Enhanced Relaxed Physical Factorization preconditioner for coupled poromechanics

Matteo Frigo\textsuperscript{a}, Nicola Castelletto\textsuperscript{b}, Massimiliano Ferronato\textsuperscript{a}

\textsuperscript{a}Department ICEA, University of Padova, Padova, IT
\textsuperscript{b}Atmospheric, Earth and Energy Division, Lawrence Livermore National Laboratory, United States

Abstract
The relaxed physical factorization (RPF) preconditioner is a recent algorithm allowing for the efficient and robust solution to the block linear systems arising from the three-field displacement-velocity-pressure formulation of coupled poromechanics. For its application, however, it is necessary to invert blocks with the algebraic form $\hat{C} = (C + \beta F F^T)$, where $C$ is a symmetric positive definite matrix, $F F^T$ a rank-deficient term, and $\beta$ a real non-negative coefficient. The inversion of $\hat{C}$, performed in an inexact way, can become unstable for large values of $\beta$, as it usually occurs at some stages of a full poromechanical simulation. In this work, we propose a family of algebraic techniques to stabilize the inexact solve with $\hat{C}$. This strategy can prove useful in other problems as well where such an issue might arise, such as augmented Lagrangian preconditioning techniques for Navier-Stokes or incompressible elasticity. First, we introduce an iterative scheme obtained by a natural splitting of matrix $\hat{C}$. Second, we develop a technique based on the use of a proper projection operator annihilating the near-kernel modes of $\hat{C}$. Both approaches give rise to a novel class of preconditioners denoted as Enhanced RPF (ERPF). Effectiveness and robustness of the proposed algorithms are demonstrated in both theoretical benchmarks and real-world large-size applications, outperforming the native RPF preconditioner.

Keywords: Preconditioning, Krylov subspace methods, Poromechanics

1. Introduction
Numerical simulation of Darcy’s flow in a porous medium coupled with quasi-static mechanical deformation is based on poromechanics theory \cite{1, 2}. We consider saturated single-phase flow of a slightly compressible fluid through a deformable medium. The set of governing equations for the displacement-velocity-pressure formulation of the poroelastic problem, e.g., \cite{3–8} reads:

\begin{align}
\nabla \cdot (C_{dr} : \nabla^t u - b p \mathbf{1}) &= 0 \quad \text{in } \Omega \times I \quad \text{(equilibrium),} \\
\mu \kappa^{-1} \cdot q + \nabla p &= 0 \quad \text{in } \Omega \times I \quad \text{(Darcy’s law),} \\
b \nabla \cdot \dot{u} + S_{\epsilon} \partial_p + \nabla \cdot q &= f \quad \text{in } \Omega \times I \quad \text{(continuity),}
\end{align}

with $\Omega \subset \mathbb{R}^3$ a bounded domain and $I = (0, t_{\text{max}})$ an open time interval. In (1), $C_{dr}$ is the rank-four elasticity tensor, $\nabla^t$ is the symmetric gradient operator, $b$ is the Biot coefficient, and $\mathbf{1}$ is the rank-two identity tensor; $\mu$ and $\kappa$ are the fluid viscosity and the rank-two permeability tensor, respectively; $S_{\epsilon}$ is the constrained specific storage coefficient, i.e., the reciprocal of Biot’s modulus, and $f$ the fluid source term. The primary unknowns are denoted by $u$ (displacement), $p$ (pressure), $\dot{u}$ (velocity), and $q$ (divergence of the fluid flux).
The initial condition can be given as $q$ (Darcy’s velocity), and $p$ (excess pore pressure). The problem is completed with proper boundary conditions:

\begin{align}
\mathbf{u} = \mathbf{u}_0 & \quad \text{on } \Gamma_u \times I, \quad \text{(prescribed displacement)} \tag{2a} \\
(C_{dr} : \nabla \mathbf{u} - b p I) \cdot \mathbf{n} = \mathbf{t} & \quad \text{on } \Gamma_t \times I, \quad \text{(prescribed traction)} \tag{2b} \\
\mathbf{q} \cdot \mathbf{n} = \bar{q} & \quad \text{on } \Gamma_q \times I, \quad \text{(prescribed discharge)} \tag{2c} \\
p = \bar{p} & \quad \text{on } \Gamma_p \times I, \quad \text{(prescribed pressure)} \tag{2d}
\end{align}

where $\mathbf{u}_0$, $\mathbf{t}$, $\bar{q}$, and $\bar{p}$ are the boundary displacement, traction, Darcy velocity and excess pore pressure, respectively, on the portions $\Gamma_u$, $\Gamma_t$, $\Gamma_q$, and $\Gamma_p$ of the frontier $\partial \Omega$ such that $\Gamma_u \cup \Gamma_t = \Gamma_q \cup \Gamma_p = \partial \Omega$ and $\Gamma_u \cap \Gamma_t = \Gamma_q \cap \Gamma_p = \emptyset$.

The initial condition can be given as

$$b \nabla \cdot \mathbf{u}(x, 0) + S_e p(x, 0) = b \nabla \cdot \mathbf{u}_0 + S_e p_0, \quad x \in \overline{\Omega}, \tag{3}$$

with $\mathbf{u}_0$ and $p_0$ the initial displacement and excess pore pressure field. While equation (3) specifies the initial fluid content of the medium [1], in practice $p_0$ is often directly measured or computed, and $\mathbf{u}_0$ is then obtained so as to satisfy the momentum balance (1a)—see, e.g., [9, 10]. The focus of this work is the iterative solution of the linear algebraic system arising from the discretization of the initial boundary value problem (1)-(3). In particular, we consider the block linear system $\mathbf{Ax} = \mathbf{F}$ obtained by combining a mixed finite element discretization in space with implicit integration in time using the $\theta$-method [11]:

$$\mathbf{A} = \begin{bmatrix}
K & 0 & -Q \\
0 & A & -B \\
Q^T & \gamma B^T & P
\end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix}
\mathbf{u} \\
\mathbf{q} \\
p
\end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix}
f_u \\
n_q \\
 f_p
\end{bmatrix}, \quad \tag{4}$$

where $\mathbf{u} \in \mathbb{R}^{n_u}$, $\mathbf{q} \in \mathbb{R}^{n_q}$ and $p \in \mathbb{R}^{n_p}$ denote the vectors containing the unknown $n_u$ displacement, $n_q$ Darcy velocity, and $n_p$ pressure degrees of freedom, respectively, at discrete time $t_{n+1}$, and $\theta = \theta \Delta t$, with $\Delta t = (t_{n+1} - t_n)$ the time integration step size and $\theta$ a real parameter ($1/2 \leq \theta \leq 1$). In (4), $K$ is the classic small-strain structural stiffness matrix, $A$ is the (scaled) velocity mass matrix, $P$ is the (scaled) pressure mass matrix, $Q$ is the poromechanical coupling block and $B$ is the Gram matrix. We employ $Q_1$ elements for the displacement field, $RT_0$ elements for the pressure field, and $P_0$ elements for the pressure field. Assuming a linear elastic law for the mechanical behavior of the porous medium leads to a symmetric positive definite (SPD) matrix $K$. Matrix $A$ is SPD as well, while $P$ is diagonal with non-negative entries. Additional details and explicit expressions for matrices in $\mathbf{A}$ and vectors in the block right-hand side $\mathbf{F}$ can be found in [11, 12].

The two-field displacement-pressure formulation discretized by the continuous Galerkin (CG) finite elements is probably the most common technique for numerical modeling of coupled poromechanics [13]. In contrast to the CG method, the selected three-field formulation yields locally mass-conservative velocity fields and is robust for highly heterogeneous permeability fields, a key requirement in several geoscience applications, e.g., [14, 15]. We note that an enriched Galerkin (EG) method can be also used to fix this drawback of the CG method [16, 17]. If the selected combination of discretization spaces does not intrinsically satisfy the inf-sup stability in the undrained limit [18], proper stabilization strategies can be introduced to eliminate the spurious oscillation modes arising in the pressure solution. These techniques can be subdivided into two classes, based on whether they: (i) enrich the discrete variable spaces so that the Ladyzhenskaya-Babuška-Brezzi (LBB) condition is met, e.g., [19, 20], or (ii) introduce a stabilization term to guarantee the solvability of the resulting saddle-point problem, e.g., [21, 22]. The first strategy is mathematically elegant and robust, but can modify the algebraic structure of problem (4). The selected $Q_1$-$RT_0$-$P_0$ mixed discretization can be effectively stabilized following the second strategy by adding a proper term to the mass balance equation [23]. This approach has the advantage of having a minor impact on the algebraic structure of the problem (4), so that the solution algorithms developed in this work can be used independently of the presence of such a contribution.

Our goal is the efficient iterative solution of the linear system (4) with the aid of preconditioned Krylov subspace methods. Since $\mathbf{A}$ is non-symmetric, or indefinite if properly symmetrized, a global Bi-CGSTab [24] or GMRES [25] algorithm can be used as a solver. Over the past decade, a growing interest has regarded the implicit solution of coupled poromechanics, with the introduction of a number of methods built on both fully-algebraic approaches and
discretization-based strategies. The proposed algorithms can be grouped into two main categories: (i) sequential-implicit methods, in which the primary variables are updated one at a time, iterating between governing equations [9, 26–32]; and (ii) fully-coupled approaches, which solve the global system of equations simultaneously for all the primary unknowns, with an emphasis on preconditioned Krylov solvers with scalable and parameter-robust methods [11, 12, 18, 33–46]. Sequential-implicit methods exhibit a linear convergence, but can take advantage of using distinct, and specialized, codes for the structural and the flow models. In contrast, fully-coupled approaches ensure unconditional stability with a super-linear convergence, but require the design of dedicated preconditioning operators. This research field, in particular, has attracted an increasing interest from the scientific community, with the introduction of a number of different algorithms in the last few years. Just to cite a few recent examples of preconditioners for coupled poromechanics, we mention spectrally-equivalent block diagonal and block triangular strategies [18, 33, 38, 41, 43, 45], multigrid-based scalable methods [39, 40, 46], or block preconditioners combining physically-based and algebraic tricks to construct different Schur complement approximations [11, 12, 23, 44]. A recent novel approach belonging to the category of fully-coupled methods is the relaxed physical factorization (RPF) preconditioner [47]. The distinctive feature of the RPF algorithm is that it does not rely on accurate sparse approximations of the Schur complements, with the convergence accelerated by setting a nearly-optimal relaxation parameter $\alpha$. Similar approaches were originally developed for the solution of the linear system arising from the discretization of the Navier-Stokes equations [48, 49].

