq \max_{\substack{\mathbf{v}\in\mathcal{U} \\ \|\mathbf{v}\|_2=1}} |\mu|\|C\mathbf{v}\|_2 \leq \max_{\substack{\mathbf{v}\in\mathbb{C}^n \\ \|\mathbf{v}\|_2=1}} |\mu|\|C\mathbf{v}\|_2 = \sigma_1|\mu|$$

From the left and right extremes we obtain the lower bound

$$|\mu| \geq \frac{\lambda_r}{\sigma_1}$$

□

An interesting result follows in the conforming case.

Corollary 5.1

Let μ be an eigenvalue of the generalized eigenvalue problem

$$K\mathbf{u} = \mu C\mathbf{u} \quad \mathbf{u} \in \mathcal{U}$$

then, in the context of a conforming mesh, it satisfies the following inequalities:

$$-\frac{\lambda_1}{2} \leq \mu \leq -\frac{\lambda_r}{2}$$

Proof. We already know from Lemma 5.1 that in the conforming case the eigenvalues μ are negative. In fact, all the singular values of C are known in the conforming case. Without loss of generality, we may assumed the matrix C has been reordered such that it has the following structure

$$C = \begin{pmatrix} 0 & 0 \\ 0 & C_{22} \end{pmatrix} \text{ with } C_{22} = \begin{pmatrix} -I_m & I_m \\ I_m & -I_m \end{pmatrix}$$

Reorderings do not effect the singular values. The matrix C is symmetric negative semidefinite. Therefore, its singular values are the absolute value of its eigenvalues. Obviously, any vector expressed as a linear combination of the vectors of the canonical basis $\{\mathbf{e}_i\}$ $i = 1, 2, \dots, n-2m$ belongs to the kernel of C. The only interesting part of this matrix is C_{22}. It is easy to see that any vector $\mathbf{y}$ of the form $\mathbf{y}^T = (\mathbf{y}_1^T, \mathbf{y}_1^T)$ with $\mathbf{y}_1 \in \mathbb{R}^m$ belongs to the kernel of C_{22}. We have therefore identified all vectors in the kernel. It is also easy to see that all remaining eigenvectors are expressed as $\mathbf{y}^T = (\mathbf{y}_1^T, -\mathbf{y}_1^T)$. Indeed,

$$\begin{pmatrix} -I_m & I_m \\ I_m & -I_m \end{pmatrix} \begin{pmatrix} \mathbf{y}_1 \\ -\mathbf{y}_1 \end{pmatrix} = -2 \begin{pmatrix} \mathbf{y}_1 \\ -\mathbf{y}_1 \end{pmatrix}$$

Therefore, all the nonzero eigenvalues of C are equal to -2. Consequently, all the nonzero singular values of C are equal to 2. □

Theorem 5.1 provides an upper bound based on which we can choose $|\alpha|$. However, it is not a very practical result. Indeed, the computation of σ_1 and σ_m may become expensive for very large problems. Moreover, they can only be computed using Krylov subspace methods for eigenvalue problems because the matrix C is not explicitly available. Only matrix-vector multiplies can be done with this matrix. It is therefore worthwhile investigating if theoretical results are available for the singular values of C. Let us first make a few observations.

In general, when choosing $W = M_1$, C_{22} is expressed as

$$C_{22} = \begin{pmatrix} -M_1 \\ M_2 R_{21} \end{pmatrix} M_1^{-1} \begin{pmatrix} I_m & -R_{12} \end{pmatrix} = -\begin{pmatrix} I_m \\ Q_1 \end{pmatrix} \begin{pmatrix} I_m & Q_2^T \end{pmatrix} = -Y_1 Y_2^T \tag{5.5}$$

where we have defined

$$Y_1 = \begin{pmatrix} I_m \\ Q_1 \end{pmatrix} \quad Y_2 = \begin{pmatrix} I_m \\ Q_2 \end{pmatrix}$$

and the matrices $Q_1 = -M_2 R_{21} M_1^{-1}$ and $Q_2 = -R_{12}^T$. The matrices Y_1 and Y_2 are submatrices of the reordered factors U_1 and U_2 respectively. Therefore, we already have a low-rank factorization for the matrix C_{22}. We already know that C_{22} has rank m and we would like to find an expression for its truncated singular value decomposition. For this purpose, we need orthonormal bases for Y_1 and Y_2. The next result from Kressner (2020) will be useful.

Lemma 5.2

Let $Y \in \mathbb{C}^{n\times m}$ and $Q \in \mathbb{C}^{(n-m)\times m}$ with $n \geq m$ be given by

$$Y = \begin{pmatrix} I_m \\ Q \end{pmatrix}$$

then, the columns of $U = \begin{pmatrix} I_m \\ Q \end{pmatrix} (I_m + Q^* Q)^{-1/2}$ form an orthonormal basis for Y.

Proof. Clearly, the columns of U and Y span the same subspace. The small matrix $(I_m + Q^*Q)^{-1/2}$ is for the change of basis. Moreover, the columns of U are orthonormal. Indeed, by a direct computation

$$U^*U = (I_m + Q^*Q)^{-1/2} \begin{pmatrix} I_m & Q^* \end{pmatrix} \begin{pmatrix} I_m \\ Q \end{pmatrix} (I_m + Q^*Q)^{-1/2} = (I_m + Q^*Q)^{-1/2}(I_m + Q^*Q)(I_m + Q^*Q)^{-1/2} = I_m$$

□

A direct application of the previous lemma shows that

$$U = \begin{pmatrix} I_m \\ Q_1 \end{pmatrix} (I_m + Q_1^T Q_1)^{-1/2} = Y_1 (I_m + Q_1^T Q_1)^{-1/2} \text{ and } V = \begin{pmatrix} I_m \\ Q_2 \end{pmatrix} (I_m + Q_2^T Q_2)^{-1/2} = Y_2 (I_m + Q_2^T Q_2)^{-1/2}$$

are orthonormal bases for Y_1 and Y_2 respectively. Substituting in equation (5.5), we obtain

$$C_{22} = -U(I_m + Q_1^T Q_1)^{1/2} (I_m + Q_2^T Q_2)^{1/2} V^T$$

Let us now denote $Z_1 = (I_m + Q_1^T Q_1)^{1/2}$ and $Z_2 = (I_m + Q_2^T Q_2)^{1/2}$. These matrices are symmetric positive definite. We are only interested in the singular values of C_{22}, not the left and right singular vectors. Clearly, the last expression shows that the singular values of C_{22} are the singular values of $Z_1 Z_2$. The next theorem provides bounds on those singular values.

Theorem 5.2

Let $\lambda_1(M_2)$ and $\lambda_m(M_1)$ denote the largest eigenvalue of M_2 and smallest eigenvalue of M_1 respectively and let σ be a nonzero singular value of C. Then, with the choice $W = M_1$, it satisfies the following inequalities:

$$1 \leq \sigma \leq \sqrt{\left(1 + \frac{\lambda_1^2(M_2)}{\lambda_m^2(M_1)}\|R_{21}\|_2^2\right)\left(1 + \|R_{12}\|_2^2\right)}$$

Proof. To identify bounds on the singular values, we recall that the singular values of a matrix A are the square root of the eigenvalues of $A^T A$ or AA^T depending on the size of the matrix. Then, notice that for all $\mathbf{v} \in \mathbb{R}^m$

$$\frac{\mathbf{v}^T Z_2 Z_1^2 Z_2 \mathbf{v}}{\|\mathbf{v}\|_2^2} = \frac{\mathbf{y}^T Z_1^2 \mathbf{y}}{\mathbf{y}^T Z_2^{-2} \mathbf{y}}$$

Recalling that Z_1 and Z_2 are symmetric matrices, the result follows immediately

$$\sigma_m^2(Z_1)\sigma_m^2(Z_2) = \lambda_m(Z_1^2)\lambda_m(Z_2^2) = \frac{\lambda_m(Z_1^2)}{\lambda_1(Z_2^{-2})} \leq \frac{\mathbf{y}^T Z_1^2 \mathbf{y}}{\mathbf{y}^T Z_2^{-2} \mathbf{y}} \leq \frac{\lambda_1(Z_1^2)}{\lambda_m(Z_2^{-2})} = \lambda_1(Z_1^2)\lambda_1(Z_2^2) = \sigma_1^2(Z_1)\sigma_1^2(Z_2)$$

From which we conclude that if σ is a nonzero singular value of C_{22}, then

$$\sigma_m(Z_1)\sigma_m(Z_2) \leq \sigma \leq \sigma_1(Z_1)\sigma_1(Z_2) \tag{5.6}$$

We do not necessarily have to compute these singular values. In fact, finding lower bounds on $\sigma_m(Z_1)$ and $\sigma_m(Z_2)$ and upper bounds on $\sigma_1(Z_1)$ and $\sigma_1(Z_2)$ is enough for our intended applications. Since

$$\sigma_i(Z_1) = \sqrt{1 + \lambda_i(Q_1^T Q_1)} = \sqrt{1 + \sigma_i^2(Q_1)}$$
$$\sigma_i(Z_2) = \sqrt{1 + \lambda_i(Q_2^T Q_2)} = \sqrt{1 + \sigma_i^2(Q_2)}$$

we obtain straightforwardly

$$1 \leq \sigma_i(Z_1) \leq \sqrt{1 + \|Q_1\|_2^2} \leq \sqrt{1 + \|M_1^{-1}\|_2^2 \|M_2\|^2 \|R_{21}\|_2^2} = \sqrt{1 + \frac{\lambda_1^2(M_2)}{\lambda_m^2(M_1)}\|R_{21}\|_2^2}$$
$$1 \leq \sigma_i(Z_2) \leq \sqrt{1 + \|Q_2\|_2^2} = \sqrt{1 + \|R_{12}\|_2^2}$$

from which we deduce lower and upper bounds on the smallest and largest singular values respectively of Z_1 and Z_2. The result of the theorem follows from the inequalities in (5.6). □

As a corollary, we obtain an already established result. Indeed, in the conforming case $Q_1 = Q_2 = -I_m$. Therefore,

$$\sigma_i(Z_1) = \sigma_i(Z_2) = \sqrt{1 + \lambda_i(I_m)} = \sqrt{2} \quad i = 1, 2, \ldots, m$$

Hence, all the nonzero singular values of C are equal to 2 which is exactly the result we obtained previously.

Combining Theorem 5.1 and 5.2 leads to a very interesting and advantageous result:

$$|\mu| \leq \frac{\lambda_1}{\sigma_m} \leq \lambda_1$$

Although the analysis was not entirely trivial, the result is extremely simple. In addition, the upper bound may be very cheaply approximated using the Gershgorin circles. A much better estimate could be achieved with a few iterations of Lanczos or even with the power method. However, a tight upper bound is not required. Thus, we may use the cheapest method available.

Let us recall that from FEM theory, the mesh size dependency of the largest eigenvalue of the stiffness matrix is known and depends on the dimension of the problem. For 2D problems, the largest eigenvalue is bounded by a constant independent of the mesh size whereas for 3D problems it will decrease linearly with the mesh size according to Theorem 3.3. Setting $|\alpha| = \lambda_u$ ($\alpha = -\lambda_u$) with λ_u an upper bound of λ_1 should lead to a highly performing preconditioner. Optionally, one could set $\alpha = -w\lambda_u$ with $w > 1$ to add a safety margin. However, due to the numerous inequalities used in the proof, this is absolutely not necessary and λ_u is already a safe upper bound. We must also warn against choosing $|\alpha|$ too large: as $|\alpha|$ increases, the condition number of the $(1, 1)$ block of the preconditioner increases as well. Thus, solving linear systems with the preconditioner might get increasingly challenging. Therefore, we recommend setting $\alpha = -\lambda_u$ without any safeguarding.

Interestingly, the result indicates that the preconditioner should be unaffected by the conditioning of the INTERNODES matrix. In particular, the criterion does not explicitly depend on the matrices Q_1 and Q_2. Indeed, numerical experiments well verified this observation. An increased condition number of several orders of magnitude only led to a very small increase of the number of iterations.

5.6 Solving linear systems with the preconditioning matrix

Now that we have a criterion for the choice of α, we must investigate how to solve linear systems with the preconditioning matrix efficiently.

We begin by substituting β and γ by their expression depending on α, that is to say, $\beta = -2\alpha$ and $\gamma = \alpha$. We set $k = 2$ and consider the ideal preconditioner

$$M = \begin{pmatrix} K + \alpha B W^{-1} \tilde{B} & kB \\ 0 & -\alpha^{-1} W \end{pmatrix}$$

However, we will instead consider the practical preconditioner

$$M = \begin{pmatrix} \tilde{K} + \alpha B W^{-1} \tilde{B} & kB \\ 0 & -\alpha^{-1} W \end{pmatrix}$$

where K has been replaced by $\tilde{K} = K + \epsilon I_n$ with $\epsilon > 0$. Adding the term ϵI_n to K leads to a symmetric positive definite (and therefore invertible) matrix. Hence, $\tilde{K}$ admits a Cholesky factorization $\tilde{K} = LL^T$ where L is a lower triangular matrix. The Cholesky factorization is unique for symmetric positive definite matrices.

During the iterations of GMRES, we must solve linear systems with the preconditioning matrix: $M\mathbf{x} = \mathbf{y}$. Explicitly

$$\begin{pmatrix} \tilde{K} + \alpha B W^{-1} \tilde{B} & kB \\ 0 & -\alpha^{-1} W \end{pmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix}$$

Therefore,

$$\begin{aligned} (\tilde{K} + \alpha B W^{-1} \tilde{B})\mathbf{x}_1 + kB\mathbf{x}_2 &= \mathbf{y}_1 \\ -\alpha^{-1} W \mathbf{x}_2 &= \mathbf{y}_2 \end{aligned}$$

The second equation is trivial and yields $\mathbf{x}_2 = -\alpha W^{-1} \mathbf{y}_2$. The only difficulty is solving

$$(\tilde{K} + \alpha B W^{-1} \tilde{B})\mathbf{x}_1 = \mathbf{y}_1 + \alpha k B W^{-1} \mathbf{y}_2 = \tilde{\mathbf{y}}_1$$

The strategy is to use the sparsity of the blocks B and $\tilde{B}$. At the heart of the method is the following theorem.

Theorem 5.3

Let the matrix $\tilde{K} + \alpha B W^{-1} \tilde{B}$ be ordered such that

$$\tilde{K} + \alpha B W^{-1} \tilde{B} = \begin{pmatrix} \tilde{K}_{11} & \tilde{K}_{12} \\ \tilde{K}_{21} & \tilde{K}_{22} \end{pmatrix} - \alpha \begin{pmatrix} 0 & 0 \\ 0 & Y_1 Y_2^T \end{pmatrix}$$

with Y_1 and Y_2 defined in the previous section. Moreover, let

$$L = \begin{pmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix}$$

be the Cholesky factor of $\tilde{K}$ and let us denote $G = I - \alpha L_{22}^{-1} Y_1 Y_2^T L_{22}^{-T}$. Then, $\tilde{K} + \alpha B W^{-1} \tilde{B}$ admits a block LDL^T factorization given by

$$\tilde{K} + \alpha B W^{-1} \tilde{B} = \begin{pmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & G \end{pmatrix} \begin{pmatrix} L_{11}^T & L_{21}^T \\ 0 & L_{22}^T \end{pmatrix} = LDL^T$$

Proof. Since $\tilde{K} = LL^T$, a simple substitution leads to

$$\tilde{K} + \alpha B W^{-1} \tilde{B} = LL^T + \alpha B W^{-1} \tilde{B} = L(I + \alpha L^{-1} B W^{-1} \tilde{B} L^{-T}) L^T$$

The sparsity of the matrices B and $\tilde{B}$ can be used to simplify the computation of $L^{-1} B W^{-1} \tilde{B} L^{-T}$. Indeed,

$$\begin{aligned} L^{-1} B W^{-1} \tilde{B} L^{-T} &= \begin{pmatrix} L_{11}^{-1} & 0 \\ -L_{22}^{-1} L_{21} L_{11}^{-1} & L_{22}^{-1} \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & -Y_1 Y_2^T \end{pmatrix} \begin{pmatrix} L_{11}^{-T} & -L_{11}^{-T} L_{21}^T \\ 0 & L_{22}^{-T} \end{pmatrix} \\ &= \begin{pmatrix} 0 & 0 \\ 0 & -L_{22}^{-1} Y_1 Y_2^T L_{22}^{-T} \end{pmatrix} \end{aligned}$$

Therefore, we obtain

$$\tilde{K} + \alpha B W^{-1} \tilde{B} = \begin{pmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & I - \alpha L_{22}^{-1} Y_1 Y_2^T L_{22}^{-T} \end{pmatrix} \begin{pmatrix} L_{11}^T & L_{21}^T \\ 0 & L_{22}^T \end{pmatrix}$$

□

Consequently, assuming $G = I - \alpha L_{22}^{-1} Y_1 Y_2^T L_{22}^{-T}$ is invertible, the inverse of $\tilde{K} + \alpha B W^{-1} \tilde{B}$ admits the explicit expression

$$(\tilde{K} + \alpha B W^{-1} \tilde{B})^{-1} = \begin{pmatrix} L_{11}^{-T} & -L_{11}^{-T} L_{21}^T L_{22}^{-T} \\ 0 & L_{22}^{-T} \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & G^{-1} \end{pmatrix} \begin{pmatrix} L_{11}^{-1} & 0 \\ -L_{22}^{-1} L_{21} L_{11}^{-1} & L_{22}^{-1} \end{pmatrix} = L^{-T} D^{-1} L^{-1}$$

We will denote b_1 and b_2 the sizes of L_{11} and L_{22} respectively with $b_2 << b_1$. Consequently, solving a linear system with $\tilde{K} + \alpha B W^{-1} \tilde{B}$ only requires the solution to two large triangular systems and another very small linear system with the matrix G. In linear elasticity without any remeshing, the stiffness matrix K does not change during the iterations of the contact algorithm. Thus, we may conveniently compute only a single Cholesky factorization and reuse it during all subsequent iterations of the contact algorithm for solves with the preconditioning matrix. On the other hand, the matrix G is completely dense. However, if the problem is small enough, it can be computed explicitly and direct methods should be used to solve these small linear systems. In such circumstances, one should not even think of using iterative methods.

