31st International Conference
ENGINEERING MECHANICS 2025
Medlov, Czech Republic, May 12 –14, 2025
Paper **#31**, pp. 61–64, doi: 10.21495/em2025-61

IMPLEMENTATION OF BLOCK PRECONDITIONERS FOR PIEZOELASTICITY IN SFEPY

Cimrman R.[1]

Abstract: *Motivated by the need to model 3D printed electroactive metamaterials, an approach is presented to improve the computational efficiency of the SfePy package for the simulation of structurally complex piezoelectric lattice-like structures. Enhancements to the iterative solver interface allow the use of state-of-the-art iterative solvers and Schur complement-based preconditioners. Their performance is illustrated by a numerical study where the convergence of the iterative solvers is observed for several levels of uniform mesh refinement.*

Keywords: piezoelasticity, iterative solution, block preconditioners, numerical study, SfePy

1. Introduction

SfePy (Simple Finite Elements in Python, https://sfepy.org) is an Open Source software for solving systems of partial differential equations by various flavors of the finite element (FE) method. It uses fast vectorized operations on the Python level (Cimrman, 2021) for speed and interfaces a number of efficient external solvers such as the direct solver MUMPS (Amestoy et al., 2019) or the many iterative solvers and preconditioners available in PETSc (Balay et al., 2024). While it has been used for multiscale modelling of metamaterials including piezoelectric (PZ) effects in past (Cimrman et al., 2019), the iterative solver interface has been rather limited for multiphysical problems. Our group's interest in modeling the behaviour of 3D printed lattice-like electroactive metamaterials motivated this contribution, which presents an approach that allows efficient simulation of large PZ structures in SfePy.

2. Static Piezoelastic Problem and Preconditioning

A piezoelastic body $\Omega \subset \mathbb{R}^3$ is, under linear assumptions, described by the constitutive relations

$$\boldsymbol{\sigma} = \boldsymbol{C}\boldsymbol{\varepsilon} - \boldsymbol{e}^T\boldsymbol{E} \;, \quad \boldsymbol{d} = \boldsymbol{e}\boldsymbol{\varepsilon} + \boldsymbol{\kappa}\boldsymbol{E} \;, \quad \boldsymbol{\varepsilon} = \frac{1}{2}(\boldsymbol{\nabla}\boldsymbol{u} + \boldsymbol{\nabla}^T\boldsymbol{u}) \;, \quad \boldsymbol{E} = -\boldsymbol{\nabla}\varphi \;, \tag{1}$$

where the mechanical stress $\boldsymbol{\sigma}$ (in Voigt notation) and the electric displacement $\boldsymbol{d}$ are proportional to the mechanical strain $\boldsymbol{\varepsilon}$ and the electric field vector $\boldsymbol{E}$, $\boldsymbol{u}$ is the mechanical displacement vector, φ the electric potential, $\boldsymbol{C}$ the matrix of elastic properties under constant electric field intensity, $\boldsymbol{e}$ the piezoelectric modulus and $\boldsymbol{\kappa}$ the permittivity under constant deformation.

Let $V_0^u(\Omega) = \{\boldsymbol{u} \in [H^1(\Omega)]^3, \boldsymbol{u} = \boldsymbol{0} \text{ on } \Gamma_u\}$, $V_0^\varphi = \{\varphi \in H^1(\Omega), \varphi = 0 \text{ on } \Gamma_p\}$, where Γ_u, Γ_p are the boundaries where the displacement and potential Dirichlet boundary conditions are prescribed. The governing equations in the static case, i.e. the static mechanical equilibrium and the Gauss's law, lead, for zero Neumann boundary conditions, to the following weak formulation: Find $\boldsymbol{u}$, φ such that

