A cheap preconditioner for thermoelastic problems in IGA

Joaquin Eduardo Cornejo Fuentes, David Dureisseix, Arnaud Duval, Thomas Elguedj

CSMA 2024
16ème Colloque National en Calcul des Structures
13-17 Mai 2024, Presqu'île de Giens (Var)

A cheap preconditioner for thermoelastic problems in IGA

J. Cornejo Fuentes[1], A. Duval[1], D. Dureisseix[1], T. Elguedj[1]

[1] *Univ Lyon, INSA Lyon, CNRS, LaMCoS, UMR5259, {joaquin.cornejo-fuentes,arnaud.duval,david.dureisseix,thomas.elguedj}@insa-lyon.fr*

Abstract — This paper employs IsoGeometric Analysis to address both heat conduction and elasticity problems. The main goal of this study is to introduce a preconditioning approach to improve the convergence rate of iterative solver. This method is based on Fast Diagonalization method which exploits the tensor structure of basis functions. We demonstrate that our preconditioner offers efficiency, cost–effectiveness and simplicity in implementation. Furthermore, it exhibits robustness concerning polynomial degree and mesh size parameters. Numerical experiments, which support our theoretical results, are presented.
Mots clés — IsoGeometric analysis, Fast Diagonalization, easy-to-code.

1 Research context

A few decades ago, IsoGeometric Analysis (IGA) was introduced by Hughes et al.[8] as a numerical method for solving partial differential equations (PDEs). This approach extends the conventional Finite Element Method by employing the basis functions from Computer-Aided Design to represent both the geometry and the solution field. Over the last decade, its efficiency has been demonstrated in structural dynamics, large deformation plasticity and other fields (see [6, 9]). However, numerical simulations, especially using IGA, may sometimes require many hours of computation or high–performance tools to handle more complex models. In recent years, some techniques like Weighted Quadrature, Matrix Free and Fast Diagonalization have been developed to reduce CPU time and memory while maintaining quality of the results (see [4]). The purpose of this paper is to introduce our preconditioner, based on Fast Diagonalization method, to address both heat conduction and elasticity problems when using an iterative solver. We demonstrate theoretically and numerically its efficiency, cost–effectiveness and simplicity.

2 Preliminaries

2.1 B–splines

Given the integers $p \geq 0$ and $m > 0$, an open knot–vector in the domain $[0, 1]$ is defined as $\Xi = \{\xi_1, \ldots, \xi_{m+p+1}\}$, with $0 = \xi_1 = \ldots = \xi_{p+1} < \xi_{p+2} \leq \ldots \leq \xi_m < \xi_{m+1} = \ldots = \xi_{m+p+1} = 1$, where knots can be repeated up to multiplicity p [5]. We restrict our study to the case where all knots have multiplicity 1 except the first and last knots. Then, from the knot–vector Ξ, we denote with $\hat{b}_{A,p}$, $A = 1, \ldots, m$, the univariate B–splines basis functions of degree p that are computed recursively through Cox–De Boor formulas [5]. It is important to remark that B–splines have important mathematical properties which are useful in design and analysis: they are nonnegative, form a partition of unity and have local support [5]. Moreover, multivariate B–splines are defined as the tensor product of univariate B–splines. In this case, the parametric domain is defined as $\hat{\Omega} = [0, 1]^d$, where d is the number of space dimensions. Let $\Xi_l = \{\xi_{l,1}, \ldots, \xi_{l,m+p+1}\}$, for $l = 1, \ldots, d$, be the knot–vectors following the l–dimension, then the multivariate B–spline functions are defined as $\hat{N}_{\boldsymbol{A},\boldsymbol{p}}(\boldsymbol{\xi}) = \hat{b}_{A_1,p_1}(\xi_1) \ldots \hat{b}_{A_d,p_d}(\xi_d)$, where the multi–indices $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_d)$, $\boldsymbol{A} = (A_1, \ldots, A_d)$ and $\boldsymbol{p} = (p_1, \ldots, p_d)$ are used. Multivariate B–splines inherit the same properties as univariate B–splines. In the problems we intend to solve, we consider a physical domain Ω, a finite region of $\mathbb{R}^d$, with a piecewise smooth boundary $\partial\Omega$. In IGA, the physical domain Ω is given by a spline parameterization. We restrict our work to the case when Ω is given by a single–patch spline parameterization, *i.e.*, we assume that there exist a conforming parameterization F that maps each point $\boldsymbol{\xi} \in \hat{\Omega}$ to a point $\boldsymbol{x} \in \Omega$. We also consider that F is a smooth mapping such that the Jacobian matrix

of F, defined as $J_F = \left[\partial x_i / \partial \xi_j\right]$, and its inverse are unambiguous in Ω. Then,

$$\boldsymbol{x} = F(\boldsymbol{\xi}) = \sum_{A \in \eta} \boldsymbol{C}_A \hat{N}_A(\boldsymbol{\xi}), \quad \boldsymbol{C}_A \in \mathbb{R}^d, \tag{1}$$

where $\boldsymbol{C}_A$ are the control points and $\hat{N}_A$ are the multivariate B–spline basis functions. Hereafter, the multi–index $\boldsymbol{A} = (A_1, \ldots, A_d)$ is identified by the scalar index $A = A_1 + \sum_{k=2}^{d} \left[(A_k - 1) \prod_{l=1}^{k-1} n_l\right]$ where n_l is the number of control points in the l–dimension. Furthermore, $\eta = 1, \ldots, n_{cp}$ denotes the set of control points numbers and n_{cp} is the total number of control points, *i.e.*, $n_{cp} = \prod_{l=1}^{d} n_l$.

