Journal of Computational Physics 230 (2011) 8835–8851

Contents lists available at SciVerse ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Development of a Stokes flow solver robust to large viscosity jumps using a Schur complement approach with mixed precision arithmetic

Mikito Furuichi [a,*], Dave A. May [b], Paul J. Tackley [b]

[a] Institute for Research on Earth Evolution, Japan Agency for Marine-Earth Science and Technology (JAMSTEC), 3173-25 Showa-machi, Kanazawa, Yokohama, Japan
[b] Institute of Geophysics, Department of Earth Sciences, ETH Zurich, CH-8092 Zurich, Switzerland

ARTICLE INFO

Article history:
Received 30 June 2010
Received in revised form 24 January 2011
Accepted 1 September 2011
Available online 12 September 2011

Keywords:
Stokes flow
Mantle convection
Viscosity jump
Preconditioner
Double–double
High precision
Multi grid

ABSTRACT

We develop an iterative solution technique for solving Stokes flow problems with smooth and discontinuous viscosity structures using a three dimensional, staggered grid finite difference discretization. Two preconditioned iterative methodologies are applied to the saddle point arising from the discrete Stokes problem. They consist of a velocity–pressure coupled approach (FC) and a decoupled, Schur complement approach (SC). Within both of these methods, we utilize either the scaled BFBt, or an identity matrix scaled by the local cell viscosity (LV) to define a preconditioner for the Schur complement. Additionally, we propose to use a mixed precision Krylov kernel to improve the convergence by reducing round-off error. In this approach, standard double precision is used during the application of the preconditioner, whilst higher precision arithmetic is used to define the matrix vector product, dot products and norms required by the Krylov method. In our Krylov kernel, we utilize quad precision arithmetic which is emulated via the double–double precision method. We consider several simplified geodynamic problems with a viscosity contrast to demonstrate the robustness and scalability of our solution methods. Through a careful choice of stopping conditions, we are able to quantitatively compare the residuals between the SC and FC approaches. We examine the trade-off relationship between the number of outer iterations required for convergence, and the computational cost per iteration, for the each solution methods. We find that it is advantageous to use the FC approach utilizing relaxed tolerances for solution of the sub-problems, combined with the LV preconditioner. We also observed that in general, the SC approach is more robust than FC and that BFBt is more robust than LV when used in our numerical experimental. In addition, our mixed precision method produces improved convergence rates of Arnoldi type Krylov subspace methods without a drastic increasing the computational time. The usage of a high precision Krylov kernel is found to be useful for the solver associated with the velocity sub-problem.

1. Introduction

The development of an efficient and robust Stokes flow solver for problems with large and discontinuous viscosity contrasts $\Delta\eta$, is an interesting challenge in computational mathematics. One of the main applications of this problem is the study of the long time scale dynamics of the Earth's convecting mantle, and the formation and subsequent evolution of plate tectonics. Such processes can be suitably described via Stokes equations as rocks intrinsically possess a strongly tem-

* Corresponding author. Tel.: +81 45 778 5854; fax: +81 45 778 5490.
E-mail address: m-furuic@jamstec.go.jp (M. Furuichi).

0021-9991/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2011.09.007

perature-dependent viscosity and also a high absolute viscosity (10^{19}–10^{22} Pa s), thereby allowing the infinite Prandtl number approximation to be made [1].

Robustness and scalability of a solver with respect to large $\Delta\eta$ is important for treating realistic problems in geodynamical modeling. For example, a large viscosity contrast is expected between the upper mantle and the tectonic plates as a first-order approximation to their rheology [2]. Another example arises in the treatment of a free surface by the sticky air (i.e. very low viscosity, low density material) method (e.g. [3]). Here the large viscosity contrast will occur over one to several grid cells, if the boundary interface is represented, for example, by a low diffusion advection scheme [4,5]. Furthermore, the modeling of the mantle dynamics with a temperature, pressure and strain-rate (or stress) dependent viscosity [6,7], can produce viscosity contrasts up to $O(10^6)$ over a very narrow zone.

In earlier studies, several groups have reported the development of their own Stokes flow solver for the mantle convection problem (e.g. [8–19]). Nevertheless, despite these progresses it still remains a challenging problem to handle locally and highly varying viscosity structures in three dimensions using an efficient, scalable (e.g. multigrid) iterative solution technique, which is necessary to solve three dimensional problems with very high grid resolution.

In this study, we design and investigate an iterative Stokes flow solver which can handle problems with the extremely large contrast in viscosity in three dimensions. Our strategy is based on two key techniques: the pressure Schur complement and mixed precision arithmetic.

The Schur complement has been used to solve the Stokes saddle point problem in mantle convection simulations. For example, SIMPLE(R) like methods [12,13,20] use an approximate solution of the Schur complement as a part of their iteration process to derive the pressure corrections. In general, the convergence of solution obtained using this type of Schur complement approach degrades with increasing viscosity contrast. In the solution strategy used here, the Schur complement appears in the pressure–velocity coupled or decoupled approaches used to solve the saddle point problem, and is solved with a preconditioned Krylov subspace method. Recently two types of Schur complement preconditioners, scaled BFBt [21–24] and weighted mass matrix preconditioner [18,25–27], are argued to be an effective approach for strongly variable viscosity problems. Numerical experiments from earlier studies, for example [18,21,23,27], have demonstrated the performance of various preconditioning methods including one of them for the variable viscosity problems, typically using a finite element discretization. Here, we have implemented these two preconditioning techniques, and analyzed their performance with respect to viscosity contrast using the finite difference discretization on a staggered grid in three dimensions.