For very large problems, resorting to a Cholesky factorization or even forming the matrix G becomes infeasible. Fortunately, the same strategy can be used with only a few adjustments. First of all, an incomplete Cholesky factorization can be used instead of the full one. Secondly, linear systems with the small matrix G may be solved iteratively. However, using an incomplete Cholesky factorization will destroy the nice properties of the preconditioner. Numerical testing in a later section will assess the impact on the quality of the preconditioner.

For now, let us investigate how linear systems with the matrix G may be solved iteratively. The matrix G^{-1} can be expressed as

$$\begin{aligned} G^{-1} &= (I - \alpha L_{22}^{-1} Y_1 Y_2^T L_{22}^{-T})^{-1} \\ &= L_{22}^T (L_{22} L_{22}^T - \alpha Y_1 Y_2^T)^{-1} L_{22} \\ &= L_{22}^T V^{-1} L_{22} \end{aligned}$$

where we have denoted $V = L_{22} L_{22}^T - \alpha Y_1 Y_2^T$. Matrix-vector products with V are clearly affordable. Instead, matrix-vector products with G would require solving two triangular systems.

Unfortunately, GMRES again converged very slowly for these inner iterations. In fact, convergence was not far from the worst case scenario where GMRES converges only at the last iteration, when the dimension of the Krylov basis reaches the size of the matrix. Thus, inner preconditioning is required. Before discussing it, some additional tools must be introduced.

5.7 Sparse approximate inverse techniques

All the most successful solution techniques we will discuss are based on the approximation of the dense blocks contained in the low-rank term $Y_1 Y_2^T$ by some sparse approximation. We recall that the matrices are expressed as

$$Y_1 = \begin{pmatrix} I_m \\ Q_1 \end{pmatrix} \quad Y_2 = \begin{pmatrix} I_m \\ Q_2 \end{pmatrix}$$

The matrices $Q_1 = -M_2 R_{21} M_1^{-1}$ and $Q_2 = -R_{12}^T$ are very dense. Thus, similarly to B and $\tilde{B}$, the matrices Y_1 and Y_2 are never formed explicitly for large problems. However, they admit an exceedingly good sparse approximation which can be formed explicitly. Before discussing it, we must recall some of the theory behind sparse approximate inverse techniques. One of the most successful techniques was developed by Grote and Huckle (1997) and we will follow their discussion in this section. The problem consists in finding a good sparse approximate inverse M to a sparse matrix A. Let S be a set containing pairs of indices (i, j) which define the sparsity structure of a matrix. Then, we introduce the set of sparse matrices $\mathcal{S} = \{M \in \mathbb{R}^{n\times n} : m_{ij} = 0 \ (i, j) \notin S\}$ and consider the minimization problem

$$\min_{M \in \mathcal{S}} \|AM - I\|_F^2$$

Historically, sparse approximate inverses were designed for being used as preconditioners. More specifically, the matrix M is intended here as a right preconditioner. However, our use of sparse approximate inverses will be somewhat different. It must be recognized that the minimization problem defined above is not a guarantee for building an effective preconditioner. It does not always provide a reliable measure for the convergence of iterative methods. However, there are many computational advantages. The minimization in the Frobenius norm allows to decouple the problem and solve a least squares problem for each column of M separately. For this reason, it is also well suited for parallel preconditioning. The problem becomes

$$\|AM - I\|_F^2 = \sum_{k=1}^{n} \|A\mathbf{m}_k - \mathbf{e}_k\|_2^2$$

Since the columns of M are computed independently, we may restrict the explanation to only one of them. Due to the sparsity of $\mathbf{m}_k$, many of the columns of A will not contribute. Let $\mathcal{J}$ be the set of nonzero entries in $\mathbf{m}_k$. We may further reduce the size of the problem by only considering the nonzero rows of the submatrix $A(:, \mathcal{J})$. Indeed, zero rows will not effect the least squares solution. We denote $\mathcal{I}$ the set of nonzero rows. Thus, the problem reduces to

$$\min \|A(\mathcal{I}, \mathcal{J})\mathbf{m}_k(\mathcal{J}) - \mathbf{e}_k(\mathcal{I})\|_2^2 = \min \|\hat{A}\hat{\mathbf{m}}_k - \hat{\mathbf{e}}_k\|_2^2$$

where $\hat{A}$ is the submatrix $\hat{A} = A(\mathcal{I}, \mathcal{J})$ and $\hat{\mathbf{m}}_k = \mathbf{m}_k(\mathcal{J})$ and $\hat{\mathbf{e}}_k = \mathbf{e}_k(\mathcal{I})$ are subvectors of $\mathbf{m}_k$ and $\mathbf{e}_k$ respectively. The greater the sparsity of the matrices, the smaller the size of the least squares problem which is solved exactly using a QR factorization. The procedure is repeated for each column of M. The difficult part is finding a good sparsity structure $\mathcal{S}$. For this purpose, Grote and Huckle considered an adaptive strategy. One defines an initial sparsity structure which is then improved within a few iterations. The initial sparsity structure is arbitrary. However, the better it is chosen, the fewer the number of iterations. For our problem, choosing a diagonal sparsity structure is a very good choice since the approximate inverse is not expected to be too different from a diagonal matrix.

To determine how the sparsity structure should be augmented, the residual vector $\mathbf{r} = A(:, \mathcal{J})\hat{\mathbf{m}}_k - \mathbf{e}_k$ is formed and the one-dimensional minimization problem $\min_{\mu_j} \|\mathbf{r} + \mu_j \mathbf{a}_j\|_2$ is solved for each column j in a set of potential candidates. The idea is to find out where fill-in should be allowed such that the nonzeros added will decrease most the residual. In other words, determining which are the most profitable indices. Once these indices have been found, they are added to the set $\mathcal{J}$ and the procedure is repeated until a tolerance on the residual is reached. To avoid excessive fill-in, a threshold on the maximum number of nonzeros per column is set. Typically, this threshold will not be reached if the matrix admits a good sparse approximate inverse. The resulting algorithm the authors propose is called SPAI (SParse Approximate Inverse). Chow and Saad (1998) proposed different strategies based on descent type algorithms and Newton iterations to minimize the residual. These methods are coupled with strategies for numerical dropping as the iterations proceed to avoid excessive fill-in. We refer to Grote and Huckle (1997), Chow and Saad (1998) and Benzi and Tuma (1999) for all the implementation details. Using an adaptive strategy for computing the sparsity pattern may be expensive and several alternative techniques were proposed to make such strategies more efficient.

The idea of sparse approximate inverses dates as far back as the 1970s but research was most intense in the late 1990s. These techniques are especially effective for small matrices. For large problems, they are acknowledged to be more expensive than incomplete factorizations. Nevertheless, since most of them apply to general nonsymmetric matrices, they offer a valuable alternative when incomplete LU factorizations fail due to instabilities. Such instabilities are most frequently encountered for highly nonsymmetric or indefinite matrices. It was shown experimentally that sparse approximate inverses could solve difficult problems for which incomplete LU factorizations failed. Nonetheless, one must bare in mind that these techniques can only be successful provided the inverse is well approximated by a sparse matrix.

In this project, we will not use them to compute a sparse approximate inverse of the entire INTERNODES matrix. Instead, they will be used to find very good sparse approximations to the inverse of the matrices Φ_{11} and Φ_{22} which will serve as building blocks for the sparse preconditioner of a larger matrix.

In order to compute a sparse approximation of Y_1 and Y_2, the matrices $Q_1 = -M_2 R_{21} M_1^{-1}$ and $Q_2 = -R_{12}^T$ will be approximated. In fact, they admit an exceedingly good sparse approximation which can be formed explicitly. To understand why, we must further recall the definition of the matrices R_{12} and R_{21}.

$$R_{12} = D_{11}^{-1} \Phi_{12} \Phi_{22}^{-1}$$
$$R_{21} = D_{22}^{-1} \Phi_{21} \Phi_{11}^{-1}$$

where D_{ii}, $i = 1, 2$ is a diagonal matrix containing the rescaling factors and $\Phi_{ij}, \quad i, j = 1, 2$ are radial basis function interpolation matrices. These matrices are all extremely sparse and the matrices $\Phi_{ii} \quad i = 1, 2$ are in addition strictly diagonally dominant by rows. Due to the definition of the Wendland C^2 basis functions, the entries of these matrices decay very fast away from the diagonal. Thanks to this property, we can expect very good sparse approximate inverses. In fact, the simplest approximation consists of extracting the diagonal only and taking the inverse of the diagonal entries. The resulting matrix is provably a good approximation (Yan, 2019). However, we will build better approximations using the SPAI algorithm. The sparse approximation of the matrices R_{12} and R_{21} is expressed as

$$\hat{R}_{12} = D_{11}^{-1} \Phi_{12} \Psi_{22}$$
$$\hat{R}_{21} = D_{22}^{-1} \Phi_{21} \Psi_{11}$$

where Ψ_{11} and Ψ_{22} are the approximations of Φ_{11}^{-1} and Φ_{22}^{-1} respectively computed with the SPAI algorithm. We will purposely never write approximate inverses as the true inverse of an approximate matrix. Indeed, it is difficult in general to prove the non-singularity of a computed sparse approximate inverse unless using a very strict (and unpractical) tolerance (see Corollary 3.1 of Grote and Huckle). Although it is highly unlikely in practice to encounter a singular approximate inverse, we cannot prevent it in an easy way.

The matrices Q_1 and Q_2 are then approximated by

$$\hat{Q}_1 = -\hat{M}_2 \hat{R}_{21} \hat{M}_1^{-1}$$
$$\hat{Q}_2 = -\hat{R}_{12}^T$$

where the mass matrices M_1 and M_2 are approximated by using their diagonal only. Such an approximation is based on the fact that the diagonal of a mass matrix is provably a good preconditioner (Wathen, 1987). Finally, the sparse approximation of the factors Y_1 and Y_2 is naturally defined as

$$\hat{Y}_1 = \begin{pmatrix} I_m \\ \hat{Q}_1 \end{pmatrix} \quad \hat{Y}_2 = \begin{pmatrix} I_m \\ \hat{Q}_2 \end{pmatrix}$$

We are now ready to discuss ideas for inner preconditioning.

5.8 Inner preconditioning

The first step will be to substitute $L_{22}L_{22}^T$ by $\tilde{K}_{22} - L_{21}L_{21}^T$ such that V can be expressed as

$$V = \tilde{K}_{22} - \alpha Y_1 Y_2^T - L_{21}L_{21}^T$$

and we note that $\tilde{K}_{22}$ is sparse as it is a submatrix of $\tilde{K}$. Moreover, we have just seen that $Y_1 Y_2^T$ can be very well approximated by a sparse matrix. Thus, only the third term must now be dealt with. The idea is fairly simple and one of the two following strategies can be applied:

1. All the entries of $L_{21}L_{21}^T$ which do not fall within the sparsity pattern of $\tilde{K}_{22}$ or $\hat{Y}_1\hat{Y}_2^T$ are dropped (*nofill* strategy).
2. All the entries smaller in absolute value to a certain tolerance are dropped (*droptol* strategy).

Let us denote $\hat{L}_{21}\hat{L}_{21}^T$ the resulting sparse approximation. The sparse matrix

$$\hat{V} = \tilde{K}_{22} - \alpha\hat{Y}_1\hat{Y}_2^T - \hat{L}_{21}\hat{L}_{21}^T$$

can be formed and an LU factorization can be pre-computed at the beginning of each iteration of the contact algorithm. It is used as preconditioner for the inner iterations. As expected, the quality of the preconditioner is closely linked to the number of entries being dropped. On the one hand, the fewer the number of entries dropped, the better the quality of the preconditioner but the more expensive is the cost of applying the preconditioner. Therefore, there is a trade-off which can be decided on based on numerical experiments. Unfortunately, the quality of the preconditioner is also expected to decrease for increasingly large matrices. This feature is a consequence of the increasing number of entries discarded. Incomplete factorizations also suffer from it.

For these reasons, a different strategy was considered. The objective is to design a preconditioner such that its quality is independent of the size of the matrix but may depend on other parameters.

Let us reconsider the matrix $M = L_{22}L_{22}^T - \alpha\hat{Y}_1\hat{Y}_2^T$. To clarify the explanation, we assume $\mathcal{C}^{(0)} = \mathcal{C}_0$ and consider the iteration $k = 0$. These assumptions will be relaxed later. Then, recall that

$$\hat{Y}_1 = \begin{pmatrix} I_m \\ \hat{Q}_1 \end{pmatrix} \quad \hat{Y}_2 = \begin{pmatrix} I_m \\ \hat{Q}_2 \end{pmatrix}$$

The matrices $\hat{Y}_1$ and $\hat{Y}_2$ have (coincidentally) a trapezoidal pattern which coincides with the first few columns of L_{22}. The pattern of the matrices is shown in Figure 5.1.

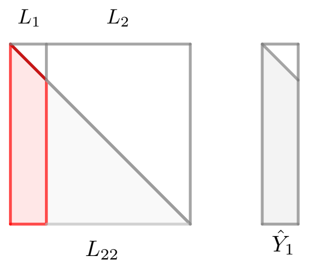

Figure 5.1: Replacement of the first m columns of L_{22} by $\hat{Y}_1$

Notice that L_{22} can be partitioned as $L_{22} = (L_1, L_2)$ where L_1 consists of the first leading m columns of L_{22} and

L_2 of the $b_2 - m$ remaining columns. Then, we obtain

$$\begin{aligned} M &= L_1 L_1^T + L_2 L_2^T - \alpha \hat{Y}_1 \hat{Y}_2^T \\ &\approx L_2 L_2^T - \alpha \hat{Y}_1 \hat{Y}_2^T \\ &= (-\alpha \hat{Y}_1 \; L_2)(\hat{Y}_2 \; L_2)^T \\ &= \hat{L}\hat{U} \end{aligned}$$

Thus, $\hat{L}$ is obtained after replacing the first m columns of L_{22} by $-\alpha\hat{Y}_1$ and $\hat{U}$ is obtained after replacing the first m rows of L_{22}^T by $\hat{Y}_2^T$. In fact, the assumptions stated previously can be relaxed: the replacement can always be done provided the row indices of the first nonzeros in each column of Y_1 (or Y_2) are distinct. Surprisingly, numerical experimentation indicated that the preconditioner performed better when omitting the $-\alpha$. The reason remains unclear.

There are many advantages with this preconditioner. First of all, the cost of computing the factorization is negligible. Only a few columns must be replaced. Secondly, the number of columns replaced does not depend on the size of the matrix. It only depends on the number of interface degrees of freedom of the first body ($m = m_1$). Therefore, for a fixed m, the quality of the preconditioner should not depend on the size of the matrix. Finally, the implementation is very easy and does not require reorderings.

However, there are a few minor drawbacks. First, the quality of the preconditioner cannot be controlled as we did with the previous preconditioner based on a drop tolerance. Secondly, applying the preconditioner could be more expensive depending on how much fill-in is left in the incomplete factors.

5.9 Numerical experiments

In this section, we collect a few numerical experiments designed to test the quality of the preconditioners designed in the previous sections. We will consider as benchmark problem the classic Hertzian contact problem between two elastic bodies. Two different variants will be tested. The geometry and boundary conditions are shown in Figure 5.2.

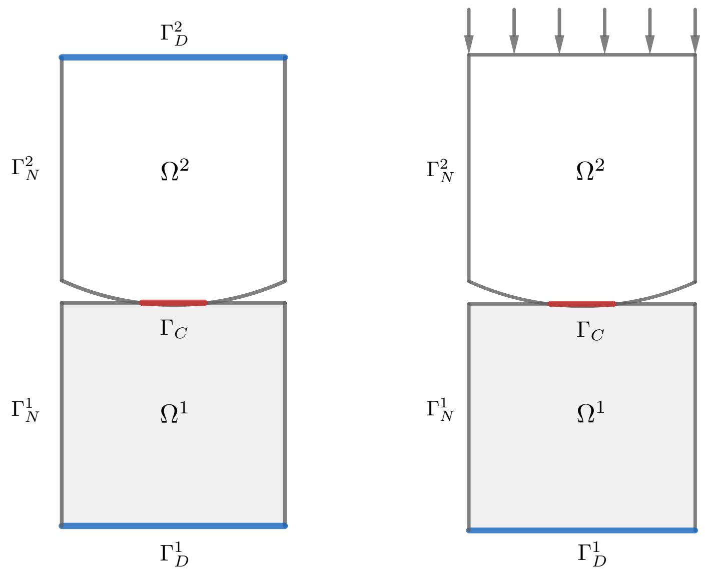

Figure 5.2: Hertzian contact problems

The first body is represented in gray and the second one in white. Between them lies the potential contact interface Γ_C represented in red. The first body is subjected to homogeneous Dirichlet boundary conditions on its base in blue in the figure. The only difference between our two variants lies in the boundary conditions of the second body. In the first case (Figure 5.2, left), we prescribe non-homogeneous Dirichlet boundary conditions on its top edge. In the second case (Figure 5.2, right), a uniform traction is instead applied on this same edge. Therefore, the second body is only subjected to Neumann boundary conditions. This will result in a singularity of the stiffness matrix.