2.2 Heat conduction problem

In the considered heat conduction problem, it is assumed that the material has an anisotropic homogeneous thermal conductivity $\boldsymbol{k}$ and the internal heat generated by the body is described by the given function $f : \Omega \to \mathbb{R}$. Moreover, we consider that on $\partial\Omega$ a homogeneous Dirichlet boundary condition is imposed. Finally, the strong form of the problem, as written in [7], is:

$$(S) = \begin{cases} \text{Find } T : \Omega \to \mathbb{R}, \text{ such that:} & \\ \nabla \cdot (\boldsymbol{k} \nabla T) + f = 0 & \text{in } \Omega; \\ T = 0 & \text{on } \partial\Omega. \end{cases} \tag{2}$$

In this study case, we define the function space $\mathcal{V} = \{w \,|\, w \in H^1(\Omega),\ w = 0 \text{ on } \partial\Omega\}$. Then, the corresponding weak form of Eq. (2) is given by:

$$(W) = \begin{cases} \text{Find } T \in \mathcal{V}, \text{ such that for every } w \in \mathcal{V}: & \\ \displaystyle\sum_{i=1}^{d} \sum_{j=1}^{d} \int_{\Omega} \frac{\partial T(\boldsymbol{x})}{\partial x_i} k_{ij} \frac{\partial w(\boldsymbol{x})}{\partial x_j} d\Omega = \int_{\Omega} f(\boldsymbol{x})\, w(\boldsymbol{x}) d\Omega. & \end{cases} \tag{3}$$

Moreover, we may construct the finite–dimensional approximation of $\mathcal{V}$, denoted $\mathcal{V}^h$, as $\mathcal{V}^h = \left\{ w^h | w^h = \sum_{A \in \eta} N_A(\boldsymbol{x}) w_A, \text{ with } w_A = 0 \text{ for all } A \in \eta_g. \right\}$. Here $N_A(\boldsymbol{x}) = \hat{N}_A \circ F^{-1}(\boldsymbol{x})$ and η_g denotes the subset of η that contains those control points on $\partial\Omega$. The number of control points in $\eta - \eta_g$ is $n_{dof} = \prod_{l=1}^{d} n_{dof,l}$. Then, we may approximate the temperature field by $T^h = \sum_{A \in \eta - \eta_g} N_A(\boldsymbol{x}) d_A$ where the non-trivial scalar values d_A are computed from the matrix formulation:

$$\mathsf{K}\boldsymbol{d} = \mathsf{F}, \text{ with } \boldsymbol{d} \in \mathbb{R}^{n_{dof}}.$$

Here the conductivity matrix K and the internal heat flux vector F are defined as follows:

$$[\mathsf{K}]_{AB} = \sum_{l=1}^{d} \sum_{m=1}^{d} \int_{\hat{\Omega}} \frac{\partial}{\partial \xi_l} \hat{N}_A(\boldsymbol{\xi}) D_{lm}(\boldsymbol{\xi}) \frac{\partial}{\partial \xi_m} \hat{N}_B(\boldsymbol{\xi}) d\hat{\Omega},$$
$$\text{with } D(\boldsymbol{\xi}) = J_F^{-1}(\boldsymbol{\xi}) \left[\boldsymbol{k} \det J_F(\boldsymbol{\xi})\right] J_F^{-\top}(\boldsymbol{\xi}); \tag{4a}$$

$$[\mathsf{F}]_A = \int_{\hat{\Omega}} \hat{N}_A(\boldsymbol{\xi}) \hat{f}(\boldsymbol{\xi}) \det J_F(\boldsymbol{\xi}) d\hat{\Omega}. \tag{4b}$$