$$\int_\Omega \boldsymbol{\varepsilon}(\boldsymbol{v})^T\boldsymbol{C}\boldsymbol{\varepsilon}(\boldsymbol{u}) + \int_\Omega \boldsymbol{\varepsilon}(\boldsymbol{v})^T\boldsymbol{e}^T\boldsymbol{\nabla}\varphi - \int_\Omega \boldsymbol{v}\cdot\boldsymbol{f} = 0 \quad \forall \boldsymbol{v} \in V_0^u(\Omega) \;, \tag{2}$$

$$-\int_\Omega (\boldsymbol{\nabla}\psi)^T\boldsymbol{e}\boldsymbol{\varepsilon}(\boldsymbol{u}) + \int_\Omega (\boldsymbol{\nabla}\psi)^T\boldsymbol{\kappa}\boldsymbol{\nabla}\varphi = 0 \quad \forall \psi \in V_0^\varphi(\Omega) \;. \tag{3}$$

[1] Robert Cimrman, Institute of Thermomechanics of the CAS, Prague, CZ, cimrman@it.cas.cz & New Technologies — Research Centre and Faculty of Applied Sciences, University of West Bohemia, Plzeň, CZ, cimrman3@ntc.zcu.cz

The FE discretization of (2)–(3) uses the approximations

$$\boldsymbol{u}(\boldsymbol{\xi}) = \mathbf{N}_u(\boldsymbol{\xi})\mathbf{u}\ , \quad \varphi(\boldsymbol{\xi}) = \mathbf{N}_\varphi(\boldsymbol{\xi})\mathbf{p}\ , \tag{4}$$

where $\mathbf{N}_u(\boldsymbol{\xi})$, $\mathbf{N}_\varphi(\boldsymbol{\xi})$ are the shape functions of the displacements and the potential, respectively, evaluated in numerical quadrature points $\boldsymbol{\xi}$, and leads to a linear system with the following structure:

$$\begin{bmatrix} \mathbf{K} & \mathbf{B}^T \\ -\mathbf{B} & \mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{0} \end{bmatrix}\ , \tag{5}$$

where $\mathbf{K}$ is the stiffness matrix, $\mathbf{B}$ is the piezoelastic coupling matrix, $\mathbf{C}$ is the electrostatic potential matrix and $\mathbf{f}$ the volume forces vector.

The system (5) can be solved using a direct solver for problems of small to moderate size. For large problems, iterative solvers are needed due to lower memory requirements and more efficient parallelization possibilities. Here we consider the flexible generalized minimal residual method (FGMRES) as implemented in PETSc using the petsc4py Python interface (Dalcin et al., 2011), together with two preconditioners:

1. a simple block diagonal (BD) preconditioner

$$\mathbf{P}_{BD} = \begin{bmatrix} \mathbf{K} & \mathbf{0} \\ \mathbf{0} & \mathbf{C} \end{bmatrix} \tag{6}$$

2. the cell-by-cell Schur complement (CBCS) preconditioner (Cao and Neytcheva, 2021)

$$\mathbf{P}_{CBCS} = \begin{bmatrix} \mathbf{K} & \mathbf{0} \\ -\mathbf{B} & \mathbf{S}_{CBC} \end{bmatrix}\ , \text{ where } \mathbf{S}_{CBC} = \sum_{T_e \in \Omega_h} \mathbf{K}_e + \mathbf{B}_e^T \mathbf{C}_e^{-1} \mathbf{B}_e\ , \tag{7}$$

   and T_e are the elements of the FE domain Ω_h.

Both were recently reported (Wegert et al., 2024) to work well with PZ material models.

3. Implementation of the Preconditioners and PETSc Support Enhancements

The BD preconditioner did not require any special implementation effort as it was directly available in PETSc via its PCFieldSplit preconditioner.