The geometry has been discretized with increasingly smaller mesh sizes and $\mathbb{P}_1$ finite elements. The mesh sizes and the corresponding block sizes have been reported in Table 5.1 and 5.2.

h	n	m
0.1	530	18
0.05	1 956	34
0.01	46 484	114
0.005	184 134	88

Table 5.1: Mesh sizes and block sizes for variant 1

h	n	m
0.1	552	18
0.05	1 998	34
0.01	46 686	114
0.005	184 536	88

Table 5.2: Mesh sizes and block sizes for variant 2

The block size n depends on the total number of unknowns. Thus, the second variant leads to a slightly larger n due to the Neumann boundary conditions. However, the block size m is only controlled by radial basis function interpolation requirements and does not depend on the boundary conditions.

We will solve the contact problem for these two variants using a right preconditioned GMRES method for solving the linear systems. In this case, the unpreconditioned residual is readily available. Note that due to the rescaling, the so-called unpreconditioned residual is in fact the residual of the rescaled system. The decrease of the residual norm for the outer iterations corresponding to the first iteration of the contact algorithm is reported in Figure 5.3 and 5.4 for variants 1 and 2 respectively. The same preconditioner was also tested using left preconditioned GMRES. The curves for the unpreconditioned residual norm were very similar to those shown in Figure 5.3 and 5.4 and are omitted. In particular, the same iteration counts were recovered. Interestingly, when the preconditioned residual was used instead in the stopping criterion, the results were more pessimistic. According to this residual more iterations were required to meet the tolerance. Because of the mismatch between the preconditioned and unpreconditioned residual, we decided to use right preconditioned GMRES. In all numerical experiments, an absolute stopping criterion was used on the norm of the residual vector with a tolerance of 10^{-8} and a restart value of 20.

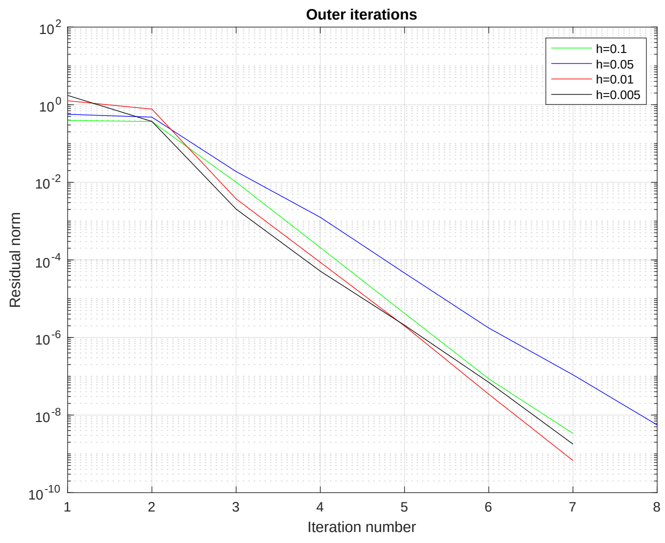

Figure 5.3: Residual norm for variant 1

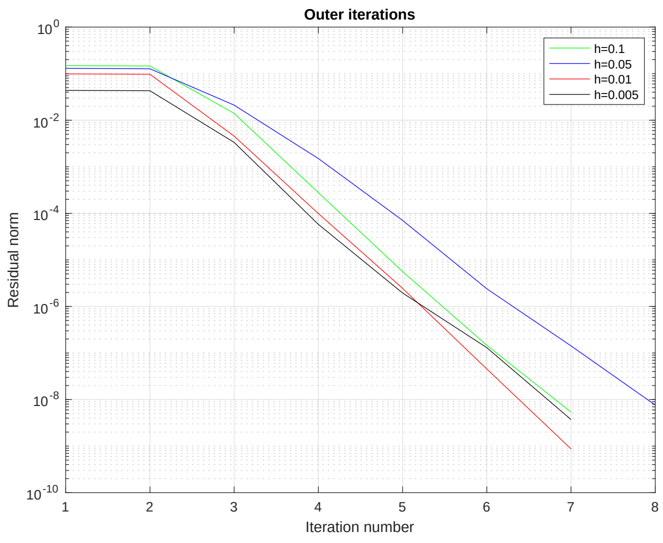

Figure 5.4: Residual norm for variant 2

The figures indicate that the number of iterations needed to reach the prescribed tolerance is almost independent of the mesh size. This is an extremely precious property when using an iterative solver such as GMRES. As its cost increases at each iteration, the method is efficient only if the iteration count remains small. Secondly, the preconditioner is robust: cases with singular or invertible stiffness matrices are handled equally well. This observation does not come as a surprise. The preconditioner was designed to handle singular stiffness matrices.

Moreover, thanks to the rescaling, the quality of the preconditioner is not effected by the elastic modulus. This is well verified in Figure 5.5 for a range of elastic moduli spanning ten orders of magnitude. The experiment was carried out using variant 1 and an intermediate mesh size of $h = 0.05$.

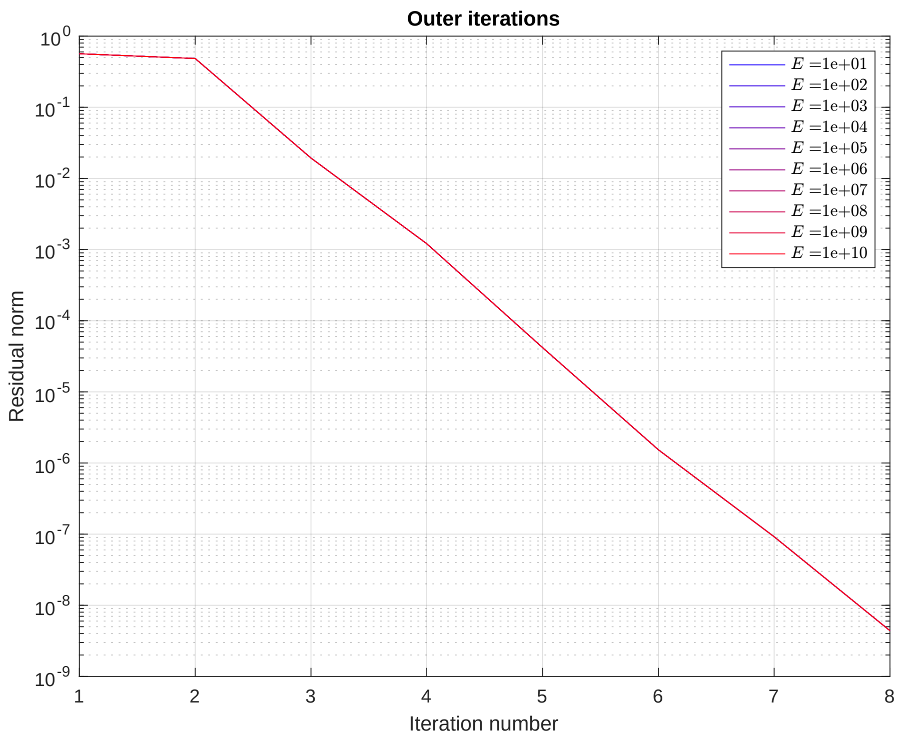

Figure 5.5: Residual norm for different elastic moduli

We will now experiment with incomplete Cholesky factorizations coupled with inner iterations. The same geometry and mesh sizes will be used in the experiments. In practice, these techniques would not be used for such small problem sizes. It is nevertheless interesting to compare them here with a method for medium sized problems. In the experiments, an incomplete Cholesky factorization is computed instead of the full factorization. A drop tolerance of 10^{-4} is used. The results for both variants are shown in Figure 5.6 and 5.7. For the first variant, using an incomplete factorization still leads to a very good preconditioner. However, the iteration count increases with the size of the matrix as could be expected.

This time, serious issues are encountered for the variant with a singular stiffness matrix. Computing the incomplete factorization with the perturbed stiffness matrix leads to nonpositive (most likely zero) pivots and the factorization fails. There is an easy explanation for this failure. The perturbed stiffness matrix is defined as $\tilde{K} = K + \epsilon I_n$ with $\epsilon = 10^{-8}$ in the computations. However, using a drop tolerance of 10^{-4} will wipe out the perturbation ϵI_n which guarantees positive definiteness of the matrix. Without the perturbation term, the matrix is singular and the factorization fails. An easy workaround is to increase the magnitude of the perturbation by choosing ϵ larger than the drop tolerance. It successfully removes the issue but also considerably degrades the quality of the preconditioner when comparing with the first variant.

This is a serious drawback of our approach. Although the Cholesky factorization exists for every symmetric positive definite matrix, the incomplete factorization may fail. Existence results for incomplete factorizations are unfortunately limited. In particular, the existence is guaranteed for arbitrary sparsity patterns for M-matrices and the larger class of H-matrices with positive diagonal entries (see for instance Meijerink and Van Der Vorst (1977), Theorem 2.4 and Manteuffel (1980), Corollary 3.3). Unfortunately, finite element matrices for problems in structural mechanics typically do not satisfy these properties. Various strategies have been developed to compute robust incomplete Cholesky factorizations. Here again the literature is vast and we will not cover it all. One such method is an incomplete LDL^T factorization developed by Benzi and Tůma (2003) which generally leads to good preconditioners while having little storage requirements. It is based on an A-orthogonalization process which, even incomplete, cannot lead to pivot breakdowns.

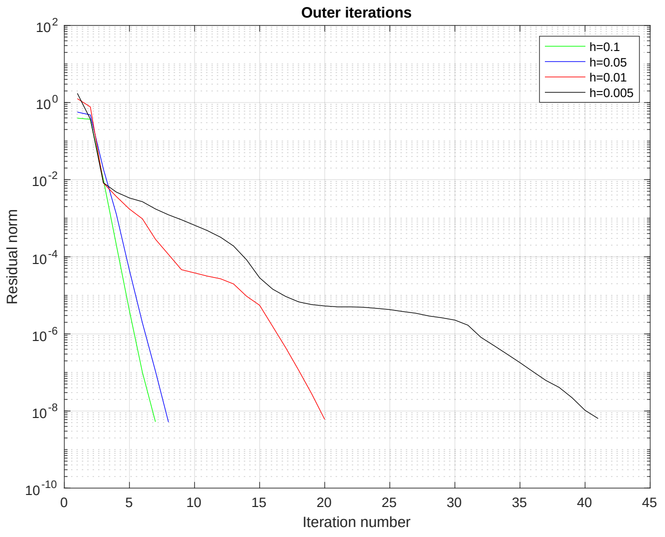

Figure 5.6: Residual norm for variant 1 using an incomplete Cholesky factorization

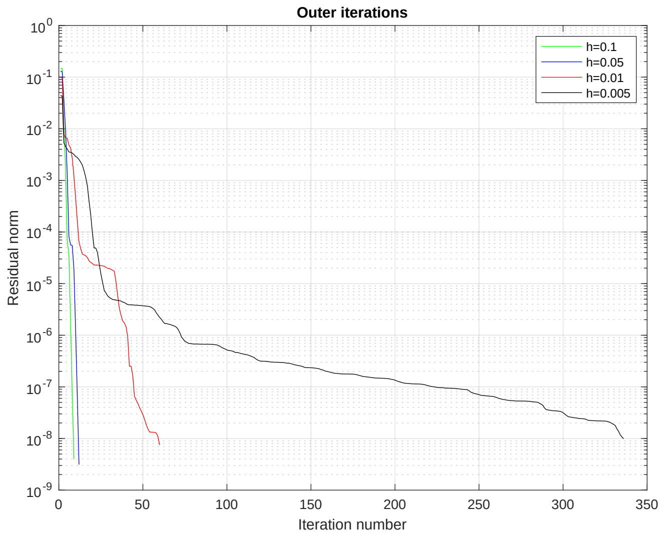

Figure 5.7: Residual norm for variant 2 using an incomplete Cholesky factorization

We now compare the different strategies for the inner iterations. These strategies include:

1. The sparse LU preconditioner with *nofill* strategy.
2. The sparse LU preconditioner with *droptol* $= 10^{-2}$ strategy.
3. The LU preconditioner with column replacement.

For these inner iterations, we again use the right preconditioned GMRES method. The risk of using inner iterations is that potential errors committed in the inner solves affect the outer solves by delaying convergence. Inexact inner solves might lead to a stagnation of the outer iterations. This situation is particularly unfavorable given the cost of each outer iteration. It can be avoided by solving the small linear systems to high accuracy. Thus, an absolute stopping criterion with a small tolerance of 10^{-11} is used. Due to the small matrix size, the restart threshold can be increased. However, it was kept relatively small by setting it to 100. The sizes of the different blocks are reported in Table 5.3 and 5.4. We recall that b_2 is the size of the small linear system. In the computations, the intersection radius is chosen such that $\mathcal{C}_0 \subset \partial\Omega$. Obviously, the closer $\mathcal{C}_0$ is to the final contact interface, the better the performance of the method. Fine tuning the intersection radius can have a tremendous impact on performance. Such tuning was never done in our experiments. It was chosen based on a rough estimate of where contact was expected to occur.

h	n	m	b_1	b_2
0.1	530	18	478	52
0.05	1 956	34	1 856	100
0.01	46 484	114	45 990	494
0.005	184 134	88	183 144	990

Table 5.3: Mesh sizes and block sizes for variant 1

h	n	m	b_1	b_2
0.1	552	18	500	52
0.05	1 998	34	1 898	100
0.01	46 686	114	46 192	494
0.005	184 536	88	183 546	990

Table 5.4: Mesh sizes and block sizes for variant 2

The residual norm for the first strategy is shown in Figure 5.8 and 5.9 for variant 1 and 2 respectively. The results are for the first iteration of the contact algorithm and the first outer iteration.

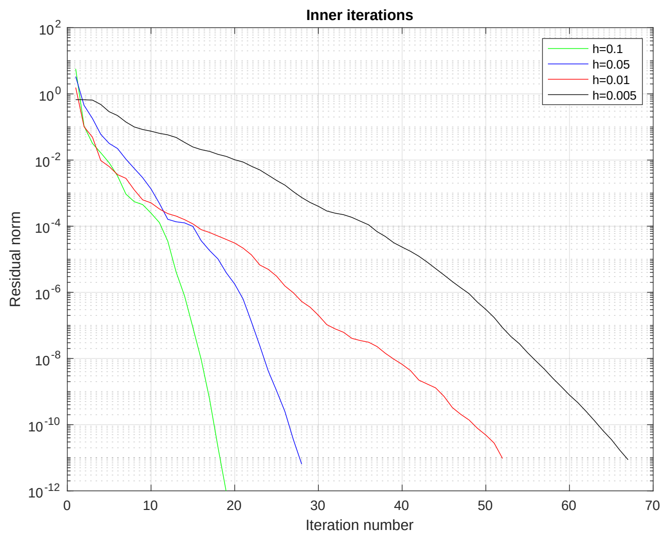

Figure 5.8: Residual norm for variant 1 using the *nofill* strategy

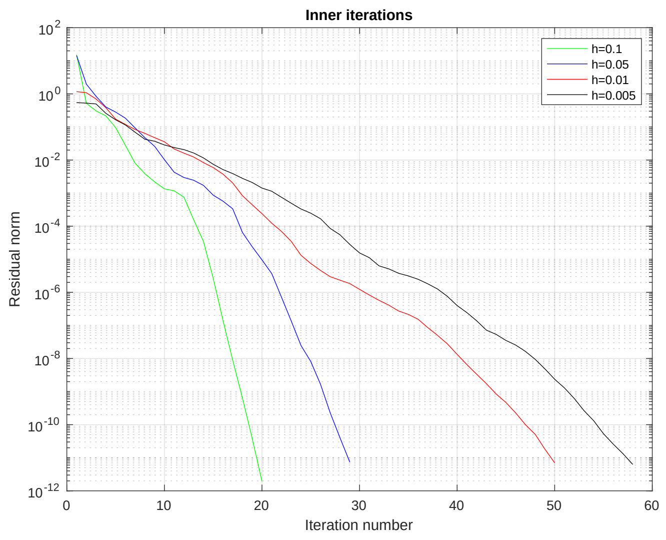

Figure 5.9: Residual norm for variant 2 using the *nofill* strategy

Surprisingly, the iteration counts for the second variant are smaller than those for the first variant for large matrices. However, we are especially concerned with the increase of the iteration number with the size of the matrix. The matrix size is controlled by the intersection radius. An upper bound is given by the number of degrees of freedom on the boundary of both bodies. Thus, the matrix may get quite large. According to Figure 5.10 and 5.11, using the strategy with drop tolerance improves the iteration count. However, it also induces more fill-in in the preconditioner. Thus, only a comparison of the computational times could conclude which strategy is most efficient. Overall, the two strategies behave very similarly with an increase of the iteration count for larger matrices.