Using idealized model setups which are characteristic of geodynamic applications, we performed a number of numerical experiments in which we compare (i) coupled vs. decoupled solution strategies for solving the saddle point problem, and (ii) the strength of two different Schur complement preconditioners. The model problems represent two classes of viscosity structure, namely discontinuous structures (i.e. localized contrast $\sim O(10^6)$) and diffuse structures which have any enormous contrast $\sim O(10^{12})$ over the whole domain. By examining a wide range of solver/preconditioning strategies for these two classes of viscosity structure, our performance analysis and comparison provides a guide as to which technique may prove beneficial to use with future model setups. Choosing a suitable method is not always straightforward without understanding the robustness of the approach. For instance, it is not always clear where the cross over point is between methods which require many iterations but the cost per iteration is cheap, versus a method which requires few iterations which are expensive.

In addition, we partially incorporated high precision arithmetic into Krylov subspace method to improve the convergence behavior. We are especially interested in applying this technique to Krylov methods preconditioned with geometric multigrid, which are used to solve the momentum equation. Here, the double–double precision algorithms proposed by Bailey [28] are employed to emulate quad precision arithmetic using a pair of double precision values [29].

All the solution methods presented here are specifically designed for vector processing machine especially for the Earth Simulator 2 (based on the NEC SX-9). Although the numerical experiments were performed on a vector machine, the performance analysis of this study is expected to show similar general trends if implemented on a scalar CPU architecture.

In the following section, we explain the Stokes flow model and the spatial discretization used throughout in this study. In Section 3, we will introduce our solver design for the saddle point problem and describe the preconditioners used for the Schur complement, and sub-problems of the Stokes flow system. The mixed precision strategy for double–double precision is given in Section 4. The numerical experiments for Stokes solvers and sub-problem solvers are provided in the Section 5. Finally, in Section 6, we provide concluding remarks.

2. Stokes flow problem

We solve for Stokes flow given by the momentum equation,

$$-\frac{\partial \tau_{ij}}{\partial x_j}+\frac{\partial p}{\partial x_i}=-\frac{\partial}{\partial x_j}\left[\eta\left(\frac{\partial u_j}{\partial x_i}+\frac{\partial u_i}{\partial x_j}\right)\right]+\frac{\partial p}{\partial x_i}=f_i, \tag{1}$$

and continuity equation

$$-\frac{\partial u_i}{\partial x_i}=h, \tag{2}$$

in a three dimensional Cartesian geometry, where u is the velocity vector with u_i in ith direction ($i = 1, 2, 3$ in three dimensions) and p is the pressure. The variables, η, τ_{ij} and f are viscosity, deviatoric stress and body force respectively. We impose the 'free-slip' (i.e. $u_i = 0$, $\partial u_{j\neq i}/\partial x_i = 0$ on the wall normal to ith direction) or 'no-slip' (i.e. $u_i = 0$ on the wall) boundary conditions. The linear (arithmetic) mean method was used for averaging viscosity to calculate the shear stress components. In our work, we only consider incompressible flow in which $h = 0$, however for completeness we leave h in the continuity equation throughout this paper. In addition, we only consider a linear (Newtonian) constitutive equation: i.e. η is independent of u and p.

We apply the finite difference discretization to (1) and (2) on a staggered grid with a size of $n = Nx \times Ny \times Nz$. The total number of velocity unknowns is denoted by m. The discretization of (1) and (2) gives a system of linear equations

$$A\vec{x} = \begin{pmatrix} K & G \\ D & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ h \end{pmatrix} = \vec{b}, \tag{3}$$

where $u \in \Re^m$ is the discrete velocity vector, $p \in \Re^n$ is the discrete pressure solution, $K \in \Re^{m\times m}$ is a positive definite matrix for the gradient of the deviatoric stress term, $G \in \Re^{m\times n}$ and $D \in \Re^{n\times m}$ provide the discretized pressure gradient and divergence of velocity respectively. In this study, we only investigate the case when the grid spacing is constant, i.e. $dx = dy = dz$. Under these conditions, $D = G^T$ is satisfied in the finite difference discretization, and the matrix A is symmetric. When the large contrast of matrix elements is caused by a large $\Delta\eta$, the matrix K becomes ill-conditioned.

3. Solver design

In this section, we will explain the examined solution techniques and the interplay between them, which are schematically summarized in Fig. 1.

3.1. Solution of the saddle point problem

We consider two general solution approaches to treat saddle point problems: they are the so-called decoupled (segregated) and fully coupled (all at once) approaches. These approaches involve the solution of simpler sub-problems for the velocity and pressure.

3.1.1. Decoupled Schur complement reduction approach (SC)

We employ Schur complement reduction (SC) to obtain a pressure equation, which is decoupled from the momentum equation [30,31]. In order to decouple the systems (3) is rewritten as,

$$\begin{pmatrix} I & 0 \\ -G^T K^{-1} & I_P \end{pmatrix} \begin{pmatrix} K & G \\ G^T & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} K & G \\ 0 & -S \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ -\hat{h} \end{pmatrix}, \tag{4}$$

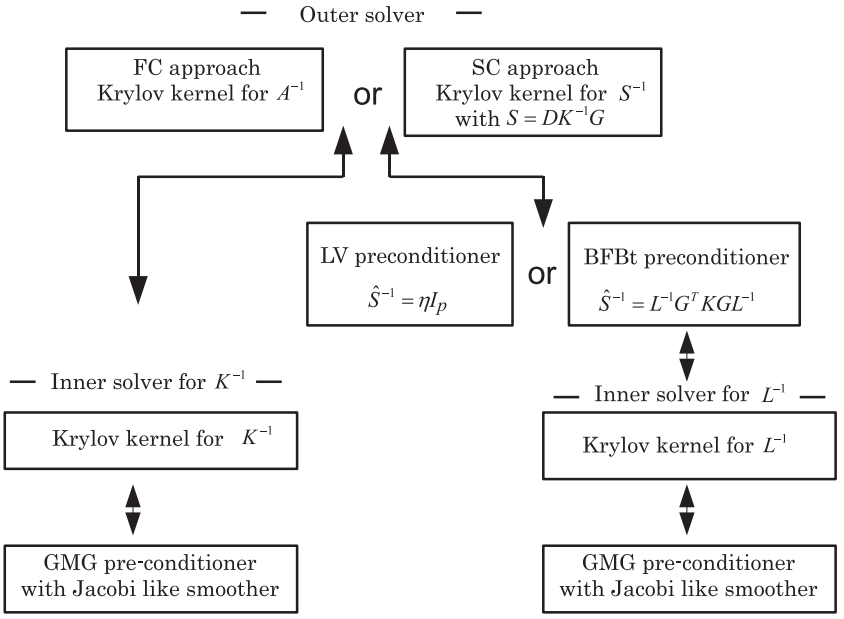

Fig. 1. Overview of total solver design of Section 3. For the sake of simplicity, only the symbolic operations are written for each module.

