On some block-preconditioners for saddle point systems and their CPU-GPU performance

Ali Dorostkar[a], Maya Neytcheva[a], Björn Lund[b]

[a]*Department of Information Technology, Uppsala University, Sweden*
[b]*Department of Earth Sciences, Uppsala University, Sweden*

Abstract

In this work we emphasize some aspects of the numerical and computational performance of block preconditioners for systems with matrices of saddle point form. We discuss the quality of a sparse approximation of the related Schur complement for constructing an efficient preconditioner and the achieved numerical efficiency in terms of number of iterations. We also present a performance study of the computational efficiency of the corresponding preconditioned iterative solution methods, implemented using publicly available numerical linear algebra software packages, both on multicore CPU and GPU devices. We show that the presently available GPU accelerators can be very efficiently used in computer simulations involving inner-outer solution methods and hierarchical data structures. The benchmark problem originates from a geophysical application, namely, the elastic Glacial Isostatic Adjustment model, discretized using the finite element method.

Keywords: Indefinite block-preconditioners, Finite Element based element-wise sparse approximation of the Schur complement, inner-outer iterative methods, OpenMP and MPI paradigms, CPU-GPU, compressible and incompressible linear elasticity

1. Introduction

Performing large scale computer simulations of realistic models is a complex task that consists of several tightly interrelated aspects. Provided that the mathematical model is given, we need to consider the following phases: (1) choose a proper discretization method that guarantees the desired accuracy of the solution, (2) choose numerical solution techniques that are robust and efficient with respect to all involved problem and discretization parameters, (3) choose software implementation that will perform in a fast and scalable manner on current computer architectures. Furthermore, we have to take into account the interplay between these aspects since the discretization method influences the properties of the arising discrete algebraic systems to be solved. The latter impacts the choice of the solution methods by introducing additional requirements for

Email addresses: `ali.dorostkar@it.uu.se` (Ali Dorostkar), `maya.neytcheva@it.uu.se` (Maya Neytcheva), `bjorn.lund@geo.uu.se` (Björn Lund)

the program implementation and the computational efficiency of the simulations. In addition we point out that the majority of large scale computer simulations require not just one but rather a collection of algorithms and techniques, such as unstructured, adaptive or moving meshes; time-dependent processes that require the repeated solution of nonlinear and/or linear systems; inner- outer solution procedures, block preconditioners that utilize internal algebraic structures, methods as algebraic multigrid that rely on the hierarchy of data and have a recursive nature. Finally, all this has to work efficiently on the modern computer architectures. At this level of complexity, writing the code from scratch is impractical, time consuming and generally ill-advised. A better choice is to rely on the available numerical algebra software packages, that have been presumably optimized to utilize the available hardware resources to a high extent.

As we emphasize large scale and computationally heavy numerical simulations, we consider iterative solution methods to be the viable alternative. The latter automatically entails the need to choose an efficient and robust preconditioning technique, in order to accelerate the convergence of the iterative process.

In this work we focus on the class of problems that gives rise to linear systems with indefinite matrices of saddle point form and solve those using state-of-the-art block-structured preconditioners. To achieve the desired overall efficiency, taking into account that the blocks of the system matrix are very large themselves, we use inner iterative solvers. This leads to inner-outer type of solution procedures and raises the necessity to properly choose the inner stopping tolerances so that we balance between computational burden and accurate enough block solvers to ensure the fast convergence of the outer method.

In general, such preconditioners require a high quality sparse approximation of a (most often dense) Schur complement matrix. We describe the so-called *element-wise* approximation technique and analyze some classes of matrices for which the approach leads to high quality sparse Schur complement approximations. We then test both the numerical efficiency (number of iterations) and the computational efficiency (time) on multicore CPU/GPU systems using the OpenMP (shared memory paradigm) and also the MPI (distributed memory) programming model.

The implementation is based on the software packages Deal.II ([1]), Trilinos ([2]), AGMG ([3]) and PARALUTION ([4]). The iteration counts and the simulation time for the CPU and GPU platforms are compared. The execution time is also compared with that of the sparse direct solver in a high quality commercial software package ABAQUS [5].

The paper is organized as follows. In Section 2 we describe the construction of the approximate Schur complement and analyze its quality when applied to indefinite matrices of saddle point form. Section 3 comprises the description of the underlying problem of interest, its discretization and algebraic form. We present the experimental setting in Section 4 and the obtained numerical results in Section 5. Some conclusions and an outlook are given in Section 6.

2. Finite element-based sparse approximations of the Schur complement for indefinite matrices

2.1. Briefly on preconditioners for two-by-two block matrices

The scientific literature on preconditioners for two-by-two block systems is vast and here we present only some details, relevant for the cases, considered in this paper. For more details we refer to [6, 7] and the references therein.

We consider a preconditioning strategy based on the structure of the matrix, for systems in a two-by-two block form, and the discretization which in this case is a standard conforming Finite Element method (FEM).

To set up the notations, we formulate a general abstract mixed-variable problem in variational form. Let $\mathcal{V}$ and $\mathcal{Q}$ be some Hilbert spaces with norms $\|\cdot\|_{\mathcal{V}}$ and $\|\cdot\|_{\mathcal{Q}}$ and scalar products $\langle\cdot,\cdot\rangle_{\mathcal{V}}$ and $\langle\cdot,\cdot\rangle_{\mathcal{Q}}$. Consider the bilinear forms $a(u,v)$ on $\mathcal{V}\times\mathcal{V}$, $b(v,p)$ on $\mathcal{V}\times\mathcal{Q}$ and $c(p,q)$ on $\mathcal{Q}\times\mathcal{Q}$. Assume also that the bilinear forms satisfy the necessary and sufficient conditions so that the following problem has a unique solution (see, for instance [8]):
Find $(u,p)\in\mathcal{V}\times\mathcal{Q}$, such that

$$\begin{aligned} a(u,v)+b(p,v) &= \langle f,v\rangle_{\mathcal{V}} \quad \text{for all } v\in\mathcal{V} \\ b(v,q)+c(p,q) &= \langle g,q\rangle_{\mathcal{Q}} \quad \text{for all } q\in\mathcal{Q} \end{aligned} \tag{1}$$

For the discussion to follow next, we do not need to specify in detail the particular spaces $\mathcal{V}$ and $\mathcal{Q}$ or the properties of the bilinear forms.

When discretizing (1), the continuous spaces are replaced by their finite dimensional counterparts $\mathcal{V}_h$, $\mathcal{Q}_h$ and we obtain a linear (or linearized) system of algebraic equations to solve

$$\mathcal{A}\begin{bmatrix}\mathbf{u}_h\\ \mathbf{p}_h\end{bmatrix} = \begin{bmatrix}\mathbf{f}_h\\ \mathbf{g}_h\end{bmatrix}, \tag{2}$$

where $\mathcal{A}$ admits a two-by-two block structure and is assumed to be nonsingular,

$$\mathcal{A} = \begin{bmatrix}A_{11} & A_{12}\\ A_{21} & A_{22}\end{bmatrix}. \tag{3}$$

We note that u can be a vector variable and that would introduce additional block structure in the above algebraic problem.

To solve systems with $\mathcal{A}$ we consider preconditioned Krylov subspace iterative solution methods for general matrices, such as `GMRES` or `GCG` (cf. [9, 10]). Most often, the preconditioner is based on the exact block factorization of $\mathcal{A}$ and utilizes its structure. The preconditioner can be of block-multiplicative form (4, left), or of block-triangular form, say as in (4, right),

$$\mathcal{B}_F = \begin{bmatrix}A_{11} & 0\\ A_{21} & S\end{bmatrix}\begin{bmatrix}I_1 & Z_{12}\\ 0 & I_2\end{bmatrix}, \quad \mathcal{B} = \begin{bmatrix}A_{11} & 0\\ A_{21} & S\end{bmatrix}. \tag{4}$$

Here S is an approximation of the exact Schur complement $S_{\mathcal{A}}$ of $\mathcal{A}$, $S_{\mathcal{A}} = A_{22} - A_{21}A_{11}^{-1}A_{12}$, Z_{12} is either exactly equal or approximates the matrix product $A_{11}^{-1}A_{12}$ and I_1, I_2 are identity matrices of corresponding order.

As our interest is to solve primarily nonsymmetric systems and/or to use variable preconditioning, we consider in this work only the block lower-triangular form $\mathcal{B}$. The preconditioner

$$\widehat{\mathcal{B}} = \begin{bmatrix} A_{11} & 0 \\ A_{21} & S_{\mathcal{A}} \end{bmatrix}$$

is often referred to as the *ideal* preconditioner for $\mathcal{A}$ since it clusters all the eigenvalues of $\widehat{\mathcal{B}}^{-1}\mathcal{A}$ at one and iterative methods as GMRES or GCG converge in 2-3 iterations. From a practical point of view to apply $\widehat{\mathcal{B}}$ is very expensive and therefore A_{11} and $S_{\mathcal{A}}$ are approximated. However, if the approximations are not good enough, then the clustering property is weakened and even might be lost. In our study, to control the quality of the preconditioner we approximate $S_{\mathcal{A}}$ by S and use inner solvers for A_{11} and S. In this way the overall efficiency of $\mathcal{B}$ depends on how accurately we solve with A_{11} and S, which is controlled by stopping tolerances and is relatively easy, and how well S approximates $S_{\mathcal{A}}$, which is the more difficult task. Therefore we discuss mainly the construction of an approximation of $S_{\mathcal{A}}$ in a FEM framework.