The RPF technique has been shown very efficient in solving challenging problems with severe material heterogeneities and proved competitive with other well-established block preconditioners [47]. However, a performance degradation can be observed when the time-step approaches 0 (undrained conditions) or $+\infty$ (fully drained conditions). This behavior is due to the increase of the ill-conditioning of the inner blocks, which causes the worsening of the RPF efficiency and robustness. In fact, the native RPF preconditioner requires the solution of inner systems in the form $\tilde{C} = (C + \beta FF^T)$, where $C$ is SPD and $FF^T$ is rank-deficient, and $\beta$ is a scalar coefficient that may tend to infinity in the limit case of $\frac{\tau}{t_c} \ll 1$ (undrained conditions) or $\frac{\tau}{t_c} \gg 1$ (uncoupled consolidation), with $t_c$ the characteristic consolidation time [2]. Problematic issues with a very similar algebraic origin may frequently occur in several other important applications beyond coupled poromechanics and independently of the selected discretization spaces, e.g., in augmented Lagrangian approaches for the Navier-Stokes equations or mixed formulations of incompressible elasticity [50–54]. In this paper, we advance two methods to stabilize the solves with the inner blocks $\tilde{C}$. In the first approach, a natural splitting of $\tilde{C}$ is proposed to obtain a convergent sequential iterative scheme. In the second approach, a proper projection operator is introduced to annihilate the near-kernel modes of $\tilde{C}$. We would like to emphasize that the proposed two strategies are fully algebraic, hence they are naturally amenable to a generalization to other applications where such an issue might arise.

The paper is organized as follows. In Section 2 a brief review of the RPF preconditioner is provided, with a focus on the numerical challenges associated with large and small time-step values. Section 3 provides the theoretical basis for the two methods introduced for the RPF stabilization, with the related properties, performance and robustness investigated in Section 4 through both academic benchmarks and real-world applications. Finally, a few conclusive remarks close the paper in Section 5.

2. The RPF Preconditioner

In this section, we briefly revisit the RPF preconditioner [47] recalling only the aspects necessary for the remainder of this work. In factored form, the RPF preconditioner $M^{-1}$ reads:

$$M^{-1} = \alpha (M_1M_2)^{-1} = \alpha \left( \begin{array}{ccc} K & 0 & -Q \\ 0 & \alpha I_q & 0 \\ Q^T & 0 & \alpha I_p \end{array} \right) \left( \begin{array}{ccc} \alpha I_u & 0 & 0 \\ 0 & A & -B \\ 0 & \gamma B^T & \alpha I_p \end{array} \right)^{-1},$$

(5)

where $\alpha > \|P\|_\infty$ is the relaxation parameter, and $I_u$, $I_q$, and $I_p$ are the identity matrix in $\mathbb{R}^{n_u \times n_u}$, $\mathbb{R}^{n_q \times n_q}$, and $\mathbb{R}^{n_p \times n_p}$, respectively. Setting $\alpha$ is key to accelerate the convergence of the Krylov subspace method preconditioned with (5) to solve (4). Though the convergence rate of non-symmetric Krylov solvers does not depend only on the eigenvalue distribution of the preconditioned matrix, the computational practice suggests that a clustered eigenspectrum rarely leads to slow convergence. Hence, the criterion for setting $\alpha$ relies on clustering as much as possible the eigenvalues.
of the preconditioned matrix $T = M^{-1}A$ around 1. To this aim, $\alpha$ can be selected such that the trace of $T$ is as close as possible to the system size:

$$\alpha = \arg \min_{\alpha \geq 0} \left( n_u + n_q + n_p - \text{tr} [T] \right).$$

(6)

Following the arguments developed in [47], equation (6) can be rewritten as:

$$\alpha = \arg \min_{\alpha \geq 0} \left( n_p - \text{tr} [Z_\alpha] \right),$$

(7)

where

$$Z_\alpha = \alpha (\alpha I_p + S_K)^{-1} S (\alpha I_p + \gamma S_A)^{-1},$$

(8)

and

$$S_K = Q^T K^{-1} Q,$$

(9)

$$S_A = B^T A^{-1} B,$$

(10)

$$S = P + S_K + \gamma S_A.$$  

(11)

Finding analytically $\alpha$ as in (7) is not practically feasible, because the matrices (9), (10), (11) are dense. However, for the sake of $\alpha$ estimate only, we can replace $S_K$ and $S_A$ with diagonal matrices, $D_K$ and $D_A$, respectively. As far as $S_K$ is concerned, it can be effectively approximated by the so-called “fixed-stress” matrix [55, 56], also computed in a fully algebraic way [12]. For the matrix $S_A$, we follow the approach used in [57] by defining:

$$D_A = \text{diag} \left( B^T \tilde{A}^{-1} B \right),$$

(12)

where

$$\tilde{A} = \text{diag} \left( a_1, a_2, \ldots, a_n \right), \quad a_i = \left( \sum_{j=1}^{n} |A_{ij}|^2 \right)^{1/2}, \quad i \in \{1, \ldots, n\}.$$  

(13)

With the diagonal approximations $D_K$ and $D_A$ instead of $S_K$ and $S_A$, the trace of $Z_\alpha$ can be computed at a negligible cost. The value of $\alpha$ is therefore obtained from (7) as [47]:

$$\alpha = \frac{\sqrt{\gamma}}{n_p} \sum_{i=1}^{n_p} \sqrt{D_K^{(j)} D_A^{(i)}},$$

(14)

Remark 2.1. Throughout this paper, we overload the operator diag$(\cdot)$ to create a square diagonal matrix. In particular, if $v \in \mathbb{R}^n$, then diag$(v)$ returns a diagonal matrix with the elements of vector $v$ (Eq. (13)) ; otherwise, if $v \in \mathbb{R}^{n \times n}$, then diag$(V)$ returns a diagonal matrix consisting of the main diagonal of $V$ (Eq. (12)).

Remark 2.2. We emphasize that, in many approaches, the design of robust and efficient preconditioners based on accurate approximations of matrices (9), (10), and (11)—i.e., the Schur complement approximation—is the focus. Conversely, in the RPF framework cheap diagonal approximations of such matrices are enough to enable effective estimates for the parameter $\alpha$, the key component to the success of this preconditioning technique.

To apply $M^{-1}$ to a vector, it is convenient to rewrite (5) as:

$$M^{-1} = \begin{bmatrix}
I_u & 0 & -\frac{1}{\alpha} Q^T \\
0 & I_q & 0 \\
0 & 0 & I_p
\end{bmatrix} \begin{bmatrix}
\hat{K} & 0 & 0 \\
0 & \hat{A} & -B \\
0 & 0 & \alpha I_p
\end{bmatrix} \begin{bmatrix}
I_u & 0 & 0 \\
0 & I_q & 0 \\
0 & 0 & I_p
\end{bmatrix},$$

(15)

with

$$\hat{K} = K + \frac{1}{\alpha} QQ^T,$$

(16)

$$\hat{A} = A + \frac{2}{\alpha} BB^T.$$  

(17)
Equation (15) shows that the RPF application requires two inner solves with matrices \( \hat{K} \) and \( \hat{A} \) of (16)-(17). Notice that such matrices depend on \( \alpha \), whose inverse multiplies the rank-deficient matrices \( QQ^T \) and \( BB^T \). The latter could prevail on the SPD contributions \( K \) and \( A \) for relatively small values of \( \alpha \), thus affecting the accuracy and numerical stability in the application of \( \hat{K}^{-1} \) and \( \hat{A}^{-1} \). For this reason, two limiting values, \( \alpha_K \) and \( \alpha_A \), are introduced, with the definitions (16)-(17) modified as:

\[
\begin{align*}
\hat{K} &= K + \frac{1}{\max(\alpha, \alpha_K)} QQ^T, \\
\hat{A} &= A + \frac{\gamma}{\max(\alpha, \alpha_A)} BB^T.
\end{align*}
\]

(18)

(19)

Following the result of Theorem 3.1 in [47], the bounds \( \alpha_K \) and \( \alpha_A \) read:

\[
\alpha_K = \frac{\lambda_1(S_K)}{\omega_K - 1}, \quad \alpha_A = \frac{\gamma \lambda_1(S_A)}{\omega_A - 1},
\]

(20)

where \( \lambda_1(S_K) \) and \( \lambda_1(S_A) \) are the spectral radii of \( S_K \) and \( S_A \), and \( \omega_K \) and \( \omega_A \) are user-specified parameters denoting the maximum acceptable value for the ratios \( \kappa(\hat{K})/\kappa(K) \) and \( \kappa(\hat{A})/\kappa(A) \), i.e., the maximum increase of the spectral condition number of \( \hat{K} \) and \( \hat{A} \) with respect to that of \( K \) and \( A \), respectively. For the practical computation of \( \alpha_K \) and \( \alpha_A \), in equation (20) \( S_K \) and \( S_A \) can be replaced by \( D_K \) and \( D_A \), respectively.

The RPF set-up and application to the vector \( r^T = [r^T_u, r^T_q, r^T_p] \) are summarized in Algorithm 1 and 2, respectively. Of course, lines 3 and 8 of Algorithm 2 can be replaced by an approximate application of \( \hat{K}^{-1} \) and \( \hat{A}^{-1} \).

**Algorithm 1** RPF computation: \( M = \text{RPF\_SetUp}(\gamma, \omega_K, \omega_A, D_K, D_A, A) \)

1. \( \lambda_1(D_K) = \max_i D_K^{(i)} \)
2. \( \lambda_1(D_A) = \max_j D_A^{(j)} \)
3. \( \alpha_K = (\omega_K - 1)^{-1} \lambda_1(D_K) \)
4. \( \alpha_A = (\omega_A - 1)^{-1} \gamma \lambda_1(D_A) \)
5. \( \alpha = \sqrt{\gamma}^{-1} \sum_i D_K^{(i)} D_A^{(i)} \)
6. \( \hat{K} = QQ^T \)
7. \( \hat{A} = BB^T \)
8. \( \hat{K} \leftarrow K + \max(\alpha, \alpha_K)^{-1} \hat{K} \)
9. \( \hat{A} \leftarrow A + \gamma \max(\alpha, \alpha_A)^{-1} \hat{A} \)

**Algorithm 2** RPF application: \( [t^T_u, t^T_q, t^T_p] = \text{RPF\_Apply}(r_u, r_q, r_p, A, M) \)

1. \( x_u = \alpha^{-1} Q r_p \)
2. \( x_u \leftarrow r_u + x_u \)
3. Solve \( \hat{K} t_u = x_u \)
4. \( y_p = Q^T t_u \)
5. \( y_p \leftarrow r_p - y_p \)
6. \( z_q = B y_p \)
7. \( z_q \leftarrow z_q + \alpha^{-1} z_q \)
8. Solve \( \hat{A} t_q = z_q \)
9. \( t_p = B^T t_q \)
10. \( t_p \leftarrow \alpha^{-1} (y_p - \gamma t_p) \)
3. Enhanced RPF preconditioner

The use of equations (18)-(19) instead of (16)-(17) often is not sufficient to guarantee the RPF efficiency. Especially for large time-step values, a degradation of the RPF performance can be observed. There are several reasons for this behavior. First, the value of \( \alpha \) for this behavior. First, the value of \( \alpha \) in Theorem 3.1.