where the Schur complement given by

$$S = G^T K^{-1} G \in \Re^{n\times n}, \tag{5}$$

which is positive semi-definite, and

$$\hat{h} = G^T K^{-1} f - h. \tag{6}$$

In the SC method, we first solve decoupled matrix problem for the pressure

$$Sp = \hat{h}. \tag{7}$$

by the preconditioned Krylov subspace method.

As for the choice of the Krylov subspace method to solve (7), we employ the Arnoldi based method GCR (Generalized Conjugate Residual). This choice comes from the work by May and Moresi [21], in which Arnoldi type GMRES (Generalized Minimum Residual) method was demonstrated as a suitable Krylov method in comparison to Lanczos type CG (Conjugate Gradient) method for finite element discretizations when the system was solved using SC. We can expect a similar behavior when Arnoldi based methods are applied to the finite difference discretization used here.

The preconditioning strategy for Schur complement S is discussed in more detail in Section 3.2.

The Krylov subspace method to solve (7) requires a matrix–vector product of the Schur complement $y = Sx$: $(x \in \Re^n, y \in \Re^n)$ to compute a Krylov subspace $K(S,r_o) = \mathrm{Span}(r_0, Sr_0, S^2r_0, \ldots, S^{n-1}r_0)$, where $r_0 = \hat{h} - Sp_0$. Here, the explicit construction of S is expensive, since it includes the inverse matrix K^{-1}. Thus, it is preferable to calculate the matrix-vector in a matrix-free manner by the following three steps:

$$\text{compute} \quad f^* = Gx, \tag{8}$$

$$\text{solve} \quad Ku^* = f^*, \tag{9}$$

$$\text{compute} \quad y = G^T u^*. \tag{10}$$

We have also solved (9) in an iterative manner. This implies that the solve used in (7) includes the solve in (9). In this study, we refer to the solver in (7) as the 'outer solver' of the SC approach, and that used in (9) as the 'inner solver'. After solving for the pressure in (7), the velocity is obtained via solving

$$Ku = f - Gp. \tag{11}$$

The cost of obtaining an accurate velocity solution by (11) is expensive, especially for problems with a large viscosity contrast. We therefore update the velocity u at each iteration of the outer solver for Eq. (7), to find a good initial guess for the velocity to be used when we solve (11) iteratively [32]. We update the pressure in the outer solve at the Nth iteration by

$$p_{N+1} = p_N + \delta p. \tag{12}$$

Following this step, we update the velocity solution via

$$u_{N+1} = K^{-1} f - K^{-1} G(p_N + \delta p), \tag{13}$$

$$= u_N - K^{-1} G \delta p. \tag{14}$$

In this calculation, it is not necessary to solve an additional matrix problem. The second term $K^{-1}G\delta p$ of (14) is obtained during an auxiliary step required to evaluate the matrix–vector product of S (see Eqs. (8) (9)) which is required by the outer solver. Although we still need to solve (11) after we obtain a pressure solution, this technique can provide a good initial guess for the velocity, thereby decreasing the number of iterations required for convergence and thus reducing the CPU time required.

3.1.2. Fully coupled preconditioned approach (FC)

In fully coupled approach (FC), the solution is obtained by updating the velocity and pressure simultaneously with a Krylov subspace method applied directly to (3). In this study, we refer this process as an 'outer solver' of the FC approach. As for the preconditioner, we apply right preconditioning to (3),

$$A\widehat{A}^{-1}\vec{y} = \vec{b}, \quad \vec{x} = \widehat{A}^{-1}\vec{y} \tag{15}$$

with the block upper triangular preconditioner,

$$\widehat{A} = \begin{pmatrix} K & G \\ 0 & -\widehat{S} \end{pmatrix}, \tag{16}$$

where $\widehat{S}$ is the preconditioner for S. The action of the preconditioning,

$$\begin{pmatrix} z_u \\ z_p \end{pmatrix} = \widehat{A}^{-1} \begin{pmatrix} r_u \\ r_p \end{pmatrix} \tag{17}$$

is obtained as follows:

$$\text{compute} \quad z_p = -\widehat{S}^{-1} r_p, \tag{18}$$

$$\text{solve} \quad Kz_u = r_u - Gz_p. \tag{19}$$

In practice, the solution of the sub-problem in (19) is obtained using the same inner solver employed by the SC method.

3.2. Preconditioner for S

The performance of both the SC and FC approaches to solve Stokes flow problems with variable viscosity, are sensitive to the approximation of the Schur complement used. Recently, two methods have been demonstrated to be efficient preconditioning strategies for the Schur complement for problems with highly varying viscosity. We discuss these methods in the following sections.