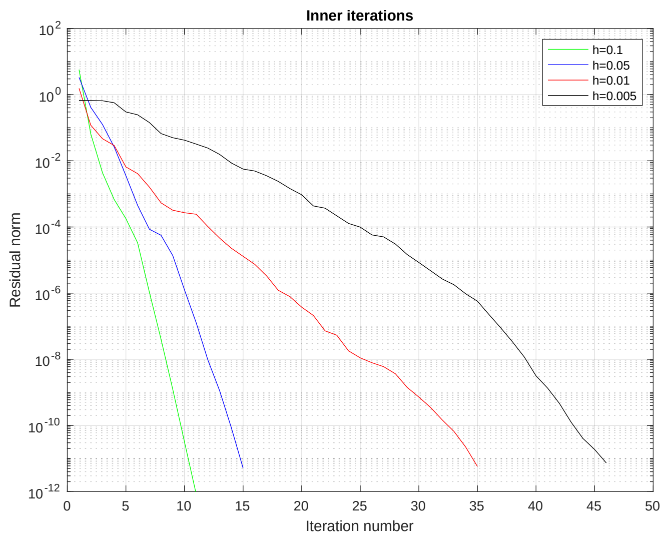

Figure 5.10: Residual norm for variant 1 using the *droptol* strategy

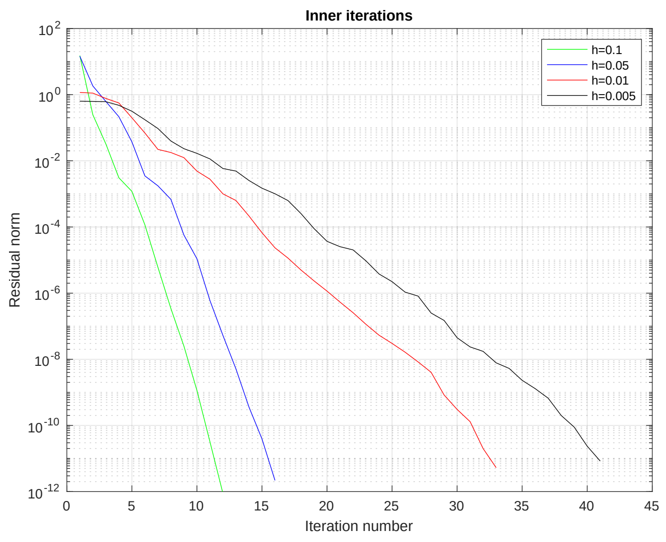

Figure 5.11: Residual norm for variant 2 using the *droptol* strategy

We finally test the third strategy in Figure 5.12 and 5.13 for variant 1 and 2 respectively. The preconditioner performs worse in terms of iteration numbers for smaller matrices in comparison to the two previous strategies. However, the iteration count grows more slowly with the size of the matrix such that it performs as well as for the previous strategy for the largest matrix. If this trend persists, we can expect the preconditioner to perform better for larger matrices.

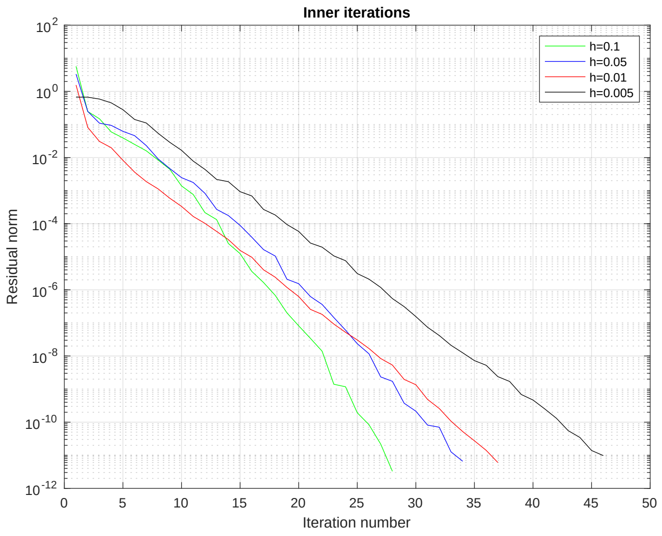

Figure 5.12: Residual norm for variant 1 using the modified LU preconditioner

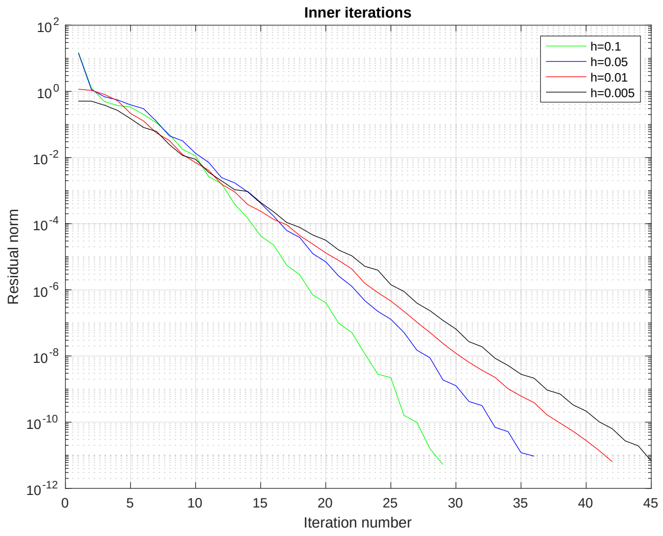

Figure 5.13: Residual norm for variant 2 using the modified LU preconditioner

The different methods are now compared in terms of computational cost. The first round of testing compares preconditioned GMRES against a direct method used to solve the linear systems. In MATLAB, a direct method is called when using the backslash command. Due to the properties of the INTERNODES matrix, it will be some state-of-the-art implementation of a sparse LU factorization. Our method uses an in-house implementation of preconditioned GMRES. We did not use the built-in *gmres* function because we lacked control over the stopping criterion. Moreover, it uses left preconditioned GMRES whereas we are using the right preconditioned version. Our implementation is very basic and its performance could certainly be improved. All the experiments are carried out on MacOS with a Dual-Core Intel Core i7 processor with 2.2 GHz of processing speed. In Figure 5.14, the computational times for the entire contact algorithm are reported for variant 1. The contact algorithm usually requires solving a sequence of linear systems and not just a single one. Moreover, these times are more instructive from a user perspective. The contact algorithm calls many different functions. It includes:

1. Computing the radiuses of radial basis functions.
2. Assembling the interpolation matrices, interface mass matrices and stiffness matrix.
3. Solving the linear systems with the INTERNODES matrix.
4. Verifying the convergence of the algorithm.

However, the only difference between the two implementations is the method for solving the linear systems. The maximum number of iterations for the contact algorithm was set to 10. Most of the time, the contact algorithm converged within two to three iterations. However, the threshold on the maximum number of iterations was reached for the smallest mesh size. In fact, the contact algorithm does not converge for the smallest mesh size. The issue is well identified but will not be discussed here. Since the iteration numbers are different for different mesh sizes, trying to find a trendline is meaningless.

Some important observations must be made. First of all, the computational times are always relatively small for both implementations and it is extremely challenging to outperform direct methods for applications where they excel. For example, for the mesh size $h = 0.01$ associated to a matrix of size over 46 000, three large linear systems are solved and the contact algorithm computes the solution in roughly 1.89 s with a direct method. Despite, their effectiveness, our preconditioned GMRES implementation always outperforms direct methods by a large margin. For the smallest matrix size, the computational times may fluctuate but they stabilize for larger matrices. The computational times are also reported in Table 5.5 as well as the speedup factor defined as the ratio of computational times for a direct and iterative method.

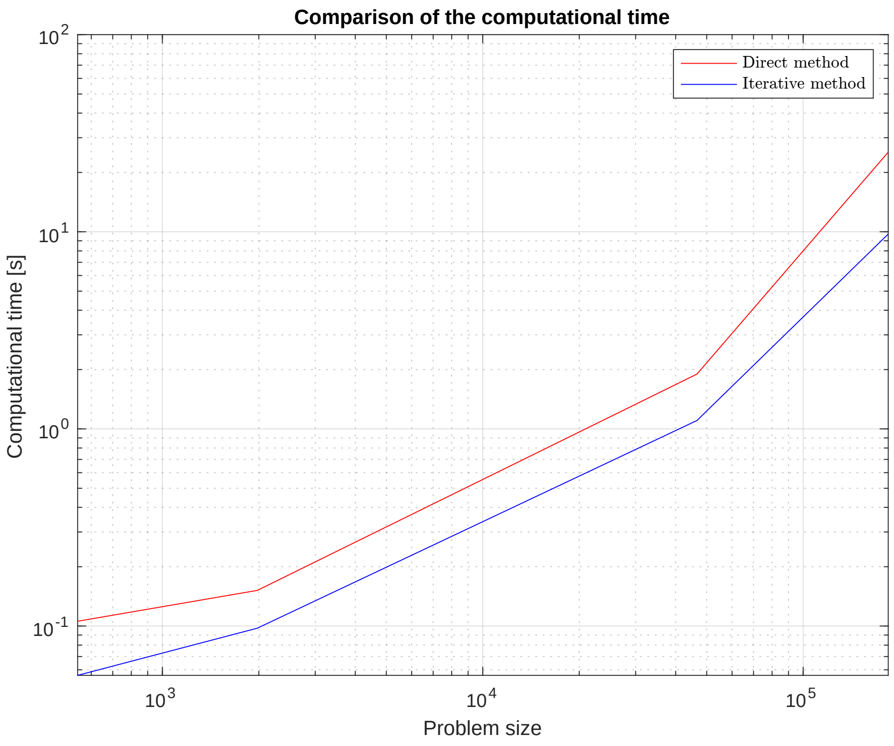

Figure 5.14: Computational times for the contact algorithm

h	0.1	0.05	0.01	0.005
Time direct [s]	0.10	0.15	1.89	25.34
Time iterative [s]	0.056	0.097	1.10	9.74
Speedup factor	1.88	1.55	1.72	2.60

Table 5.5: Computational times and speedup factors

The speedup factor is expected to increase further with the size of the matrix and the number of linear systems solved in the sequence. Since our method uses a single Cholesky factorization, its cost is amortized over several linear systems. In comparison, direct methods will recompute a factorization at each iteration.

We now compare the computational times for the different strategies used in the inner iterations. In this case, an incomplete Cholesky factorization is used. The results are reported in Figure 5.15 for the first variant. As we have already noticed, using incomplete factorizations destroys the nice properties of our preconditioner. The number of iterations needed to converge become mesh size dependent. Although the growth in the number of iterations is moderate (41 iterations instead of 7 for the smallest mesh size), this increase has dramatic consequences for the overall computational expense.

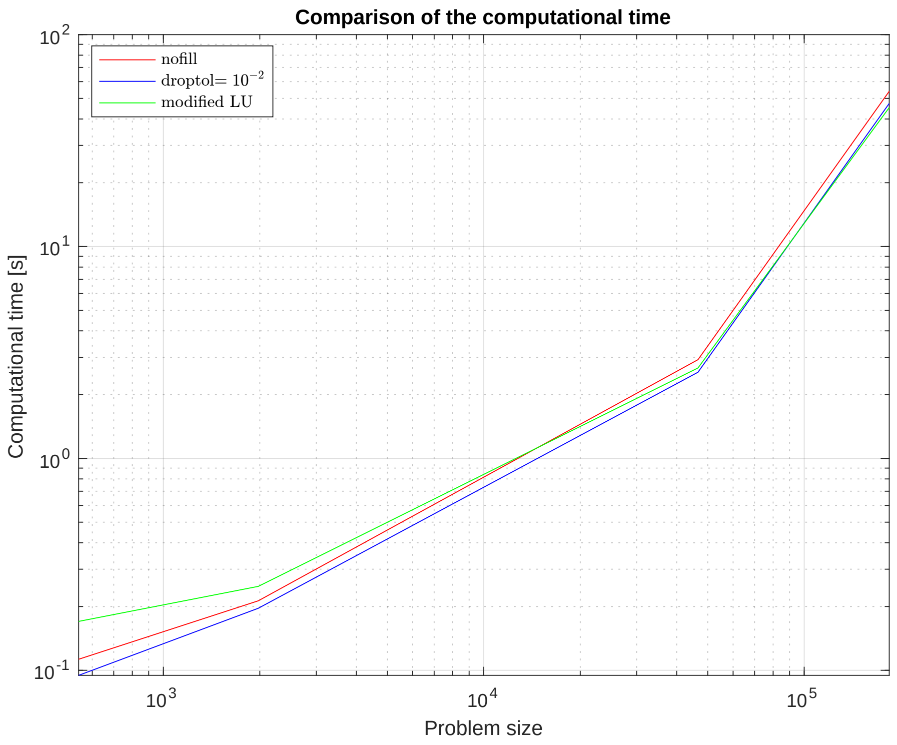

Figure 5.15: Computational times for the contact algorithm and different strategies for inner iterations

The curves are not very different from each other because the times reported are for the entire contact algorithm. Keeping track of the inner iterations only was not convenient because their cost is spread across various functions located in various files. The figure reveals the overwhelming contribution of the outer iterations. This tremendous increase is not only due to the additional triangular solves and matrix-vector multiplies but also because of the growing cost of GMRES. The cost of the inner iterations becomes negligible for large matrices. The figure nevertheless confirms our expectations. The *nofill* strategy performs well for small matrices but its cost grows rapidly for larger matrices. On the contrary, the modified LU factorization with column replacement performs poorly on small matrices but its cost increases more slowly with the size of the matrix such that it becomes competitive for larger matrices. The strategy with drop tolerance always performs well. For the largest matrix, the modified LU and drop tolerance strategies perform equally well.

These results once again highlight the benefits of having a preconditioner with mesh size independent convergence, as is the case when using the complete Cholesky factorization. None of the other preconditioners we have developed satisfy this property exactly. Although many perform very well on relatively coarse meshes, the increase in the number of iterations with mesh refinement leads to an unacceptable increase of the computational expense. Some of our other attempts which are worthwhile mentioning are collected in the Appendix.

A slight drawback of our method is linked to the ordering of the unknowns: we have assumed interface degrees of freedom appear last in the system of equations. This reordering is absolutely necessary for the efficient solution of the linear systems. It is the main assumption in Theorem 5.3. In particular, it entails that the perturbed stiffness matrix cannot be reordered according to Nested Dissection nor any other sparse reordering strategy. Consequently, more fill-in must be expected in the Cholesky factors. This drawback fortunately turns out to be only minor. Indeed, the proportion of interface degrees of freedom is very small. Moreover, although interface degrees of freedom must come last, it is completely irrelevant how degrees of freedom are ordered within each block. In practice, the ordering computed with Nested Dissection is generated and subsequently interface degrees of freedom are retrieved from the ordering and placed at the bottom. The other degrees of freedom keep their position in the ordering. Hence, the actual ordering is only a slight perturbation of the optimal ordering. This strategy was used for our benchmark problems. Let us provide an illustration for a mesh size $h = 0.01$ corresponding to a matrix of size 46 484. The sparsity pattern of the Cholesky factor corresponding to the ordering computed with Nested Dissection is shown in Figure 5.16 and compared in Figure 5.17 to the sparsity pattern corresponding to the actual ordering used in the computations. As can be noticed, the increased fill-in is fortunately limited.

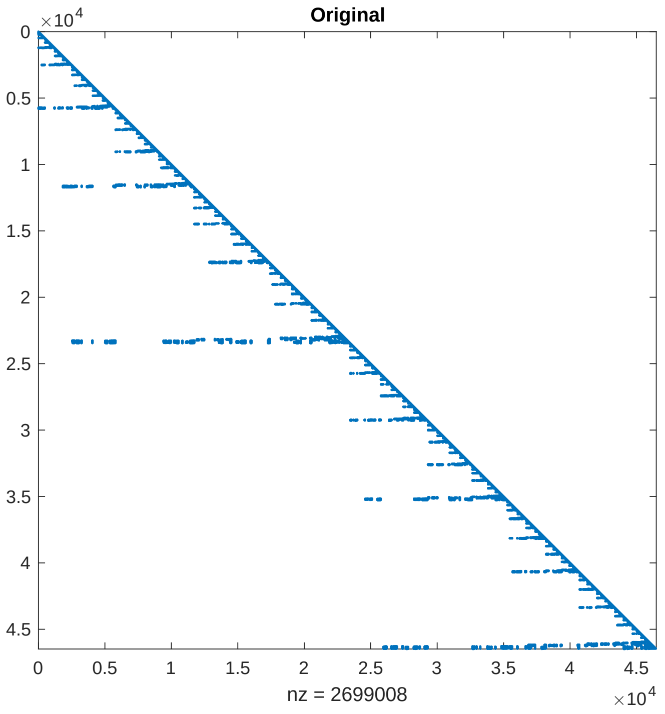

Figure 5.16: Cholesky factor for the ordering of Nested Dissection

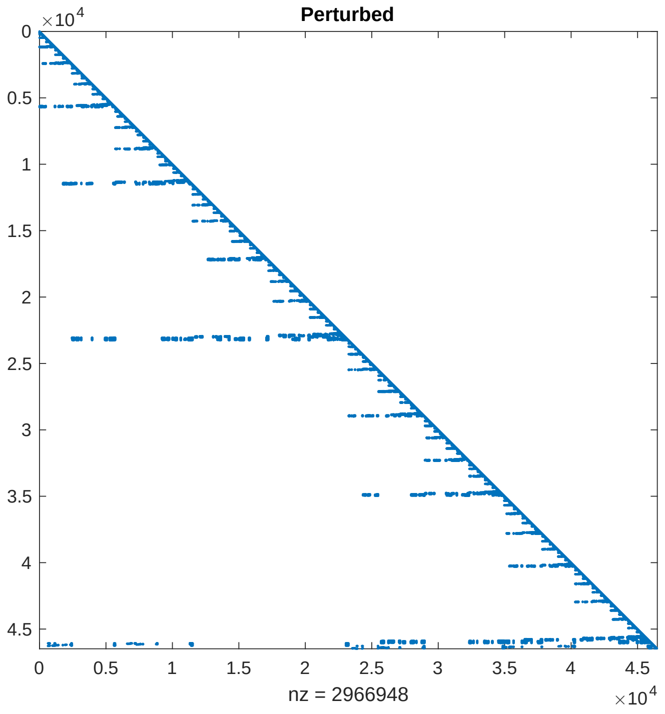

Figure 5.17: Cholesky factor for the perturbed ordering

As long as the full Cholesky factorization can be afforded, our method will lead to great savings in computational time. However, large 3D problems remain challenging for our solver. Krylov subspace methods are certainly not the only option available. Other authors have successfully implemented algebraic multigrid techniques to solve problems in contact mechanics (Adams, 2004).