2.3 Elasticity problem

Let $\boldsymbol{\sigma} \in S$ denote the Cauchy stress tensor, let $\boldsymbol{u} \in \mathbb{R}^d$ denote the displacement and let $\boldsymbol{\varepsilon} \in S$ denote the small strain tensor. Here $S = \mathbb{R}^{d \times d}_{sym}$ is the space of all symmetric second order tensors. Let us assume that the mentioned variables depend only on the spatial variable $\boldsymbol{x} \in \Omega$. Moreover, the prescribed body force per unit volume is described by the given function $\boldsymbol{f}$ and a homogeneous Dirichlet boundary condition is imposed over $\partial\Omega$. Then, the strong form of the problem is [7]:

$$(S) = \begin{cases} \text{Find } \boldsymbol{u} : \Omega \to \mathbb{R}, \text{ such that:} & \\ \nabla \cdot \boldsymbol{\sigma} + \boldsymbol{f} = \boldsymbol{0} & \text{in } \Omega; \\ \boldsymbol{u} = \boldsymbol{0} & \text{on } \partial\Omega; \\ \text{where } \boldsymbol{\sigma} = \lambda \operatorname{tr}(\boldsymbol{\varepsilon}) \mathbb{1} + 2\mu\boldsymbol{\varepsilon} \text{ and} & \boldsymbol{\varepsilon} = \nabla^s \boldsymbol{u} = \frac{1}{2}\left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^{\top}\right). \end{cases} \tag{5}$$

Here, in the constitutive equation, we assume that $\boldsymbol{\sigma}$ follows the Hooke's law for an isotropic material. The Lamé parameters λ and μ are computed from the Young's modulus E and Poisson's ratio ν.

In this study case, we define the function space $\mathcal{V} = \{\boldsymbol{w} \,|\, \boldsymbol{w} \in H^1(\Omega),\ \boldsymbol{w} = 0 \text{ on } \partial\Omega\}$. Then, the corresponding weak form of Eq. (5) is given by:

$$(W) = \begin{cases} \text{Find } \boldsymbol{u} \in \mathcal{V}, \text{ such that for every } \boldsymbol{w} \in \mathcal{V}: \\ \displaystyle\int_\Omega \boldsymbol{\sigma}(\boldsymbol{u}) \cdot \boldsymbol{\varepsilon}(\boldsymbol{w}) d\Omega = \int_\Omega \boldsymbol{f}(\boldsymbol{x}) \cdot \boldsymbol{w}(\boldsymbol{x}) d\Omega. \end{cases} \tag{6}$$

As in the previous paragraph, we may approximate the displacement field by $\boldsymbol{u}^h = \displaystyle\sum_{A \in \eta - \eta_g} N_A(\boldsymbol{x}) \boldsymbol{d}_A$ where the nontrivial vector values $\boldsymbol{d}_A$ are computed from the matrix formulation:

$$\mathsf{S}\boldsymbol{d} = \mathsf{F}, \text{ with } \boldsymbol{d} \in \mathbb{R}^{n_{dof} d}.$$

Here the stiffness S matrix and the external force vector F are defined as follows:

$$[\mathsf{S}]_{AB} = \int_\Omega \left(\lambda \nabla N_A(\boldsymbol{x}) \otimes \nabla N_B(\boldsymbol{x}) + \mu \nabla N_B(\boldsymbol{x}) \otimes \nabla N_A(\boldsymbol{x}) + \mu (\nabla N_A(\boldsymbol{x}) \cdot \nabla N_B(\boldsymbol{x}))\, \mathbb{1}\right) d\Omega, \tag{7a}$$

$$[\mathsf{F}]_A = \int_{\hat{\Omega}} \hat{N}_A(\boldsymbol{\xi}) \hat{\boldsymbol{f}}(\boldsymbol{\xi}) \det J_F(\boldsymbol{\xi}) d\hat{\Omega}. \tag{7b}$$

Adopting a tensor notation, as in [11], the stiffness matrix may be rewritten and partitioned as:

$$\mathsf{S} = \begin{bmatrix} \mathsf{S}^{(1,1)} & \dots & \mathsf{S}^{(1,d)} \\ \vdots & \ddots & \vdots \\ \mathsf{S}^{(d,1)} & \dots & \mathsf{S}^{(d,d)} \end{bmatrix} \text{ with } \left[\mathsf{S}^{(i,j)}\right]_{AB} = \sum_{l=1}^{d} \sum_{m=1}^{d} \int_{\hat{\Omega}} \frac{\partial}{\partial \xi_l} \hat{N}_A(\boldsymbol{\xi}) D^{(i,j)}_{lm}(\boldsymbol{\xi}) \frac{\partial}{\partial \xi_m} \hat{N}_B(\boldsymbol{\xi}) d\hat{\Omega}, \tag{8}$$

where $D^{(i,j)}(\boldsymbol{\xi}) = J_F^{-1}(\boldsymbol{\xi}) \left[\boldsymbol{k}^{(i,j)} \det J_F(\boldsymbol{\xi})\right] J_F^{-\top}(\boldsymbol{\xi})$. In this way, the stiffness matrix is composed by conductivity–like matrices where the terms of the tensor $\boldsymbol{k}^{(i,j)}$ are computed using the Kronecker delta δ function and the following relation:

$$\left[\boldsymbol{k}^{(i,j)}\right]_{lm} = \lambda \delta_{il} \delta_{jm} + \mu \left(\delta_{im} \delta_{jl} + \delta_{ij} \delta_{lm}\right).$$

3 Fast Diagonalization method

In order to increase the convergence rate of the iterative algorithm, it is necessary to find an efficient preconditioner for the linear system $\mathsf{A}\boldsymbol{x} = \boldsymbol{b}$, where A could be the conductivity or stiffness matrix. The Fast Diagonalization method (FD), which was first used as a technique for solving elliptic PDEs in the finite difference method, takes advantage of the tensor structure of the matrices and allows to efficiently find an approximation of their inverse [12]. To illustrate the idea behind this method, we will first explain how to construct the preconditioner in a heat conduction case where the mapping and material properties are considered to be equal to the identity.