2.2. Element-wise Schur complements approximations

The matrix $\mathcal{A}$ (3) can be assembled block by block. One notes, however, that it can be assembled based on local (macro element) matrices that admit block structure themselves, namely, $\mathcal{A} = \sum\limits_{\ell=1}^{L} {R^{(\ell)}}^T A^{(\ell)} R^{(\ell)}$, $\mathcal{A} \in \mathbb{R}^{N\times N}$, $A^{(\ell)} \in \mathbb{R}^{n\times n}$, where

$$\mathcal{A}^{(\ell)} = \begin{bmatrix} A_{11}^{(\ell)} & A_{12}^{(\ell)} \\ A_{21}^{(\ell)} & A_{22}^{(\ell)} \end{bmatrix} \begin{matrix} \}n_1 \\ \}n_2 \end{matrix}. \tag{5}$$

Here $n = n_1 + n_2$, $\ell = 1, \cdots, L$ and L denotes the number of the finite (macro-)elements in the discretization mesh. The matrices $R^{(\ell)} \in \mathbb{R}^{n\times N}$ are the standard Boolean matrices which provide the local-to-global correspondence of the numbering of the degrees of freedom.

A very appealing idea is then, based on (5), to compute exactly local Schur complements and assemble those into a global matrix to be used as an approximation of $S_{\mathcal{A}}$, i.e., let

$$S = \sum_{\ell=1}^{L} {R_2^{(\ell)}}^T S^{(\ell)} R_2^{(\ell)}, \tag{6}$$

where $S^{(\ell)} = A_{22}^{(\ell)} - A_{21}^{(\ell)} {A_{11}^{(\ell)}}^{-1} A_{12}^{(\ell)}$ and $R_2^{(\ell)}$ are the parts of $R^{(\ell)}$ corresponding to the degrees of freedom in A_{22}.

Without loss of generality we assume that all $A_{11}^{(\ell)}$ are invertible. If these are singular, we cure the problem by adding a diagonal perturbation of order h^2, where h is the characteristic discretization parameter.

The construction of $S^{(\ell)}$ can be done fully in parallel across ℓ and the resulting matrix is sparse – properties that comprise two important advantages of the method. We next address the question what is the quality of S as an approximation of $S_{\mathcal{A}}$.

Historically, this element-wise ([11]) (additive ([12]) or element-by-element ([13])) method to approximate the Schur complement has been first developed for matrices arising from discretized scalar elliptic problems, on which we impose a two-by-two block structure, based on consecutive regular mesh refinements that split the degrees of freedom into two groups - 'coarse' and 'newly appeared' or 'fine', see [14, 11]. It has been shown that for mesh-based splittings and symmetric and positive definite (spd) matrices, S and $S_{\mathcal{A}}$ are spectrally equivalent, namely, there exist constants α and β for which there holds (in a positive definite sense)

$$\alpha S_{\mathcal{A}} \leq S \leq \beta S_{\mathcal{A}}, \tag{7}$$

and α and β do not depend on the discretization parameter and on discontinuities in the problem coefficients, as long as these discontinuities are aligned with the coarsest mesh used. The method has also been shown to work well for parabolic problems, such as for the heat equation ([15]) as well as for some convection-diffusion problems that are nonsymmetric, without a rigorous theory in the latter case (cf. [13]). Recently the theory has been extended to the case of highly varying problem coefficients that are not necessarily aligned with any coarse mesh. Using certain overlapping strategies, the mesh- and coefficient-independent properties of the preconditioned matrix have been derived (cf. [16]). Results, analogous to (7), for spd problems, where the splitting is not based on meshes, are derived in [13, 17].

For the case of spd matrices the upper bound β is shown to be 1 and the derivation is based on certain minimization properties of the Schur complement that hold in this case. For details, see, for instance [12, 11]. The lower bound α has been originally derived in terms of the so-called Cauchy-Bunyakowski-Schwarz (CBS) constant, denoted usually by γ, $0 \leq \gamma < 1$, cf. [18], and the relation for two-level setting is

$$\alpha = 1 - \gamma^2.$$

As γ does not depend on the discretization parameter and on discontinuous problem coefficients aligned with some coarse mesh, S appears to be a high quality approximation of $S_{\mathcal{A}}$. An extension of this theory to more general nonsymmetric and indefinite matrices is done in [19], where a parameter analogous to γ has been introduced.

We consider now indefinite matrices, originating from some mixed FEM discretization. To this end, we specify the properties of the bilinear forms involved and of the spaces $\mathcal{V}$ and $\mathcal{Q}$.

Let $a(u,v)$ be a continuous bilinear form, not necessarily symmetric, and satisfying

$$|a(u,v)| \leq \|a\| \, \|u\|_{\mathcal{V}} \|v\|_{\mathcal{V}}, \ \forall u, v \in \mathcal{V}.$$

Consider a bounded continuous bilinear form $b(v,q)$ on $\mathcal{V} \times \mathcal{Q}$ which satisfies

$$|b(v,q)| \leq \|b\| \, \|v\|_{\mathcal{V}} \|q\|_{\mathcal{Q}}, \ \forall v \in \mathcal{V}, \forall q \in \mathcal{Q}.$$

Let also $\widetilde{m}(p,q)$ be a bilinear form on $Q \times Q$, assumed to be positive and bounded, i.e.,

$$0 \leq \widetilde{m}(p,p) \text{ and } |\widetilde{m}(p,q)| \leq \|\widetilde{m}\| \, \|p\|_{\mathcal{Q}} \|q\|_{\mathcal{Q}}, \forall p, q \in Q.$$

For the problems we discuss here, $\widetilde{m}$ is assumed to be of the form $\widetilde{m}(p,q) = \delta m(p,q)$, where $\delta \geq 0$ and $m(p,q) > 0 \quad \forall p, q \in Q$, i.e., $\widetilde{m}(p,q)$ may vanish for $\delta = 0$.

From now on, to obey the traditional vocabulary in the setting of indefinite problems, we introduce the commonly used linear operator (matrix) notations. Let A, $\widetilde{M}$ and M

be the linear operators, associated with $a(\cdot,\cdot)$, $\widetilde{m}(\cdot,\cdot)$ and $m(\cdot,\cdot)$. Denote by $\mathcal{V}', \mathcal{Q}'$ the dual spaces of $\mathcal{V}, \mathcal{Q}$ and introduce the linear operator $B : \mathcal{Q} \to \mathcal{Q}'$ and its transpose $B^T : \mathcal{Q} \to \mathcal{Q}'$, defined as

$$(Bv, q)_{\mathcal{Q}'\times\mathcal{Q}} = (v, B^T q)_{\mathcal{Q}\times\mathcal{Q}'} = b(v,q),\ \forall q \in \mathcal{Q}.$$

We pose the problem of finding $(u,p) \in \mathcal{V} \times \mathcal{Q}$, such that

$$\begin{array}{ll} a(u,v) + b(p,v) = \langle f, v\rangle_{\mathcal{V}} & \text{for all}, v \in \mathcal{V} \\ b(v,q) - c(p,q) = \langle g, q\rangle_{\mathcal{Q}} & \text{for all}, q \in \mathcal{Q} \end{array} \tag{8}$$

and its matrix form becomes then

$$\mathcal{A}\begin{bmatrix}\mathbf{u}_h\\ \mathbf{p}_h\end{bmatrix} = \begin{bmatrix}A & B^T\\ B & -\widetilde{M}\end{bmatrix}\begin{bmatrix}\mathbf{u}_h\\ \mathbf{p}_h\end{bmatrix} = \begin{bmatrix}\mathbf{f}_h\\ \mathbf{g}_h\end{bmatrix}. \tag{9}$$

As already mentioned, $\widetilde{M}$ can be a zero matrix.

It was noticed already in [20, 21] that for linear elasticity in mixed form, the element-wise computed matrix S is a very high quality approximation of the true Schur complement, even when the block A is nonsymmetric. The same result is observed in [17, 13] for the Stokes problem. There, it is also shown that the element-wise approximation S is not fully robust for the Oseen problem nor for the Cahn-Hilliard problem. The matrices arising from all those four problems are indefinite of two-by-two block structure. The block A for Stokes and Cahn-Hilliard is spd, it is nonsymmetric for the Oseen and is either symmetric or nonsymmetric for the particular elasticity problem, which is considered in [20, 21] and also in this paper.