3.2.1. Scaled BFBt preconditioner for outer solver (BFBt)

From an algebraic point of view, developing preconditioners for the Schur complement S is difficult because of the inverse matrix K^{-1} placed between the rectangular operators G^T and G. Elman et al. [24] therefore proposed an approximation consisting of two square matrices given by

$$S \approx \widehat{S} = G^T G K_p^{-1} = L K_p^{-1}, \tag{20}$$

where $K_p \in \Re^{n\times n}$ is an operator defined in the discrete pressure space satisfying

$$Z = KG - GK_p \approx 0, \tag{21}$$

where $L = (G^T G) \in \Re^{n\times n}$ is the discrete Laplacian, defined on the pressure space. Now the inverse operation of $\widehat{S}$ is simply obtained in a square matrix manner by

$$\widehat{S}^{-1} = K_p L^{-1}. \tag{22}$$

In general, it is difficult to find the operator K_p, which exactly satisfies $Z = 0$ in (18). This is the reason why $\widehat{S}$ is only an approximation of S in (17). According to the least commutator approximation [23], the operator K_p is chosen to satisfy the normal equation for GK_p of (21)

$$G^T KG - (G^T G) K_p = 0, \tag{23}$$

so as to minimize the Frobenius norm $\|Z\|_F$. The relation (23) gives

$$K_p = (G^T G)^{-1} G^T KG = L^{-1} G^T KG. \tag{24}$$

Consequently combining (22) and (24), the BFBt preconditioner proposed by Elman [22]

$$\widehat{S}^{-1} = K_p L^{-1} = L^{-1} G^T K G L^{-1}, \tag{25}$$

is obtained.

The matrix–vector product $z = \widehat{S}^{-1} r$ of this preconditioner is calculated by the following procedure:

$$\text{solve}: \ L\bar{z} = r, \tag{26}$$

$$\text{compute}: \ t_1 = G\bar{z}, \tag{27}$$

$$\text{compute}: \ t_2 = Kt_1, \tag{28}$$

$$\text{compute}: \ \bar{z} = G^T t_2, \tag{29}$$

$$\text{solve}: \ Lz = \bar{z}, \tag{30}$$

These processes require the solution of two Poisson like sub-problems for $\bar{z}$ and z. The effectiveness of the BFBt preconditioner (25) is known to depend on the size of the commutator Z. In practice, we observe that if the problem has a spatially smoothed viscosity structure, Z will be close to the zero, and using the preconditioner defined by Eq. (25) is robust. On the other hand, for a problem with a highly varying viscosity structure we find $Z \neq 0$, and the resulting preconditioner from Eq. (25) was frequently observed to yield slow convergence.

In order to improve the convergence for a non-smooth viscosity structures, we apply a symmetric block diagonal scaling,

$$\begin{pmatrix} X_u^{-1} K X_u^{-T} & X_u^{-1} G X_p^{-T} \\ X_p^{-1} G X_u^{-T} & 0 \end{pmatrix} \begin{pmatrix} X_u^T u \\ X_p^T p \end{pmatrix} = \begin{pmatrix} K_s & G_s \\ G_s & 0 \end{pmatrix} \begin{pmatrix} X_u^T u \\ X_p^T p \end{pmatrix} = \begin{pmatrix} X_u^{-1} f \\ X_p^{-1} h \end{pmatrix}, \tag{31}$$

where $X_u \in \Re^{m\times m}$ and $X_p \in \Re^{n\times n}$ are the scaling matrixes for velocity and pressure fields respectively. The scaling matrix X_u is used to normalize K, and is expressed in a diagonal matrix form

$$X_u = \mathrm{diag}(\hat{t}), \quad \hat{t}_I \equiv \sqrt{\max_J \left([K]_{IJ} \right)}, \quad \hat{t} \in \Re^m, \tag{32}$$

where $[\,\cdot\,]_{IJ}$ denotes the matrix element. As for a scaling matrix for the pressure X_p, we consider to normalize $L \in \Re^{n\times n}$. This provides constant diagonal elements

$$X_p = \mathrm{diag}(\hat{q}), \quad \hat{q}_I \equiv \frac{1}{m}\sqrt{\left(\sum_J \left[X_u^{-1} \right]_{JJ}^2 \right)(\hat{g}\cdot\hat{g})}, \quad \hat{q} \in \Re^n, \tag{33}$$

where $\hat{g}_I = \max_J \left([G]_{IJ} \right)$.

With these diagonal scaling matrices, the scaled operator of L^{-1} is defined by

$$L_s^{-1} = X_p^{-1} D X_u^{-1} X_u^{-1} G X_p^{-T}.$$

According to May and Moresi [21], row/column scaling of the relation (31) with the diagonal matrix (32) and (33) is found to be quite useful to reduce the commutation Z, even for the problems with strong viscosity variations. This fact is also confirmed in our numerical experiments using finite difference discretization (see Section 5).

3.2.2. Local viscosity diagonal matrix preconditioner (LV)

For the Stokes flow problem with a constant viscosity equal to one, it is well known that the Schur complement is spectrally equivalent to the identity operator on the pressure space I_p, therefore $\widehat{S} = I_p$ is regarded as a suitable choice for the preconditioner. In the case of a spatially variable viscosity, the choice is no longer appropriate because of the increase of the condition number of $\widehat{S}^{-1}S$ [25,26] and thus is not a suitable preconditioner to use.

In the context of finite elements, in [26] it was proven that the

$$\widehat{S} = \mathrm{diag}(\hat{\eta})^{-1} M_p \tag{34}$$

is spectrally equivalent to the Schur complement, and clustering of the eigenvalues of $\widehat{S}^{-1}S$ was numerically demonstrated by using this preconditioner. Here M_p is the pressure mass matrix and for simplicity we assume that the viscosity is piece wise constant over each element, thus $\mathrm{diag}(\hat{\eta})$ corresponds to a diagonal matrix of element defined viscosities. Some of the recent solution methods for the Stokes flow problem employ this type preconditioning scheme and demonstrate that Eq. (34) produces robust convergence for the problems with variable viscosity (e.g. [6,18,19,25,26]).