3.1 FD for the heat conduction problem

3.1.1 The classic approach

A simple preconditioner matrix P for the matrix K is given by:

$$[\mathsf{P}]_{AB} = \sum_{l=1}^{d} \int_{\hat{\Omega}} \frac{\partial}{\partial \xi_l} \hat{N}_A(\boldsymbol{\xi}) \frac{\partial}{\partial \xi_l} \hat{N}_B(\boldsymbol{\xi}) d\hat{\Omega}, \text{ for } A, B \in \eta - \eta_g; \tag{9}$$

that is obtained by considering a mapping F equal to the identity map and the conductivity tensor $\boldsymbol{k}$ equal to the identity tensor. Exploring the tensor–product structure of the basis functions, for the case $d = 3$, Eq. (9) may be written as:

$$\mathsf{P} = M_3 \otimes M_2 \otimes K_1 + M_3 \otimes K_2 \otimes M_1 + K_3 \otimes M_2 \otimes M_1.$$

Here, K_l are the univariate stiffness matrices while M_l are the univariate mass matrices. Both matrices are the same size of $n_{dof,l} \times n_{dof,l}$. These univariate matrices are defined as

$$[K_l]_{A_l B_l} = \int_0^1 \hat{b}'_{A_l}(\xi_l)\hat{b}'_{B_l}(\xi_l)d\xi_l, \qquad [M_l]_{A_l B_l} = \int_0^1 \hat{b}_{A_l}(\xi_l)\hat{b}_{B_l}(\xi_l)d\xi_l.$$

In order to find the inverse of P, the first step is to compute the generalized eigendecomposition of the matrix pencils (K_l, M_l), *i.e.*, we may write

$$K_l U_l = M_l U_l \Lambda_l,$$

where U_l is a matrix containing the generalized eigenvectors and Λ_l is a diagonal matrix containing the corresponding eigenvalues. Since M_l is a symmetric and positive definite matrix, the eigenvectors are M_l–orthogonal, *i.e.*, $U_l^\top M_l U_l = \mathbb{1}_l$, where $\mathbb{1}_l$ is the identity matrix. As a consequence, we have the following factorization

$$M_l = U_l^{-\top} U_l^{-1}, \qquad K_l = U_l^{-\top} \Lambda_l U_l^{-1}.$$

Finally, we may express the inverse of P as

$$\begin{aligned} &\mathsf{P}^{-1} = (U_3 \otimes U_2 \otimes U_1)\mathsf{Q}^{-1}(U_3 \otimes U_2 \otimes U_1)^\top, \text{ where} \\ &\quad \mathsf{Q} = \mathbb{1}_3 \otimes \mathbb{1}_2 \otimes \Lambda_1 + \mathbb{1}_3 \otimes \Lambda_2 \otimes \mathbb{1}_1 + \Lambda_3 \otimes \mathbb{1}_2 \otimes \mathbb{1}_1 \text{ is a diagonal matrix.} \end{aligned} \tag{10}$$

In the iterative solver chosen, we need to solve $\boldsymbol{s}^{(it)} = \mathsf{P}^{-1}\boldsymbol{r}^{(it)}$, where $\boldsymbol{r}^{(it)}$ is the residual at the (it)–th iteration, *i.e.*, $\boldsymbol{r}^{(it)} = \boldsymbol{b} - \mathsf{A}\boldsymbol{x}^{(it)}$. That equation may be easily solved using Eq. (10). After an initial step of performing the generalized eigendecomposition, appliying P just involves an inversion of a diagonal matrix and two matrix–vector products, where sum–factorization technique [1] can be applied. The overall computational cost is $O\left(n_{dof}^{4/3}\right)$, which does not depend on the B–spline degree p. We summarize this procedure in the following algorithm:

Algorithm 1: 3D Fast Diagonalization for the heat conduction problem

Data: The generalized eigendecomposition of pencils (K_l, M_l) for $l = 1, 2, 3$, and $\boldsymbol{r}^{(it)}$
Result: $\boldsymbol{s}^{(it)}$

1 Compute $\tilde{\boldsymbol{r}} = (U_3^\top \otimes U_2^\top \otimes U_1^\top) \cdot \boldsymbol{r}^{(it)}$.
2 Compute $\tilde{\boldsymbol{s}} = \mathsf{Q}^{-1} \cdot \tilde{\boldsymbol{r}}$.
3 Compute $\boldsymbol{s}^{(it)} = (U_3 \otimes U_2 \otimes U_1) \cdot \tilde{\boldsymbol{s}}$.