In the sequel we address this behavior and analyze it theoretically in a more general setting. We denote the element matrix as

$$\mathcal{A}^{(\ell)} = \begin{bmatrix}A^{(\ell)} & (B^{(\ell)})^T\\ B^{(\ell)} & -\widetilde{M}^{(\ell)}\end{bmatrix}\begin{matrix}\}n_1\\ \}n_2\end{matrix}.$$

To this end we utilize a framework to study the quality of the preconditioned system $S_{\widetilde{\mathcal{A}}}^{-1}S$. Define the auxiliary matrix $\widetilde{\mathcal{A}}$, introduced first in [17] and [13]:

$$\widetilde{\mathcal{A}} = \begin{bmatrix} A^{(1)} & & & & (B^{(1)})^T R_2^{(1)} \\ & A^{(2)} & & & (B^{(2)})^T R_2^{(2)} \\ & & \ddots & & \vdots \\ & & & A^{(L)} & (B^{(L)})^T R_2^{(L)} \\ {R_2^{(1)}}^T B^{(1)} & {R_2^{(2)}}^T B^{(2)} & \cdots & {R_2^{(L)}}^T B^{(L)} & -\sum\limits_{\ell=1}^{L} {R_2^{(\ell)}}^T \widetilde{M}^{(\ell)} R_2^{(\ell)} \end{bmatrix} \tag{10}$$

Clearly, the matrix $\widetilde{\mathcal{A}} = \begin{bmatrix}\widetilde{A} & \widetilde{B}^T\\ \widetilde{B} & -\widetilde{M}\end{bmatrix}$ is constructed in such a way that its Schur complement is exactly S. We note that $\mathcal{A}$ and $\widetilde{\mathcal{A}}$ have the same 22-block. Further, $\mathcal{A}$ and $\widetilde{\mathcal{A}}$ are related algebraically in the following way. Define a matrix $C = \begin{bmatrix}C_1 & 0_{12}\\ 0_{12}^T & I_2\end{bmatrix}$, where I_2

is the identity matrix of order N_2, 0_{12} is a rectangular zero matrix of order $(L\,n_1, N_2)$. The block C_1 is of order $(L\,n_1, N_1)$ and is constructed as follows

$$C_1^T = \left[{R_1^{(1)}}^T \quad {R_1^{(2)}}^T \quad \dots \quad {R_1^{(L)}}^T \right].$$

Note, that C is full-rank by construction. A straightforward computation reveals that

$$\mathcal{A} = C^T \widetilde{\mathcal{A}} C = \begin{bmatrix} C_1^T \widetilde{A} C_1 & C_1^T \widetilde{B}^T \\ \widetilde{B} C_1 & -\widetilde{M} \end{bmatrix} \begin{matrix} \}n_1 \\ \}n_2 \end{matrix}.$$

With these notations we have

$$S = \widetilde{M} + \widetilde{B}\widetilde{A}^{-1}\widetilde{B}^T \tag{11}$$

$$S_{\mathcal{A}} = \widetilde{M} + BA^{-1}B^T = \widetilde{M} + \widetilde{B}C_1 \left(C_1^T A C_1\right)^{-1} C_1^T \widetilde{B}^T, \tag{12}$$

where S and $S_{\mathcal{A}}$ are the negative Schur complements of $\widetilde{\mathcal{A}}$ and $\widetilde{\mathcal{A}}$ respectively.

Assume now that A is spd. The following auxiliary result is helpful.

For any spd matrix $A(n,n)$ and any rectangular matrix $X(n,m)$ of full rank there holds $X\left(X^*AX\right)^{-1}X^* \le A^{-1}$ in positive definite sense. Here X^* denotes the complex conjugate of X.

Proof Consider the matrix

$$\widehat{\mathcal{A}} = \begin{bmatrix} A^{-1} & X \\ X^* & X^*AX \end{bmatrix}.$$

By assumption A, A^{-1} and X^*AX are positive definite. The matrix $\widehat{\mathcal{A}}$ is positive definite by construction and it follows that its Schur complement is also positive definite. Thus, $A^{-1} - X\left(X^*AX\right)^{-1}X^* \ge 0$ and the result $A^{-1} \ge X\left(X^*AX\right)^{-1}X^*$ follows directly. ■

By assuming that $\widetilde{M}$ is symmetric positive semi-definite (spsd), and choosing $X \equiv C_1$ and $A \equiv \widetilde{A}$ we obtain that $C_1\left(C^T\widetilde{\mathcal{A}}C\right)^{-1}C_1^T \le \widetilde{A}^{-1}$, i.e.,

$$\mathbf{z}^T C_1 \left(C^T\widetilde{\mathcal{A}}C\right)^{-1} C_1^T \mathbf{z} \le \mathbf{z}^T \widetilde{A}^{-1}\mathbf{z}, \quad \forall \mathbf{z} \ne 0.$$

Then,

$$\begin{aligned} S_{\mathcal{A}} - S &= \widetilde{B}\left[C_1\left(C^T\widetilde{\mathcal{A}}C\right)^{-1}C_1^T - \widetilde{A}^{-1}\right]\widetilde{B} \\ &\le \widetilde{B}\left(\widetilde{A}^{-1} - \widetilde{A}^{-1}\right)\widetilde{B}^T = 0. \end{aligned}$$

Thus, there holds $\alpha S_{\mathcal{A}} \le S$ with $\alpha = 1$ and the result holds true for any spsd $\widetilde{M}$ or $\widetilde{M} = 0$. The bound is confirmed by the numerical experiments for the Cahn-Hilliard problem and the Stokes problem in [13, 17] and for the mixed elasticity in Section 5. For those problems $\mathcal{A}$ and $\widetilde{M}$ are spd and spsd.

Next we estimate β in $S \le \beta S_{\mathcal{A}}$. In order to derive an upper bound, we use the already imposed assumptions on $b(v,p)$ to be uniformly bounded and that a discrete

Ladyshenskaya-Babuška-Brezzi (LBB) condition holds. Below we use the formulation of the two conditions as in [22] and assume that there exist constants β_1 and β_2, $0 < \beta_1 \le \beta_2$ such that

$$\beta_1 \le \frac{p^T B A^{-1} B^T p}{p^T M_p\, p} \le \beta_2, \tag{13}$$

where M_p is the mass matrix corresponding to the variable p. Here β_1 and β_2 are independent of h and other problem parameters but may depend on the domain of definition of the differential problem, (Ω).

Clearly, relation (13) holds also on element level. We use the right inequality to obtain

$$\frac{(R_2^{(\ell)} p_m)^T B^{(\ell)} \left(A^{(\ell)}\right)^{-1} (B^{(\ell)})^T R_2^{(\ell)} p_m}{(R_2^{(\ell)} p_m)^T M_p^{(\ell)} R_2^{(\ell)} p_m} \le \beta_2^{(\ell)} \le \widetilde{\beta}_2, \tag{14}$$

where $\widetilde{\beta}_2 = \max(\beta_2^{(1)}, \cdots, \beta_2^{(L)})$ and $M_p^{(\ell)}$ are the local mass matrices. From (14) we obtain

$$\begin{aligned}\sum_{\ell=1}^{M} (R_2^{(\ell)} p_m)^T B^{(\ell)} \left(A^{(\ell)}\right)^{-1} (B^{(\ell)})^T (R_2^{(\ell)} p_m) &\le \widetilde{\beta}_2 \sum_{\ell=1}^{M} (R_2^{(\ell)} p_m)^T M_p^{(\ell)} (R_2^{(\ell)} p_m)\\ &\le d\widetilde{\beta}_2 p^T M_p p. \end{aligned}\tag{15}$$

Here d is the connectivity of the discretization mesh (maximum number of neighboring elements per mesh point). In the case of a quadrilateral mesh, $d = 4$. We next apply the left inequality in the LBB condition to (15),

$$d\widetilde{\beta}_2 p^T M_p p \le d\frac{\widetilde{\beta}_2}{\beta_1} p^T B A^{-1} B^T p. \tag{16}$$

Thus, under the same assumption as $\mathcal{A}$ and $\widetilde{\mathcal{A}}$ we obtain that

$$\sum_{\ell=1}^{N} (R_2^{(\ell)})^T B^{(\ell)} \left(A^{(\ell)}\right)^{-1} (B^{(\ell)})^T R_2^{(\ell)} \le d\,\frac{\widetilde{\beta}_2}{\beta_1} B A^{-1} B^T. \tag{17}$$

Finally, we conclude that

$$S \le d\,\frac{\widetilde{\beta}_2}{\beta_1} S_{\mathcal{A}}. \tag{18}$$

We combine the above results in a theorem.