The next chapter will conclude with an application of INTERNODES to a classic problem of contact mechanics.

Chapter 6

Comparison with analytical solutions

In this chapter, we will compare the numerical solution computed with INTERNODES to the analytical solution of Westergaard for the contact of an elastic wavy surface with a flat surface. The case study reduces to a two dimensional plane strain problem. The geometry is shown in Figure 6.1. The profile of the wavy surface is modeled by the equation

$$f(x) = \Delta\Big(1 - \cos\big(\frac{2\pi x}{L}\big)\Big)$$

where Δ is the amplitude and L is the wavelength. Due to the periodicity, only one wavelength is modeled. The boundary conditions are shown in Figure 6.2. A uniform pressure $\bar{\mathbf{p}}$ is applied on the upper body and horizontal displacements are blocked along the sides of both bodies for symmetry purposes. Moreover, vertical displacements are blocked at the bottom of the lower body. Dirichlet boundary conditions must be prescribed along the vertical of the lower body for the contact problem to be solvable. These are not consistent with the classic Westergaard solution. However, the result is asymptotically the same.

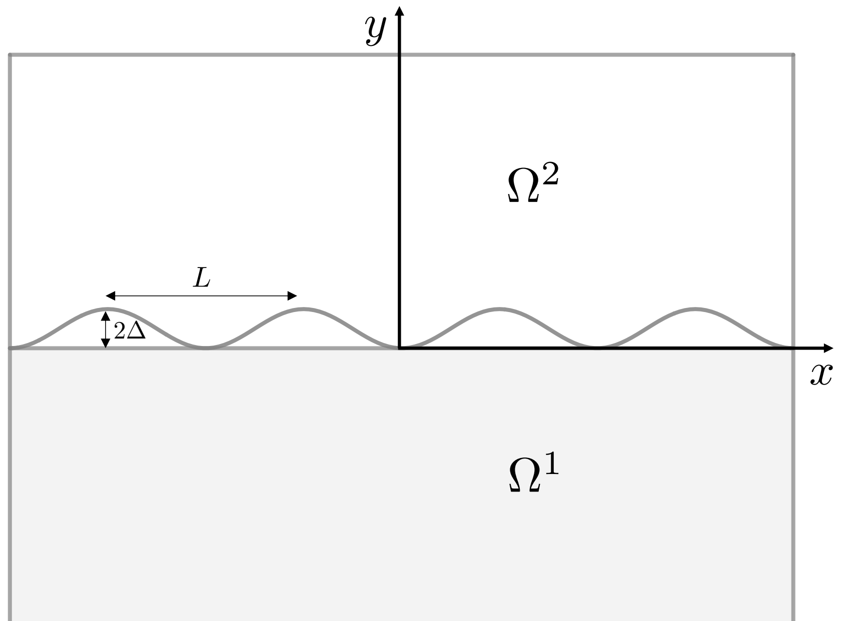

Figure 6.1: Elastic contact between a wavy surface and a flat surface

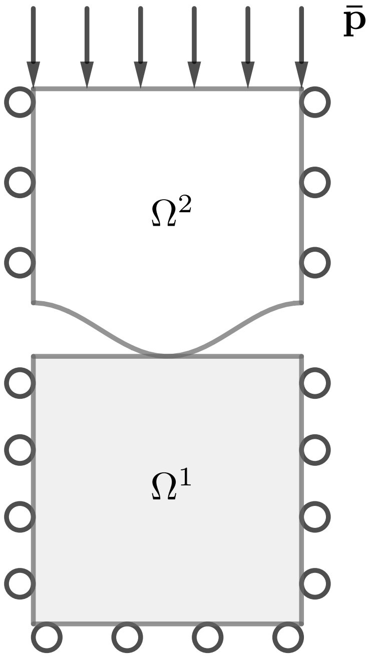

Figure 6.2: Boundary conditions

According to the Westergaard solution, the mean pressure is expressed as

$$\bar{p} = \frac{\pi E^* \sin^2(\psi_a)\Delta}{L}$$

with $\psi_a = \frac{\pi a}{L}$ and $2a$ is the contact width in the deformed configuration. The material properties of each body could be different. Thus, an equivalent elastic modulus is defined as

$$\frac{1}{E^*} = \frac{1-v_1^2}{E_1} + \frac{1-v_2^2}{E_2}$$

However, we shall straightforwardly assume the material properties of both bodies are the same such that $E^* = \frac{E}{2(1-v^2)}$. The pressure vector is then defined as $\bar{\mathbf{p}}^T = (0, -\bar{p})^T$. The contact pressure at the interface is expressed as (Johnson, 1985)

$$p(x) = \frac{2\pi E^* \Delta}{L} \cos(\psi)\sqrt{\sin^2(\psi_a) - \sin^2(\psi)} \qquad -a \leq x \leq a \tag{6.1}$$

where $\psi = \frac{\pi x}{L}$. The contact pressure must be compared with the Lagrange multipliers of the finite element solution. Table 6.1 collects the material and geometrical parameters used in the computations.

Parameter	Value
L $[m]$	1
Δ $[m]$	0.1
a $[m]$	0.1
E $[Pa]$	3×10^{10}
ν $[-]$	0.2

Table 6.1: Material and geometrical parameters

We will consider two $\mathbb{P}_1$ finite element discretizations with different degrees of nonconformity. In the first case, the wavy surface and the flat surface are discretized with the same number of elements on their boundary. In the second case, there are approximately 1.1 more elements on the flat surface. The nonconformity is more severe in the second case. The finite element solution computed with the INTERNODES method is shown in Figure 6.3

for the first case. The solution for the second case is very similar. The displacements in absolute value along the horizontal and vertical are reported in Figure 6.4. The figure indicates some sliding takes place along the interface.

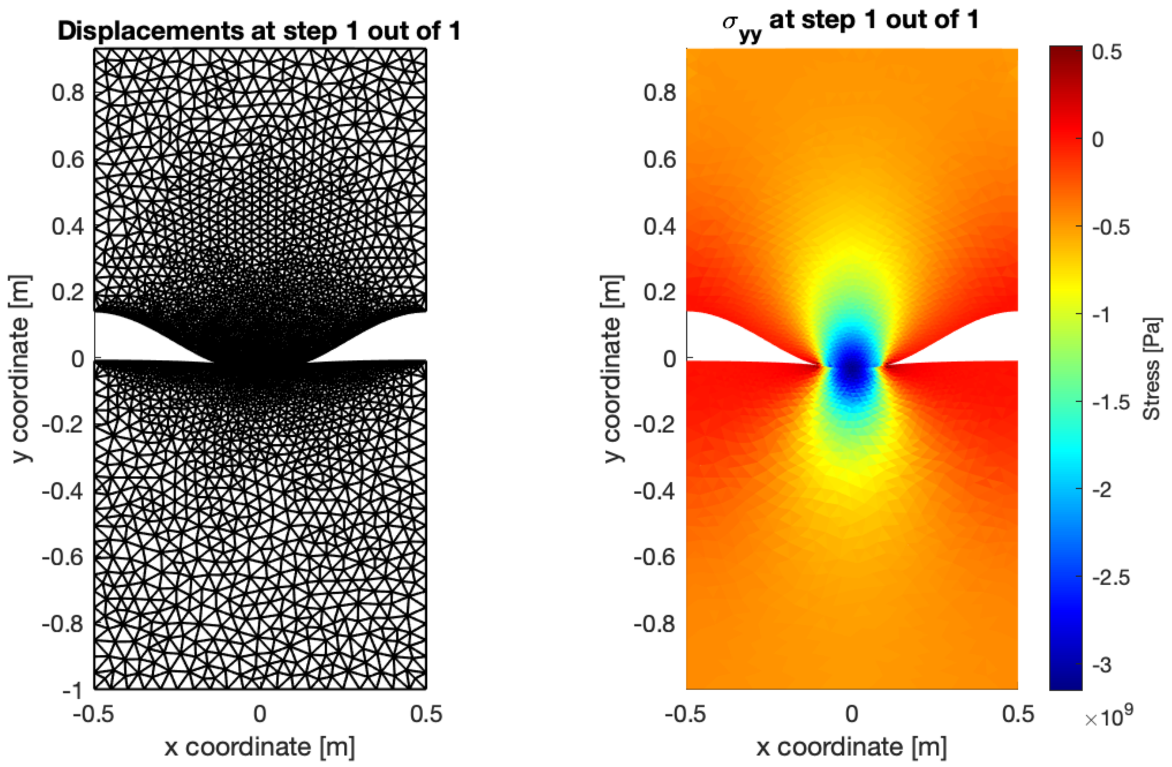

Figure 6.3: Displacements and stresses in the nearly conforming case

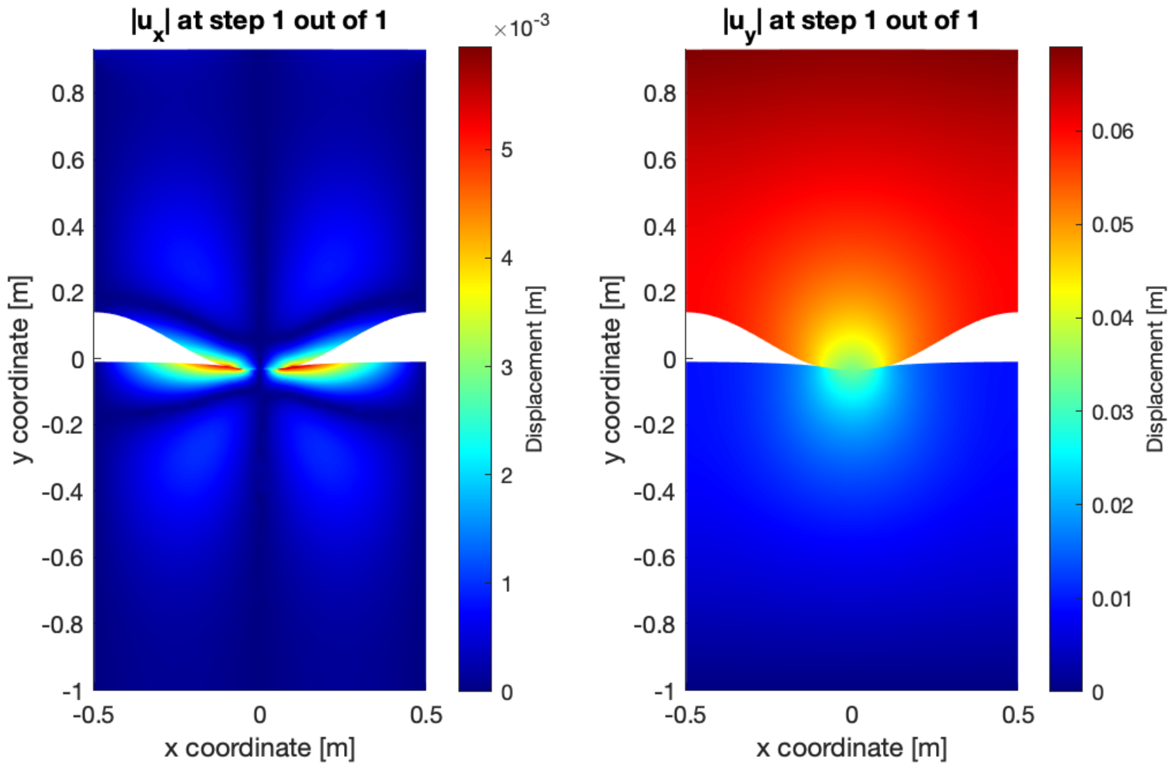

Figure 6.4: Displacements in absolute value along the horizontal and vertical for the nearly conforming case

A zoom of a section of the interface is shown in Figure 6.5 and 6.6 for cases 1 and 2 respectively. Despite the major nonconformity in the second case, the quality of the solution remains satisfying.

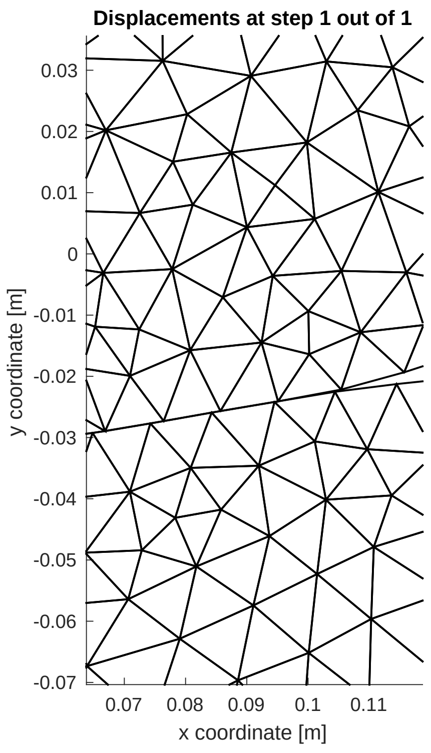

Figure 6.5: Zoom of the contact interface for case 1

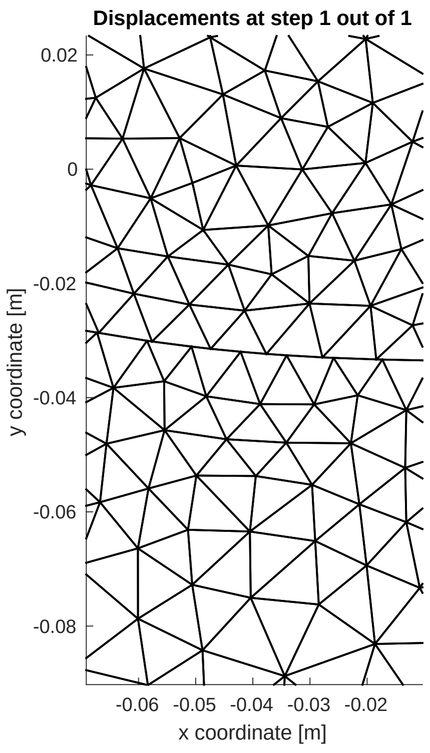

Figure 6.6: Zoom of the contact interface for case 2

Once the solution has been computed, the nodal contact pressure at the interface of body 1 or 2 is readily available through the Lagrange multipliers, depending which body is chosen as master. In our experiments, body 1 is always chosen as master. The contact pressure on the other interface is recovered through interpolation. They can be directly compared to the analytical solution in equation (6.1) by projecting them along the interface normal. In the nearly conforming case, Figure 6.7 and 6.8 compare the solutions for the interface of body 1 and 2. Figure 6.9 and 6.10 provide the same comparison for the fully nonconforming case. First of all, it must be noted that the numerical solution is quite close to the analytical one despite the different setting in the boundary conditions. When the mesh is nearly conforming, the differences in the contact pressure across the interface are very small. The fully nonconforming case is clearly more challenging. The spike appearing in Figure 6.9 is quite surprising but seems to have been smoothed out in the interpolation process.