3.1.2 Inclusion of the geometry and thermal properties

The preconditioner described before does not include any information on the geometry parametrization F nor thermal properties of the material. To improve FD method without increasing the computational cost, we decided to adopt the following strategy: add some coefficients c_l such that

$$\mathsf{P} = c_1 M_3 \otimes M_2 \otimes K_1 + c_2 M_3 \otimes K_2 \otimes M_1 + c_3 K_3 \otimes M_2 \otimes M_1.$$

In this way, if we proceed as before, we may find that the inverse of P is very similar to Eq. (10) except for the diagonal matrix Q that is

$$\mathsf{Q} = c_1 \mathbb{1}_3 \otimes \mathbb{1}_2 \otimes \Lambda_1 + c_2 \mathbb{1}_3 \otimes \Lambda_2 \otimes \mathbb{1}_1 + c_3 \Lambda_3 \otimes \mathbb{1}_2 \otimes \mathbb{1}_1,$$

Thus, we can apply the same algorithm as in the previous paragraph. Actually, the general form of the modified preconditioner matrix P may be written as

$$[\mathsf{P}]_{AB} = \sum_{l=1}^{d} \int_{\hat{\Omega}} \frac{\partial}{\partial \xi_l} \hat{N}_A(\boldsymbol{\xi}) \bar{C}_{ll} \frac{\partial}{\partial \xi_l} \hat{N}_B(\boldsymbol{\xi}) d\hat{\Omega}, \text{ where } \bar{C} = \text{diag}\left(\{c_1, \ldots, c_d\}\right). \tag{11}$$

Here, applying reverse engineering, the tensor $\bar{C}$ may be interpreted as an approximation of $D(\boldsymbol{\xi})$, defined in Eq. (4a). Then, to compute c_l, we could use the following formulas introduced by [3]:

$$c_l = \int_{\hat{\Omega}} D_{ll}(\boldsymbol{\xi})\, d\hat{\Omega} = \sum_{p=1}^{d}\sum_{q=1}^{d} \int_{\hat{\Omega}} \left[J_F^{-1}\right]_{lp} [\boldsymbol{k}]_{pq} \det J_F \left[J_F^{-1}\right]_{lq} d\hat{\Omega}.$$

To keep it simple and knowing that we need only an approximation and not exact value of the integrals, we can compute them using the trapezoidal rule technique. For this purpose, we choose conveniently the quadrature points that are in positions 0, 0.5 and 1 in each dimension, so that we obtain a hypercubic grid of 3^d points within the parametric domain $\hat{\Omega}$. In a 3D case, we can apply the following quadrature rule:

$$c_l \approx \frac{1}{8} \sum_{q=1}^{3^3} \omega_q D_{ll}\left(\tilde{\xi}_q\right), \tag{12}$$

with ω_q being 8 for the interior points, 4 for the interior points of the boundary surfaces, 2 for the boundary edge points and 1 for the corner points [10].

3.1.3 Spectral properties

The convergence rate of most iterative algorithms depends on spectral properties of the matrix $\mathsf{P}^{-1}\mathsf{K}$ [2]. It turns out that the convergence rate increases if the spectral condition number, $\kappa\left(\mathsf{P}^{-1}\mathsf{K}\right)$, tends to 1. For most cases, computing the exact value of κ is too expensive, but we could provide an upper bound using the following proposition.

Proposition 1. *It holds*

$$\kappa\left(\mathsf{P}^{-1}\mathsf{K}\right) \leq \sup_{\hat{\Omega}} \lambda_{\max}(\bar{D}) \Big/ \inf_{\hat{\Omega}} \lambda_{\min}(\bar{D}), \tag{13}$$

where $\bar{D} = \bar{C}^{-1/2} D(\boldsymbol{\xi}) \bar{C}^{-1/2}$, while $\lambda_{\max}(\bar{D})$ and $\lambda_{\min}(\bar{D})$ denote respectively the maximum and minimum eigenvalue of $\bar{D}$.