Theorem 2.1 *Consider the mixed variational problem (8) and the resulting algebraic system of equations (9). Let S, computed as in (6), be the element-wise approximation of the iterative Schur complement $S_{\mathcal{A}}$ of the matrix $\mathcal{A}$ in (9). Under the assumption that the bilinear form $a(u, v)$ is coercive and bounded, thus the block A is symmetric and positive definite and that the bilinear form $\widetilde{m}(p, q)$ is bounded and nonnegative, i.e., $\widetilde{M}$ spsd, there holds that*

$$S_{\mathcal{A}} \le S.$$

Under the assumption that the bilinear form $b(u,p)$ in (8) is bounded and the variational problem is discretized with a stable pair of finite element spaces, thus, the relations (14) hold true, then

$$S \leq \beta S_{\mathcal{A}},$$

where $\beta = d\,\frac{\widetilde{\beta}_2}{\beta_1}$, d is the maximum number of neighboring elements in the discretization mesh, β_1 is the discrete LBB constant and $\widetilde{\beta}_2 = \max(\beta_2^{(1)}, \cdots, \beta_2^{(L)})$ with $\beta_i^{(\ell)}$ being the upper bound constants for the local element-wise representation of $b(u,p)$.

Remark 2.2 *Discussion regarding the upper bound in (18)*
The theoretical estimate of the upper bound β is based on the local and global LBB constants. These are independent of the discretization parameter but depend on the domain of the problem. Therefore, as the exact values of the discrete LBB constants are in general now known the question regarding the sharpness of the bound arises naturally. There are some available estimates of the LBB constants for the continuous problem but only for special domains, such as ovals and discs. For rectangular domains in two dimensions, as in our case, there exist an upper bound,

$$0 < \beta_1 \leq \frac{1}{2} - \frac{1}{\pi} \approx 0.18169.$$

Further, it is shown that for elongated rectangular domains with aspect ratio $0 \leq r \leq 1$, that

$$sin^2(\frac{1}{2}arctan\, r) \leq \beta_1 \leq 1 - \frac{sinh\,\rho}{\rho\, cosh\,\rho}$$

with $\rho = \frac{r\pi}{1}$, which means that the LBB constant approaches zero as $O(r^2)$ as $r \to 0$. For polyhedral domains with corner angles $\omega_c \in (0, 2\pi)$ the continuous LBB constant satisfies relations of the form

$$0 < \beta_1 \leq \min_c \left(\frac{1}{2} - \frac{sin\,\omega_c}{2\omega_c}\right)^{1/2}.$$

For more details and a recent study on the values of the LBB constant in the continuous case, we refer to [23] and the references therein. We also refer to [24], where the dependence of the LBB condition on the domain geometry is studied.

The above results indicate that in certain cases the LBB constant can be very small, as for instance for elongated or anisotropic meshes. Thus, the bound we show in (18) can become very large, indicating that S may not be a good approximation of $S_{\mathcal{A}}$. On the other side, such meshes have to be used with precautions as these may cause a loss of accuracy in the discrete solution. For the meshes used in the numerical tests, we see that in all cases the real part of the max eigenvalue of $S_{\mathcal{A}}^{-1}S$ is below 1.5.

For the case when A is not symmetric, we provide only numerical evidence in Figures 3 and 4.

Remark 2.3 *When tested for the discrete Cahn-Hilliard problem.(cf. [13, 17]), it is noticed that while the lower bound is one, the upper bound seems to depend, although not that pronounced, on the discretization parameter.*

Those can be related to the fact that the problem there is not discretized using LBB-stable spaces. The element-wise Schur complement turns out to perform unsatisfactory for the Oseen problem too, in particular, when the convection becomes dominating.

For Stokes problem the element-wise Schur approximation is spectrally equivalent to the true Schur complement. However, the same holds for the mass matrix, which is cheaper to compute.

3. The model problem

The problem at hand, which gives rise to the large scale linear systems of algebraic equations to be solved, is the so-called glacial isostatic adjustment (GIA) model. It comprises the response of the solid Earth to redistribution of mass due to alternating glaciation and deglaciation periods. The processes that cause subsidence or uplift of the Earth surface are active today and performing numerical simulations of these phenomena is an increasingly important facet in the study of how the Earth responds to the ongoing global warming trend.

3.1. Mathematical model

The mathematical description of GIA processes employs a generalized visco-elastic rheology that accounts for viscous flow as the main mechanism of stress relaxation. Models, describing the response of the solid Earth to load, are based on the concept of isostacy, where the weight of the load is compensated for by adjustment in, and buoyancy of, the viscous mantle. The applied problem in its full complexity is described in [25], showing that the part, tested in this study, appears as a building block in the extended simulation context.

In the numerical experiments we deal with a simplified elastic GIA model of modest mathematical difficulty. We neglect the self-gravitation effects and use a two dimensional model, where the Earth is a flat elastic homogeneous material body, that can be modeled as compressible or incompressible. Namely, we consider the momentum equation for quasi-static perturbations of a homogeneous, elastic continuum in a constant gravity field

$$-\nabla \cdot \sigma - \nabla(\mathbf{u} \cdot \nabla p_0) + (\nabla \cdot \mathbf{u})\nabla p_0 = \mathbf{f} \text{ in } \Omega \subset \mathbb{R}^2 \tag{19a}$$

with boundary conditions

$$\sigma(\mathbf{u}) \cdot \mathbf{n} = \boldsymbol{\ell} \text{ on } \Gamma_L, \qquad \sigma(\mathbf{u}) \cdot \mathbf{n} = \mathbf{0} \text{ on } \Gamma_N, \tag{19b}$$

$$\mathbf{u} = \mathbf{0} \text{ on } \Gamma_D, \qquad u_1 = 0,\ \partial_x u_2 = 0 \text{ on } \Gamma_S. \tag{19c}$$

Here σ denotes stress and $\mathbf{u}$ denotes the displacement vector. The geometry of the problem is shown in Figure 1. We express Equation (19a) in terms of displacements only, using the following simplified formulation:

$$-\mu\Delta\mathbf{u} - (\mu + \lambda)\nabla(\nabla \cdot \mathbf{u}) - \nabla(\mathbf{u} \cdot \nabla p_0) + (\nabla \cdot \mathbf{u})\nabla p_0 = \mathbf{f}, \tag{20}$$

where μ and λ are the Lamé coefficients, related to the Young's modulus E and the Poisson's ratio ν as

$$\mu = \frac{E}{2(1+\nu)}, \quad \lambda = \frac{2\mu\nu}{(1-2\nu)}. \tag{21}$$

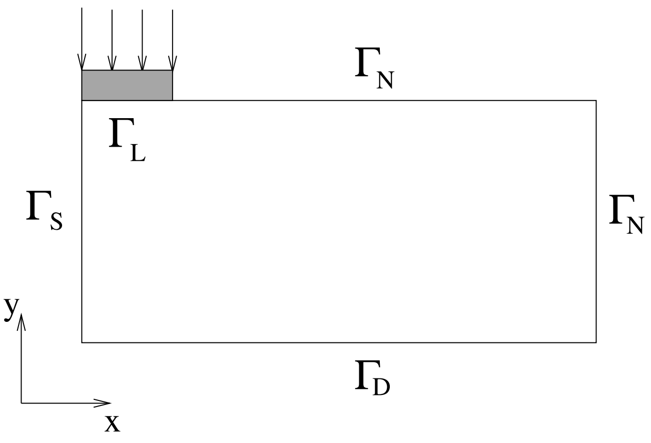


Figure 1: The geometry of the problem

Remark 3.1 *Replacing $\nabla\cdot\sigma$ by $\mu\Delta\mathbf{u}+(\mu+\lambda)\nabla(\nabla\cdot\mathbf{u}$ is correct only for smooth problems, and homogeneous materials, which is not true for the GIA model as the Earth has a layered structure. We use the formulation only for simplicity of the presentation.*

In Equation (19a), p_0 is the so-called *pre-stress*, $\mathbf{f}$ is a body force, and $\boldsymbol{\ell}$ is a surface load. The third term on the left hand side of Equation (19a) describes the *buoyancy* of the compressed material, and it vanishes for purely incompressible material since $\nabla\cdot\mathbf{u}=0$. This term is also excluded in the numerical tests.

In order to compensate for neglecting self-gravitation effects, we need to model fully incompressible materials, i.e., for which $\nu = 0.5$. Note, however, that for fully incompressible linear elastic solid the problem is not well-posed, since λ becomes unbounded. Thus, when $\nu \to 0.5$, the problem in Equation (20) becomes ill-posed, and the corresponding discrete analogue becomes extremely ill-conditioned. This is the mathematical formulation of the phenomenon known as *volumetric locking*, which may lead to erroneous results when solving the discretized Equation (20) in the nearly incompressible limit. See, for example, [26], for further details on the locking effect.