The generalized eigenvalue problem (23) can be rewritten as:

\[
(C + \beta FF^T) \lambda_i v_i = (C + \beta_i FF^T) \lambda_i v_i
\]

are bounded in the interval \([1, \lambda_1]\), where:

\[
\lambda_1 = \frac{\beta \mu_1(S_C) + 1}{\beta \mu_1(S_C) + 1},
\]

\( S_C = F^T C^{-1} F \), and \( \mu_1(S_C) \) is the spectral radius of \( S_C \).

Proof. The generalized eigenvalue problem (23) can be rewritten as:

\[
\left[ C(1 - \lambda_i) + (\beta - \beta_i \lambda_i) FF^T \right] v_i = 0.
\]

Setting \( H = C^{-1} FF^T \) and \( \mu_i = (\lambda_i - 1)/(\beta - \beta_i \lambda_i) \), equation (25) reads:

\[
H v_i = \mu_i v_i.
\]

The eigenvalues \( \mu_i \) of \( H \) are either 0, with multiplicity \( n - n_p \), or equal to those of \( S_C \). Furthermore, the eigenvalues \( \lambda_i \) are related to \( \mu_i \) as follows:

\[
\lambda_i = \frac{\beta \mu_i + 1}{\beta \mu_i + 1}.
\]

Since \( \mu_i \in [0, \mu_1(S_C)] \) and \( \lambda_i \) in (27) increases monotonically with \( \mu_i \) for \( \beta > \beta_i \), the eigenvalues \( \lambda_i \) belong to the interval \( \mathcal{I} \):

\[
\mathcal{I} = \left\{ \lambda_i \in \mathbb{R} \mid 1 \leq \lambda_i \leq \frac{\beta \mu_1(S_C) + 1}{\beta \mu_1(S_C) + 1} \right\}.
\]

Let us denote with \( \hat{C}_t \) the matrix

\[
\hat{C}_t = C + \beta_i FF^T,
\]

obtained with the lower-bound value \( \beta_i \) for \( \beta \), and use \( \hat{C}_t^{-1} \) as a preconditioner for the inner solve with \( \hat{C} \) in the RPF application (Algorithm 2). Theorem 3.1 shows that the spectral conditioning number of \( \hat{C}_t^{-1} \) grows indefinitely with \( \beta \), making the inner solve with \( \hat{C} \) more difficult and expensive as \( \beta | \beta_i \to \infty \). If the solution to the inner system (21) is obtained inexactly, as usually necessary for the sake of computational efficiency in large-size applications, the accuracy of \( w \) is expected to decrease when the separation between \( \beta \) and the limiting value \( \beta_i \) grows. This may lead to an overall degradation of the RPF performance that might even prevent the outer Krylov solver from converging.
In order to stabilize the global RPF behavior, it is therefore important to restore equations (16) and (17) in the preconditioner construction and enhance the accuracy in the local solve with \( \hat{C} \). To this aim, we introduce here two different approaches:

- **Method 1**: exploits the structure of \( \hat{C} \) by defining a natural splitting that can be used to develop an effective preconditioner for the inner problem;
- **Method 2**: uses an appropriate projection operator in order to get rid of the near-null space of \( \hat{C} \).

The use of Method 1 and Method 2 gives rise to the Enhanced RPF preconditioners (ERPF1 and ERPF2, respectively). We notice here that both Method 1 and Method 2 are fully algebraic and are developed for a general SPD matrix \( C \) and rank-deficient term \( FF^T \). Hence, their generalization to applications beyond the one focused in this work is reasonably straightforward.

### 3.1. Method 1

A weighted splitting of matrix \( \hat{C} \) is adopted to obtain a stationary iterative scheme. Let us write \( \hat{C} \) as follows:

\[
\hat{C} = \left( 1 - \frac{\beta}{\beta_L} \right) C + \frac{\beta}{\beta_L} \left( C + \beta_L FF^T \right) = \left( 1 - \frac{\beta}{\beta_L} \right) C + \frac{\beta}{\beta_L} \hat{C}_L, \tag{30}
\]

and consider the following stationary iteration to solve (21):

\[
w_{k+1} = w_k + \frac{\beta}{\beta_L} \hat{C}_L^{-1} r_k, \tag{31}
\]

with \( r_k = (b - \hat{C} w_k) \) the residual at iteration \( k \). The iteration matrix \( G \) associated with scheme (31) reads:

\[
G = I - \frac{\beta}{\beta_L} \hat{C}_L^{-1} \hat{C} = \left( 1 - \frac{\beta}{\beta_L} \right) \hat{C}_L^{-1} C. \tag{32}
\]

**Lemma 3.2.** The stationary scheme (31) for the solution of the system \( \hat{C} w = b \) of equation (21), with \( \beta > \beta_L > 0 \), is unconditionally convergent with rate \( R = \log(1 - \beta_L/\beta) \).

**Proof.** The spectral radius \( \lambda_1(G) \) of the iteration matrix is given by:

\[
\lambda_1(G) = \left( 1 - \frac{\beta}{\beta_L} \right) \lambda_1(N), \tag{33}
\]

with \( N = \hat{C}_L^{-1} C \) and \( \lambda_1(N) \) its spectral radius. Since \( N \) reads

\[
N = \left( C + \beta_L FF^T \right)^{-1} C = \left[ C \left( I + \beta_L C^{-1} FF^T \right)^{-1} \right]^{-1} C = (I + \beta_L H)^{-1}, \tag{34}
\]

its spectral radius is:

\[
\lambda_1(N) = \frac{1}{1 + \beta_L \lambda_n(H)}, \tag{35}
\]

where \( \lambda_n(H) \) denotes the smallest eigenvalue of \( H = C^{-1} FF^T \). Being \( H \) similar to the symmetric positive semidefinite matrix \( C^{-\frac{1}{2}} FF^T C^{-\frac{1}{2}} \), \( \lambda_n(H) = 0 \) and

\[
\lambda_1(G) = 1 - \frac{\beta}{\beta_L} < 1. \tag{36}
\]

Hence, the stationary scheme (31) is unconditionally convergent with rate \( R = \log(1 - \beta_L/\beta) \). \( \square \)

The idea of Method 1 is to solve the system of equation (21) with the scheme (31). Since in the RPF preconditioner application such system is solved inexactly, we carry out a pre-defined number of stationary iterations, \( n_{in} \), and replace
Algorithm 3 Method 1: \( w = \text{MET1\_APPLY}\left(n_{in}, \beta, \beta, \beta, \hat{C}, \hat{M}^{-1}_{\hat{C}}\right) \)

1: \( w = 0 \)
2: \( \text{for } k = 1, \ldots, n_{in} \text{ do} \)
3: \( r = b - \hat{C}w \)
4: \( \text{Apply } \hat{M}^{-1}_{\hat{C}} \text{ to } r \text{ to get } v \)
5: \( w \leftarrow w + \left(\frac{\beta}{\beta_{\ell}}\right)v \)
6: \( \text{end for} \)

the exact application of \( \hat{C}_{\ell}^{-1} \) with an inexact one, say the inner preconditioner \( \hat{M}^{-1}_{\hat{C}} \). This procedure is summarized in Algorithm 3.

The ERPF1 preconditioner is obtained by replacing lines 3 and 8 of Algorithm 2 with the function in Algorithm 3 whenever \( \alpha < \alpha_{K} \) and \( \alpha < \alpha_{A} \), respectively. For each inner iteration \( k \), Method 1 involves one matrix-vector product, one application of \( \hat{M}^{-1}_{\hat{C}} \) and two vector updates, hence the algorithm cost can grow up quickly with \( n_{in} \). More precisely, denoting by \( n \) the number of rows of \( \hat{C} \), the complexity \( \chi_{1} \) of Algorithm 3 reads:

\[
\chi_{1} = n_{in} \times \left[ \chi\left(\hat{M}^{-1}_{\hat{C}}\right) + O(n) \right],
\]

where \( \chi(\hat{M}^{-1}_{\hat{C}}) \) is the complexity of the inner preconditioner application. For this reason, \( n_{in} \) should be kept as low as possible, say 2 or 3.

Remark 3.1. Equation (36) shows that the convergence rate of the iteration (31) degrades to 0 as \( \beta \to \infty \). In our application, \( \beta \) is either \( 1/\alpha \) (equation (16)) or \( \gamma/\alpha \) (equation (17)). In a full poromechanical simulation, the condition \( \gamma/\alpha \to \infty \) is more likely to occur and usually happens at the final stage of the process, when the solution approaches steady-state conditions and \( \Delta t \), hence \( \gamma \), progressively increases. In this situation, Method 1 is expected to lose efficiency because it may require too many inner iterations to obtain a sufficiently accurate result.

Remark 3.2. The stationary scheme (31) is a Richardson iteration preconditioned by \( (\beta_{\ell}/\beta)\hat{C}_{\ell}^{-1} \). Since \( \hat{C} \) is SPD, it could be also effectively replaced by a Conjugate Gradient scheme with the same preconditioner. Stopping the inner iterations after \( n_{in} = 2 \) or 3 steps, however, usually does not provide a significantly different outcome with respect to the use of Algorithm 3.