Here we adapt Eq. (34) to finite difference discretizations. This is readily achieved since the finite difference analog of the mass matrix M_p is the identity matrix I_p. The finite difference form of (34) is

$$\widehat{S} = \mathrm{diag}(\hat{\eta})^{-1} I_p, \quad \hat{\eta} \in \Re^n, \tag{35}$$

where $\mathrm{diag}(\hat{\eta})$ is a diagonal matrix with each entry corresponding to viscosity defined at the pressure node, which in our discretization is the cell viscosity. Since the operators in (35) are all diagonal, we define

$$\widehat{S}^{-1} = \hat{\eta} I_p, \tag{36}$$

which we refer to as a local viscosity (LV) Schur complement preconditioner.

Compared with the BFBt preconditioner, the action of the preconditioner (36) does not require the solution of two subproblems involving L. Therefore, the LV preconditioning has an advantage in saving memory and computational cost per iteration loop compared to the BFBt preconditioner.

3.3. Inner solver for K^{-1}

Here, we explain the solution method for the sub-problems involving K (or K_s) in (9), (11) and (19), which appear during each iteration of SC or FC. These systems are iteratively solved by a preconditioned Krylov subspace method. The choice of Krylov method and the details pertaining to the preconditioner we use are explained below.

3.3.1. Multigrid preconditioning for K^{-1}

For the purpose of dealing with high resolution meshes, a multigrid technique is employed as the preconditioner for the inner solver. Multigrid methods provide an optimal method for elliptic problems, because they ideally exhibit mesh independent convergence behavior.

The problem domains we consider can be defined via a cube, thus we can discretize our domain using a structured grid. Due to the geometric simplicity of our model, we employ a standard geometric multigrid (GMG) technique for staggered grid schemes [33]. In the GMG method, the transfer operators (restriction and interpolation) between the fine and coarse grids are defined using trilinear interpolation. Our coarse grid operators are defined by re-discretizing Stokes equations on each coarser grid level. To define the coarse grid viscosity, we first evaluate the viscous law on the finest grid, and then use tri-

linear interpolation to restrict the fine grid viscosity onto the next coarsest grid. The restriction of the viscosity field onto the coarse grids is applied recursively.

As for the cycling scheme of the multigrid method, we use V-cycle. Other cycling strategies (e.g. F or W cycle), may provide more robust and accurate solutions (e.g. [10,12–14]) for particular problems. From the point of using GMG as a fast preconditioning operation, we decided to only employ one V-cycle of multigrid per Krylov iteration.

The multigrid levels l are defined from the finest grid level l_{top} to the coarsest grid level $l_{bot} = 1$. The number of the grid levels l_{top} is chosen to satisfy $(Nx)_1 \times (Ny)_1 \times (Nz)_1 \leqslant 64$ where $(\cdot)_1$ denotes the size at the ($l_{bot} = 1$)th grid level in each direction. The number of pre-and post-smoothing iterations applied on each grid level during one V-cycle are given by $2 \times N_K(l)$ and $N_K(l)$, respectively at grid level l, where $N_K(l) = \overline{N}_K \times 2^{(ltop-l)}$. In this study, we use $\overline{N}_K = 20$.

One of the well-known difficulties of the GMG preconditioner is for the problems with discontinuous coefficients. The error component regarding the discontinuous coefficients also has a discontinuous character, which is typically not sensitive to the geometrically smooth corrections given by GMG preconditioning. In addition, such an error is slow to converge by simple smoothing methods (e.g. Jacobi or SOR type relaxation) in the direction of weakly coupling of matrix form. In practice, when the simple GMG preconditioner is applied to problems containing a jump in viscosity, the interplay between corrections from coarse grid solution and smoothing sometimes fail to reduce the error residual. Under such conditions, this may result in the residual stagnating, or in the worst case, breakdown of calculation due to residual divergence [15]. In this study, we minimize these problems by only using GMG as a preconditioner for a robust Krylov method, rather than as a standalone solver.

Other already proposed remedies for the problem regarding discontinuous profile of GMG preconditioning is to employ the Galerkin-based coarse grid operator with operator-dependent interpolation, instead of simply re-discretizing the coarse grid operator. Galerkin coarse grid operators satisfy the variational principle, which whilst theoretically say nothing about the efficiency of GMG convergence, they do guarantee more robust convergence since they produce an algebraically smooth GMG correction. As a result, combined with an efficient interpolation scheme to transfer the discontinuous information properly, a Galerkin-based GMG method may lead to an effective mutlilevel preconditioner for problems with highly variable coefficients. However, since it is non-trivial to implement such an algebraic technique for velocity problems efficiently, we did not utilize Galerkin coarser grid operators in this study. It may be worthwhile to examine the benefits of this kind of technique applied to variable viscosity problems in the future.

3.3.2. Smoothing method for K^{-1}

We employ weighted Jacobi method as the basic smoothing iteration method,

$$u_n = u_n + \omega K_d^{-1}(f - Ku_n), \tag{37}$$

where K_d, f, ω and u_n are the diagonal of K, right hand side vector, a scalar weighting factor and solution at the nth iteration step respectively. Empirically we use $\omega = 0.6$ for K^{-1} problem. The smoothing operations consume most of the computational time in our multigrid implementation, therefore the optimization of the smoothing application for specific computer architecture is imperative. This simple Jacobi like iteration is suitable for a vector machine environment such as the Earth Simulator, which is our target architecture [16]. At the coarsest grid level l_{bot} of the K^{-1} calculation, a direct solver (Gaussian elimination) is applied to obtain the solution exactly up to the double precision limit.

As we discussed in Section 3.3.1, the relaxation property of the smoother is also important for solving discontinuous problems when using a GMG preconditioner. One may consider employing stronger smoothers, such as line-relaxation, cell-relaxation, or an ILU-type method, instead of the simple point-wise relaxation scheme used here (e.g. [34]). However, even though these more complex smoothers methods may result in faster convergence, it is not clear whether they would also reduce the overall CPU time due to the additional overhead for these methods [10]. In addition, exploiting the hardware level optimization provided by vector based architectures within the implementation of these more complex algorithms is not simple. As a result, we only consider simple point wise smoothers in this study.