A known remedy to the locking problem is to introduce the so-called *kinematic pressure* $p = \frac{\lambda}{\mu}\nabla\cdot\mathbf{u}$, and reformulate Equation (20) as a coupled system of PDEs, which yields

$$-\mu\Delta\mathbf{u} - \mu\nabla(\nabla\cdot\mathbf{u}) - \nabla(\mathbf{u}\cdot\nabla p_0) + (\nabla\cdot\mathbf{u})\nabla p_0 - \mu\nabla p = \mathbf{f} \text{ in } \Omega \tag{22a}$$

$$\mu\nabla\cdot\mathbf{u} - \frac{\mu^2}{\lambda}p = 0 \text{ in } \Omega \tag{22b}$$

with boundary conditions (using $\varepsilon = \frac{1}{2}\left(\nabla\mathbf{u}+\nabla\mathbf{u}^T\right)$ for short notations)

$$\begin{array}{ll} [2\mu\varepsilon(\mathbf{u})+\mu pI]\cdot\mathbf{n} = \ell \text{ on } \Gamma_L & \mathbf{u}=\mathbf{0} \text{ on } \Gamma_D \\ {[2\mu\varepsilon(\mathbf{u})+\mu pI]}\cdot\mathbf{n} = \mathbf{0} \text{ on } \Gamma_N & u_1 = 0, \partial_x u_2 = 0 \text{ on } \Gamma_S. \end{array}$$

3.2. Discretization and algebraic formulation

Next the problem is discretized using finite elements. As the variational formulation is straightforward, we omit it. Since we use mixed form, we choose quadrilateral mesh and a stable finite element pair of spaces for the displacements and the pressure that satisfies the LBB condition $Q2 - Q1$. In brief, we state the finite dimensional problem in a matrix- vector form:

find the displacements $\mathbf{u}_h$ and the pressure $\mathbf{p}_h$ by solving the linear system

$$\mathcal{A}\begin{bmatrix}\mathbf{u}_h\\ \mathbf{p}_h\end{bmatrix} = \begin{bmatrix}\mathbf{r}^{(1)}\\ \mathbf{r}^{(2)}\end{bmatrix}, \quad \mathcal{A} = \begin{bmatrix}A & B^T\\ B & -\widetilde{M}\end{bmatrix}, \tag{23}$$

where $A \in \mathbb{R}^{N_u \times N_u}$ is non-symmetric, sparse and in general might be indefinite, $\widetilde{M}^{N_p \times N_p} = \frac{\mu^2}{\lambda} M_p$ and M_p is the pressure mass matrix, which is positive definite.

The problem has been earlier studied in [13, 27, 28], where independence of the convergence of the preconditioned iterative method of possible discontinuous problem coefficient is illustrated, using different discretization and different ingredients in the preconditioning strategy.

3.3. Numerical solution method and preconditioning

The system (23) is solved by flexible GMRES (FGMRES), cf. [29], referred to as the *outer solver*. The preconditioner is

$$\mathcal{B} = \begin{bmatrix}[A] & 0\\ B & -[S]\end{bmatrix}. \tag{24}$$

The notation $[\cdot]$ means that the block is solved by an inner preconitioned method. Systems with A and S are solved by inner iterations, again using FGMRES and an algebraic Multigrid (AMG) method as a preconditioner. The choice of the outer method is due to the fact that the block A may be nonsymmetric and that we use inner solvers which result in variable preconditioning.

Remark 3.2 *There are several options how to solve systems with A. Most straightforwardly we can construct an AMG preconditioner for the whole block A, which we do in this paper.*

Alternatively, we can use a block-diagonal preconditioner for A. An ordering of the degrees of freedom of the displacements first in x-direction and then in y-direction reveals a block two-by-two structure of the matrix A itself. For the purely elastic problem (with no advection terms), the diagonal blocks correspond to (mildly) anisotropic discrete Laplacians. From Korn's inequality we obtain that the block-diagonal part of A is spectrally equivalent to the whole block. For an explanation, see for instance, [9]. The block-diagonal part of A can still be used in the presence of advection, as long as it is not dominating. Again, the diagonal blocks are solved by an AMG-preconditioned iterative method. The latter approach is used in [13, 27]. The reasons not to use this procedure here are two. First of all, the block-diagonal preconditioner for A does not reduce the iterations more than what is reported here. The second reason is related to the ease of implementation. The current functionality of Deal.II does not easily support handling of additional block-structures in the block A itself. In the cited earlier works the functionality was achieved by using PETSc, which is avoided in the current experiments.

4. Implementation details

We describe next the software used to implement the solution procedure for (23), preconditioned by (24). The implementation is developed mostly in *C++*, however, it links some additional solvers that are available in FORTRAN.

4.1. Deal.II

Differential Equations Analysis Library (Deal.II) is used as the main finite element software toolbox. Deal.II is a finite element library aimed at solving systems of partial differential equations, publicly available under an Open Source license.

Deal.II provides an abstraction level that makes it feasible to change the problem parameters, such as the problem dimension, with minimal amount of effort. The functionality of the library is expanded by providing interfaces to other packages such as Trilinos, METIS [30] and others. The interfaces act as an encapsulation(wrapper) around the libraries. The wrappers have a unified signature and provide default configuration for the corresponding library.

In this study, we use Deal.II to generate the mesh, handle the degrees of freedom(DOFs) and assemble the matrix blocks, needed for the system matrix $\mathcal{A}$ and the preconditioner. For more details regarding Deal.II, see [1].

4.2. Trilinos

Trilinos is a vast collection of algorithms within an object oriented framework aimed at scientific computing problems, see [2] for details. In this study, only a small subset of the available functionalities is accessed through the Deal.II wrapper. More specifically we use, *Epetra* as sparse matrix and vector storage, *Teuchos* to pass method parameters to the solvers and the preconditioner, *ML* for multigrid preconditioning and *AZTEC* for the iterative solver (FGMRES).

Deal.II configures the Algebraic Multigrid (AMG) preconditioner from Trilinos using the following default settings: Chebyshev smoother with two pre- and post-smoothing steps, uncoupled aggregation with threshold of 0.001 and one multigrid cycle. We refer to [1] for a detailed description of the settings.

4.3. AGMG

AGMG is an aggregation-based algebraic multigrid method [3]. It provides tools for constructing preconditioner and to solve linear system of equations. AGMG aims at being efficient for large systems arising from scalar second order elliptic PDEs. The software package provides subroutines, written in FORTRAN 90, which implement the method described in [31], with further improvements from [32, 33].

Scalable parallel performance is shown for up to 370000 cores. However, the parallel implementation is only available with MPI. In this study, we compare only its serial performance.

The default settings of AGMG are the following. The coarsening is by double pairwise aggregation, with quality control, performed separately on the two components of the displacement vector. One forward and one backward Gauss-Seidel is performed for pre- and post-smoothing respectively. At each intermediate level, two Krylov accelerated iterations are performed ([34]). The main iterative solver inbuilt in the package, is the Generalized Conjugate Residual (GCR) method, [35], which works well with variable preconditioning.

4.4. PARALUTION

PARALUTION is a sparse linear algebra library with focus on exploring fine-grained parallelism, targeting modern processors and accelerators including multi/many-core CPU and GPU platforms. The main goal of the PARALUTION project is to provide a portable library containing iterative methods and preconditioners for linear systems with sparse matrices, to be run on state-of-the-art hardware. Details can be found in [4]. The library provides build-in plug-in to exports and imports Deal.II data types.

To solve the linear problem, we use the preconditioner $\mathcal{B}$, and AMG from PARALUTION. Here AMG is set to work with one V-cycle, coarse grid size of 2000 and coupling strength of 0.001. The coarsening type is set to smoothed aggregation. Multi-colored Gauss-Seidel smoother is used with a relaxation parameter set to 1.3 (cf. [36]), one pre- and and two post-smoothing steps are performed.

The AMG is constructed entirely on the CPU, while the execution can be performed on the CPU or the GPU without any code modification.

4.5. ABAQUS

ABAQUS is a general-purpose finite element analysis program, most suited for numerical modeling of the structural response. It handles various stress problems, both with static and dynamic response. The program is designed to ease the solution of complex problems, and has a simple input language, with comprehensive data checking, as well as a wide range of pre- and post- processing options and is often used for applications in Geophysics. However, enhanced numerical simulations of GIA problems are not straightforwardly performed with ABAQUS since important terms in the continuous model, such as prestress advection, cannot be added directly, leading to the necessity to modify the model in order to be able to use the package. A discussion on these issues is provided in [27, 37]. Further, in particular for small displacement elasticity formulation, ABAQUS does not handle purely incompressible materials in the sense that ν cannot be set to 0.5 but to some closer value, such as 0.4999, for instance.

ABAQUS offers a sparse direct solver in 2D which shows nearly optimal computational complexity, see the results in [27] and the performance figures in Section 5.

The tests in this study are performed with ABAQUS 6.12. We note that in the existing version of the package, iterative methods can be tested only on 3D problems. For further details, we refer to [5].

5. Numerical experiments and performance analysis

Jiekna	CPU: Intel(R) Xeon(R) 1.6GHz 12 cores
K40	CPU: Intel(R) Core(TM) i5-3550 CPU 3.30GHz 4 cores
	GPU: NVIDIA K40, 12G, 2880 cores
Tintin	CPU: 160 AMD Opteron 6220 Bulldozer 3.30GHz 2560 cores
	RAM: 64 or 128 GByte RAM per node.
	Network: Dual Gigabit Ethernet and QDR Infiniband.

The performance results for ABAQUS, Deal.II, Trilinos and AGMG are obtained using the computer resource Jiekna. To exploit parallelism on Jiekna, we employ the built-in functionality of the packages to use OpenMP. The executions are carried out on one, four and eight threads. Both the CPU and GPU tests with PARALUTION (ver 0.6.1) are executed on K40. The CPU implementation is parallelized internally in PARALUTION using OpenMP and the number of threads are set to four (without hyperthreading). The MPI implementation of the problem is tested on Tintin with up to 512 cores.