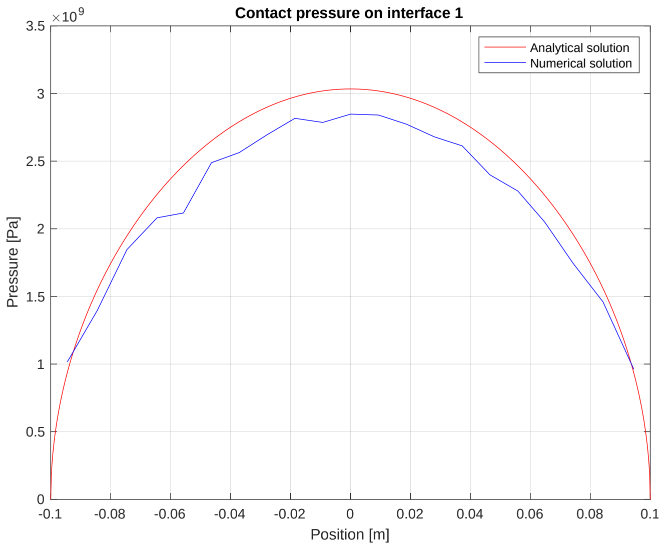

Figure 6.7: Contact pressure on the interface of body 1 for case 1

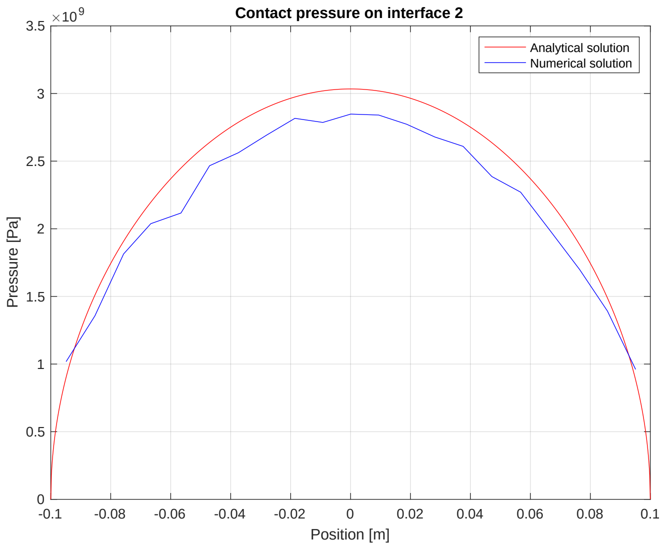

Figure 6.8: Contact pressure on the interface of body 2 for case 1

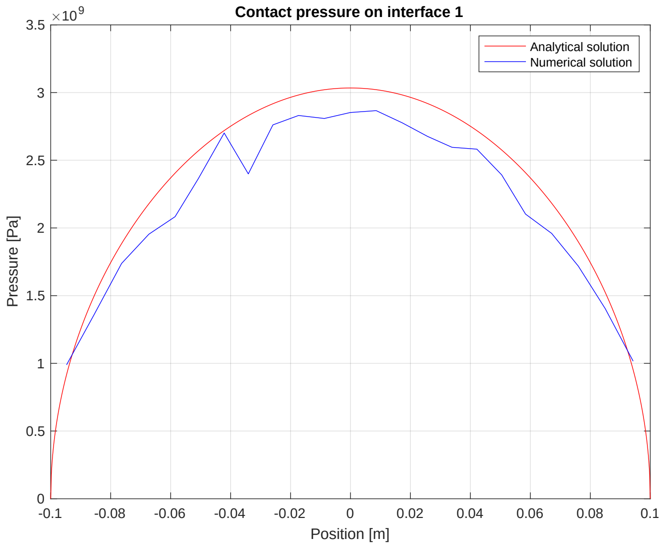

Figure 6.9: Contact pressure on the interface of body 1 for case 2

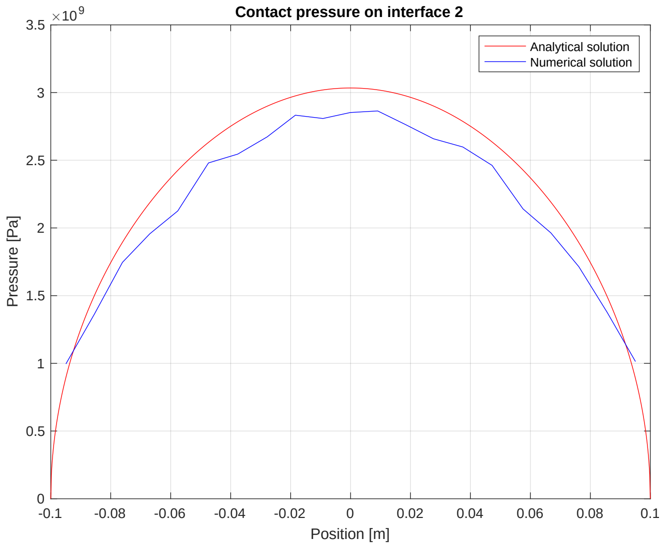

Figure 6.10: Contact pressure on the interface of body 2 for case 2

For a continuous problem, the normal component of the stress vector is expressed as $\sigma_n = \mathbf{n} \cdot \sigma \mathbf{n}$. In the discrete setting, the normal vector is usually not well defined at nodes due to the angles between finite elements. In practice, a weighted average is taken of the surface normals of the neighboring finite elements. The weights are proportional to the lengths (or areas) of the finite elements. However, for the finite element mesh considered in this application, the lengths of neighboring finite elements were almost equal. Thus, a weighted average or a simple average does not make much of a difference. Let $\tilde{n}$ be this average. Moreover, regardless of the finite element order, the stress field is discontinuous between the elements. For $\mathbb{P}_1$ discretizations, we may simply consider the average of the stress inside the finite elements to which the node belongs. We name this quantity $\tilde{\sigma}$. Thus, in the discrete setting, we compute $\tilde{\sigma}_n = \tilde{\mathbf{n}} \cdot \tilde{\sigma} \tilde{\mathbf{n}}$. The normal component of the stress vector should be continuous across the interface. It is verified experimentally in Figure 6.11 and 6.12 for the nearly conforming and fully nonconforming cases respectively. As could be expected, the relation is well satisfied when the meshes are nearly conforming. Nevertheless, the error remains fairly small even in the nonconforming case.

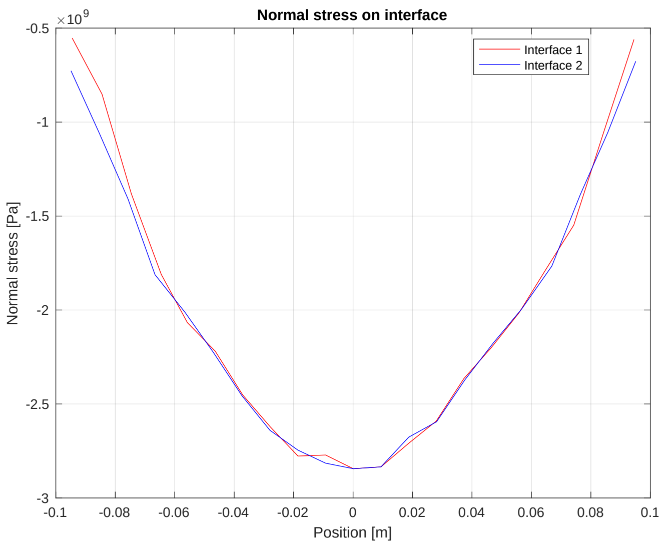

Figure 6.11: Normal component of the stress vector in the nearly conforming case

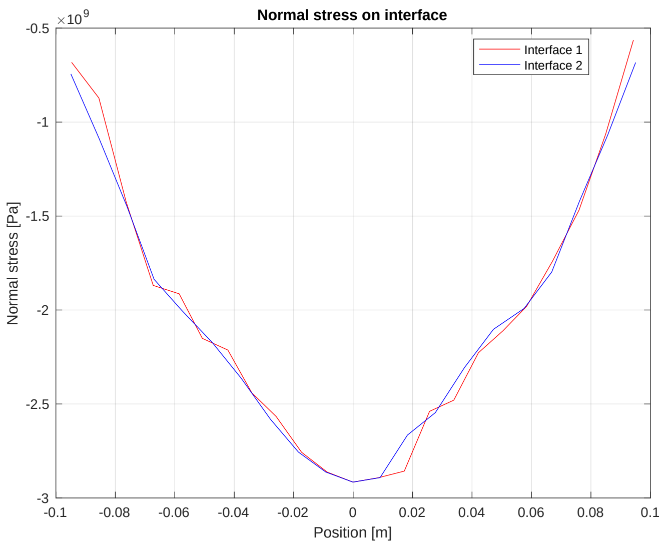

Figure 6.12: Normal component of the stress vector in the fully nonconforming case

These experiments confirm the value of the INTERNODES method for solving problems in contact mechanics.

Conclusion

In this work, we have designed a highly efficient preconditioner for solving the sequence of linear systems arising from the application of the INTERNODES method to linear elastic problems in contact mechanics. The saddle point type structure inherited from the variational problem and the properties of the stiffness matrix naturally led to considering an augmented Lagrangian preconditioner. We have discussed how the preconditioner could be suitably tuned and we have proved properties related to the spectrum of the preconditioned matrix. Taking advantage of the sparsity structure of the matrices, we then proposed an algorithm to efficiently solve linear systems with the preconditioning matrix. Assuming a linear elastic constitutive model, our strategy relies on a unique Cholesky factorization which is computed once and for all. Resorting to a complete factorization allows the implementation of a nearly ideal preconditioner. Numerical experiments confirmed the quality and robustness of the preconditioner. In addition, they revealed that the convergence rate was independent of the mesh size and our preconditioned iterative scheme outperformed state-of-the-art sparse direct solvers.

To avoid excessive memory usage, we have then discussed possible extensions to incomplete factorizations coupled with inner iterations. We have proposed several preconditioners for the inner iterations and we have shown experimentally that incomplete factorizations could still lead to an efficient preconditioner. However, pivot breakdowns were also occasionally experienced during the factorization process. Such breakdowns are well known for applications in structural mechanics and robust incomplete factorizations could provide a possible remedy. Yet, another unfortunate feature common to all incomplete factorizations is the increase of the iteration count with the problem size. Thus, also of interest are other solution techniques which were briefly examined and are listed in the appendix. Some of these Cholesky-free methods seem very promising for large 3D problems. Those problems were not tested in the project and there is still much empirical work left in determining the best strategy. Cholesky-free methods would also certainly be favored in case the stiffness matrix must be reassembled at each iteration of the contact algorithm, for example in case of nonlinear constitutive models.

With effective preconditioning and a good accuracy, the INTERNODES method could compete with more established methods used for problems in contact mechanics. However, much research is still needed before reaching a definite conclusion. The work done in this thesis remains largely in the design phase. In particular, the project raised concerns over the stability of the method in close connection with the radiuses of the radial basis functions. Fortunately, the requirement for diagonal dominance of the interpolation matrices indirectly forces a constraint on the radiuses. This constraint restricts the choice of radiuses and instabilities seem less likely to occur. Future work should investigate a potential connection between diagonal dominance of the interpolation matrices and stability. Secondly, convergence issues might be encountered on very fine meshes. These issues are due to infeasibilities in the update of the contact interface without violation of the diagonal dominance requirement.

Bibliography

Adams, M. F. (2004). Algebraic multigrid methods for constrained linear systems with applications to contact problems in solid mechanics. *Numerical linear algebra with applications*, 11(2-3):141–153.

Bennett, J. M. (1965). Triangular factors of modified matrices. *Numerische Mathematik*, 7(3):217–221.

Benzi, M. (2002). Preconditioning techniques for large linear systems: a survey. *Journal of computational Physics*, 182(2):418–477.

Benzi, M., Golub, G. H., and Liesen, J. (2005). Numerical solution of saddle point problems. *Acta numerica*, 14:1–137.

Benzi, M. and Olshanskii, M. A. (2011). Field-of-values convergence analysis of augmented Lagrangian preconditioners for the linearized Navier-Stokes problem. *SIAM Journal on Numerical Analysis*, 49(2):770–788.

Benzi, M. and Tůma, M. (2003). A robust incomplete factorization preconditioner for positive definite matrices. *Numerical Linear Algebra with Applications*, 10(5-6):385–400.

Benzi, M. and Tuma, M. (1999). A comparative study of sparse approximate inverse preconditioners. *Applied Numerical Mathematics*, 30(2-3):305–340.

Bochev, P. and Lehoucq, R. (2011). Energy principles and finite element methods for pure traction linear elasticity. *Computational Methods in Applied Mathematics*, 11(2):173–191.

Buhmann, M. D. (2000). Radial basis functions. *Acta numerica*, 9:1–38.

Cao, Z.-H. (2008). Augmentation block preconditioners for saddle point-type matrices with singular (1, 1) blocks. *Numerical Linear Algebra with Applications*, 15(6):515–533.

Carr, A., de Sturler, E., and Gugercin, S. (2021). Preconditioning parametrized linear systems. *SIAM Journal on Scientific Computing*, 43(3):A2242–A2267.

Chow, E. and Saad, Y. (1998). Approximate inverse preconditioners via sparse-sparse iterations. *SIAM Journal on Scientific Computing*, 19(3):995–1023.

Davis, T. A. (2006). Direct methods for sparse linear systems.

Deparis, S., Forti, D., Gervasio, P., and Quarteroni, A. (2016). INTERNODES: an accurate interpolation-based method for coupling the Galerkin solutions of PDEs on subdomains featuring non-conforming interfaces. *Computers & Fluids*, 141:22–41.

Deparis, S., Forti, D., and Quarteroni, A. (2014). A rescaled localized radial basis function interpolation on non-cartesian and nonconforming grids. *SIAM Journal on Scientific Computing*, 36(6):A2745–A2762.

Elman, H., Howle, V. E., Shadid, J., Shuttleworth, R., and Tuminaro, R. (2006). Block preconditioners based on approximate commutators. *SIAM Journal on Scientific Computing*, 27(5):1651–1668.

Elman, H., Silvester, D., and Wathen, A. (2014). *Finite Elements and Fast Iterative Solvers: with Applications in Incompressible Fluid Dynamics*. Numerical Mathematics and Scientific Computation. Oxford University Press, Oxford.

Ern, A. (2002). *Eléments finis : théorie, applications, mise en oeuvre*. Mathématiques & applications 36. Springer, Paris.

Gershgorin, S. A. (1931). Über die Abgrenzung der Eigenwerte einer Matrix. *Bulletin de l'Académie des Sciences de l'URSS*, (6):749–754.

Gervasio, P. and Quarteroni, A. (2018). Analysis of the INTERNODES method for non-conforming discretizations of elliptic equations. *Computer Methods in Applied Mechanics and Engineering*, 334:138–166.

Golub, G. H. and Greif, C. (2003). On solving block-structured indefinite linear systems. *SIAM Journal on Scientific Computing*, 24(6):2076–2092.

Greenbaum, A. (1997). *Iterative methods for solving linear systems*. Frontiers in Applied Mathematics. Society for Industrial and Applied Mathematics.

Greenbaum, A., Pták, V., and Strakoš, Z. e. k. (1996). Any nonincreasing convergence curve is possible for gmres. *Siam journal on matrix analysis and applications*, 17(3):465–469.

Grote, M. J. and Huckle, T. (1997). Parallel preconditioning with sparse approximate inverses. *SIAM Journal on Scientific Computing*, 18(3):838–853.

Günther-Hanssen, O. (2020). Finite Element Method INTERNODES for Contact Mechanics A study on condition number and iterative solver performance. *Infoscience EPFL*.

Hiptmair, R. (2006). Operator preconditioning. *Computers & Mathematics with Applications*, 52(5):699–706.

Jin, F., Wan, Q., and Guo, X. (2016). A double-Westergaard model for adhesive contact of a wavy surface. *International Journal of Solids and Structures*, 102:66–76.

Johnson, K. L. (1985). *Contact Mechanics*. Cambridge University Press, New York.

Kikuchi, N. (1988). *Contact problems in elasticity : a study of variational inequalities and finite element methods*. SIAM studies in applied mathematics ; 8. Society for Industrial and Applied Mathematics SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104, Philadelphia, Pa.

Kressner, D. (2020). Computational linear algebra. Lecture notes.

Liu, Q. (2013). New preconditioners for nonsymmetric saddle point systems with singular (1,1) block. *International Scholarly Research Notices*, 2013.

Liu, Q., Morgan, R. B., and Wilcox, W. (2015). Polynomial Preconditioned GMRES and GMRES-DR. *SIAM Journal on Scientific Computing*, 37(5):S407–S428.

Manteuffel, T. A. (1980). An incomplete factorization technique for positive definite linear systems. *Mathematics of computation*, 34(150):473–497.

Meijerink, J. A. and Van Der Vorst, H. A. (1977). An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix. *Mathematics of computation*, 31(137):148–162.

Nash, S. G. and Sofer, A. (1996). Preconditioning reduced matrices. *SIAM Journal on Matrix Analysis and Applications*, 17(1):47–68.

Pearson, J. W. and Pestana, J. (2020). Preconditioners for Krylov subspace methods: An overview. *GAMM-Mitteilungen*, 43(4):e202000015.

Pearson, J. W. and Wathen, A. J. (2012). A new approximation of the Schur complement in preconditioners for PDE-constrained optimization. *Numerical Linear Algebra with Applications*, 19(5):816–829.

Rozložník, M. (2018). *Saddle-point problems and their iterative solution*. Springer.

Saad, Y. (1981). Krylov subspace methods for solving large unsymmetric linear systems. *Mathematics of computation*, 37(155):105–126.

Saad, Y. (2003). *Iterative methods for sparse linear systems*. SIAM.

Saad, Y. and Schultz, M. H. (1986). GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on scientific and statistical computing*, 7(3):856–869.

Sofonea, M. (2012). *Mathematical models in contact mechanics*. London mathematical society lecture note series ; 398. Cambridge University Press, New York.

Stange, P., Griewank, A., and Bollhöfer, M. (2006). On the efficient update of rectangular LU factorizations subject to low rank modifications.

Van der Vorst, H. A. (1992). Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM Journal on scientific and Statistical Computing*, 13(2):631–644.

Varga, R. S. (1976). On diagonal dominance arguments for bounding $\|a^{-1}\|_\infty$. *Linear Algebra and its applications*, 14(3):211–217.

Voet, Y. (2020). Nonlinear finite elements in dynamics. *Infoscience EPFL.*

Wathen, A. J. (1987). Realistic eigenvalue bounds for the Galerkin mass matrix. *IMA Journal of Numerical Analysis*, 7(4):449–457.

Wathen, A. J. (2015). Preconditioning. *Acta Numerica*, 24:329–376.