3.2. Method 2

Another strategy to solve (21) relies on using an appropriate projection operator. In particular, the idea is to project the system (21) onto a space \( \mathcal{Z} \) such that:

\[
\mathbb{R}^{n} = \ker(F^T) \oplus \mathcal{Z},
\]

so as to annihilate the components of \( w \) lying in the kernel of \( F^T \). Let us define the \( \hat{C} \)-orthogonal projector \( P \) as:

\[
P = Z \left[ Z^T \left( \hat{C}Z \right) \right]^{-1} Z^T \hat{C},
\]

where \( Z = [z_1, \ldots, z_{n_p}] \in \mathbb{R}^{n \times n_p} \) is the generator of \( \mathcal{Z} \):

\[
\mathcal{Z} = \text{span}\{z_1, \ldots, z_{n_p}\}.
\]

The operator \( P \) is the linear mapping that projects a vector \( w \) onto \( \mathcal{Z} \) orthogonally to \( \mathcal{L} \):

\[
\mathcal{L} = \{ y_i \in \mathbb{R}^n \mid y_i = \hat{C}z_i, \ z_i \in \mathcal{Z} \}.
\]

Notice that the projector \( P \) is \( \hat{C} \)-orthogonal according to the following definition [58]:

Definition 3.3. A projector \( P \) onto a subspace \( \mathcal{Z} \) is \( \hat{C} \)-orthogonal if and only if it is self-adjoint with respect to the \( \hat{C} \)-inner product.
Of course, we have \( \text{Ran}(P) = \text{Ran}(Z) \). The solution \( w \) of (21) can be decomposed as the sum of two contributions:

\[
w = Pw + (I - P)w = w_Z + w_S,
\]

where \( w_Z \in Z \) and \( w_S \in S = \text{Ran}(I - P) \). Applying the projector (39), the component \( w_Z \) is given by:

\[
w_Z = Pw = ZE^{-1}Z^T \hat{C}w = ZE^{-1}Z^T b,
\]

where \( E = Z^T \hat{C}Z \) is the projection of \( \hat{C} \) onto \( Z \). In order to obtain \( w_S \), let us consider the following projected system:

\[
(I - P^T) \hat{C}w = (I - P^T)b.
\]

Recalling that \( (I - P^T) \hat{C} = \hat{C}(I - P) \), equation (44) can be rewritten as:

\[
\hat{C}w_S = (I - P^T)b.
\]

Since \( w_S \perp L \) of equation (41), the following condition holds:

\[
Z^T \left( C + \beta FF^T \right) w_S = 0.
\]

If we set \( Z = C^{-1}F \), equation (46) becomes:

\[
(I + \beta SC) F^T w_S = 0,
\]

which implies \( w_S \in \ker(F^T) \) because of the regularity of the matrix \( (I + \beta SC) \), with \( SC = F^T C^{-1}F \) (see proof of Theorem 3.1). Hence, by setting \( Z = \text{Ran}(C^{-1}F) \) equation (45) simplifies to:

\[
Cw_S = (I - P^T)b,
\]

which no longer requires the inversion of \( \hat{C} \). Observe that the contribution \( P^T b \) at the right-hand side of (48) reads:

\[
\hat{C}ZE^{-1}Z^T b = \hat{C}w_Z,
\]

e.g., \( P^T b = \hat{C}w_Z = b_Z \) is the projected right-hand side onto \( Z \). The solution \( w \) to the system (21) is therefore given by the following set of equations:

\[
\begin{align*}
  w_Z &= ZE^{-1}Z^T b, \\
  w_S &= C^{-1}(b - b_Z), \\
  w &= w_Z + w_S.
\end{align*}
\]

The matrix \( ZE^{-1}Z^T \) reads:

\[
ZE^{-1}Z^T = C^{-1}F \left[ F^T C^{-1}CC^{-1}F + \beta \left( F^T C^{-1}FF^T C^{-1}F \right) \right]^{-1} F^T C^{-1}
\]

\[
= C^{-1}F \left[ SC + \beta S_C^2 \right]^{-1} F^T C^{-1}
\]

\[
= C^{-1}FS_C^{-1} \left[ I + \beta SC \right]^{-1} F^T C^{-1}.
\]

Introducing (51) in the first equation of (50), we have:

\[
\begin{align*}
  w_Z &= C^{-1}FS_C^{-1}S_C^{-1} F^T C^{-1} b,
\end{align*}
\]
with $\hat{S}_C = I + \beta S_C$. The global solution $w$ can be rewritten as:

$$w = w_S + w_Z = C^{-1}(b - b_Z) + w_Z = C^{-1}(b - \hat{C}w_Z + Cw_Z) = C^{-1}(b - \beta FF^Tw_Z),$$

and using $w_Z$ of equation (52) finally yields:

$$w = C^{-1}(b - \beta FF^T w_Z) = C^{-1}(b - \beta F\hat{S}_C^{-1}F^T C^{-1}b).$$

Remark 3.3. The result (54) can be also obtained applying directly the Sherman-Morrison-Woodbury identity to equation (21). Therefore, this relationship can be also regarded as the natural outcome of a particular projection operated on the matrix $\hat{C}$.

The solution of the inner system (21) through equation (54) requires two inner solves with $\hat{C}$, which is a well-defined SPD matrix, and one with $\tilde{S}_C$. The latter is not available explicitly, however it can be easily and inexpensively approximated for $\hat{K}$ and $\hat{A}$ by using, for instance, $D_K$ and $D_A$ already adopted to compute $a$ (see Section 2). Hence, an inexact solve with $\hat{S}_C$ is performed by using some approximation, say $M_s^{-1}$. Of course, also the solves with $C$ can be performed inexactly, by means of another inner local preconditioner, $M_C^{-1}$, for SPD matrices. Algorithm 4 summarizes the procedure resulting from Method 2.

**Algorithm 4 Method 2:** $w = \text{MET2}_\text{APPLY}((\beta, b, F, M_C^{-1}, M_s^{-1}))$

1. Apply $M_C^{-1}$ to $b$ to get $c$
2. $d = F^T c$
3. Apply $M_s^{-1}$ to $d$ to get $g$
4. $c = Fg$
5. $c \leftarrow b - \beta c$
6. Apply $M_C^{-1}$ to $c$ to get $w$

The ERPF2 preconditioner is obtained by replacing lines 3 and 8 of Algorithm 2 with the function in Algorithm 4 whenever $\alpha < \alpha_K$ and $\alpha < \alpha_A$, respectively. The complexity $\chi_2$ of Algorithm 2 reads:

$$\chi_2 = 2\chi(M_C^{-1}) + \chi(M_s^{-1}) + O(n),$$

where $n$ is the number of rows of $F$, and $\chi(M_C^{-1})$ and $\chi(M_s^{-1})$ denote the complexity of the inner preconditioner applications. The overall cost for the preconditioner application per iteration increases with respect to the native RPF, however the overall algorithm is expected to take benefit from the acceleration of the global Krylov solver.

Remark 3.4. The use of Method 2 for solving the inner systems with both $\hat{K}$ and $\hat{A}$ is equivalent to compute the vector $t = M^{-1}r$ using directly equation (5):

$$\begin{cases}
y = aM_1^{-1}r \\
t = M_2^{-1}y
\end{cases},$$

where $M_1$ and $M_2$ are inverted with the aid of the factorizations:

$$M_1 = \begin{bmatrix} K & 0 & -Q \\ 0 & aI_q & 0 \\ Q^T & 0 & aI_p \end{bmatrix}, \quad M_2 = \begin{bmatrix} I_a & 0 & 0 \\ 0 & A & -B \\ 0 & \gamma B^T & aI_p \end{bmatrix}$$

This leads to an alternative ERPF2 implementation, which is naturally more prone to a split preconditioning strategy. The use of the factorizations (57)-(58) into (56) whenever $\alpha < \alpha_K$ and $\alpha < \alpha_A$ provides the same numerical outcome as
the application of Algorithm 4. Details of such equivalence along with the alternative ERPF2 algorithm are provided in Appendix A.

4. Numerical Results

Two sets of numerical experiments are discussed in this section. The first set (Test 1) arises from Mandel’s problem [59], i.e., a classic benchmark of linear poroelasticity. This problem is used to verify the theoretical properties of the proposed methods. In the second set (Test 2), two real-world applications are considered in order to test the robustness and efficiency of the preconditioners.

We consider three variants of RPF, ERPF1, and ERPF2 according to the selection of the inner preconditioners. In essence, the first variant (M_i) relies on applying “exactly” the inner preconditioners using nested direct solvers and aims at investigating the theoretical properties of the different approaches. The second (M_ii) and the third (M_iii) variants introduce further levels of approximation utilizing incomplete Cholesky (ic) and algebraic multigrid (amg) preconditioners, respectively. Specifically:

- **RPF** (Algorithm 2): all exact/inexact solves involve \( \hat{K} \) in (16) and \( \hat{A} \) in (17) irrespective of the value of \( \alpha \);

- **ERPF1** (Algorithm 3): Method 1 requires inner solves with matrices \( \hat{K}_\alpha \) or \( \hat{A}_\alpha \) that depend on the relaxation parameter \( \alpha \) as follows

\[
\hat{K}_\alpha = \begin{cases} 
\hat{K}_\ell, & \text{if } \alpha < \alpha_K \\
\hat{K}, & \text{if } \alpha \geq \alpha_K 
\end{cases}
\]

\[
\hat{A}_\alpha = \begin{cases} 
\hat{A}_\ell, & \text{if } \alpha < \alpha_A \\
\hat{A}, & \text{if } \alpha \geq \alpha_A 
\end{cases}
\]

with \( \hat{K}_\ell \) and \( \hat{A}_\ell \) computed at \( \alpha_K \) and \( \alpha_A \), respectively, using expression (29);

- **ERPF2** (Algorithm 4): Method 2 is used to replace the inner solves with \( \hat{K} \) or \( \hat{A} \) whenever either \( \alpha < \alpha_K \) or \( \alpha < \alpha_A \). It requires inner solves with \( K \) and \( \hat{S}_K \), or \( A \) and \( \hat{S}_A \). For \( \hat{S}_K \) we use the diagonal matrix \( D_K \) employed in the computation of \( \alpha \) and \( \alpha_K \) (see Section 2) by defining:

\[
D_K = I_p + \frac{1}{\alpha} D_K.
\]

The application of \( \hat{S}_K^{-1} \) is approximated by \( \hat{D}_K^{-1} \). For \( \hat{S}_A \) we use the diagonal matrix \( \hat{A}^{-1} \) employed in the computation of \( \alpha \) and \( \alpha_A \) (see Section 2, equation (13)) by defining:

\[
\hat{S}_A \approx I_p + \frac{\gamma}{\alpha} B^T \hat{A}^{-1} B.
\]

The inverse of \( \hat{A} \) is used to approximate the application of \( A^{-1} \) as well.