We start by illustrating the quality of the element-wise Schur complement approximation for the test problem when the block A is symmetric and nonsymmetric, corresponding to excluding or including the term $(\nabla \cdot \mathbf{u})\nabla p_0$ in (22a). Figure 1 shows the problem geometry. We compute the eigenvalues of the generalized eigenvalue problems

$$\mathcal{A}\mathbf{x} = \lambda \mathcal{B}\mathbf{x}, \quad S\mathbf{x} = \lambda S_{\mathcal{A}}\mathbf{x}.$$

The results for a rectangular domain are shown in Figure 3. Figure 3(c) illustrates the lower bound derived in Theorem (2.1) when $\mathcal{A}$ is symmetric. However, when the matrix is nonsymmetric, Figure 3(d) shows that the lower bound does not hold any longer. Figures 3(a) and 3(b) present the quality of the preconditioner $\mathcal{B}$. We see that for both symmetric and nonsymmetric matrices the eigenvalues of the preconditioned matrix are clustered around one which illustrates the high quality of the preconditioner, also confirmed by the iteration counts.

Next we conduct the same eigenvalue analysis on a domains with distorted cells. We create a geometry as in Figure 2 with a sharp corner to test the sensitivity of the upper bound in (18). Figure 4 illustrates that the distortion in the cells does not affect the quality of the preconditioner and that of the element-wise Schur complement approximation.

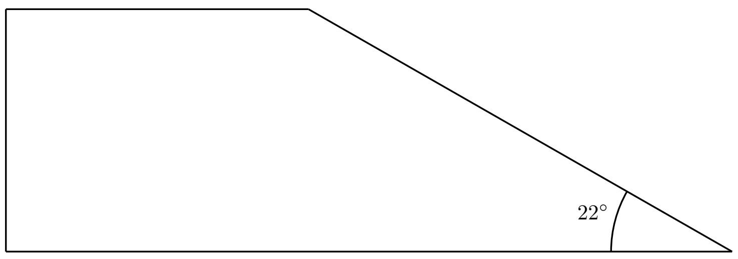
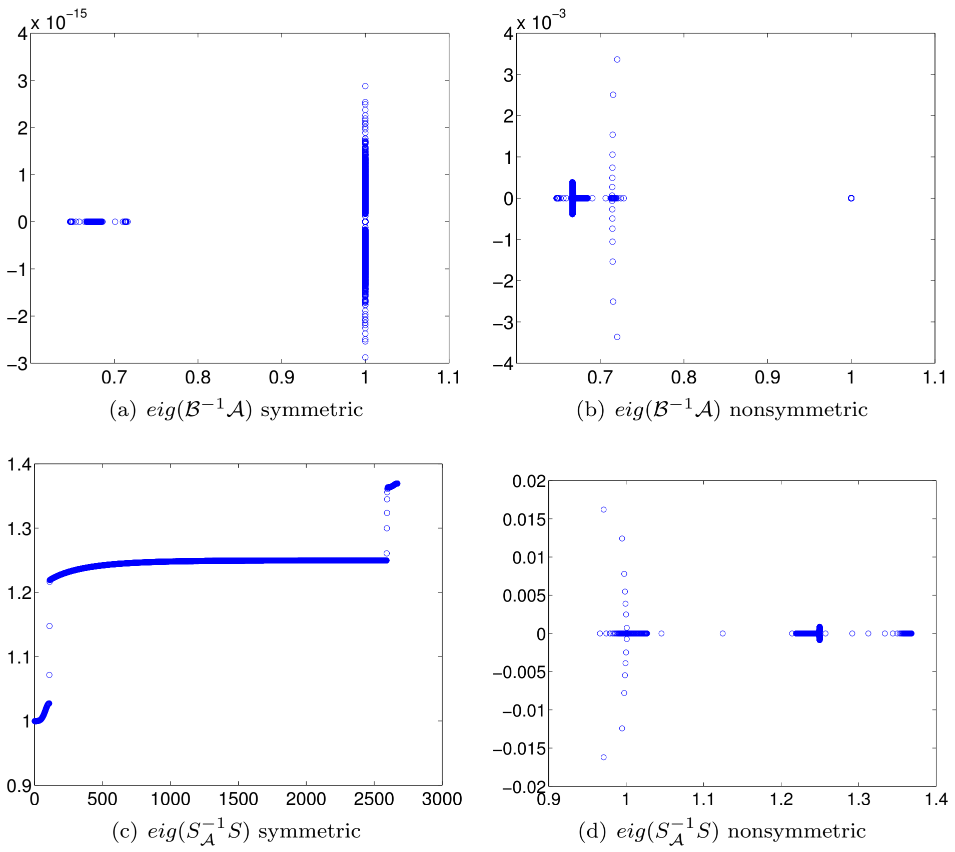


Figure 2: Domain with a small corner

Next, Table 1, provides a comparison of the simulation times between the Deal.II-Trilinos implementation and the sparse direct solver in ABAQUS. We note that due to the mixed formulation (22), while enabling the model to handle fully incompressible materials, we increase the number of degree of freedom and obtain a system of equation that is about 30% larger than what is solved in ABAQUS. The problem sizes are given in Table 1.

It is evident from Table 1 that both solution techniques scale nearly exactly linearly with the problem size. We note that although we are solving a larger system, the preconditioned iterative solver implemented in Trilinos is faster than the direct solver in ABAQUS. In the table, the number of outer iterations is followed by the number of inner iterations for A and for S, given in brackets. As is seen, the number of iterations is constant across different problem sizes, which exhibits the optimal numerical and computational efficiency of the iterative solver.

Table 2 shows the performance of the iterative solver when using Trilinos for compressible ($\nu = 0.2$) and incompressible Earth($\nu = 0.5$). The observed convergence in the



(a) $eig(\mathcal{B}^{-1}\mathcal{A})$ symmetric (b) $eig(\mathcal{B}^{-1}\mathcal{A})$ nonsymmetric

(c) $eig(S_{\mathcal{A}}^{-1}S)$ symmetric (d) $eig(S_{\mathcal{A}}^{-1}S)$ nonsymmetric

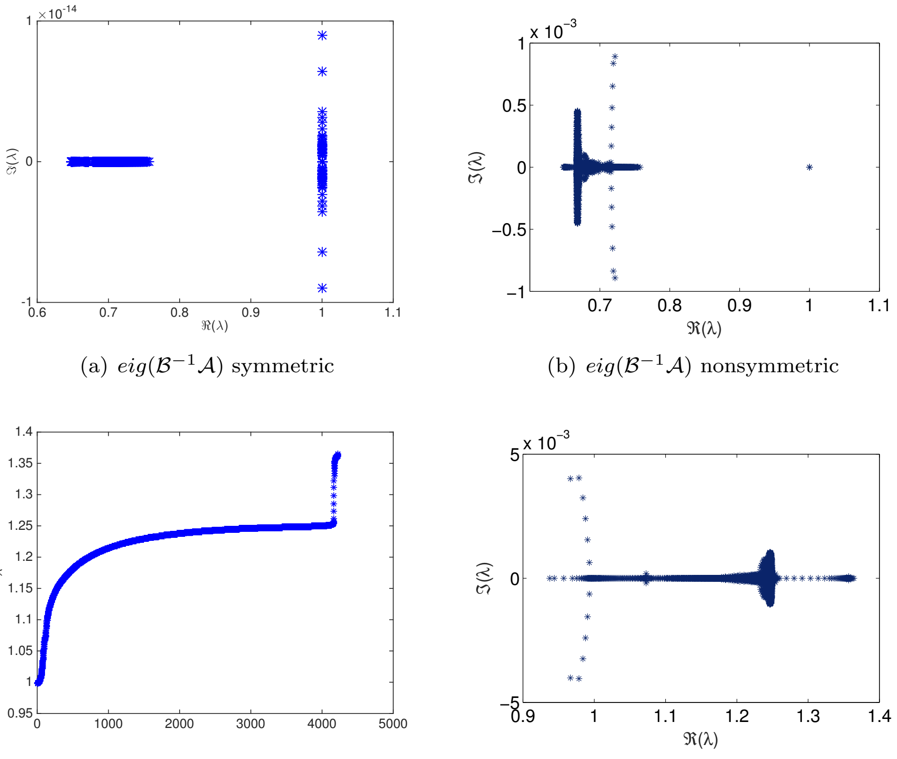
Figure 3: 90^o, Three refinements

No. of	Deal.II + Trilinos				ABAQUS		
Threads	DOFs	Iter.	Setup	Solve (2/3)	DOFs	Setup	Solve
1		10(6, 1)	5.63	34.2 (22.8)		7.44	59
4	1 479 043	10(6, 1)	5.42	23.4 (15.6)	986 626	7.49	33
8		10(6, 1)	5.41	19.1 (12.7)		7.51	28
1		10(6, 1)	23.2	150 (100.0)		29.72	269
4	5 907 203	10(6, 1)	22.3	95.5 (63.66)	3 939 330	29.93	145
8		10(6, 1)	22.1	79 (52.66)		29.94	122

Table 1: Deal.II and ABAQUS performance; compressible material

incompressible case is slightly slower due to the worse condition number of the matrix $\mathcal{A}$.