In contrast, the CBCS preconditioner achieves its efficiency by computing the approximate Schur complement locally on each element, so some support in the FE engine was required. SfePy allows the construction of equations from the individual terms (e.g. the integrals in (2), (3)) and also assembles the global linear system term by term. To avoid evaluating each term twice (once for the assembling, the second time for the preconditioner), an alternative user-defined assembling function can now be passed to the package. In our case, it not only assembles the matrix blocks, but also stores the cell-by-cell matrices of each block in a list that can then be used to build the CBCS preconditioner. Other implemented improvements are

- Attachment of rigid body modes to the elasticity block $\mathbf{K}$ so that the algebraic multigrid converges well.
- A user-defined function to construct a PETSc matrix from the sparse matrix used in SfePy. MatNest is used in our context.
- The linear system and preconditioner matrices can be different.
- The linear system can be prescaled to better balance its blocks.

Finally, the CBCS preconditioner is implemented as a Python class that uses option prefixes for its inner solvers so that they can be configured from the command line. For example,

```
-cbcs_K_ksp_type=cg -cbcs_K_pc_type=gamg -cbcs_K_ksp_rtol=1e-2
```

specifies the inner solver. preconditioner and relative tolerance for the $\mathbf{K}$ block, see also below.

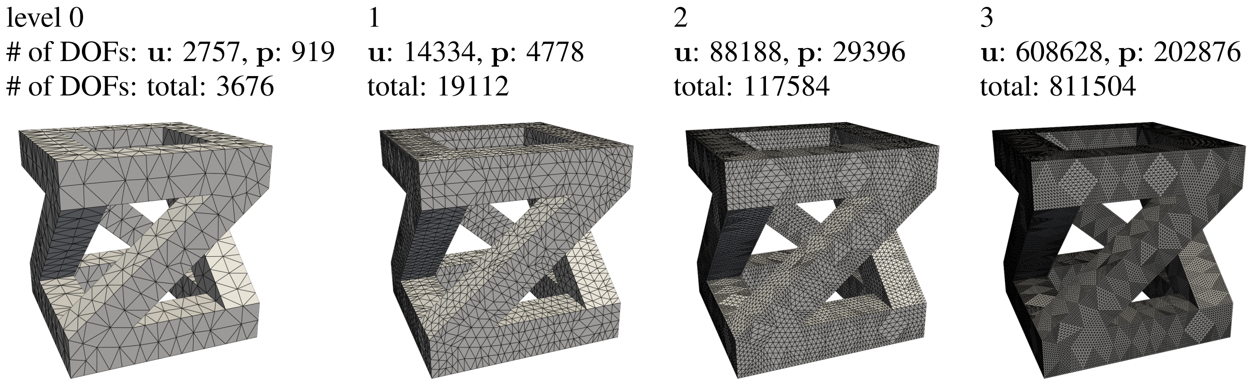

level 0
# of DOFs: **u**: 2757, **p**: 919
# of DOFs: total: 3676
1
u: 14334, **p**: 4778
total: 19112
2
u: 88188, **p**: 29396
total: 117584
3
u: 608628, **p**: 202876
total: 811504

Fig. 1: Lattice geometry, uniform mesh refinement levels and their respective DOF counts.

4. Numerical Example

To evaluate the performance of the preconditioners described above, a hollow cube with skewed “vertical” bars, see Fig. 1, made of a PZ material with properties taken from the PIC 181 sensor (company PI, https://www.piceramic.com/) was used for our benchmarks. The base mesh was uniformly refined up to three times, resulting in problem sizes in terms of degrees of freedom (DOF) shown in Fig. 1.

The lattice was fixed in the bottom plane, a voltage of 1000 V was applied to the upper surfaces of the skewed bars and 0 V to the lower surfaces of these bars. Similar structures, after appropriate scaling, could be used as reference periodic cells of (possibly 3D-printed) PZ metamaterials exhibiting flexoelectric response (Zubko et al., 2013) at the macroscale.

The performance of the direct solver MUMPS and the preconditioned iterative solvers BD-FGMRES, CBCS-FGMRES was evaluated. The flexible variant of GMRES allows to use approximate solvers for the blocks of $\mathbf{P}_{BD}$, i.e. to invert $\mathbf{K}$ and $\mathbf{C}$, or $\mathbf{P}_{CBCS}$, i.e. to invert $\mathbf{K}$ and $\mathbf{S}$. The conjugate gradient method (CG of PETSc) preconditioned by the algebraic multigrid (GAMG of PETSc) with the relative tolerance 10^{-2} was used in all simulations. The relative tolerance of FGMRES was set to 10^{-8}.