3.3.3. Choice of Krylov subspace method for K^{-1}

We employ the Krylov subspace method to accelerate the convergence of inner solver for K^{-1} with GMG preconditioner. Although we will not survey all of them, there are a lot of established Krylov subspace methods. In general, Krylov subspace methods are classified to two categories: one is with a Lanczos process (e.g. CG, BiCGSTAB, MINRES) and another one includes an Arnoldi process (e.g. GMRES, GCR).

In our study of inner solve for K^{-1}, we consider MINRES (Minimum Residual) and GCR as a Lanzos and Arnoldi method respectively, and compare their performances for the problem with high viscosity jump. The MINRES method is appropriate for symmetric matrix problems, in which acceleration of convergence using orthogonal Krylov subspace basis is obtained by short (three) terms recurrences. On the other hand, the GCR method can be applied not only for symmetric but also for non-symmetric matrix problems. The GCR method consists of the full recurrence calculation for the orthogonal Krylov subspace basis, and requires the storage for them. From the point of economy of memory and computational cost per iteration cycle, MINRES is preferred for a symmetric problem, and is therefore used by several authors for Stokes flow problems (e.g. [18,19,35]). On the other hand, in general, a three term recurrence algorithm is more sensitive to rounding error than a full term recurrence system [36]. Thus, even for the symmetric problems, if we are not limited by memory (as in our case), Ar-

noldi based methods (such as GCR) are sometimes preferable. This proves to be especially useful for the ill-conditioned systems which arise when a variable viscosity is introduced [21]. In our numerical experiments, we explore the performance of MINRES and GCR combined with a GMG preconditioner for K^{-1} calculation. We especially focus on the robustness of the Krylov solutions against the increasing jumping viscosity contrast.

3.3.4. Scaled system

In solving the scaled problem of $K_s u^* = f^*$ with large $\Delta\eta$, large non-diagonal terms of K_s are sometimes found, although small non-diagonal terms (ideally diagonal dominant matrix) are preferable for an iterative solver for a robust convergence behavior. In order to avoid to treat such problems directly by iterative method, our inner solve is applied to the non-scaled problem $K\left(X_u^{-T}u^*\right) = X_u f^*$ to obtain $X_u^{-T}u^*$, and then compute u^* by the scaling operation later. Although a normalization of the matrix by the block diagonal scaling X_u sometimes accelerates the convergence by the iterative solver, this inner solver treatment means that we give highest priority to robustness rather than speed of convergence.

3.4. Inner solver for L_s^{-1}

For the action of BFBt preconditioning, the problem of diagonally scaled Laplacian of L_s^{-1} should be solved twice as the sub-problem. The inner solver of L_s^{-1} is equipped with the GMG preconditioned GCR method. The cell-centered multigrid approach is employed with simple piecewise constant restriction and tri-linear interpolation (see, e.g. [33]). For the smoother of L_s^{-1} calculation, we use $\overline{N}_L = 20$ and $N_L(l) = \overline{N}_L \times 2^{(ltop-l)}$ for both of pre-and post-smoothing at each level.

The weighted Jacobbi relaxation with damping factor $\omega = 0.9$ is applied as smoother of GMG method.

4. Double–double precision arithmetic for the Krylov subspace method (DD)

Throughout Krylov subspace iterations, we observed that arithmetic rounding errors can cause the reduction of the residual to stagnate when the matrix possess entries which vary significantly. Such a scenario is encountered when solving variable viscosity Stokes flow. In order to avoid issues related to rounding error, we considered improving the sensitivity of Krylov subspace iteration to the preconditioned solutions by utilizing high precision floating point arithmetic.

At the present time, quad precision arithmetic (e.g. REAL*16 in Fortran) is supported by most computer architectures and compilers. In practice however, quad precision arithmetic is roughly 20 times slower than 64 bit (REAL*8) calculations. Thus, instead of standard quad precision arithmetic, we employ a double–double precision (DD) algorithm [28,29], in which a pair of double precision terms is used to emulate true quad precision.

The double–double method has almost the same accuracy as quad precision and is faster than normal quad arithmetic. The acceleration of the arithmetic calculation in double–double precision is provided by the computer architecture. The algorithm for double–double precision arithmetic inherently possess a high computational intensity (ratio of floating point operations to memory accesses). Algorithms with this attributed are readily accelerated by a vector register or cache memory. Such an acceleration for the double–double precision has been reported for scalar CPU architectures (e.g. Intel CPU equipped with SSE2) and GPU architectures (e.g. [37]). Some details of our implementation of arithmetic in double–double precision are given in Appendix A. A full description of the algorithm can be found in [28].

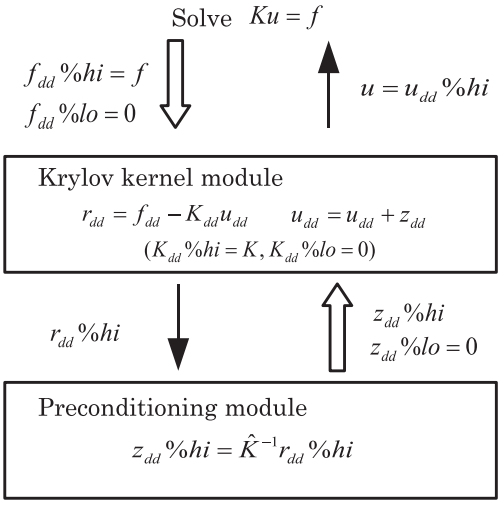

Fig. 2. Procedure for the mixed precision Krylov subspace method. For the sake of simplicity, only the symbolic operations are written for each module. Here $\hat{K}$ is the preconditioner matrix for K.

In order to minimize the usage of the expensive double–double precision arithmetic within Krylov subspace methods, we employ a mixed precision method, which is shown schematically in Fig. 2. In this mixed precision method, double–double precision arithmetic is applied throughout the Krylov subspace algorithm, except during the application of the preconditioner. This is consistent with the idea that the preconditioner is used to provide an approximation of the true solution.