Proof. Let $\boldsymbol{d} \in \mathbb{R}^{n_{\text{dof}}}$, and define $\mathcal{V}^h \ni u^h = \sum_A \hat{N}_A d_A$. Then it holds

$$\begin{aligned}
\boldsymbol{d}^\top \mathsf{K} \boldsymbol{d} &= \int_{\hat{\Omega}} \hat{\nabla} u^h \cdot \left[\left(\bar{C}^{1/2} \bar{D} \bar{C}^{1/2}\right) \hat{\nabla} u^h\right] d\hat{\Omega} \leq \int_{\hat{\Omega}} \lambda_{\max}(\bar{D}) \hat{\nabla} u^h \cdot \left[\left(\bar{C}^{1/2} \bar{C}^{1/2}\right) \hat{\nabla} u^h\right] d\hat{\Omega} \\
&\leq \sup_{\hat{\Omega}} \lambda_{\max}(\bar{D}) \int_{\hat{\Omega}} \hat{\nabla} u^h \cdot \left[\bar{C} \hat{\nabla} u^h\right] d\hat{\Omega} = \sup_{\hat{\Omega}} \lambda_{\max}(\bar{D}) \boldsymbol{d}^\top \mathsf{P} \boldsymbol{d}.
\end{aligned} \tag{14}$$

By the min–max theorem, we infer $\lambda_{\max}\left(\mathsf{P}^{-1}\mathsf{K}\right) \leq \sup_{\hat{\Omega}} \lambda_{\max}(\bar{D})$. With analogous calculations, it is possible to prove that $\lambda_{\min}\left(\mathsf{P}^{-1}\mathsf{K}\right) \geq \inf_{\hat{\Omega}} \lambda_{\min}(\bar{D})$, and hence $\kappa\left(\mathsf{P}^{-1}\mathsf{K}\right)$ is upper bounded by Eq. (13). □

Proposition 1 formalizes an almost intuitive fact: as long as the geometry is not too complex and the thermal properties of the material do not vary much along Ω, the condition number $\kappa\left(\mathsf{P}^{-1}\mathsf{K}\right)$ will be close to 1 and the preconditioner will perform quite well. In addition, it is worth noting that the upper bound in Eq. (13) does not depend on the B–spline degree p nor the mesh discretization h.

3.2 FD for the elasticity problem

3.2.1 Extension of the modified FD method

We build the preconditioner only considering the main–diagonal blocks $\left\{\mathsf{S}^{(1,1)}, \ldots, \mathsf{S}^{(d,d)}\right\}$ from Eq. (8). In that way, the inverse of the resulting block diagonal matrix P is composed of the inverse of its main–diagonal blocks; and, since each block is a conductivity–like matrix, the method described in Section 3.1 could be easily applied.

Then, the preconditioner P and its inverse are defined as:

$$\mathsf{P} = \begin{bmatrix} \mathsf{P}^{(1)} & & \\ & \ddots & \\ & & \mathsf{P}^{(d)} \end{bmatrix} \text{ and } \mathsf{P}^{-1} = \begin{bmatrix} \left(\mathsf{P}^{(1)}\right)^{-1} & & \\ & \ddots & \\ & & \left(\mathsf{P}^{(d)}\right)^{-1} \end{bmatrix}.$$

Here, for the case $d = 3$, the main–diagonal block matrix $\mathsf{P}^{(m)}$, for $m = 1, \ldots, d$, may be written as:

$$\mathsf{P}^{(m)} = c_1^{(m)} M_3 \otimes M_2 \otimes K_1 + c_2^{(m)} M_3 \otimes K_2 \otimes M_1 + c_3^{(m)} K_3 \otimes M_2 \otimes M_1.$$

Its inverse, using Fast Diagonalization method, may be written as

$$\begin{aligned} \left(\mathsf{P}^{(m)}\right)^{-1} &= (U_3 \otimes U_3 \otimes U_1) \left(Q^{(m)}\right)^{-1} (U_d \otimes U_2 \otimes U_1)^{\top}, \text{where} \\ Q^{(m)} &= c_1^{(m)} \mathbb{1}_3 \otimes \mathbb{1}_2 \otimes \Lambda_1 + c_2^{(m)} \mathbb{1}_3 \otimes \Lambda_2 \otimes \mathbb{1}_1 + c_3^{(m)} \Lambda_3 \otimes \mathbb{1}_2 \otimes \mathbb{1}_1. \end{aligned} \tag{15}$$

To compute the different coefficients $c_l^{(m)}$, we may use the following formula:

$$c_l^{(m)} = \int_{\hat{\Omega}} D_{ll}^{(m,m)} d\hat{\Omega} = \sum_{p=1}^{d} \sum_{q=1}^{d} \int_{\hat{\Omega}} \left[J_F^{-1}\right]_{lp} \left[\boldsymbol{k}^{(m,m)}\right]_{pq} \det J_F \left[J_F^{-1}\right]_{lq} d\hat{\Omega}.$$

As before, we could compute an approximation using a trapezoidal rule as in Eq. (12).