Wendland, H. (1995). Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in computational Mathematics*, 4(1):389–396.

Wriggers, P. (2006). *Computational Contact Mechanics.* Springer Berlin Heidelberg, Berlin, Heidelberg, 2nd ed. 2006. edition.

Yan, T. (2019). Approximating the inverse of a diagonally dominant matrix with positive elements. *arXiv preprint arXiv:1902.00668.*

Yannakakis, M. (1981). Computing the minimum fill-in is NP-complete. *SIAM Journal on Algebraic Discrete Methods*, 2(1):77–79.

Appendix A

Other attempts for preconditioning the INTERNODES matrix

A.1 General guidelines

The preconditioners analyzed in earlier chapters are among the most successful. However, they were not the first nor the last of a long list of attempts. Below are some of the other attempts for the preconditioning of the INTERNODES matrix. These attempts have been listed here for the record. This section may be skipped entirely. However, it might be useful to anyone wishing to extend or improve the work done in this thesis.

The attempts either consist of approximating directly the INTERNODES matrix or the $(1,1)$ block of the augmented Lagrangian preconditioner. As we have seen, this preconditioner is very efficient and leads to mesh size independent convergence when implemented exactly. Thus, we can hopefully expect small changes to this preconditioner to still yield an effective preconditioner. In fact, it is precisely what we did when adding the small perturbation to the stiffness matrix appearing in the $(1,1)$ block.

We have classified these attempts based on the solution technique used. When building efficient preconditioners, one should bare in mind the following guidelines:

1. The block structure of the matrix must be taken into account.
2. The most critical components of the matrix should be preserved.

As a rule of thumb, one should expect the following behavior: dumping entirely one or more blocks of a matrix may lead to a good preconditioner for coarse meshes. However, the iteration count may rapidly increase for finer meshes. As we have already discussed, such a feature is particularly troublesome when using a nonsymmetric Krylov subspace method such as GMRES since the cost of storage and the number of floating point operations increase at each iteration. Hence, preconditioners which achieve iteration counts independent of the mesh size are extremely precious for dealing with large applications. None of the attempts discussed in the subsequent sections preserve this property exactly. However, iteration counts that increase very slowly with mesh refinement are still very valuable.

A.2 Easy preconditioners

The first and easy preconditioners one may try are based on the approximation of the stiffness matrix by some suitable sparse matrix. As expected, approximating the stiffness matrix based on its diagonal or triangular part led to very poor preconditioners even on coarse meshes. On the other hand, Jacobi or Gauss-Seidel preconditioners based on the entire INTERNODES matrix ignore the block structure and fail for our application. One could try easy fixes in order to guarantee the invertibility of the preconditioner but numerical experiments rapidly concluded such preconditioners were worthless (Günther-Hanssen, 2020).

A.3 Techniques based on matrix factorizations

There are many equivalent ways of factorizing a 2×2 block matrix. Here again, the factorization can be done either directly on a modification of the INTERNODES matrix or on the $(1, 1)$ block of the preconditioner.

A.3.1 Factorization of a modified coefficient matrix

For large applications, the INTERNODES matrix cannot be formed explicitly due to a few dense blocks inside the matrices B and $\tilde{B}$. These blocks are therefore replaced by sparse approximations. These approximations are computed similarly to those used when forming the sparse matrices $\hat{Y}_1$ and $\hat{Y}_2$. Moreover, the stiffness matrix being singular in general, it is replaced with $\tilde{K}$ as we have done for the preconditioner. We will therefore consider the matrix

$$M = \begin{pmatrix} \tilde{K} & \hat{B} \\ \hat{\tilde{B}} & 0 \end{pmatrix}$$

In linear elasticity, only the blocks $\hat{\tilde{B}}$ and $\hat{B}$ are changing during the iterations of the contact algorithm. Hence, it could be interesting to pre-compute an LU or Cholesky factorization of the matrix $\tilde{K}$ and reconstruct a factorization for the matrix M. For obvious computational and storage reasons, one should always favor a Cholesky over an LU factorization of $\tilde{K}$. Assuming $\tilde{K}$ has been factorized as $\tilde{K} = LL^T$, we will denote $S = -\hat{\tilde{B}}\tilde{K}^{-1}\hat{B} = -\hat{\tilde{B}}L^{-T}L^{-1}\hat{B}$ the Schur complement of $\tilde{K}$ in M. Then, it can easily be verified that M admits the equivalent factorizations

$$\begin{aligned} M = \begin{pmatrix} \tilde{K} & \hat{B} \\ \hat{\tilde{B}} & 0 \end{pmatrix} &= \begin{pmatrix} L & 0 \\ \hat{\tilde{B}}L^{-T} & I \end{pmatrix} \begin{pmatrix} L^T & L^{-1}\hat{B} \\ 0 & S \end{pmatrix} \\ &= \begin{pmatrix} \tilde{K} & 0 \\ \hat{\tilde{B}} & I \end{pmatrix} \begin{pmatrix} I & \tilde{K}^{-1}B \\ 0 & S \end{pmatrix} \\ &= \begin{pmatrix} I & 0 \\ \hat{\tilde{B}}\tilde{K}^{-1} & I \end{pmatrix} \begin{pmatrix} \tilde{K} & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} I & \tilde{K}^{-1}B \\ 0 & I \end{pmatrix} \end{aligned}$$

The first factorization is almost an LU factorization for the entire matrix. It would be an LU factorization if we had used an LU factorization of $\tilde{K}$. These factorizations are all equivalent in the sense that:

1. They all require solving two linear systems with the matrix $\tilde{K}$ (or equivalently four triangular systems).
2. They all require solving a linear system with the Schur complement matrix S.

The first point is already a major drawback. Recall that our best solution method only requires solving two triangular systems. Secondly, the Schur complement matrix cannot be formed in general. First of all, computing $\hat{\tilde{B}}L^{-T}$ or $L^{-1}\hat{B}$ becomes expensive when m becomes large and will lead to much fill-in in the solution matrix. Hence, even if $\hat{\tilde{B}}L^{-T}L^{-1}\hat{B}$ can be computed, the resulting matrix will be completely dense in general. It is also unclear how to approximate the Schur complement matrix. Schur complement preconditioning is very challenging in general. A few successful techniques have been developed for applications in fluid dynamics. However, almost all of them failed for our application.

When modifying the blocks B and $\tilde{B}$ in order to build a preconditioner, some attention must be paid to the invertibility of the resulting preconditioner. If the preconditioner has a saddle point structure, the same conditions for the invertibility of the INTERNODES matrix apply to the preconditioner. These conditions were summarized in Theorem 3.1. For instance, the preconditioner

$$M = \begin{pmatrix} K & \hat{B} \\ \tilde{B} & 0 \end{pmatrix}$$

where

$$\hat{B} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ M_2 R_{21} \end{pmatrix}$$

is singular in any of the following cases:

1. $|\mathcal{D}_\Gamma^1| > |\mathcal{D}_\Gamma^2|$
2. The first block of the stiffness matrix is singular

In the first case, $\operatorname{rank}(B) = m_2 < m$ and the first condition of Theorem 3.1 is violated. In the second case, there exists a vector $\mathbf{x} \in \{\ker(K) \cap \ker(B^T)\}$ and the second condition of Theorem 3.1 is violated. Both cases were encountered in experiments.

A.3.2 Factorization of the $(1, 1)$ block of the preconditioner

As we have seen, our preconditioner is highly performing when it is implemented exactly. We have discussed previously the difficulties in solving the linear systems with coefficient matrix $\tilde{K} + \alpha B W^{-1} \tilde{B}$. An idea is to replace again the dense blocks inside the matrix $\alpha B W^{-1} \tilde{B}$ by sparse approximations and compute a factorization for the approximation of $\tilde{K} + \alpha B W^{-1} \tilde{B}$. Let us denote M the resulting approximation. Assuming the matrix is reordered, it is expressed as

$$M = \begin{pmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} - \alpha \hat{Y}_1 \hat{Y}_2^T \end{pmatrix}$$

First idea

An idea is to dump either K_{12} or K_{21} in order to recover a triangular block matrix. Subsequently, linear systems with coefficient matrices K_{11} and $K_{22} - \alpha \hat{Y}_1 \hat{Y}_2^T$ must be solved, both of which can be done very efficiently by computing a sparse Cholesky factorization for K_{11} and a sparse LU factorization for $K_{22} - \alpha \hat{Y}_1 \hat{Y}_2^T$. This preconditioner turned out to be performing quite well for relatively coarse meshes. However, the number of iterations increased rapidly with mesh refinement. This could be expected because we are dumping entirely a block of the matrix M.

Second idea

Instead of dumping blocks, first note that M can be factorized as

$$M = \begin{pmatrix} I & 0 \\ K_{21} K_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} K_{11} & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} I & K_{11}^{-1} K_{12} \\ 0 & I \end{pmatrix}$$

with the Schur complement matrix $S = K_{22} - \alpha \hat{Y}_1 \hat{Y}_2^T - K_{21} K_{11}^{-1} K_{12}$. Thus, solving linear systems with this preconditioner requires solving two linear systems with K_{11} and one smaller linear system with S. Unfortunately, S cannot be formed because $K_{21} K_{11}^{-1} K_{12}$ is expensive to compute and is very dense. Therefore, an idea is to consider instead the preconditioner

$$M = \begin{pmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} - \alpha \hat{Y}_1 \hat{Y}_2^T + K_{21} K_{11}^{-1} K_{12} \end{pmatrix}$$

and notice that the matrix can be factorized as

$$M = \begin{pmatrix} I & 0 \\ K_{21} K_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} K_{11} & 0 \\ 0 & \hat{S} \end{pmatrix} \begin{pmatrix} I & K_{11}^{-1} K_{12} \\ 0 & I \end{pmatrix}$$

where $\hat{S} = K_{22} - \alpha \hat{Y}_1 \hat{Y}_2^T$ is the resulting Schur complement matrix. Clearly, the approximation is motivated by the fact that linear systems with this matrix are much easier to solve for. The advantage of this technique is that we have only introduced a small change in one of the blocks of the matrix. The perturbation $K_{21} K_{11}^{-1} K_{12}$

added to the $(2,2)$ block changes the sparsity pattern of the block. However, we found out experimentally that its entries are much smaller in magnitude in comparison to those of $K_{22} - \alpha\hat{Y}_1\hat{Y}_2^T$. The resulting preconditioner performed very well. However, solving linear systems with this preconditioner requires solving two linear systems with the large matrix K_{11}. Thus, not only is the iteration count greater than for our most successful method but also the cost per iteration. Nevertheless, the preconditioner does not assume the availability of any factorization of the matrix K_{11}. The matrix K_{11} is symmetric positive definite and very efficient solvers such as the conjugate gradient method or algebraic multigrid methods may be used to solve linear systems with this matrix (together with efficient preconditioners). Thus, the preconditioner could become interesting for large 3D applications. Of course, the preconditioner is even better if $K_{21}K_{11}^{-1}K_{12}$ can be approximated in some way. Experimentally, using the simple approximation $K_{21}\hat{K}_{11}^{-1}K_{12}$ with $\hat{K}_{11} = \mathrm{diag}(K_{11})$ already significantly improved the quality of the preconditioner.

A.4 Inner preconditioning

Here, the idea is to use the exact augmented Lagrangian preconditioner but use a few inner iterations to solve linear systems with the coefficient matrix $\tilde{K} + \alpha BW^{-1}\tilde{B}$. These inner solves once again require inner preconditioning.

First idea

An idea is to use the Cholesky factorization of $\tilde{K}$ directly as preconditioner. It does not require updating the preconditioner during the iterations of the contact algorithm. However, the low-rank term is entirely ignored. Thus, the greater the magnitude of the entries of $\alpha BW^{-1}\tilde{B}$, the poorer the quality of the preconditioner. Unfortunately, the magnitude of the low-rank term cannot be controlled because there is already a constraint tied on α. A parametric study confirmed the rapid decrease of the quality of the preconditioner for increasing values of $|\alpha|$. The number of inner iterations was too large for the preconditioner to be of any practical use.

Second idea

Another idea is to precondition the inner iterations with the matrix $\tilde{K} + \alpha BW^{-1}\tilde{B}$. Its inverse can be expressed using the Woodbury matrix identity:

$$(\tilde{K} + \alpha BW^{-1}\tilde{B})^{-1} = [I - \tilde{K}^{-1}B(\alpha^{-1}W + \tilde{B}\tilde{K}^{-1}B)^{-1}\tilde{B}]\tilde{K}^{-1}$$

But we are again facing the same issues encountered with other preconditioners:

1. Two large linear systems with the matrix $\tilde{K}$ must be solved at each iteration.
2. The Schur complement matrix cannot be formed and attempts at approximating it have been unsuccessful.

A.5 Low-rank updating

The techniques in this section can either be used as inner preconditioning for solving linear systems with $K + \alpha BW^{-1}\tilde{B}$ or can directly be used to approximate the $(1,1)$ block of the preconditioner. The second usage is generally cheaper. The solution technique again begins with sparse approximations. More specifically, the matrix $\alpha BW^{-1}\tilde{B}$ is approximated by $-\alpha\hat{U}_1\hat{U}_2^T$ with the matrices

$$\hat{U}_1 = \begin{pmatrix} 0 \\ I_m \\ 0 \\ \hat{Q}_1 \end{pmatrix} \quad \hat{U}_2 = \begin{pmatrix} 0 \\ I_m \\ 0 \\ \hat{Q}_2 \end{pmatrix}$$

Assuming an LU decomposition of $\tilde{K}$ has already been computed, we denote

$$M = \tilde{K} - \alpha\hat{U}_1\hat{U}_2^T = LU - \alpha\hat{U}_1\hat{U}_2^T$$

M is expressed as a sparse low-rank update of the matrix $\tilde{K}$.

First idea

Here, our goal is to find the LU factorization of M by updating the LU factorization of $\tilde{K}$. In fact, there exists algorithms specifically for this purpose. However, these algorithms require an update of each of the columns of L and rows of U. Most of these algorithms deal with rank 1 updates. However, some can be extended to deal with higher rank updates. The algorithm from Bennett (1965) was successfully implemented using the pseudo-code of Stange et al. (2006). Unfortunately, despite several attempts, our implementations were far more expensive than recomputing an entire factorization from scratch. Even though the performance of the implementation could be further improved, it is difficult to imagine it could compete with state-of-the-art implementations of the LU factorization.

Second idea

The second idea assumes the availability of either a Cholesky or an LU decomposition of $\tilde{K}$. However, whenever given the choice, we will naturally opt for a Cholesky decomposition. The strategy is similar to building the modified LU preconditioner for the inner iterations. It stems from noticing that L can be partitioned into $L = (L_1, L_2)$ where L_1 consists of the first leading m columns of L and L_2 of the $n - m$ remaining columns. Then, we obtain

$$M = L_1 L_1^T + L_2 L_2^T - \alpha \hat{U}_1 \hat{U}_2^T$$

If we assume the degrees of freedom are ordered such that those from the interface of body 1 are placed before all the others, then the $m \times m$ identity matrix is contained within the first m rows of $\hat{U}_1$ and $\hat{U}_2$. Approximate factors L and U can be formed by replacing the first m columns of L by $-\alpha\hat{U}_1$ and replacing the first m rows of L^T by the $\hat{U}_2^T$. Hence, we consider the approximation

$$M \approx L_2 L_2^T - \alpha \hat{U}_1 \hat{U}_2^T = (-\alpha \hat{U}_1 \; L_2)(\hat{U}_2 \; L_2)^T = \hat{L}\hat{U}$$

Thanks to the ordering considered, the factors $\hat{L}$ and $\hat{U}$ are invertible. As could be expected, the smaller m is the better the quality of the preconditioner. However, the quality of the sparse approximation to the low-rank term also plays a role. The main advantage of this preconditioner is that it comes for free and it requires solving only two triangular systems at each iteration. Moreover, since the triangular systems are solved sequentially, the factors $\hat{L}$ and $\hat{U}$ must not be stored simultaneously. The preconditioner turned out to perform very well even on finer meshes. The iteration numbers were 18, 37 and 57 for mesh sizes $h = 0.1$, $h = 0.05$ and $h = 0.01$ and a tolerance of 1×10^{-7} for inner preconditioning usage.

In fact, the assumption on the ordering can be slightly relaxed. It is sufficient to ensure that the interface degrees of freedom of body 1 come before those of body 2. However, their relative position with respect to all the other degrees of freedom is irrelevant. In particular, the ordering we have used extensively where the interface degrees of freedom come last satisfy the condition. For this ordering, the preconditioner performed even better. The number of iterations was 15, 21, 28 for the same mesh sizes as before. This finding indicates that the ordering impacts the performance.

The major shortcoming in its performance results from the dumping of m columns of the Cholesky factor L. However, as m is expected to decrease during the iterations of the contact algorithm, we can expect this preconditioner to become even more efficient as the iterations proceed.