In Fig. 2, we designate a double–double precision value as $(\cdot)_{dd}$. For example x_{dd} has two double precision components x_{dd} *%hi* the higher (bigger) and x_{dd} *%lo* the lower (smaller) term. In order to solve $Ku = f$, we first project the vector f from double precision to double–double precision via (f_{dd} *%hi* $= f$, f_{dd} *%lo* $= 0$). Then the algebraic operations for the Krylov subspace $K(K_{dd}, (r_{dd})_o)$, where (K_{dd} *% hi* $= K$, K_{dd} *%lo* $= 0$), are performed in double–double precision arithmetic. For the preconditioning calculation, standard double precision arithmetic is used to calculate the trial solution vector z_{dd} *%hi* from the residual vector r_{dd} *%hi*. The vector z_{dd} *%hi* is used for updating the solution u_{dd}. Once the solution u_{dd} is deemed to have converged, the Krylov method returns the higher component of the solution via $u = u_{dd}$ *%hi*.

5. Numerical experiments

Here we perform some numerical experiments to demonstrate the performance of the solution strategies discussed in Sections 3 and 4.

5.1. General remarks

5.1.1. Hardware

All the numerical experiments in this study were performed using a single processor of the Earth Simulator 2 (based on the NEC SX-9) with a maximum of 128 GByte of memory. This amount of memory is sufficient to solve Stokes problems via the strategies described here on grids of a resolution of upto $256 \times 256 \times 256$. The CPU time given in this section is reported by the FTRACE available on Earth Simulator, and includes negligible small setup time ($\sim$0.016[s]) of the Stokes flow problem.

5.1.2. Implementation of GCR method

In this test, the Arnoldi type GCR method is equipped with the maximum of 60 stored basis. In general, in order to avoid the error regarding memory size, the GCR method is equipped with restarting. As a side effect, however we sometimes observe that restarting of the GCR method avoids the problem of stagnation of convergence by the rounding error. We do not want to take into account such phenomena in our performance analysis in order to make the discussion as simple as possible. So, restarting technique of Krylov method is not used in all inner/outer solves in this study. If the Krylov iteration counts exceed maximum stored step 60, we regard the iterative process as having stagnation of convergence.

5.1.3. Stopping conditions for Schur complement reduction (SC) and Fully coupled method (FC)

For the inner solver on K^{-1}, stopping conditions are given by $\|(r_K)_n\|/\|f\| \leqslant \varepsilon^K$ and $\|(r_K)_n\|/\|(r_K)_0\| \leqslant \varepsilon_r^K$ in the SC and FC approaches respectively, where r_K is a residual and $(\cdot)_n$ is the value at nth inner iteration step. These two types of stopping conditions come from the role of K^{-1} calculation in each approach. In the SC approach, K^{-1} calculation appears in the action of Schur complement, and should be solved with same accuracy during the iterative calculation. Then we use norm of initial force vector f as an criteria of the accuracy of solution. On the other hand, in the FC approach, K^{-1} calculation appears in the block preconditioning. In this case, relative residual ε_r^K can be regarded as a solution parameter for the stopping conditions. The inner solver for L_s^{-1} in BFBt preconditioning employs the stopping condition by a relative residual, $\|(r_L)_N\|/\|(r_L)_0\| \leqslant \varepsilon^L$, where $(r_L)_0$ is the right hand side vector of the problems of (26) or (30).

Next, we consider the stopping conditions of outer solve calculation for a fair comparison between SC and FC approaches. In the FC method, the norm of the coupled residual,

$$\|\vec{r}_N\| = \left\|\vec{b} - A\vec{x}_N\right\| \leqslant \left\|\vec{b}\right\| \times \varepsilon^S \tag{38}$$

is generally used for the stopping condition, where $(\cdot)_N$ is the values at the Nth outer solve iteration. The coupled residual of the solution can be rewritten by $\vec{r}_N \equiv ((r_u)_N, (r_p)_N)$, where $(r_u)_N = f - Ku_N - Gp_N$ and $(r_p)_N = h - G^T u_N$ are residual of momentum equation (velocity residual) and continuity equation (pressure residual) respectively.

In the SC approach, the residual $(r_u)_N$ comes from inverse matrix operation K^{-1} of (11) to obtain u_N. By using a smaller stopping condition ε^K for (11), we can obtain a smaller value of the residual $(r_u)_N$. This means that the residual $(r_u)_N$ is not the essential residual in SC approach. On the other hand, the residual of outer solve (7) in the SC approach is relevant to the residual of continuity equation, since the relation (4) leads to

$$(r_S)_N = \hat{h} - Sp_N = -(r_P)_N + G^T K^{-1}(r_u)_N, \tag{39}$$

where r^S is the residual for outer solve (7). Now we assume $(r_u)_N \approx 0$ therefore $\vec{r}_N \approx (0, (r_p)_N) \approx (0, -(r_S)_N)$ by using small ε^K in the SC approach and derive the stopping condition of the Schur complement residual,

$$\|\vec{r}_N\| \approx \|(r_S)_N\| \leqslant \|\vec{b}\| \times \varepsilon^S, \tag{40}$$

which is consistent with that of FC approach. An alternative to this form stopping condition is to use $\|(r_S)_N\| \leqslant \|\hat{h}\| \times \varepsilon^S$. It however is not suitable as the tolerance to analyze the performance of the solver with different viscosity contrast problems, because the value $\|\hat{h}\|$ itself also depends on the viscosity profile.