3.2.2 Spectral properties

The upper bound of $\kappa\left(\mathsf{P}^{-1}\mathsf{S}\right)$ is quite similar to the one given in Proposition 1, *i.e.*, $\kappa\left(\mathsf{P}^{-1}\mathsf{S}\right) \leq \sup_{\hat{\Omega}} \lambda_{\max}(\bar{D}) \Big/ \inf_{\hat{\Omega}} \lambda_{\min}(\bar{D})$, with $\bar{D} = \bar{C}^{-1/2} D(\boldsymbol{\xi}) \bar{C}^{-1/2}$. However, the block matrices are defined by:

$$D(\boldsymbol{\xi}) = \begin{bmatrix} D^{(1,1)}(\boldsymbol{\xi}) & \ldots & D^{(1,d)}(\boldsymbol{\xi}) \\ \vdots & \ddots & \vdots \\ D^{(d,1)}(\boldsymbol{\xi}) & \ldots & D^{(d,d)}(\boldsymbol{\xi}) \end{bmatrix} \text{ and } \bar{C} = \begin{bmatrix} \bar{C}^{(1)} & & \\ & \ddots & \\ & & \bar{C}^{(d)} \end{bmatrix} \text{ with } \bar{C}^{(m)} = \operatorname{diag}\left(\left\{c_1^{(m)}, \ldots, c_d^{(m)}\right\}\right).$$

4 Results

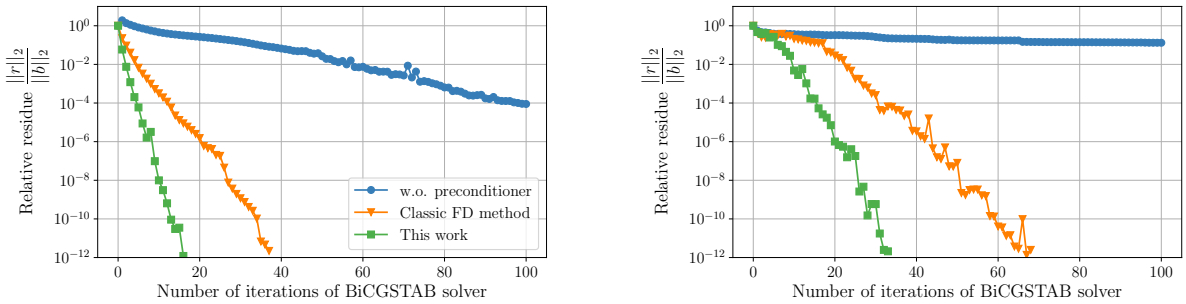
The results presented here were computed on a laptop equipped with Intel i7-11850H processors running at 2.5 GHz and with 16 GB of RAM. The tests were programmed in Python, version 3.8.10.; but, the core algorithms were developed in Fortran90, compiled into a static library, and called in Python using *f2py* module. In our benchmark, we do not incorporate parallelization and all computations were restricted to a single computational thread. We applied our preconditioner to both a heat conduction problem and an elasticity problem. In both cases, the physical domain is a quarter of an annulus delimited by $\Omega = \{\boldsymbol{x} = (x_1, x_2) \in \mathbb{R}^2 | 0 \leq x_1,\ 0 \leq x_2,\ 1 \leq x_1^2 + x_2^2 \leq 16\}$. For the heat conduction problem, we consider zero–temperature on the entire boundary $\partial\Omega$, and we fix the internal heat source $f = -\nabla \cdot (\boldsymbol{k} \nabla T)$ such that the exact temperature is given by $T(\boldsymbol{x}) = \sin(\pi x_1)\sin(\pi x_2)(x_1^2 + x_2^2 - 1)(x_1^2 + x_2^2 - 16)$. Besides, for the elasticity problem, we solve a well–known problem in literature: an infinite plate with circular hole loaded with far–field uniaxial tension $T_x = 1$. For the sake of simplicity, this problem is modelled as a quarter of the initial geometry with the outer edge circular shaped. Concerning the boundary conditions, we set $\boldsymbol{\sigma} \cdot \boldsymbol{n} = 0$ in the inner edge, where $\boldsymbol{n}$ is the unit outward normal vector and, on the outer edge, we imposed the exact traction $\boldsymbol{g}$ that, in polar coordinates (r, θ), is given by:

$$\boldsymbol{g} = \begin{cases} g_1 = \dfrac{T_x}{2}\left(2\cos\theta - \dfrac{R_{in}^2}{r^2}(2\cos\theta + 3\cos 3\theta) + \dfrac{3R_{in}^4}{r^4}\cos 3\theta\right); \\ g_2 = \dfrac{3T_x}{2}\sin 3\theta \left(\dfrac{R_{in}^4}{r^4} - \dfrac{R_{in}^2}{r^2}\right); \text{ where } R_{in} \text{ is the inner radius.} \end{cases}$$

For the elasticity problem, we consider an isotropic material with Young's modulus $E = 10^3$ and Poisson's ratio $\nu = 0.3$; while for the heat conduction problem, we consider an anisotropic homogeneous material with thermal conductivity given by

$$\boldsymbol{k} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 2 \end{bmatrix}.$$

The iterative solver chosen is the biconjugate gradient stabilized method (BiCGSTAB). The initial guess is the null vector and the stopping criteria is defined by $||\boldsymbol{r}^{(it)}||_2 \leq 10^{-12}||\boldsymbol{b}||_2$.



