# Response to Reviewer Comments

Manuscript: "Efficient Uzawa algorithms with projection strategies for geodynamic Stokes flow"

by Jang, Lee, Thieulot, Choi and So

Dear Editor and Professor Stadler,

We sincerely thank Professor Stadler for the thorough and constructive review of our manuscript. We appreciate the positive feedback regarding the clarity of writing and careful preparation, and the insightful comments have helped us significantly improve the manuscript. Below, we provide detailed point-by-point responses to all comments and describe the corresponding changes that will be reflected in the revised manuscript. All modifications will be highlighted in blue in the revised manuscript.

## Response to Reviewer Comments

### General Comments

**Reviewer:** *As the authors acknowledge in the introduction, Uzawa methods are no longer regarded as state-of-the-art for Stokes problems; contemporary approaches typically handle velocity and pressure simultaneously and generally achieve superior convergence. The authors argue that Uzawa methods are simple, modular, and therefore easy to implement. I am not entirely persuaded by this claim. Given the solver technology available in modern FEM and linear algebra libraries (such as FEniCS and PETSc), implementing a full-space Stokes solver with a block-diagonal preconditioner is not particularly difficult.*

**Response:**

We thank the reviewer for this observation. Classical Uzawa iterations are indeed no longer the standard approach. However, our reformulation through variational principles enables conjugate gradient methods for the Schur complement, achieving robust convergence for highly heterogeneous viscosity problems. The key insight is that the inverse $\eta$-weighted operations in our solver correspond directly to $L^2$ inner products in the finite element space, making the approach a natural extension of the variational framework rather than an ad hoc modification. Additionally, we introduce a projection post-processing step that reduces error accumulation in time-dependent problems at minimal computational cost. Our numerical experiments demonstrate competitive performance across various benchmarks, and we believe the proposed method is competitive with modern full-space approaches while retaining the modularity of segregated solvers. We have revised the introduction accordingly.

**Changes in manuscript:**

The abstract and introduction have been revised to explicitly state that the proposed method reformulates the Uzawa iteration through weighted $L^2$ inner products, yielding a robust preconditioned conjugate gradient iteration for the Schur complement system, and to position this contribution as a complementary algorithmic perspective rather than a replacement for modern block-preconditioned Krylov solvers.

## Specific Comments

### Comment 1: Line 71 - Vector Laplacian and boundary conditions

**Reviewer:** *As for the statement that the constant-viscosity Stokes system reduces to the vector Laplacian: disregarding boundary conditions this is certainly true, and it also holds for Dirichlet conditions. But is it still generally valid for more complex boundary conditions?*

**Response:**

The reviewer is correct to point out this important distinction. The reduction to vector Laplacian is exact for Dirichlet boundary conditions, but for Neumann or mixed boundary conditions, the pressure gradient term must be handled differently and the pressure-velocity coupling must be retained. We have qualified the statement in line 71 to avoid confusion.

**Changes in manuscript:**

The statement on the reduction to the vector Laplacian has been qualified to: *"In the case of constant viscosity with Dirichlet boundary conditions, the momentum equation reduces to a vector Laplacian."*

### Comment 2: $\eta$-weighted preconditioner notation

**Reviewer:** *Throughout the paper, the authors refer to the $\eta$-weighted version of the preconditioner. Is this weight not actually the inverse viscosity?*

**Response:**

The reviewer is absolutely correct. As shown in Eq. (18), $\langle \frac{1}{\eta} w^k, r \rangle = \langle q^k, r \rangle$, the weighting is indeed by $1/\eta$ (inverse viscosity). We have added clarification in the manuscript to avoid confusion while keeping the notation CD-U-$\eta$ for brevity.

**Changes in manuscript:**

A clarification has been added noting that the weighting is by $1/\eta$ (inverse viscosity), denoted as $\eta$-weighted for brevity: *"Note that the weighting is by $1/\eta$, i.e., the inverse of viscosity, which we denote as $\eta$-weighted for brevity throughout this paper."* The notation CD-U-$\eta$ remains unchanged throughout.