Table 1 summarizes the different variants illustrated above. For the incomplete Cholesky factorization, we use an algorithm implementing a fill-in technique based on the selection of a user-specified row-wise number of entries in addition to those of the original matrix, as proposed in [60] and [61]. A classic algebraic multigrid method [62] is used as implemented in the HSL_MI20 package [63]. Of course, several other algebraic options for the inner blocks are possible.

The objective of this section is to demonstrate numerically that the proposed enhancements ERPF1 and ERPF2 overcome the drawbacks of the native RPF preconditioner. A comparison with other well-established block preconditioners, which has already been done for RPF in [47], is not performed here. Test 1 shows the stability and robustness of ERPF1 and ERPF2 to a variation of the mesh and timestep size for tightly coupled poromechanical conditions. Tests 2 give the performance in real-world heterogeneous problems, where the robustness to significant jumps in the hydromechanical parameters and the computational efficiency measured in terms of CPU time reduction are the main issues.

In all test cases, Bi-CGSTab [24] is selected as a Krylov subspace method. Our focus is the robustness and computational efficiency in the solution of a single linear system. Hence, following standard practice [58], all tests
are performed by setting a random solution vector $x$ and computing the corresponding right-hand side $F = Ax$. The iterations are stopped when the ratio between the 2-norm of the real residual vector and the 2-norm of the right-hand side is smaller than $\tau = 10^{-6}$. The computational performance is evaluated in terms of number of iterations $n_{it}$, CPU time in seconds for the preconditioner set-up $T_p$, and for the solver to converge $T_s$. The total time is denoted by $T_t = T_p + T_s$. All computations are performed using a code written in FORTRAN90 on an Intel(R) Xeon(R) CPU E5-1620 v4 at 3.5 GHz Quad-Core with 64 GB of RAM memory.

4.1. Test1: Mandel’s problem

This is a classical benchmark for validating coupled poromechanical models. The problem consists of a porous slab with section $2a \times 2b$, discretized by a cartesian grid, sandwiched between rigid, frictionless, and impermeable plates (Figure 1). For the sake of simplicity, we set $a = b$. A detailed description of the test case is reported in [12, 47]. This test case has mainly a theoretical value and is used to investigate the properties of ERPF1 and ERPF2.

![Figure 1](image1.jpg)

Figure 1: Test 1, Mandel’s problem: sketch of the geometry (a) and the computational model (b).
Table 2: Test 1, Mandel's problem: grid refinement and problem size.

| $a/h$ | number of elements | $n_u$ | $n_r$ | $n_p$ |
|-------|--------------------|-------|-------|-------|
| 10    | $10 \times 1 \times 10$ | 726   | 420   | 100   |
| 20    | $20 \times 2 \times 20$ | 3,969 | 2,880 | 800   |
| 40    | $40 \times 4 \times 40$ | 25,215| 21,120| 6,400 |
| 80    | $80 \times 8 \times 80$ | 177,147| 161,280| 51,200|

Figure 2: Test 1, Mandel’s problem ($a/h = 10$): eigenspectrum of $\hat{K}^{-1}\hat{K}$ and $\hat{A}^{-1}\hat{A}$ varying $\alpha/\alpha_K$ (a) and $\alpha/\alpha_A$ (b).

Table 3: Test 1, Mandel’s problem ($a/h = 10$): condition number $\kappa$ of $\hat{K}$ and $\hat{A}$ varying $\alpha/\alpha_\ell$, $\alpha_\ell$ being either $\alpha_K$ or $\alpha_A$. The condition number of $K$ and $A$ are: $\kappa(K) = 2.14 \cdot 10^6$, $\kappa(A) = 9.84 \cdot 10^9$.

| $\alpha/\alpha_\ell$ | $\kappa(\hat{K})$ | $\kappa(\hat{A})$ |
|-----------------------|--------------------|--------------------|
| 1.00                  | 8.47 · 10^6        | 1.21 · 10^{12}     |
| 0.50                  | 1.67 · 10^7        | 2.42 · 10^{12}     |
| 0.20                  | 4.15 · 10^7        | 6.05 · 10^{12}     |
| 0.05                  | 1.65 · 10^6        | 2.41 · 10^{13}     |
The preconditioner variant \( \text{M}^{-1}_{\text{ff}} \) uses nested direct solvers, hence it has mainly a theoretical value. It is useful to isolate the impact of the proposed schemes (Algorithms 3 and 4) on the convergence behavior by varying the discretization steps in both space and time. We recall on passing that \( \gamma = \theta \Delta t \), with \( \theta \in [0.5, 1] \), and that \( \alpha \) is proportional to \( \sqrt{\gamma} \). \( \alpha \) is constant with \( \gamma \), and \( \alpha_A \) depends linearly on \( \gamma \) (see Section 2). Hence:

\[
\lim_{\Delta t \to 0} \frac{\alpha}{\alpha_K} = 0, \quad \lim_{\Delta t \to +\infty} \frac{\alpha}{\alpha_A} = 0,
\]

i.e., the conditions \( \alpha < \alpha_K \) and \( \alpha < \alpha_A \) are encountered for small and large time-step sizes, respectively, and are not likely to be satisfied simultaneously.

Table 4 provides the outer iteration count obtained in Mandel’s problem with ERPF1 by varying the mesh-size, the time-step and the number of inner iterations \( n_{in} \). As expected, the number of outer iterations decreases when growing the number of inner iterations. Since the computational cost for ERPF1 application increases with \( n_{in} \), setting \( n_{in} = 2 \) or \( n_{in} = 3 \) appears to be already a good trade-off between solver acceleration and computational cost per iteration.

The ERPF1 effectiveness is actually dependent on the value of \( \alpha/\alpha_K \) or \( \alpha/\alpha_A \).Lemma 3.2 shows that the convergence rate of the inner stationary method used with \( \hat{K} \) or \( \hat{A} \) tends to 0 with the ratio \( \alpha/\alpha_K \) or \( \alpha/\alpha_A \). This is confirmed by Table 4, which shows a performance deterioration for both \( \alpha/\alpha_K \) and \( \alpha/\alpha_A \) approaching 0. Such a deterioration, however, appears to be much more pronounced with \( \alpha/\alpha_A \), i.e., for large values of the time-step \( \Delta t \) (see Remark 3.1). Table 4 shows also a mild dependency on the mesh-size \( h \), with the outer iteration count increasing moderately only for large \( \Delta t \) values.

A similar analysis is carried out for ERPF2 with the \( \text{M}^{-1}_{\text{ff}} \) variant. Table 5 provides the results in terms of outer iteration count by varying \( \Delta t \) and \( h \). This approach turns out to be stable with respect to variations of both \( \Delta t \) and \( h \), also for extreme values of the time-step size. A comparison with Table 4 reveals also that EPRF2 appears to be more effective than EPRF1, as far as the number of outer iterations is concerned, especially when \( \alpha < \alpha_A \), i.e., large time steps. For this reason, we can conclude that ERPF2 should be usually preferred as a full poromechanical simulation proceeds towards steady-state conditions.

Of course, the results of Table 4 and 5 can change when the nested direct solvers are replaced by inner preconditioners, such as with the variants \( \text{M}^{-1}_{\text{M}} \) and \( \text{M}^{-1}_{\text{ff}} \). For example, Figure 4 and 5 compare the convergence profile for \( a/h = 10 \) and \( a/h = 20 \), respectively, and very large values of the time step (\( \Delta t = 10^2 \Delta t_c \) and \( \Delta t = 10^6 \Delta t_c \)) obtained with the original RPF (\( \text{M}^{-1}_{\text{M}} \), \( \text{M}^{-1}_{\text{ff}} \), and \( \text{M}^{-1}_{\text{ff}} \) variants) and with EPRF2 (\( \text{M}^{-1}_{\text{M}} \) and \( \text{M}^{-1}_{\text{ff}} \) variants). All incomplete Cholesky factorizations are computed in this case with zero fill-in, while algebraic multigrid is applied keeping the default parameters. It can be observed that, even in this academic example, the original RPF preconditioner with an inexact solve for \( \hat{K} \) or \( \hat{A} \) might not converge with both \( \text{M}^{-1}_{\text{M}} \) and \( \text{M}^{-1}_{\text{ff}} \) variants. Hence, this outcome appears to be an issue related to the nature of \( \hat{K} \) and \( \hat{A} \), rather than the choice of the inner preconditioner. By distinction, the ERPF2 performance is just moderately affected by the use of inexact solves, exhibiting a very stable convergence profile with respect to a \( \Delta t \) variation. Hence, the proposed algorithms are able to enhance not only the RPF performance, but also...
Table 4: Test 1, Mandel’s problem: number of outer iteration with ERPF1 and $M^{-1}$ by varying the mesh-size $h$, the time-step $\Delta t$ and the inner iterations $n_{in}$. The time-step size is relative to the characteristic consolidation time $t_c = 900 \, s$ [12, 47].

| $a/h$ | $\Delta t/t_c$ | $\alpha/\alpha_K$ | $\alpha/\alpha_A$ | # of outer iterations |
|-------|----------------|-------------------|-------------------|----------------------|
| $5.6 \cdot 10^{-7}$ | 0.10 | $> 1$ | 12 | 9 | 7 | 5 | 3 |
| $3.4 \cdot 10^{-6}$ | 0.25 | $> 1$ | 10 | 7 | 4 | 3 | 3 |
| $1.7 \cdot 10^{-5}$ | 0.50 | $> 1$ | 7 | 4 | 4 | 3 | 3 |
| 10 | $1.0 \cdot 10^{-3}$ | $> 1$ | 8 | -- | -- | -- | -- |
| $5.0 \cdot 10^{-1}$ | 0.50 | 14 | 10 | 8 | 8 | 8 | 8 |
| $1.7 \cdot 10^{-1}$ | 0.25 | 22 | 12 | 10 | 9 | 8 | 8 |
| $5.6 \cdot 10^{-1}$ | 0.10 | 24 | 15 | 11 | 11 | 10 | 10 |

| $a/h$ | $\Delta t/t_c$ | $\alpha/\alpha_K$ | $\alpha/\alpha_A$ | # of outer iterations |
|-------|----------------|-------------------|-------------------|----------------------|
| $5.6 \cdot 10^{-7}$ | 0.10 | $> 1$ | 12 | 9 | 6 | 4 | 3 |
| $3.4 \cdot 10^{-6}$ | 0.25 | $> 1$ | 9 | 7 | 4 | 3 | 2 |
| $1.7 \cdot 10^{-5}$ | 0.50 | $> 1$ | 7 | 4 | 3 | 3 | 3 |
| 20 | $1.0 \cdot 10^{-3}$ | $> 1$ | 7 | -- | -- | -- | -- |
| $5.0 \cdot 10^{-1}$ | 0.50 | 17 | 12 | 12 | 11 | 11 | 11 |
| $1.7 \cdot 10^{-1}$ | 0.25 | 29 | 16 | 13 | 13 | 12 | 12 |
| $5.6 \cdot 10^{-1}$ | 0.10 | 38 | 22 | 16 | 14 | 13 | 13 |