(a) In the heat conduction problem

(b) In the elasticity problem

Figure 1: Evolution of the relative residue when $p = 6$ and $h^{-1} = 64$ using different preconditioner methods: without preconditioner, classic Fast Diagonalization problem and our preconditioner.

Table 1: Performance of our preconditioner with a BiCGSTAB solver in the heat conduction problem.

	Iterations / CPU time (s)		
h^{-1}	$p = 4$	$p = 5$	$p = 6$
64	18 / 0.125	17 / 0.125	18 / 0.172
128	19 / 0.484	18 / 0.516	18 / 0.484
256	18 / 1.516	18 / 1.531	18 / 1.593
512	19 / 8.625	18 / 8.594	18 / 8.984

Table 2: Performance of our preconditioner with a BiCGSTAB solver in the elasticity problem.

	Iterations / CPU time (s)		
h^{-1}	$p = 4$	$p = 5$	$p = 6$
64	33 / 0.390	33 / 0.453	35 / 0.578
128	35 / 1.609	35 / 1.734	35 / 2.001
256	36 / 8.656	37 / 9.594	37 / 10.48
512	38 / 43.41	38 / 47.17	38 / 47.22

Fig. 1a and Fig. 1b show the evolution of the relative residue using different preconditioning methods for the heat conduction problem and elasticity problem, respectively. We can observe the larger advantage of our preconditioner over the classic FD method and even more if no preconditioner is used. The explanation for this difference is clearly because our preconditioner takes into account material properties and geometry. This implies that by computing certain scalar parameters, whose computational cost is negligible, the number of iterations required for the same level of precision can be reduced to one half.

Table 1 and Table 2 show the performance of the iterative solver using our method as preconditioner for both cases. Both examples are experimental evidence that support the theoretical results: the condition number $\kappa(\mathsf{P}^{-1}\mathsf{A})$ only depends on the complexity of geometry and material properties. It is robust with respect to the degree p and mesh discretization h.

5 Conclusion

In this work we have introduced the different bricks needed to implement a good preconditioner for both heat conduction and elasticity problems. The novelty of our preconditioner, based on Fast Diagonalization method, is its coding simplicity and remarkable efficiency in achieving very good results with a reduced number of iterations. Moreover, we have proved theoretically and experimentally that our preconditioner is robust with respect to the polynomial degree and mesh discretization. As observed in the experimental results, this implies that its performance only depends on the complexity of the geometry and material properties.

References

[1] P. Antolin, A. Buffa, F. Calabrò, M. Martinelli, and G. Sangalli. Efficient matrix computation for tensor–product isogeometric analysis: The use of sum factorization. *Computer Methods in Applied Mechanics and Engineering*, 285:817–828, 2015.

[2] R. Barrett, M. Berry, T.F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst. *Templates for the solution of linear systems: building blocks for iterative methods*. SIAM, 1994.

[3] P.D. Brubeck and P.E. Farrell. A scalable and robust vertex–star relaxation for high–order FEM. *SIAM Journal on Scientific Computing*, 44(5):A2991–A3017, 2022.

[4] J. Cornejo Fuentes, T. Elguedj, D. Dureisseix, and A. Duval. A cheap preconditioner based on fast diagonalization method for matrix-free weighted-quadrature isogeometric analysis applied to non-linear transient heat transfer problems. *Computer Methods in Applied Mechanics and Engineering*, 414:116157, 2023.

[5] J.A. Cottrell, T.J.R. Hughes, and Y. Bazilevs. *Isogeometric Analysis: Toward integration of CAD and FEA*. Wiley, 2009.

[6] T. Elguedj, Y. Bazilevs, V.M. Calo, and T.J.R. Hughes. $\bar{B}$ and $\bar{F}$ projection methods for nearly incompressible linear and non–linear elasticity and plasticity using higher–order NURBS elements. *Computer Methods in Applied Mechanics and Engineering*, 197(33):2732–2762, 2008.

[7] T.J.R. Hughes. *The Finite Element Method: Linear static and dynamic Finite Element analysis*. Dover Civil and Mechanical Engineering. Dover Publications, 2000.

[8] T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194(39):4135–4195, 2005.

[9] T.J.R. Hughes, A. Reali, and G. Sangalli. Duality and unified analysis of discrete approximations in structural dynamics and wave propagation: Comparison of p–method finite elements with k–method NURBS. *Computer Methods in Applied Mechanics and Engineering*, 197(49):4104–4124, 2008.

[10] J.N. Lyness. Symmetric integration rules for hypercubes I. Error coefficients. *Mathematics of Computation*, 19(90):260–276, 1965.

[11] J. Planas, I. Romero, and J.M. Sancho. B free. *Computer Methods in Applied Mechanics and Engineering*, 217-220:226–235, 2012.

[12] G. Sangalli and M. Tani. Isogeometric preconditioners based on fast solvers for the Sylvester equation. *SIAM Journal on Scientific Computing*, 38(6):A3644–A3671, 2016.