### Comment 3: Line 148 - Divergence-free condition and projection

**Reviewer:** *The authors argue that the divergence-free condition is slightly violated due to inexact solves and that this is corrected by the Helmholtz projection. I can see this being the case if Taylor–Hood elements are also used in the projection step, which appears to be what is done. Would this issue vanish entirely if more iterations were used so that the system is solved more accurately? This could be investigated further in the numerical experiments.*

**Response:**

The reviewer raises two important points, both of which are correct.

First, regarding the projection step, the reviewer is correct that we use Taylor-Hood (P2-P1) elements throughout, including in the projection. The projection solves a Poisson equation for the scalar potential $\phi$ in the P1 pressure space, then updates the velocity through $u^{k+1} = \hat{u} + \nabla\phi$ in the P2 velocity space. While projection is theoretically designed to enforce $\nabla \cdot u = 0$

exactly, the measured divergence residual does not reach machine zero in practice. This happens because the discrete Laplacian operator, which is assembled from the variational form $\langle \nabla\phi, \nabla\psi \rangle$, does not exactly equal the composition $BG$, where $B : V_h \to Q_h$ and $G : Q_h \to V_h$ denote the discrete divergence and gradient operators, respectively. The effect of projection in a single time step is small. However, in long time-dependent simulations with thousands of steps, these small improvements accumulate to produce substantial differences in accuracy.

Second, the reviewer correctly notes that with sufficiently many iterations, projection becomes unnecessary. In principle, if we iterate until very tight tolerances, the divergence constraint is satisfied without projection. However, we deliberately use relaxed tolerances in time-dependent simulations to maintain efficiency. This raises a key question: which is more cost-effective—more iterations or fewer iterations with projection?

To answer this, we tested both strategies on two time-dependent benchmarks: mantle convection and block sinking. In the results below, "CD-U-$\eta$($n$iter)" denotes $n$ iterations of the CD-U-$\eta$ solver per time step, and "+Proj" indicates projection post-processing. Numbers in parentheses show the projection overhead.

**Mantle convection (5000 steps, $\Delta t = 10^{-6}$):**

| | Uzawa time | $E_u$ | $R_{\text{div}}$ | $E_T$ |
|---|---|---|---|---|
| CD-U-$\eta$(2iter) | 1277s | 0.0177 | 0.0138 | 0.00447 |
| CD-U-$\eta$(2iter)+Proj | 1355s (+78s) | 0.000633 | 0.00144 | 0.000120 |
| CD-U-$\eta$(3iter) | 1704s | 0.00775 | 0.00975 | 0.00187 |
| CD-U-$\eta$(3iter)+Proj | 1783s (+79s) | 0.0000970 | 0.000263 | 0.0000178 |

**Block sinking (1500 steps, $\Delta t = 5 \times 10^{-6}$):**

For this benchmark, we used a practical strategy: the initial step iterates to tolerance, while subsequent steps use fixed iteration counts with the previous solution as initial guess.

| | Uzawa time | $E_u$ | $R_{\text{div}}$ | $E_\rho$ |
|---|---|---|---|---|
| CD-U-$\eta$(5iter) | 1253s | 0.0449 | 0.0234 | 0.0236 |
| CD-U-$\eta$(5iter)+Proj | 1278s (+25s) | 0.00200 | 0.00384 | 0.00127 |
| CD-U-$\eta$(6iter) | 1463s | 0.0336 | 0.0198 | 0.0160 |
| CD-U-$\eta$(6iter)+Proj | 1485s (+22s) | 0.00875 | 0.00608 | 0.00113 |

Both benchmarks show that projection adds minimal overhead (2-6% of total runtime) while improving accuracy by 1-2 orders of magnitude. This efficiency comes from the cost difference: each additional iteration requires solving the momentum equation, while projection requires only one additional Poisson solve per time step. Although projection can only be applied once per time step, it can enhance any iterative Stokes solver as a post-processing step.

**Changes in manuscript:**

An appendix has been added presenting a comparison of projection post-processing versus additional iterations for two time-dependent benchmarks (mantle convection and block sinking), demonstrating that projection achieves superior accuracy at lower computational cost than simply increasing the iteration count.