Table 5: Test 1, Mandel’s problem: number of outer iterations with ERPF2 and $M^{-1}$ by varying the mesh-size $h$ and the time-step $\Delta t$.

| $a/h$ | $\Delta t/t_c$ | $\alpha/\alpha_K$ | # of iter. | $\Delta t/t_c$ | $\alpha/\alpha_A$ | # of iter. |
|-------|----------------|-------------------|------------|----------------|-------------------|------------|
| $10^{-6}$ | $1.3 \cdot 10^{-1}$ | 5 | $10^2$ | $3.5 \cdot 10^{-2}$ | 8 | |
| $10^{-6}$ | $4.3 \cdot 10^{-2}$ | 6 | $10^3$ | $1.1 \cdot 10^{-2}$ | 8 | |
| $10^{-6}$ | $1.4 \cdot 10^{-2}$ | 6 | $10^4$ | $3.5 \cdot 10^{-3}$ | 7 | |
| $20^{-6}$ | $3.0 \cdot 10^{-1}$ | 6 | $10^2$ | $1.9 \cdot 10^{-2}$ | 8 | |
| $20^{-6}$ | $9.5 \cdot 10^{-2}$ | 6 | $10^3$ | $6.0 \cdot 10^{-3}$ | 7 | |
| $20^{-6}$ | $3.0 \cdot 10^{-2}$ | 6 | $10^4$ | $1.9 \cdot 10^{-3}$ | 7 | |
| $40^{-6}$ | $1.9 \cdot 10^{-1}$ | 5 | $10^2$ | $9.5 \cdot 10^{-3}$ | 9 | |
| $40^{-6}$ | $6.0 \cdot 10^{-2}$ | 6 | $10^3$ | $3.0 \cdot 10^{-3}$ | 7 | |
| $40^{-6}$ | $1.8 \cdot 10^{-2}$ | 6 | $10^4$ | $9.5 \cdot 10^{-4}$ | 7 | |
| $80^{-6}$ | $7.9 \cdot 10^{-1}$ | 6 | $10^2$ | $6.2 \cdot 10^{-3}$ | 11 | |
| $80^{-6}$ | $2.5 \cdot 10^{-1}$ | 6 | $10^3$ | $1.9 \cdot 10^{-3}$ | 8 | |
| $80^{-6}$ | $8.0 \cdot 10^{-2}$ | 6 | $10^4$ | $6.2 \cdot 10^{-4}$ | 7 | |
Figure 4: Test 1, Mandel’s problem: Convergence profiles for $\Delta t = 10^2 t_c$ (a) and $\Delta t = 10^6 t_c$ (b) with $a/h = 10$.

Figure 5: Test 1, Mandel’s problem: the same as Figure 4 for $a/h = 20$. 
Table 6: Test 1, Mandel’s problem: number of outer iterations with ERPF2 and $M^0$ by varying the mesh-size $h$ and the time-step $\Delta t$.

| $a/h$ | $\Delta t/t_c$ | $\alpha/\alpha_K$ | # of iter. | $\Delta t/t_c$ | $\alpha/\alpha_K$ | # of iter. |
|-------|----------------|-------------------|------------|----------------|-------------------|------------|
| 10$^{-6}$ | 1.3 · 10$^{-1}$ | 14 | 10$^2$ | 3.5 · 10$^{-2}$ | 12 |
| 10$^{-7}$ | 4.3 · 10$^{-2}$ | 14 | 10$^3$ | 1.1 · 10$^{-2}$ | 14 |
| 10$^{-8}$ | 1.4 · 10$^{-2}$ | 14 | 10$^4$ | 3.5 · 10$^{-3}$ | 14 |
| 10$^{-6}$ | 3.0 · 10$^{-1}$ | 19 | 10$^2$ | 1.9 · 10$^{-2}$ | 13 |
| 10$^{-7}$ | 9.5 · 10$^{-2}$ | 19 | 10$^3$ | 6.0 · 10$^{-3}$ | 12 |
| 10$^{-8}$ | 3.0 · 10$^{-2}$ | 19 | 10$^4$ | 1.9 · 10$^{-3}$ | 14 |
| 10$^{-6}$ | 1.9 · 10$^{-1}$ | 22 | 10$^2$ | 9.5 · 10$^{-3}$ | 11 |
| 10$^{-7}$ | 6.0 · 10$^{-2}$ | 19 | 10$^3$ | 3.0 · 10$^{-3}$ | 12 |
| 10$^{-8}$ | 1.8 · 10$^{-2}$ | 17 | 10$^4$ | 9.5 · 10$^{-4}$ | 11 |
| 10$^{-6}$ | 7.9 · 10$^{-1}$ | 12 | 10$^2$ | 6.2 · 10$^{-3}$ | 16 |
| 10$^{-7}$ | 2.5 · 10$^{-1}$ | 19 | 10$^3$ | 1.9 · 10$^{-3}$ | 14 |
| 10$^{-8}$ | 8.0 · 10$^{-2}$ | 19 | 10$^4$ | 6.2 · 10$^{-4}$ | 11 |

its robustness.

The use of incomplete Cholesky factorizations with a partial fill-in as inner preconditioners of SPD blocks is efficient in sequential computations, but can create concerns in parallel environments. Moreover, as it is well-known, it can prevent from obtaining an optimal weak scalability with respect to the mesh size $h$. For instance, this can be observed in Figure 4 and 5. ERPF2 profiles, where the number of iterations is not constant with $h$ by using incomplete Cholesky factorizations. The scalability with $h$ can be restored by using scalable inner preconditioners, such as algebraic multigrid methods. The same analysis as Table 5 is here carried out by using the $M^0$ variant. Table 6 provides the outer iteration counts by varying $\Delta t$ and $h$. As expected, the scalability with $h$ is much improved, since the number of iterations is quite stable between the progressive refinements.

4.2. Test2: Real-world applications

The ERPF performance is finally tested in two large-size challenging cases, with unstructured grids and severe material anisotropy and heterogeneity. In particular, the solver performance is investigated in presence of strong variability of both mechanical, i.e. compressibility, and hydraulic, i.e. permeability and porosity, material parameters, with severe jumps occurring between adjacent cells. This condition poses particular challenges as to the algorithmic robustness of the proposed methods. We have considered two real-world applications:

- Test 2a: Chaobai. This model is used to predict land subsidence due to a shallow multi-aquifer system over-exploitation in the Chaobai River alluvial fan, North of Beijing, China [64]. The strong heterogeneity of the alluvial fan, which covers an overall area extent of more than 1,100 km$^2$, is accounted for by means of a statistical inverse framework in a multi-zone transition probability approach [65, 66]. Figure 6a shows a reconstruction of the lithofacies distribution. Details on the model implementation are provided in [64];

- Test 2b: SPE10. This is a typical petroleum reservoir engineering application reproducing a well-driven flow in a deforming porous medium. The model setup is based on the 10$^{th}$ SPE Comparative Solution Project [67], a well-known severe benchmark in reservoir applications, assuming a poroelastic mechanical behavior with incompressible fluid and solid constituents. The porous medium is populated with homogeneous elastic properties, namely, Young’s modulus $E = 8.3 \cdot 10^3$ MPa, Poisson’s ratio $\nu = 0.3$, and Biot’s coefficient $b = 1.0$. The model is limited to the top 35 layers, which are representative of a shallow-marine Tarbert formation characterized by extreme permeability variations (Figure 6b), covering an areal extent of 365.76 × 650.56 m$^2$ and for 21.34 m in the vertical direction. One injector and one production well, located at opposite corners of
The domain, penetrate vertically the entire reservoir and drive the porous fluid flow. The reader can refer to [68] for additional details.

The size and the number of non-zeros of the matrices arising from Test 2a and 2b are provided in Table 7.

Table 7: Test 2, real-world applications: size and number of non-zeros of the test matrices.

| Test | \(n_u\)  | \(n_q\)  | \(n_p\)  | \# non-zeros |
|------|----------|----------|----------|-------------|
| 2a: Chaobai | 2,152,683 | 2,132,612 | 707,600 | 143,359,342 |
| 2b: SPE10 | 1,455,948 | 1,409,000 | 462,000 | 94,317,731 |

The size and the number of non-zeros of the matrices arising from Test 2a and 2b are provided in Table 7.

The size of the real-world problems addressed in Test 2 prevents from the use of the \(M^{-1}_{ii}\) variant with nested direct solvers. Therefore, we compare the performance of the original RPF with ERPF1 and ERPF2 in the \(M^{-1}_{ii}\) variant, which makes use of incomplete Cholesky factorizations with partial fill-in as inner preconditioners. We employ a limited-memory implementation [61], where the user can specify the row-wise number of entries \(\rho\) to be retained in addition to the row-wise number of non-zeros of the original matrix. The parameters of the inner preconditioners are set up to minimize the total CPU time \(T_t\) with the RPF preconditioner and intermediate values of the time-step size \(\Delta t\) (\(10^{-4} < \Delta t < 10^6\) day for Test 2a and \(10^{-6} < \Delta t < 10^6\) day for Test 2b).

Table 8 provides the results obtained for Test 2a in terms of iteration count, solver CPU time \(T_s\) and total CPU time \(T_t\). For small \(\Delta t\) values, i.e., when \(\alpha < \alpha_K\), ERPF1 outperforms ERPF2, providing smaller iteration counts and total CPU times. In this test case, ERPF1 turns out to be comparable with the original RPF, which generally proves, however, less robust. As \(\Delta t\) increases, the iteration count with RPF quickly grows and soon becomes unacceptable. With ERPF1, the iterations to converge increase as well, though at a lesser extent, while they remain stable when using ERPF2. In other words, inspection of Table 8 reveals that, in a full transient problem where the time-step size typically increases as the simulation proceeds towards the steady state, the most efficient strategy consists of switching from ERPF1 when \(\alpha < \alpha_K\) to ERPF2 when \(\alpha < \alpha_A\), preserving the original RPF for the intermediate steps.