A.6 Sparse approximate map

During the iterations of the contact algorithm, the blocks B and $\tilde{B}$ of the INTERNODES matrix are changing. These changes are of course carried over to the preconditioner. In this section, we will make the dependency on the iteration number explicit by writing B_k and $\tilde{B}_k$ for the matrices B and $\tilde{B}$ respectively at iteration k of the contact algorithm. On the other hand, assuming a linear elastic constitutive model, the stiffness matrix does not change. Thus, solving linear systems with the preconditioner requires solving a sequence of linear systems of the type

$$(K + B_k W_k^{-1} \tilde{B}_k)\mathbf{x}_k = \mathbf{b}_k$$

We will denote $X_k = K + B_k W_k^{-1} \tilde{B}_k$. Assuming a good preconditioner, say P_0 has been computed upon initialization for the coefficient matrix X_0, one might reasonably question whether it is possible to recycle this good preconditioner for later linear solves knowing that X_k for $k \geq 1$ will differ from X_0 only in a few rows and columns given the sparsity of $B_k W_k^{-1} \tilde{B}_k$. The answer is yes and is based on research from Carr et al. (2021). The idea is to map the current matrix X_k for $k \geq 1$ to X_0 (for which there exists a good preconditioner). The new preconditioner P_k is then constructed by applying the map to the old preconditioner. More specifically, let us consider a right preconditioner such that $X_0 P_0$ yields fast convergence. Note that

$$X_0 P_0 = X_0 N_k^{-1} N_k P_0$$

and we set $X_k = X_0 N_k^{-1}$ or equivalently $X_k N_k = X_0$. This requirement alone defines the map N_k. Applying N_k to P_0 defines the new preconditioner P_k as $P_k = N_k P_0$. Iterative solvers such as GMRES only require matrix-vector multiplies. Hence, P_k is never formed explicitly but applying P_k now requires two successive matrix-vector multiplications. In practice, the map N_k is not computed exactly but approximately such that $X_k N_k \approx X_0$. In order to induce computational savings, the approximate map $\hat{N}_k$ is a sparse approximation of N_k. For this reason, the authors called their method the Sparse Approximate Map (SAM). Given a subspace of sparse matrices $\mathcal{S} \subseteq \mathbb{R}^{n \times n}$, the method naturally leads to considering the following minimization problem

$$\min_{\hat{N}_k \in \mathcal{S}} \|X_k \hat{N}_k - X_0\|_F^2$$

Note that if the jth column of X_k and X_0 are the same, then the jth column of $\hat{N}_k$ is simply $\mathbf{e}_j$, the jth vector of the canonical basis of $\mathbb{R}^n$. The problem we are facing is very similar to computing sparse approximate inverses. The only difference is that we are solving the least squares problems with a right-hand side vector $\mathbf{x}_{0,j}$ instead of the vector $\mathbf{e}_j$. Consequently, the SPAI algorithm can be extended straightforwardly. Since the matrices X_k and X_0 only differ in a few columns, the algorithm does not iterate over all columns but only over those that have changed. In other words, only the columns associated to degrees of freedom in the active set must be processed. This feature leads to a tremendous reduction of the computational expense. The sparse map $\hat{N}_k$ essentially consists in columns of the identity matrix expect for those associated to degrees of freedom in the active set.

However, contrary to the applications considered by Carr et al., the sparsity pattern of the matrices X_k is changing in an unpredictable way. Hence, we once again used adaptive strategies to capture automatically the sparsity pattern. Such an operation is expensive but necessary, otherwise the quality of the resulting approximate map is very poor.

The method was implemented and tested on our application. The initial preconditioner was an LU factorization of X_0. The matrix was kept small such that X_0 could be formed explicitly. This was for testing purposes only. For larger problems, we would consider an incomplete factorization of a sparse approximation of X_0.

Experimentally, we found that the map updating led to very good results. The number of GMRES iterations when using the updated preconditioner P_k grew very slowly with the iterations of the contact algorithm. However, the computation of the map was fairly expensive due to the changing sparsity pattern. Even though our implementation of the SAM algorithm could certainly be improved, we believe recomputing an entire factorization from scratch would still be a lot cheaper for our application. Therefore, this approach was abandoned.

A.7 The null space method

We end this list of attempts with another very promising method called the null space method. It is a well-established method for the solution of saddle point systems. It belongs to the class of segregated methods which compute the solution vectors $\mathbf{u}$ and $\boldsymbol{\lambda}$ one after the other contrary to coupled methods, which compute both vectors simultaneously. Direct or iterative methods may be used to solve the linear systems arising in the method. For our large scale applications, iterative methods will naturally be used. The method is usually presented in the literature for symmetric saddle point systems. However, its extension to the nonsymmetric case is straightforward.

Let us reconsider the original saddle point system

$$\begin{pmatrix} K & B \\ \tilde{B} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{d} \end{pmatrix} \tag{A.1}$$

From the equation $\tilde{B}\mathbf{u} = \mathbf{d}$, the solution vector $\mathbf{u}$ can be expressed as $\mathbf{u} = \hat{\mathbf{u}} + V_{\tilde{B}}\mathbf{y}$ where $\hat{\mathbf{u}}$ is a particular solution, $V_{\tilde{B}}$ is a basis for the kernel (or null space) of $\tilde{B}$ and $\mathbf{y}$ is some vector of $\mathbb{R}^{n-m}$. Substituting back in the first equation, we obtain $K(\hat{\mathbf{u}} + V_{\tilde{B}}\mathbf{y}) = \mathbf{f} - B\boldsymbol{\lambda}$. Let V_B be a basis for the kernel of B^T. Then, pre-multiplying the equation by V_B^T, moving known terms to the right-hand side and noticing that $V_B^T B = (B^T V_B)^T = 0$, we finally obtain

$$V_B^T K V_{\tilde{B}} \mathbf{y} = V_B^T(\mathbf{f} - K\hat{\mathbf{u}})$$

Recall from the invertibility conditions of Theorem 3.1 that $V_B^T K V_{\tilde{B}}$ is invertible. Solving this linear system yields the vector $\mathbf{y}$ from which we immediately deduce $\mathbf{u}$. The solution vector $\boldsymbol{\lambda}$ can be computed from the least squares problem

$$B\boldsymbol{\lambda} = \mathbf{f} - K\mathbf{u}$$

or equivalently from the normal equations

$$B^T B\boldsymbol{\lambda} = B^T(\mathbf{f} - K\mathbf{u})$$

This small linear system can be solved very easily with a direct method if the size of the problem is small enough. For large problems, iterative methods must again be used. The coefficient matrix is symmetric positive definite and in such circumstances, the conjugate gradient method is the method of choice. It could be preconditioned with the sparse approximation $\hat{B}^T\hat{B}$ constructed using once again sparse approximate inverse techniques.

In our case, recalling that $\mathbf{d} = R_{12}\mathbf{r}_{\Gamma_2} - \mathbf{r}_{\Gamma_1}$, an obvious particular solution of $\tilde{B}\mathbf{u} = \mathbf{d}$ is

$$\hat{\mathbf{u}} = \begin{pmatrix} 0 \\ -\mathbf{r}_{\Gamma_1} \\ 0 \\ -\mathbf{r}_{\Gamma_2} \end{pmatrix}$$

Usually, a major difficulty in applying the method consists in finding bases V_B and $V_{\tilde{B}}$ for the kernels of B^T and $\tilde{B}$ respectively. A great body of literature has been dedicated to this problem. However, for our problem such bases can be immediately deduced from the very simple structure of the matrices B^T and $\tilde{B}$ which are expressed as

$$\begin{aligned} B^T &= \begin{pmatrix} 0 & -M_1 & 0 & R_{21}^T M_2 \end{pmatrix} \\ \tilde{B} &= \begin{pmatrix} 0 & I & 0 & -R_{12} \end{pmatrix} \end{aligned}$$

Thanks to their great sparsity, a basis for their kernels can be constructed straightforwardly. Denoting $q_k = |\mathcal{D}_\Omega^k|$ and $m_k = |\mathcal{D}_\Gamma^k|$ for $k = 1, 2$, we have

$$V_B = \begin{pmatrix} I_{q_1} & 0 & 0 \\ 0 & 0 & M_1^{-1}R_{21}^T M_2 \\ 0 & I_{q_2} & 0 \\ 0 & 0 & I_{m_2} \end{pmatrix} \quad V_{\tilde{B}} = \begin{pmatrix} I_{q_1} & 0 & 0 \\ 0 & 0 & R_{12} \\ 0 & I_{q_2} & 0 \\ 0 & 0 & I_{m_2} \end{pmatrix}$$

Both bases have dimension $n - m$ (with $m = m_1$) but are not orthonormal bases. In fact, such bases are called fundamental bases. For a conforming mesh, $M_1 = M_2 = M$ and $R_{12} = R_{21} = I$ such that $\ker(B^T) = \ker(\tilde{B})$.

After a suitable permutation of the rows and columns we obtain the nice looking structure

$$PV_B Q = \begin{pmatrix} I_{q_1} & 0 & 0 \\ 0 & I_{q_2} & 0 \\ 0 & 0 & I_{m_2} \\ 0 & 0 & Q_B \end{pmatrix} \quad PV_{\tilde{B}} Q = \begin{pmatrix} I_{q_1} & 0 & 0 \\ 0 & I_{q_2} & 0 \\ 0 & 0 & I_{m_2} \\ 0 & 0 & Q_{\tilde{B}} \end{pmatrix}$$

with $Q_B = M_1^{-1}R_{21}^T M_2$ and $Q_{\tilde{B}} = R_{12}$ and P and Q are suitable permutation matrices. As usual, the matrices Q_B and $Q_{\tilde{B}}$ are very dense and will never be formed explicitly.

Therefore, when solving linear systems with coefficient matrix $V_B^T K V_{\tilde{B}}$, we must use iterative methods which avoid explicitly assembling the bases matrices V_B and $V_{\tilde{B}}$. Interestingly, numerical experiments indicated that the spectral condition numbers of $V_B^T K V_{\tilde{B}}$ and K were very similar. This can be expected provided the interpolation is good enough. Some insight can be drawn from a bound on the spectral norm of the matrix. Indeed,

$$\|V_B^T K V_{\tilde{B}}\|_2 \le \|K\|_2 \|V_B\|_2 \|V_{\tilde{B}}\|_2 = \lambda_1(K)\sqrt{(1 + \|Q_B\|_2^2)(1 + \|Q_{\tilde{B}}\|_2^2)}$$

Nevertheless, GMRES still converged relatively slowly. Therefore, solving such linear systems will in general require preconditioning. For this purpose, it is interesting to note that the basis matrices V_B and $V_{\tilde{B}}$ can be exceedingly well approximated by sparse matrices $\hat{V}_B$ and $\hat{V}_{\tilde{B}}$ using once again sparse approximate inverse techniques. The matrix $\hat{V}_B^T K \hat{V}_{\tilde{B}}$ can be formed explicitly and was found to be a very good preconditioner. However, solving linear systems with this matrix is not trivial. The size of the matrix is changing at each iteration of the contact algorithm and recomputing a factorization will be too expensive.

Of course, the structure of $V_B^T K V_{\tilde{B}}$ could be used to build preconditioners. Assuming, the matrices have been reordered, the product can be expressed as

$$V_B^T K V_{\tilde{B}} = \begin{pmatrix} I & 0 \\ 0 & X_B^T \end{pmatrix} \begin{pmatrix} K_{\Omega\Omega} & K_{\Omega\Gamma} \\ K_{\Gamma\Omega} & K_{\Gamma\Gamma} \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & X_{\tilde{B}} \end{pmatrix} = \begin{pmatrix} K_{\Omega\Omega} & K_{\Omega\Gamma} X_{\tilde{B}} \\ X_B^T K_{\Gamma\Omega} & X_B^T K_{\Gamma\Gamma} X_{\tilde{B}} \end{pmatrix}$$

with

$$X_B = \begin{pmatrix} I_{m_2} \\ Q_B \end{pmatrix} \quad X_{\tilde{B}} = \begin{pmatrix} I_{m_2} \\ Q_{\tilde{B}} \end{pmatrix}$$

Preconditioners for nonsymmetric saddle point systems could then be used. However, one of the major drawbacks of this approach is linked to the change of size of the bases between each iteration of the contact algorithm. Thus, the size of $K_{\Omega\Omega}$ and all other matrices changes. The stiffness matrix would have to be repartitioned at each iteration of the contact algorithm and once again matrix factorizations would be too costly for solutions to linear systems with coefficient matrix $K_{\Omega\Omega}$.

Thus, preconditioning strategies which are more "black box" must be sought. Nash and Sofer (1996) proposed several preconditioners for reduced symmetric positive definite matrices of the form $Z^T K Z$. Such linear systems frequently arise in applications in optimization. One of their simplest preconditioners was based on the approximation $(Z^T K Z)^{-1} \approx W^T \tilde{K}^{-1} W$ where W^T is a left inverse for Z such that $W^T Z = I$ and $\tilde{K}$ is a symmetric positive definite approximation of K. We assume in addition W is such that $WZ^T \approx I$. Obviously, equality is impossible because WZ^T is low rank. Then,

$$\begin{aligned} W^T \tilde{K}^{-1} W Z^T K Z &\approx W^T \tilde{K}^{-1} K Z \\ &\approx W^T Z \\ &= I_{n-m} \end{aligned}$$

Finding a suitable matrix W satisfying the first requirement is straightforward. For instance choosing $W^T = (Z^T Z)^{-1} Z^T$ will always satisfy $W^T Z = I$. However, we will consider an even simpler choice. Even though the matrix $V_B^T K V_{\tilde{B}}$ arising in our application is nonsymmetric, we will still consider a symmetric positive definite preconditioner. Using a different partitioning for the reordered bases V_B and $V_{\tilde{B}}$, they can be expressed as

$$V_B = \begin{pmatrix} I \\ R_B \end{pmatrix} \quad V_{\tilde{B}} = \begin{pmatrix} I \\ R_{\tilde{B}} \end{pmatrix}$$

with

$$R_B = \begin{pmatrix} 0 & 0 & Q_B \end{pmatrix} \quad R_{\tilde{B}} = \begin{pmatrix} 0 & 0 & Q_{\tilde{B}} \end{pmatrix}$$

For the matrix W, we will consider the very simple choice

$$W = \begin{pmatrix} I \\ 0 \end{pmatrix}$$

Note that W^T is a left inverse for both V_B and $V_{\tilde{B}}$ ($W^T V_B = W^T V_{\tilde{B}} = I$). Alternatively, we could consider the nonsymmetric approximation $(V_B^T K V_{\tilde{B}})^{-1} \approx W_1^T \tilde{K}^{-1} W_2$ with

$$\begin{aligned} W_1 &= V_{\tilde{B}} (V_{\tilde{B}}^T V_{\tilde{B}})^{-1} \\ W_2 &= V_B (V_B^T V_B)^{-1} \end{aligned}$$

which leads to the approximation

$$(V_B^T K V_{\tilde{B}})^{-1} \approx (V_{\tilde{B}}^T V_{\tilde{B}})^{-1} V_{\tilde{B}}^T K^{-1} V_B (V_B^T V_B)^{-1}$$

In fact, this preconditioner has been proposed by Elman et al. (2006) in the context of Schur complement preconditioning. Such strategies are used for approximating a Schur complement matrix $\tilde{B} G^{-1} B$ or its inverse where G is some square invertible matrix. As we have seen, such matrices frequently appear in block preconditioners.

In fact, our much simpler preconditioner performed far better. Moreover, applying the preconditioner is very cheap. Matrix-vector multiplies with W and W^T almost come for free. Only one large linear system must be solved for $\tilde{K}$. If K is invertible, then obviously $\tilde{K} = K$. If K is singular, we would advise different strategies depending on the size of the problem.

- For medium sized problems, linear systems with $\tilde{K}$ can be most efficiently solved by setting $\tilde{K} = K + \epsilon I_n$ with $\epsilon > 0$ small and pre-computing a Cholesky factorization of $\tilde{K}$ at the beginning of the contact algorithm. This strategy is similar to the one used for our most efficient solution technique.
- For large sized problems, $\tilde{K}$ can be any efficient preconditioner which has already been developed for elasticity problems.

We must warn against solving iteratively linear systems with the matrix $\tilde{K} = K + \epsilon I_n$ in the event of a singular stiffness matrix. Indeed, if K is singular $\tilde{K}$ is invertible but very badly conditioned for ϵ small. This reason being that $\tilde{K} = K + \epsilon I_n$ is simply a shifted matrix. Its eigenvalues are given by $\lambda(\tilde{K}) = \lambda(K) + \epsilon$. From now let us distinguish two cases:

- Assume K is invertible, then
$$\kappa(\tilde{K}) = \frac{\lambda_1 + \epsilon}{\lambda_n + \epsilon} \approx \frac{\lambda_1}{\lambda_n}$$
provided $\lambda_n >> \epsilon$. Going for iterative methods in this case would probably be fine.
- Assume K is singular, then
$$\kappa(\tilde{K}) = \frac{\lambda_1 + \epsilon}{\epsilon}$$
The condition number is extremely large for $\epsilon = 10^{-8}$, the smallest eigenvalue being ϵ away from zero. Therefore, an iterative method such as conjugate gradient will surely converge very slowly.

Some numerical testing was done with the preconditioner $M = W^T \tilde{K}^{-1} W$ for an invertible stiffness matrix. The right-hand side vector was the vector of all ones and the initial guess was zero. Although extremely simple, the preconditioner performed remarkably well. Right preconditioned GMRES converged in 18, 27, 35 and 39 iterations for mesh sizes $h = 0.1$, $h = 0.05$, $h = 0.01$ and $h = 0.005$ respectively. The restart was fixed to 40 iterations. A smaller restart of 20 slowed down convergence substantially. GMRES made very little progress in the first few iterations before progressing rapidly. The very slow increase in the iteration count is very promising for large scale applications when methods based on Cholesky factorizations are no longer affordable.