### Comment 4: Mesh refinement studies

**Reviewer:** *Figure 1 (and, in fact, all experiments) use a *fixed* discretization. Yet, one of the most important properties of numerical methods is their behavior under mesh refinement. It would be valuable to show results for at least one level of finer mesh.*

**Response:**

We have conducted mesh refinement studies using the SolKz benchmark from Duretz et al. (2011). We chose SolKz over the SolCx benchmark presented in our manuscript because SolCx's discontinuous viscosity at $x = 0.5$ reduces solution regularity, preventing observation of optimal convergence rates with standard finite elements. SolKz provides smooth exponentially varying viscosity $\eta(y) = \exp(cy)$ with dynamic ratio $10^6$, enabling proper verification of convergence rates using P2-P1 Taylor-Hood elements.

We tested resolutions from 32×32 to 1024×1024 with reference solutions obtained from the Underworld software. The following table shows the mesh refinement results. Our method achieves the optimal convergence rates for Taylor-Hood elements (order 3 for velocity and order 2 for pressure).

| Resolution | $E_u$ | Order | $E_p$ | Order |
|---|---|---|---|---|
| 32×32 | $1.877\times10^{-3}$ | - | $1.591\times10^{-2}$ | - |
| 64×64 | $2.316\times10^{-4}$ | 3.02 | $3.970\times10^{-3}$ | 2.00 |
| 128×128 | $2.888\times10^{-5}$ | 3.00 | $9.872\times10^{-4}$ | 2.01 |
| 256×256 | $3.610\times10^{-6}$ | 3.00 | $2.401\times10^{-4}$ | 2.04 |
| 512×512 | $4.514\times10^{-7}$ | 3.00 | $5.845\times10^{-5}$ | 2.04 |
| 1024×1024 | $5.648\times10^{-8}$ | 3.00 | $1.490\times10^{-5}$ | 1.97 |

**Adaptive AAR strategy for high-resolution cases.**

At high resolution (1024×1024), the conjugate gradient iteration occasionally stagnates. This occurs when the denominators in the step-size and conjugacy coefficients become too small (near machine precision), causing the updates to effectively stop since we set the coefficients to zero to avoid division by near-zero values. To overcome this, we introduce an adaptive AAR parameter that scales the preconditioner weights. Instead of using $1/\eta(x)$ directly in the $\eta$-weighted mass matrix, we use $\mathrm{AAR}/\eta(x)$.

The general strategy is as follows. We start from AAR $= 1.0$ and reduce it when stagnation is detected. However, as AAR is progressively reduced, the weights can become excessively small in high-viscosity regions, which introduces numerical instability. Therefore, we enforce a pointwise lower bound on the weights.

For the SolKz benchmark, we implement this strategy as follows. When $|R_{\mathrm{div}}^{k+1} - R_{\mathrm{div}}^{k}| < 10^{-10}$ for 2 consecutive iterations, AAR is reduced by a factor of 0.1. To maintain numerical stability, we enforce $\min_x(\mathrm{AAR}/\eta(x)) \geq 10^{-6}$.

The following table illustrates the adaptive AAR strategy for the 1024×1024 case. The solver initially converges normally (iterations 4-7) but stagnates at iteration 8. Adaptive AAR reduction immediately resumes progress, ultimately achieving $E_u \sim 5.6 \times 10^{-8}$ and $R_{\mathrm{div}} \sim 1.6 \times 10^{-8}$ by iteration 500.

| Iteration | AAR | AAR change | $E_u$ | $E_p$ |
|---|---|---|---|---|
| 4 | 1.0 | – | 1.99e-03 | 2.03e-01 |
| 5 | 1.0 | – | 2.55e-04 | 1.77e-02 |
| 6 | 1.0 | – | 2.57e-05 | 1.22e-03 |
| 7 | 1.0 | – | 2.61e-07 | 3.10e-05 |
| 8 | 1.0 | – | 2.61e-07 | 3.10e-05 |
| 9 | 1.0 | – | 2.61e-07 | 3.10e-05 |
| 10 | 0.1 | 1.0 → 0.1 | 5.75e-08 | 2.17e-05 |
| 13 | 0.01 | 0.1 → 0.01 | 5.75e-08 | 2.17e-05 |
| 15 | 0.001 | 0.01 → 0.001 | 5.67e-08 | 2.16e-05 |
| 388 | 0.0001 | 0.001 → 0.0001 | 5.65e-08 | 1.50e-05 |
| 500 | 0.0001 | – | 5.65e-08 | 1.49e-05 |

For resolutions up to 256×256, the adaptive strategy was not triggered and the solver converged with AAR= 1.0 throughout. At 512×512, AAR adjustment occurred occasionally. At 1024×1024, the adaptive strategy was actively used and proved essential for convergence. We also apply this technique in the multisinker benchmark discussed in Comment 5.