Table 9 provides the same results as Table 8 for Test 2b. In this case, the condition \(\alpha < \alpha_K\) is never met for time-step sizes with a relevant physical meaning. Therefore, we report only the performance obtained by the original RPF and the ERPF2 methods, being ERPF1 more convenient for small \(\Delta t\) only. As in Test 2a, ERPF2 always outperforms the original RPF preconditioner, proving much more robust and practically time-step independent.
The proposed methods can be generalized to applications where a similar algebraic issue arises, such as in the use of an augmented Lagrangian approach for Navier-Stokes or incompressible elasticity. In the context of coupled poromechanics, they give rise to an Enhanced RPF preconditioner, ERPF1 and ERPF2, respectively, which has been tested in both theoretical benchmarks and challenging real-world large-size applications. The following results are worth summarizing.

5. Conclusions

The Relaxed Physical Factorization introduced in [47] has proved an efficient preconditioning strategy for the three-field formulation of coupled poromechanics with respect to domain heterogeneity, anisotropy and grid distortion. However, a performance degradation was observed for the values of time-step size that are typically required at the beginning of a full transient simulation and toward steady-state conditions. In fact, at the beginning of the poromechanical process very small time steps are necessary to capture accurately the pressure and deformation evolution in almost undrained conditions, while larger and larger steps are convenient when proceeding towards the steady state. The origin of such a drawback stems from the need of inverting, also inexactly, inner blocks in the form $\hat{A}^{-1}$ and $\hat{A}_{ic}^{-1}$ is $\hat{\rho}_S = 40$.

Two fully algebraic methods are presented to address the RPF issues:

1. a natural splitting of $\hat{C}$ is introduced to define a particular stationary scheme. The scheme is unconditionally convergent and does not require the inversion of $\hat{C}$. Hence, the inner solve with $\hat{C}$ is replaced by a few iterations of such a scheme;

2. an appropriate projection operator is developed in order to annihilate the components of the solution vector of $\hat{C}\mathbf{w} = \mathbf{b}$ lying in the near-null space of $\hat{C}$. The projected system is solved in a stable way, while the remaining part of the solution vector can be computed requiring the inversion of the regular SPD matrix $C$ only, instead of $\hat{C}$.

The proposed methods can be generalized to applications where a similar algebraic issue arises, such as in the use of an augmented Lagrangian approach for Navier-Stokes or incompressible elasticity. In the context of coupled poromechanics, they give rise to an Enhanced RPF preconditioner, ERPF1 and ERPF2, respectively, which has been tested in both theoretical benchmarks and challenging real-world large-size applications. The following results are worth summarizing.

---

### Table 8: Test 2a, Chaobai: iteration count and CPU times for a variable time-step size by using the $M_S^{-1}$ variant as a preconditioner (Table 1). The fill-in parameter for: $K_K^{-1}$, $K_{ic}^{-1}$, and $K_{ic}$ is $\rho_K = 60$; $\hat{A}^{-1}$ and $\hat{A}_{ic}^{-1}$ is $\rho_A = 50$; $\hat{S}_{Aic}$ is $\rho_S = 40$. The number of inner iteration for ERPF1 is 2.

| $\Delta t$ [day] | $a/\alpha_K$ | $a/\alpha_A$ | $n_0$ | $T_s$ [s] | $T_i$ [s] | $n_0$ | $T_s$ [s] | $T_i$ [s] | $n_0$ | $T_s$ [s] | $T_i$ [s] |
|------------------|--------------|--------------|-------|----------|----------|-------|----------|----------|-------|----------|----------|
| $10^{-6}$        | 1.0 · 10^{-2} | > 1          | 49    | 124.9    | 203.9    | 25    | 102.2    | 181.4    | 58    | 216.0    | 286.6    |
| $10^{-5}$        | 2.8 · 10^{-2} | > 1          | 45    | 114.2    | 192.4    | 32    | 133.8    | 212.9    | 54    | 201.4    | 271.8    |
| $10^{-4}$        | 9.1 · 10^{-2} | > 1          | 45    | 114.7    | 193.1    | 29    | 118.2    | 197.2    | 46    | 170.1    | 240.1    |
| $10^0$           | > 1          | 2.8 · 10^{-1} | 247   | 629.7    | 709.1    | 179   | 582.9    | 661.4    | 134   | 274.4    | 334.0    |
| $10^1$           | > 1          | 9.4 · 10^{-2} | > 500 | -        | -        | 273   | 884.0    | 962.4    | 127   | 260.3    | 321.2    |
| $10^2$           | > 1          | 2.8 · 10^{-2} | > 500 | -        | -        | 386   | 1247.5   | 1326.1   | 135   | 277.4    | 337.2    |

### Table 9: Test 2b, SPE10: the same as Table 8 for RPF and ERPF2. The fill-in parameter for: $K_K^{-1}$ is $\rho_K = 30$; $\hat{A}^{-1}$ is $\rho_A = 40$; $\hat{S}_{Aic}$ is $\rho_S = 40$.

| $\Delta t$ [day] | $a/\alpha_K$ | $a/\alpha_A$ | $n_0$ | $T_s$ [s] | $T_i$ [s] | $n_0$ | $T_s$ [s] | $T_i$ [s] | $n_0$ | $T_s$ [s] | $T_i$ [s] |
|------------------|--------------|--------------|-------|----------|----------|-------|----------|----------|-------|----------|----------|
| $10^0$           | > 1          | 9.2 · 10^{-4} | 348   | 467.7    | 503.3    | 84    | 79.8     | 102.2    |       |          |          |
| $10^1$           | > 1          | 2.9 · 10^{-4} | > 1000| -        | -        | 87    | 82.7     | 104.5    |       |          |          |
| $10^2$           | > 1          | 9.2 · 10^{-5} | > 1000| -        | -        | 47    | 44.8     | 67.3     |       |          |          |
• Both methods are effective in improving the performance and robustness of the original RPF algorithm in the most critical situations, i.e., $\Delta t \to 0$ and $\Delta t \to \infty$, stabilizing the iteration counts to convergence independently of the time step size.

• ERPF1 is usually more efficient for small time step values, while ERPF2 outperforms, also by a large amount, RPF and ERPF1 for large time step values. Therefore, the most convenient approach in a full transient poromechanical application appears to be switching from ERPF1 when $\alpha < \alpha_k$ at the simulation beginning, to RPF when $\alpha > \alpha_k$ and $\alpha > \alpha_A$ with intermediate steps, to ERPF2 when $\alpha < \alpha_A$ approaching steady state conditions.

• The use of nested direct solvers ensures a scalable behavior of ERPF2 with respect to both the mesh and time spacings. Such a property is generally lost by using inexact inner solves, for instance by incomplete factorizations, but can be restored with scalable inner preconditioners, such as algebraic multigrid methods.

Acknowledgments

Partial funding was provided by TotalEnergies S.A. through the FC-MAELSTROM Project. Portions of this work were performed within the 2020 INdAM-GNCS project “Optimization and advanced linear algebra for PDE-governed problems”. Portions of this work were performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. The authors are grateful to two anonymous reviewers, whose comments significantly helped improve the presentation.

Appendix A. Alternative ERPF2 application

With the ERPF2 approach, the inner solution to $\hat{\mathbf{A}} \mathbf{w} = \mathbf{b}$ is obtained through equation (54) whenever $\beta > \beta_f$. With specific reference to Algorithm 2, this can happen twice at each preconditioner application either:

1. for $\alpha < \alpha_k$, having set $C = K, F = Q$, and $\beta = \alpha^{-1}$, or
2. for $\alpha < \alpha_A$, having set $C = A, F = B$, and $\beta = \gamma/\alpha$.

For the sake of brevity, let us consider the case no. 2, which typically occurs more often in practical applications, e.g., for large values of the time-step size. Similar considerations hold for the case no. 1.

Consider Algorithm 2. Recalling that $\mathbf{z}_q = \mathbf{r}_q + \alpha^{-1} \mathbf{B}_p \mathbf{y}_p$ (row 7), the application of $\hat{\mathbf{A}}^{-1}$ on $\mathbf{z}_q$ to get $\mathbf{t}_q$ using equation (54) yields:

$$
\mathbf{t}_q = \hat{\mathbf{A}}^{-1} \mathbf{z}_q \\
= \mathbf{A}^{-1} \mathbf{r}_q + \alpha^{-1} A^{-1} B \mathbf{B}_p - \beta A^{-1} B \mathbf{S}_A^{-1} B^T A^{-1} \mathbf{r}_q - \beta A^{-1} B \mathbf{S}_A^{-1} S_A \mathbf{y}_p \\
= \mathbf{A}^{-1} \mathbf{r}_q + \alpha^{-1} A^{-1} B \mathbf{S}_A^{-1} \left( \tilde{S}_A - \beta S_A \right) \mathbf{y}_p - \beta A^{-1} B \mathbf{S}_A^{-1} B^T A^{-1} \mathbf{r}_q \\
= \mathbf{A}^{-1} \mathbf{r}_q + \alpha^{-1} A^{-1} B \mathbf{S}_A^{-1} \mathbf{y}_p - \beta A^{-1} B \mathbf{S}_A^{-1} B^T A^{-1} \mathbf{r}_q \\
= \mathbf{A}^{-1} \mathbf{r}_q + A^{-1} B \mathbf{S}_A^{-1} \left( \alpha \mathbf{y}_p - \beta B^T A^{-1} \mathbf{r}_q \right) \\
= \mathbf{A}^{-1} \left[ \mathbf{r}_q + B \mathbf{S}_A^{-1} \left( \alpha \mathbf{y}_p - \beta B^T A^{-1} \mathbf{r}_q \right) \right].
$$

(A.1)

Recalling that $\beta = \gamma/\alpha$, lines 9 and 10 of Algorithm 2 can be rearranged as follows:

$$
\mathbf{t}_p = \alpha^{-1} \mathbf{y}_p - \beta B^T \mathbf{r}_q \\
= \alpha^{-1} \mathbf{y}_p - \beta B^T A^{-1} \mathbf{r}_q - \beta B^T A^{-1} B \mathbf{S}_A^{-1} \left( \alpha \mathbf{y}_p - \beta B^T A^{-1} \mathbf{r}_q \right) \\
= \left( \mathbf{I} - \beta B \mathbf{S}_A^{-1} \right) \left( \alpha \mathbf{y}_p - \beta B^T A^{-1} \mathbf{r}_q \right) \\
= \mathbf{S}_A^{-1} \left( \alpha \mathbf{y}_p - \beta B^T A^{-1} \mathbf{r}_q \right).
$$