Moreover, Table 2 shows that the iterative method is numerically optimal not only across problem sizes but also across varying number of threads. Note that the solver for block A is somewhat more stable in the compressible case than in the incompressible



(a) $eig(\mathcal{B}^{-1}\mathcal{A})$ symmetric

(b) $eig(\mathcal{B}^{-1}\mathcal{A})$ nonsymmetric

(c) $eig(S_{\mathcal{A}}^{-1}S)$ symmetric

(d) $eig(S_{\mathcal{A}}^{-1}S)$ nonsymmetric

Figure 4: 22^o, three refinements

case.

On the other hand, Table 2 shows that the implementation does not scale to more than four threads which can be attributed to the memory bandwidth utilization.

Further analysis of the performance, not included in this paper, reveals that most of the time (about 90%) is spent on executing various parts of the Trilinos-AMG preconditioner for the block A. Several attempts have been done to speed this up. As a first option, the inner solver has been replaced by just one action of AMG, Second, A has been preconditioned by a block-diagonal preconditioner, as mentioned in Remark 3.2. Both approaches turn out to be inefficient as they destroy the outer convergence rate.

At last, Trilinos-AMG has been replaced by an alternate state-of-the-art AMG solver, namely, by AGMG. Table 3 presents the results of the simulation using AGMG on one thread. We see that AGMG behaves in a robust and efficient manner and it is about twice as fast.

A comparison between Tables 2 and 3 reveals that AGMG is more efficient than Trilinos-AMG. The fast convergence of AGMG allows for smaller stopping tolerance which means that for the same computational cost the system can be solved more accurately which in turn reduces the number of outer iterations and overall execution time.

Next, we show the effect of hardware accelerators (GPU) on the solution time. The

Threads	DOFs	$\nu = 0.2$			$\nu = 0.5$		
		Iterations	Setup	Solve	Iterations	Setup	Solve
1		$10(4,1)$	0.0597	0.335	$17(4,1)$	0.0587	0.567
4	23 603	$10(4,1)$	0.0596	0.186	$17(4,1)$	0.0586	0.283
8		$10(4,1)$	0.136	0.151	$17(4,1)$	0.135	0.237
1		$10(5,1)$	0.228	2.06	$18(5,1)$	0.227	3.21
4	93 283	$10(5,1)$	0.205	1.16	$18(5,1)$	0.204	1.93
8		$10(5,1)$	0.280	1.03	$18(5,1)$	0.281	1.78
1		$10(5,1)$	0.854	9.41	$18(6,1)$	0.864	14.5
4	370 883	$10(5,1)$	0.819	4.9	$18(6,1)$	0.839	10.3
8		$10(5,1)$	0.947	4.23	$18(6,1)$	0.957	8.44
1		$10(6,1)$	3.43	34.2	$18(7,1)$	3.53	65.8
4	1 479 043	$10(6,1)$	3.04	23.4	$18(7,1)$	3.07	43.8
8		$10(6,1)$	3.00	19.1	$18(7,1)$	3.03	37.4
1		$10(6,1)$	15.4	150	$18(8,1)$	15.5	324
4	5 907 203	$10(6,1)$	14.2	95.5	$18(8,1)$	14.5	217
8		$10(6,1)$	14.1	79	$18(8,1)$	14.4	177

Table 2: Trilinos performance; compressible and incompressible material

	$\nu = 0.2$			$\nu = 0.5$		
DOFs	Itr.	Setup	Solve	Itr.	Setup	Solve
23 603	$9(4,1)$	0.029	0.235	$16(4,1)$	0.249	0.412
93 283	$9(4,1)$	0.128	1.02	$16(4,1)$	1	1.76
370 883	$9(5,1)$	0.515	4.62	$16(5,1)$	3.98	7.83
1 479 043	$9(5,1)$	2.09	19.4	$16(6,1)$	16	37.1
5 907 203	$9(6,1)$	8.69	90.7	$16(6,1)$	64	164

Table 3: AGMG performance; compressible and incompressible material

simulation is run on CPU and GPU using PARALUTION on K40. The results are shown in Table 4. We see that both the CPU and the GPU implementation of PARALUTION are numerically stable. Furthermore, the GPU implementation is up to four times faster compared to the other used libraries. Comparing Table 4 with Tables 2 and 3, we see that the CPU implementation of PARALUTION is as fast as Trilinos and the time to setup AMG is smaller. One design characteristic of the implementation of the solvers and preconditioners in PARALUTION has been to keep the complexity relatively low in order to gain efficiency on GPU. The effect of this is indeed a superior performance of the GPU on the cost of convergence of the iterative method at a slower rate.

We note that the discretization of the problem, corresponding to 1 479 043 degrees of freedom, on the surface of the computational domain agrees with the placement of the surface sensors that gather data from geophysical experiments. This size fits on the GPU and the performance is fastest. Trying to solve problems with larger computational domain in 2D or 3D problems might fail due to insufficient memory on the currently available GPUs.

Finally, we present the result of running the model problem on distributed systems.

DOFs	Outer	Time on CPU (s)		Time on GPU (s)	
	iter.	Setup	Solve	Setup	Solve
23 603	24	0.04	0.39	0.16	2.8
93 283	24	0.18	1.26	0.27	2.0
370 883	24	0.75	5.07	0.9	3.8
1 479 043	25	2.9	22.4	3.7	5.8
5 907 203	25	18.4	91.5	out of memory	

Table 4: PARALUTION performance on CPU and GPU; compressible material

The necessity of this test comes from the fact that with larger sizes and realistic 3D models, the lack of memory becomes the dominating issue. The distributed memory implementation uses Deal.II and Trilinos-AMG. Table 5 shows the result of this simulation for both compressible and incompressible materials on Tintin with the number of cores ranging from 1 to 512.

We see that the numerical method is stable and the number of iterations are almost constant when changing the number of cores or the problem size. We gain speed by adding more cores but because the process is memory bound we do not fully achieve perfect speed up.

6. Conclusion

The contribution of this paper is two-fold. We consider the element-wise sparse approximation of the negative Schur complement matrix for indefinite problems and study its quality. For saddle point matrices with symmetric positive (semi-)definite blocks we show that the negative Schur complement is spectrally equivalent to the so-constructed approximation and derive spectral equivalence bounds. We also illustrate the quality of the approximation for nonsymmetric problems, where we observe the same good numerical efficiency.

Further, we provide performance results of the simulations of the elastic GIA model on various parallel computer platforms, multi-CPU, GPU and a distributed memory cluster. The implementation is based only on open source finite element and numerical linear algebra packages for sparse matrices. The major outcome of the performance study is as follows.

(i) Large-scale coupled problems can be successfully implemented using publicly available numerical linear algebra software. Compared with highly specialized and optimized commercial software, the open source libraries, included in this study, allow to enhance the mathematical model and make it more realistic, adding features that are not straightforwardly incorporated when using commercial software.

(ii) Open source numerical libraries successfully compete with highly efficient commercial packages in terms of overall simulation time even in 2D and show better price-performance ratio.

Cores	DOFs	$\nu = 0.2$			$\nu = 0.5$		
		Iterations	Setup	Solve	Iterations	Setup	Solve
1		$10(8,1)$	25.3	86.9	$17(9,1)$	25.2	149
4		$10(8,1)$	7.11	23.2	$17(9,1)$	6.91	39.4
8	1 479 043	$10(8,1)$	3.62	12.9	$17(9,1)$	3.56	20.9
16		$10(8,1)$	2.19	9.51	$17(8,1)$	2.16	22.1
32		$10(8,1)$	1.26	6.8	$17(8,1)$	2.34	11.4
1		$10(9,1)$	101	435	$17(10,1)$	101	641
4		$10(9,1)$	29.1	99.2	$17(10,1)$	29.1	257
8	5 907 203	$10(8,1)$	14.6	49.8	$17(10,1)$	14.4	92.4
16		$10(9,1)$	8.68	41.1	$17(10,1)$	8.68	93.5
32		$10(9,1)$	4.27	30.9	$17(10,1)$	4.7	49.7
64		$10(9,1)$	5.22	21.1	$17(10,1)$	4.79	42.1
4		$10(11,1)$	165	563	$17(11,1)$	169	909
8		$10(11,1)$	65.4	267	$17(11,1)$	66.2	768
16		$10(10,1)$	36.9	298	$17(11,1)$	36.6	392
32	23 610 883	$10(11,1)$	18.4	120	$17(11,1)$	17.4	197
64		$10(11,1)$	9.69	75.8	$17(12,1)$	10.8	146
128		$10(11,1)$	9.57	60.8	$17(12,1)$	7.48	83.9
256		$10(11,1)$	6.61	33.3	$17(11,1)$	6.14	63.4
32		$10(12,1)$	82.1	536	$17(13,1)$	79.8	1130
64		$10(12,1)$	40	507	$17(13,1)$	38	769
128	94 407 683	$10(12,1)$	21.5	272	$17(13,1)$	21.5	486
256		$10(11,1)$	14	98.9	$17(12,1)$	14.1	249
512		$10(11,1)$	9.79	72.9	$17(13,1)$	11	137

Table 5: MPI performance and scalibility results using Deal.II and Trilinos-AMG; compressible and incompressible material

(iii) For large enough problem sizes that fit into the memory of the GPU, the GPU implementation of PARALUTION performs noticeably faster than the other tested CPU implementations.