**Changes in manuscript:**

A mesh refinement study using the SolKz benchmark has been added, demonstrating convergence across increasing resolutions up to $1024 \times 1024$. The adaptive AAR strategy is described in the main text, and the corresponding numerical results are provided in an appendix.

### Comment 5: More challenging benchmarks

**Reviewer:** *I would conjecture that the benchmarks in Sec. 3.3.1 and 3.3.2 have rather smooth solutions, so I am not convinced that these are the most informative problems to consider. The May/Moresi sinker example (Sec. 3.4.2) is more demanding, although a multi-sinker setup (e.g., May et al., SC Proceedings, 2014; Rudi et al., SISC 2017) is known to be even more challenging and would definitely be insightful.*

**Response:**

We thank the reviewer for this insightful suggestion. Following the recommendation, we conducted multisinker benchmark experiments based on the setup described in Rudi et al. (2017, SISC, "Weighted BFBT preconditioner for Stokes flow problems with highly heterogeneous viscosity").

**Experimental setup:**

- Domain: unit square $\Omega = [0,1]^2$ with 8 high-viscosity sinkers at prescribed locations
- Viscosity field: $\mu(x) = (\mu_{\max} - \mu_{\min})(1 - \chi_8(x)) + \mu_{\min}$, where $\chi_8(x) = \prod_{i=1}^{8}[1 - \exp(-\delta \max(0, |c_i - x| - \omega)^2)]$ with $\delta = 200$, $\omega = 0.075$
- Body force: $\mathbf{f}(x, y) = (0, \beta(\chi_8(x, y) - 1))$ with $\beta = 10^5$
- Dynamic ratios: $\mathrm{DR}(\mu) = 10^2$ to $10^{10}$, with $\mu_{\min} = \mathrm{DR}^{-1/2}$ and $\mu_{\max} = \mathrm{DR}^{1/2}$
- Discretization: $256 \times 256$ triangular P2-P1 Taylor-Hood elements
- Reference solution: MUMPS direct solver

- Adaptive AAR strategy: same as Comment 4, with stagnation threshold $10^{-8}$ and lower bound $\min_x(\mathrm{AAR}/\eta(x)) \geq 10^{-8}$.

**Results:** Since this multisinker problem has no analytical solution, we compute reference solutions using the MUMPS direct solver to evaluate convergence. Unlike the original study by Rudi et al. (2017), which used residual reduction as the stopping criterion, we measure convergence using relative errors in velocity and pressure. This approach provides a more direct assessment of solution accuracy. The following table presents the number of CD-U-$\eta$ iterations required to achieve three different accuracy levels ($10^{-3}$, $10^{-6}$, and $10^{-9}$) across dynamic ratios ranging from $10^2$ to $10^{10}$. The reported iteration count is the number of iterations needed for both velocity and pressure relative errors to simultaneously fall below the specified threshold.

| DR($\mu$) | $E_u, E_p < 10^{-3}$ | $E_u, E_p < 10^{-6}$ | $E_u, E_p < 10^{-9}$ |
|---|---|---|---|
| $10^2$ | 20 | 37 | 52 |
| $10^4$ | 65 | 83 | 93 |
| $10^6$ | 84 | 93 | 103 |
| $10^8$ | 83 | 94 | 108 |
| $10^{10}$ | 83 | 95 | 109 |

**Changes in manuscript:**

Multisinker benchmark results have been added, including a figure illustrating the viscosity field and the corresponding velocity and pressure solutions, to demonstrate solver robustness under strongly heterogeneous viscosity configurations.