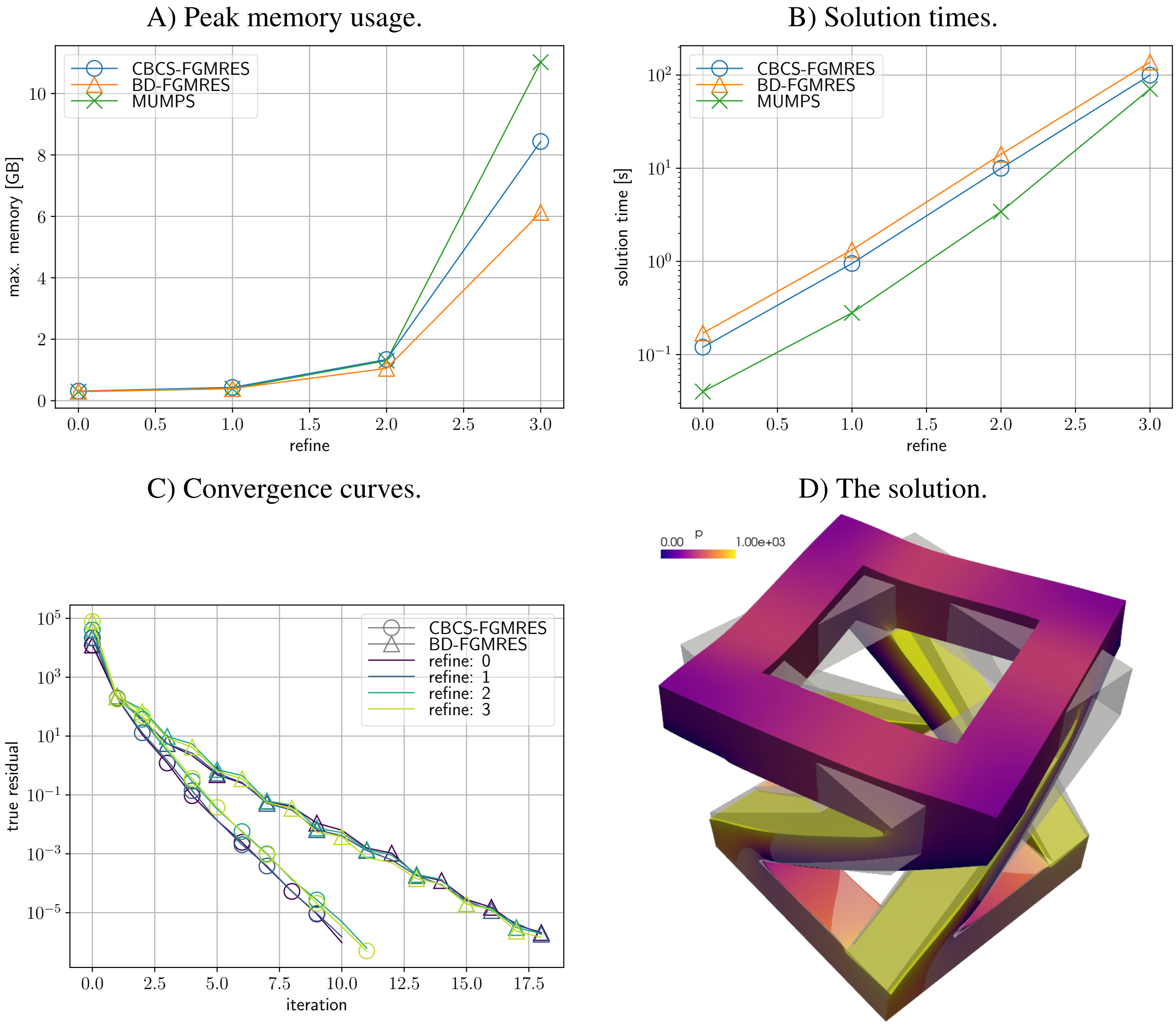
The peak memory consumption when using these solvers is shown in Fig. 2-A and supports one of the motivations for using an iterative solver, i.e. lower memory requirements. However, the solution times shown in Fig. 2-B indicate that the direct solver wins in terms of speed for small problems, but the gap narrows as the problem size increases. Fine-tuning the inner solvers and their parameters (not done here) could still significantly improve the performance of the iterative solvers. The number of iterations of both the global (FGMRES) and inner (CG+GAMG) solvers (not shown here) are in agreement with the performance reported in (Wegert et al., 2024). Fig. 2-C shows the convergence curves for the two preconditioners: $\mathbf{P}_{BD}$ needs about 18 iterations, while $\mathbf{P}_{CBCS}$ only needs about 10, making the latter faster (but more memory demanding). In both cases the number of global iterations does not depend on the size of the problem. Finally, the resulting deformation and potential are illustrated in Fig. 2-D.