(iv) None of the tested OpenMp-based CPU implementations show satisfactory scalability. The reason is that the application is memory-bound.

(v) The performance tests (with Trilinos-AMG) are also extended using MPI and confirm the expected scalable behavior.

Acknowledgments

The authors acknowledge the work done by Dimitar Lukarski implementing the inner-outer solution method in PARALUTION, developing the interface between Deal.II and PARALUTION, as well as for his continuous interest and support while carrying out the performance study. We acknowledge the help by Yvan Notay who provided the block-version of his AGMG code, that is particularly suited for systems with vector unknowns, as in linear elasticity.

Parts of the computations were performed on resources provided by SNIC through Uppsala Multidisciplinary Center for Advanced Computational Science (UPPMAX) under Projects p2009040 and p2011076.

[1] W. Bangerth, G. Kanschat, R. Hartmann, deal.II differential equations analysis library.
URL http://www.dealii.org

[2] M. A. Heroux, J. M. Willenbring, Trilinos users guide, Tech. Rep. SAND2003-2952, Sandia National Laboratories (2003).
URL http://trilinos.sandia.gov

[3] Y. Notay, AGMG software and documentation.
URL http://homepages.ulb.ac.be/~ynotay/AGMG/

[4] D. Lurkarski, Paralution project.
URL http://www.paralution.com

[5] Abaqus FEA.
URL http://www.3ds.com/

[6] M. Benzi, G. H. Golub, J. Liesen, Numerical solution of saddle point problems, Acta Numerica 14 (2005) 1–137. doi:10.1017/S0962492904000212.

[7] O. Axelsson, M. Neytcheva, Eigenvalue estimates for preconditioned saddle point matrices, Numerical Linear Algebra with Applications 13 (4) (2006) 339–360. doi:10.1002/nla.469.

[8] F. Brezzi, On the existence, uniqueness and approximation of saddle-point problems arising from Lagrangian multipliers, Rev. Française Automat. Informat. Recherche Opérationnelle Sér. Rouge 8 (R-2) (1974) 129–151.

[9] O. Axelsson, Iterative Solution Methods, Cambridge University Press, 1994.

[10] Y. Saad, M. H. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Statist. Comput. 7 (3) (1986) 856–869. doi:10.1137/0907058.

[11] O. Axelsson, R. Blaheta, M. Neytcheva, Preconditioning of boundary value problems using elementwise Schur complements, SIAM J. Matrix Anal. Appl. 31 (2) (2009) 767–789. doi:10.1137/070679673.

[12] J. K. Kraus, Algebraic multilevel preconditioning of finite element matrices using local Schur complements, Numerical Linear Algebra with Applications 13 (1) (2006) 49–70. doi:10.1002/nla.462.

[13] M. Neytcheva, On element-by-element schur complement approximations, Linear Algebra and its Applications 434 (11) (2011) 2308 – 2324, special Issue: Devoted to the 2nd NASC 08 Conference in Nanjing (NSC). doi:http://dx.doi.org/10.1016/j.laa.2010.03.031.

[14] J. Kraus, Additive Schur complement approximation and application to multilevel preconditioning, SIAM J. Sci. Comput. 34 (6) (2012) A2872–A2895. doi:10.1137/110845082.

[15] P. Boyanova, S. Margenov, M. Neytcheva, Robust AMLI methods for parabolic Crouzeix-Raviart FEM systems, Journal of Computational and Applied Mathematics 235 (2) (2010) 380 – 390, Computational Algorithms. doi:http://dx.doi.org/10.1016/j.cam.2010.05.039.

[16] J. Kraus, Additive Schur Complement Approximation for Elliptic Problems with Oscillatory Coefficients, Vol. 7116, Springer Berlin Heidelberg, 2012, pp. 52–59.

[17] M. Neytcheva, M. Do-Quang, H. Xin, Element-by-Element Schur Complement Approximations for General Nonsymmetric Matrices of Two-by-Two Block Form, Vol. 5910, Springer Berlin Heidelberg, 2010, pp. 108–115.

[18] V. Eijkhout, P. Vassilevski, The role of the strengthened Cauchy–Buniakowskii–Schwarz inequality in multilevel methods, SIAM Review 33 (3) (1991) 405–419. arXiv:http://dx.doi.org/10.1137/1033098, doi:10.1137/1033098.

[19] O. Axelsson, M. Neytcheva, Eigenvalue estimates for preconditioned saddle point matrices, Numerical Linear Algebra with Applications 13 (4) (2006) 339–360. doi:10.1002/nla.469.

[20] M. Neytcheva, E. Bängtsson, Preconditioning of nonsymmetric saddle point systems as arising in modelling of viscoelastic problems, Electronic Transactions on Numerical Analysis 29 (2008) 193–211.

[21] E. Bängtsson, M. Neytcheva, An agglomerate multilevel preconditioner for linear isostasy saddle point problems, in: Large-scale scientific computing, Vol. 3743 of Lecture Notes in Comput. Sci., Springer, Berlin, 2006, pp. 113–120. doi:10.1007/11666806_11.

[22] A. Klawonn, G. Starke, Block triangular preconditioners for nonsymmetric saddle point problems: field-of-values analysis, Numerische Mathematik 81 (4) (1999) 577–594.

[23] M. Costabel, M. Crouzeix, M. Dauge, Y. Lafranche, The inf-sup constant for the divergence on corner domainsarXiv:1402.3659.

[24] E. V. Chizhonkov, M. A. Olshanskii, On the domain geometry dependence of the LBB condition,

ESAIM: Mathematical Modelling and Numerical Analysis 34 (2000) 935–951. doi:10.1051/m2an:2000110.

[25] A. Dorostkar, D. Lukarski, B. Lund, M. Neytcheva, Y. Notay, P. Schmidt, Parallel performance study of block-preconditioned iterative methods on multicore computer systems, Tech. Rep. 2014-007, Department of Information Technology, Uppsala University (Mar. 2014).

[26] D. Braess, Finite elements, Theory, fast solvers, and applications in elasticity theory, 3rd Edition, Cambridge University Press, Cambridge, 2007. doi:10.1017/CBO9780511618635.

[27] E. Bängtsson, B. Lund, A comparison between two solution techniques to solve the equations of glacially induced deformation of an elastic Earth, International Journal for Numerical Methods in Engineering 75 (4) (2008) 479–502. doi:10.1002/nme.2268.

[28] E. Bängtsson, M. Neytcheva, Numerical simulations of glacial rebound using preconditioned iterative solution methods, Appl. Math. 50 (3) (2005) 183–201. doi:10.1007/s10492-005-0013-3.

[29] Y. Saad, A flexible inner-outer preconditioned GMRES algorithm, SIAM J. Sci. Comput. 14 (2) (1993) 461–469. doi:10.1137/0914028.

[30] G. Karypis, V. Kumar, MeTis: Unstructured Graph Partitioning and Sparse Matrix Ordering System, Version 4.0 (2009).
URL http://www.cs.umn.edu/~metis

[31] Y. Notay, An aggregation-based algebraic multigrid method, Electron. Trans. Numer. Anal. 37 (2010) 123–146.

[32] A. Napov, Y. Notay, An algebraic multigrid method with guaranteed convergence rate, SIAM J. Sci. Comput. 34 (2) (2012) A1079–A1109. doi:10.1137/100818509.

[33] Y. Notay, Aggregation-based algebraic multigrid for convection-diffusion equations, SIAM J. Sci. Comput. 34 (4) (2012) A2288–A2316. doi:10.1137/110835347.

[34] Y. Notay, P. S. Vassilevski, Recursive Krylov-based multigrid cycles, Numer. Lin. Alg. Appl. 15 (2008) 473–487.

[35] S. C. Eisenstat, H. C. Elman, M. H. Schultz, Variational iterative methods for nonsymmetric systems of linear equations 20 (1983) 345–357.

[36] D. Lukarski, Parallel Sparse Linear Algebra for Multi-core and Many-core Platforms – Parallel Solvers and Preconditioners, Ph.D. thesis, Karlsruhe Institute of Technology (Jan. 2012).

[37] P. Wu, Using commercial finite element packages for the study of earth deformations, sea levels and the state of stress, Geophysical Journal International 158 (2) (2004) 401–408. doi:10.1111/j.1365-246X.2004.02338.x.