(A.2)
Algorithm 5 Alternative ERPF2 application: \([t'_u, t'_q, t'_p] = \text{AltERPF2\_Apply}(r_u, r_q, r_p, A, M)\)

1: if \(\alpha > \alpha_K\) then
2: \(x_u = \alpha^{-1}Qr_p\)
3: \(x_u \leftarrow r_u + x_u\)
4: Apply \(M^{-1}_S\) to \(x_u\) to get \(t_u\)
5: \(y_p = Q^T t_u\)
6: \(y_p \leftarrow r_p - y_p\)
7: else
8: Apply \(M^{-1}_S\) to \(r_u\) to get \(x_u\)
9: \(x_p = Q^T x_u\)
10: \(x_p \leftarrow r_p - x_p\)
11: Apply \(M^{-1}_S\) to \(x_p\) to get \(y_p\)
12: \(x_u = Qy_p\)
13: \(x_u \leftarrow \alpha x_u + x_u\)
14: Apply \(M^{-1}_S\) to \(x_u\) to get \(t_u\)
15: end if
16: if \(\alpha > \alpha_A\) then
17: \(z_q = By_p\)
18: \(z_q \leftarrow r_q + \alpha^{-1}z_q\)
19: Apply \(M^{-1}_A\) to \(z_q\) to get \(t_q\)
20: \(t_p = B^T t_q\)
21: \(t_p \leftarrow \alpha^{-1}\left(y_p - \gamma t_q\right)\)
22: else
23: Apply \(M^{-1}_A\) to \(r_q\) to get \(z_q\)
24: \(z_p = B^T z_q\)
25: \(y_p \leftarrow \alpha^{-1}\left(y_p - \gamma z_q\right)\)
26: Apply \(M^{-1}_A\) to \(y_p\) to get \(t_p\)
27: \(z_q = Bt_p\)
28: \(z_q \leftarrow r_q + z_q\)
29: Apply \(M^{-1}_A\) to \(z_q\) to get \(t_q\)
30: end if

The vectors \(t_p\) and \(t_q\) are therefore computed by:

\[
\begin{align*}
    t_p & = S_A^{-1}\left(\alpha^{-1}y_p - \beta B^T A^{-1}r_q\right), \\
    t_q & = A^{-1}\left(r_q + Bt_p\right).
\end{align*}
\] (A.3) (A.4)

Equations (A.3) and (A.4) shows that Method 2 used for the inner solve with \(\hat{A}\) is equivalent to apply the inverse of the following block factorization for \(M_2\):

\[
\begin{bmatrix}
    I_u & 0 & 0 \\
    0 & A & -B \\
    0 & \gamma B^T & \alpha I_p
\end{bmatrix}
\begin{bmatrix}
    I_u & 0 & 0 \\
    0 & A & 0 \\
    0 & \gamma B^T & S_A
\end{bmatrix}
\begin{bmatrix}
    I_u & 0 & 0 \\
    0 & I_q & -A^{-1}B \\
    0 & 0 & \alpha I_p
\end{bmatrix}
\] (A.5)

In fact, solving the block system \(M_2t = y\) with the aid of (A.5) yields:

\[
\begin{bmatrix}
    t_u \\
    t_q \\
    t_p
\end{bmatrix}
\begin{bmatrix}
    I_u & 0 & 0 \\
    0 & I_q & \alpha^{-1}A^{-1}B \\
    0 & 0 & \alpha^{-1}I_p
\end{bmatrix}
\begin{bmatrix}
    I_u & 0 & 0 \\
    0 & A^{-1} & 0 \\
    0 & -\gamma S_A^{-1}B^T A^{-1} & S_A^{-1}
\end{bmatrix}
\begin{bmatrix}
    y_u \\
    y_q \\
    y_p
\end{bmatrix}
\] (A.6)
which provides equations (A.3)-(A.4) for $y_q = r_q$.

With similar arguments, after some algebra it can be easily proved that Method 2 for the inner solve with $\tilde{K}$ is equivalent to apply the inverse of the following factorization of $M_1$:

\[
\begin{bmatrix}
K & 0 & -Q \\
0 & \alpha I_q & 0 \\
0 & 0 & \alpha I_p \\
\end{bmatrix} = \begin{bmatrix}
K & 0 & 0 \\
0 & \alpha I_q & 0 \\
0 & 0 & \alpha I_p \\
\end{bmatrix} \begin{bmatrix}
I_u & 0 & -K^{-1}Q \\
0 & I_q & 0 \\
0 & 0 & \tilde{K} \\
\end{bmatrix}.
\] (A.7)

The solution to the system $aM_1y = r$ using (A.7) reads:

\[
\begin{bmatrix}
y_u \\
y_q \\
y_p \\
\end{bmatrix} = a \begin{bmatrix}
I_u & 0 & -\alpha^{-1}K^{-1}Q \\
0 & I_q & 0 \\
0 & 0 & \alpha^{-1}I_p \\
\end{bmatrix} \begin{bmatrix}
K^{-1} & 0 & 0 \\
0 & \alpha^{-1}I_q & 0 \\
0 & 0 & \tilde{S}_K \\
\end{bmatrix} \begin{bmatrix}
r_u \\
r_q \\
r_p \\
\end{bmatrix},
\] (A.8)

which provides:

\[
y_p = \tilde{S}_K^{-1}(r_p - Q^{-1}K^{-1}r_u),
\] (A.9)

\[
y_u = K^{-1}(ar_u + Qy_p),
\] (A.10)

with $y_q = r_q$.

As already noticed in Section (3.2), the inner solves with $K$, $A$, $\tilde{S}_K$, and $\tilde{S}_Q$ can be conveniently replaced by inexact solves with the aid of inner preconditioners, say $M_K^{-1}$, $M_A^{-1}$, $\tilde{S}_K^{-1}$, and $M_K^{-1}$, respectively. The equivalent ERPF2 application obtained by using the factorizations above is finally summarized in Algorithm 5.

References

[1] M. A. Biot, General theory of three-dimensional consolidation, J. Appl. Phys. 12 (1941) 155–164. doi:10.1063/1.1712886.

[2] O. Coussy, Poromechanics, Wiley, Chichester, UK, 2004. doi:10.1002/0470092718.

[3] A. J. H. Frijns, A four-component mixture theory applied to cartilaginous tissues: numerical modelling and experiments, PhD thesis, Technische Universiteit Eindhoven, The Netherlands (2000). doi:10.6100/IR537990.

[4] P. J. Phillips, M. F. Wheeler, A coupling of mixed and continuous Galerkin finite element methods for poroelasticity I: the continuous in time case, Comput. Geosci. 11 (2) (2007) 131–144. doi:10.1007/s10596-007-9045-y.

[5] P. J. Phillips, M. F. Wheeler, A coupling of mixed and continuous Galerkin finite element methods for poroelasticity II: the discrete-in-time case, Comput. Geosci. 11 (2) (2007b) 145–158. doi:10.1007/s10596-007-9044-z.

[6] M. Bause, F. A. Radu, U. Köcher, Space–time finite element approximation of the Biot poroelasticity system with iterative coupling, Comput. Meth. Appl. Mech. Eng. 320 (2017) 745–768. doi:10.1016/j.cma.2017.03.017.

[7] X. Hu, C. Rodrigo, F. J. Gaspar, L. T. Zikatanov, A nonconforming finite element method for the Biot's consolidation model in poroelasticity, J. Comput. Appl. Math. 310 (2017) 143–154. doi:10.1016/j.cam.2016.06.003.

[8] J. W. Both, K. Kumar, J. M. Nordbotten, F. A. Radu, Anderson accelerated fixed-stress splitting schemes for consolidation of unsaturated porous media, Comput. Math. Appl. 77 (6) (2019) 1479–1502. doi:10.1016/j.camwa.2018.07.033.

[9] V. Girault, K. Kumar, M. F. Wheeler, Convergence of iterative coupling of geomechanics with flow in a fractured porous medium, Comput. Geosci. 20 (5) (2016) 997–1011. doi:10.1007/s10596-016-9678-4.

[10] Girault, Vivette, Wheeler, Mary F., Almani, Tameem, Dana, Saumik, A priori error estimates for a discretized poro-elastic-system solved by a fixed-stress algorithm, Oil Gas Sci. Technol. - Rev. IFP Energies nouvelles 74 (2019) 24. doi:10.2516/ogst/2018071.

[11] M. Ferronato, N. Castelletto, G. Gambolati, A fully coupled 3-D mixed finite element model of Biot consolidation, J. Comput. Phys. 229 (12) (2010) 4813–4830. doi:10.1016/j.jcp.2010.03.018.

[12] N. Castelletto, J. A. White, M. Ferronato, Scalable algorithms for three-field mixed finite element coupled poromechanics, J. Comput. Phys. 327 (2016) 894–918. doi:10.1016/j.jcp.2016.09.063.

[13] R. W. Lewis, B. A. Schrefler, The Finite Element Method in the Static and Dynamic Deformation and Consolidation of Porous Media, 2nd Edition, Wiley, Chichester, UK, 1998.

[14] N. Castelletto, M. Ferronato, G. Gambolati, Thermo-hydro-mechanical modeling of fluid geological storage by Godunov-mixed methods, Int. J. Numer. Meth. Eng. 90 (8) (2012) 988–1009. doi:10.1002/nme.3352.

[15] B. Jha, R. Juanes, Coupled multiphase flow and poromechanics: A computational model of pore pressure effects on fault slip and earthquake triggering, Water Resour. Res. 5 (2014) 3776–3808. doi:10.1002/2013WR015175.

[16] J. Choo, S. Lee, Enriched galerkin finite elements for coupled poromechanics with local mass conservation, Computer Methods in Applied Mechanics and Engineering 341 (2018) 311 – 332. doi:https://doi.org/10.1016/j.cma.2018.06.022.

[17] J. Choo, Large deformation poromechanics with local mass conservation: An enriched galerkin finite element framework, International Journal for Numerical Methods in Engineering 116 (1) (2018) 66–90. doi:10.1002/nme.5915.

[18] K. Lipnikov, Numerical Methods for the Biot Model in Poroelasticity, PhD thesis, University of Houston (2002).