5. Conclusion

The weak formulation of a static piezoelastic problem and its FE discretization were presented with regard to the resulting linear system structure and suitable iterative solvers and preconditioners. Then, the implementation aspects of the PETSc-based block diagonal and cell-by-cell Schur complement preconditioners within the finite element package SfePy were discussed. Finally, a numerical convergence study was performed to demonstrate the soundness of the SfePy implementation. While the direct solver wins in terms of solution time for small problems, the suitably preconditioned iterative solvers have lower memory requirements, allow fine-tuning and have much higher potential for parallelization and are therefore promising for large problems involving PZ materials.

Acknowledgments

This work has been supported by the grant 23-06220S of the Czech Science Foundation within institutional support RVO:61388998.



A) Peak memory usage.

B) Solution times.

C) Convergence curves.

D) The solution.

Fig. 2: Dependence of A) memory and B) time requirements of the direct (crosses) and iterative solvers on the uniform mesh refinement level. C) Convergence curves of the iterative solvers. D) The finest mesh solution: the displacements indicated by the magnified deformation, the potential is given by color. The original undeformed geometry is in gray.

References

Amestoy, P. R., Buttari, A., L'Excellent, J.-Y., and Mary, T. (2019) Performance and scalability of the block low-rank multifrontal factorization on multicore architectures. *ACM Transactions on Mathematical Software*, 45, 1, pp. 2:1–2:26.

Balay, S., Abhyankar, S., Adams, M. F., Benson, S., and et al., J. B. (2024) PETSc Web page. https://petsc.org/.

Cao, Y. and Neytcheva, M. (2021) Cell-by-cell approximate schur complement technique in preconditioning of meshfree discretized piezoelectric equations. *Numerical Linear Algebra with Applications*, 28, 4, pp. e2362.

Cimrman, R. (2021) Fast evaluation of finite element weak forms using Python tensor contraction packages. *Advances in Engineering Software*, 159, pp. 103033.

Cimrman, R., Lukeš, V., and Rohan, E. (2019) Multiscale finite element calculations in Python using SfePy. *Advances in Computational Mathematics*, 45, 4, pp. 1897–1921.

Dalcin, L. D., Paz, R. R., Kler, P. A., and Cosimo, A. (2011) Parallel distributed computing using python. *Advances in Water Resources*, 34, 9, pp. 1124–1139.

Wegert, Z. J., Roberts, A. P., and Challis, V. J. (2024) Level set-based inverse homogenisation of three-dimensional piezoelectric materials. arXiv:2410.03148 [cs].

Zubko, P., Catalan, G., and Tagantsev, A. K. (2013) Flexoelectric effect in solids. *Annual Review of Materials Research*, 43, 1, pp. 387–421.