5.2. Benchmark test for the sinking block problem (SINKER)

Let us consider to solve the sinking block problem (SINKER) [21], the schematic of which is illustrated in Fig. 3. The calculations were performed in the unit box domain with the origin at the center of the box. A cube with a viscosity $\eta_1 = \Delta\eta$ and density $\rho_1 = 1$ was placed at the middle of the domain defined by

$$-0.15 \leqslant x_1 \leqslant 0.15, \quad -0.15 \leqslant x_2 \leqslant 0.15, \quad -0.15 \leqslant x_3 \leqslant 0.15. \tag{41}$$

Material properties for the cube were defined on the finite difference grid by using the location of centroid associated with each control volume. The material surrounding the cube had the properties $\eta_0 = 1$ and $\rho_0 = 0$. The body force of the momentum equation was taken as $f = (0, 0, -\rho g)$, with $g = 1$ and the right hand side of the continuity equation was taken as $h = 0$. Along all walls on the domain, free-slip boundary conditions were employed. In this setup, the force balance equation includes a viscosity jump within one grid length. If the viscosity contrast $\Delta\eta$ is high enough, the surrounding material behaves as 'sticky air' and this is known as one of the most difficult problems to solve by iterative techniques. As we will show later, the robustness of the inner solve for K^{-1} is a critical factor in obtaining solutions for u, p via iterative methods. Then we examine the mixed precision method for the inner solve for K^{-1} in order to improve the convergence of Krylov method.

5.2.1. Performance of inner solve

First, we present the performance of inner solve of $Ku = f$ for various solution techniques. Table 1 summarizes number of inner iterations and CPU times (shown in brackets only for GCR method) to reach a given accuracy $\varepsilon^K = 10^{-6}$ for several different viscosity contrasts $\Delta\eta$, using a mesh size of $64 \times 64 \times 64$. In the MINRES calculation, the true residual $r = f - Ku_n$ is calculated at each iteration, in addition to the original algorithm so that the same stopping condition was employed by GCR. For this reason, we do not report the CPU times.

For the small viscosity contrast cases, both GCR and MINRES perform similarly, however this correlation quickly breaks down as the viscosity contrast increases and the problem becomes more ill-conditioned. From the point of view of the Krylov method, this experiment clearly supports that GCR is more robust than MINRES. It should also be emphasized that this result shows the robustness of GMG preconditioned Krylov method applied to high viscosity contrast problem. Furthermore, by using the GMG preconditioner with a robust Krylov method, the convergence behavior of the inner solve provides scalable performance with increasing problem size as shown in Table 2.

When we focus on the limitation of the inner solver for K^{-1} with respect to large $\Delta\eta$ in Table 1 again, the impact of mixed precision method of Section 4 (denoted by '+DD') is obvious for the GCR method. The Fig. 4 shows the history of the residual for the problem with $\Delta\eta = 10^4$. In the standard double precision arithmetic, additional inner GCR iterations fail to reduce the residual (stagnation of convergence) around $\|r\|/\|f\| = 10^{-3}$. On the other hands, GCR with the DD arithmetic is found to allow continued residual reduction, and avoids stagnation as occurred in the double precision calculation. It also should be noted

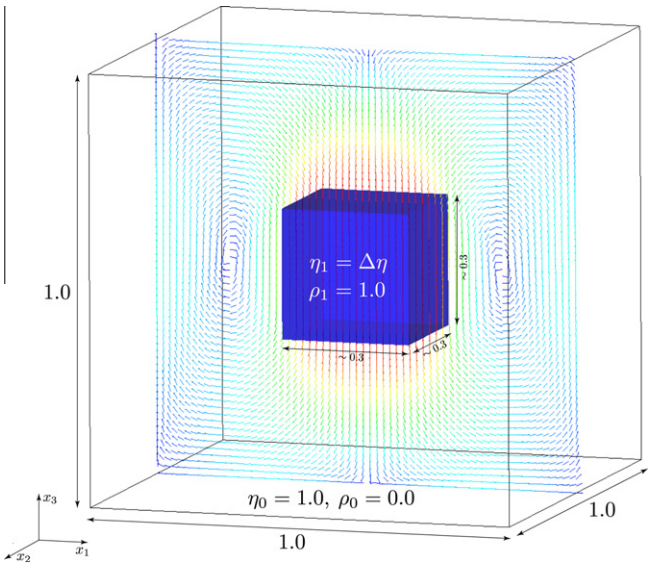

Fig. 3. Simulation setup for the 3D falling block (SINKER) problem. The vectors represent computed flow.

Table 1
Number of iterations required to solve $Ku = f$ (inner solver) for the SINKER problem with $\varepsilon^K = 10^{-6}$. The grid size used was $64 \times 64 \times 64$, see Section 5.2.1 for further details. For the GCR method, the CPU time [s] is given in brackets. An asterisk (*) is used to denote residual stagnation, which we defined as occurring if convergence did not occur within 60 Krylov iterations.

Parameters	GCR	GCR + DD	MINRES	MINRES + DD
Iterations: (*CPU time* [*s*])				
$\Delta\eta = 10^0$	3 :(0.672)	3 :(0.757)	4	4
$\Delta\eta = 10^1$	6 :(1.175)	6 :(1.357)	8	8
$\Delta\eta = 10^2$	8 :(1.507)	8 :(1.768)	*	*
$\Delta\eta = 10^3$	12 :(2.173)	12 :(2.621)	*	*
$\Delta\eta = 10^4$	*	29 :(6.69)	*	*
$\Delta\eta = 10^5$	*	49 :(12.76)	*	*
$\Delta\eta = 10^6$	*	*	*	*

Table 2
Grid size dependence of GMG preconditioned inner solve for $Ku = f$ in the SINKER problem with increasing $\Delta\eta$. Number of iterations required to reach $\varepsilon^K = 10^{-6}$ are presented.

Grid size	GCR + DD : $\Delta\eta = 10^0$	GCR + DD : $\Delta\eta = 10^2$	GCR+DD : $\Delta\eta = 10^4$	GCR+DD : $\Delta\eta = 10^5$
Iterations				
$32 \times 32 \times 32$	3	9	29	41
$64 \times 64 \times 64$	3	8	29	49
$128 \times 128 \times 128$	3	10	33	50
$256 \times 256 \times 256$	3	9	32	57

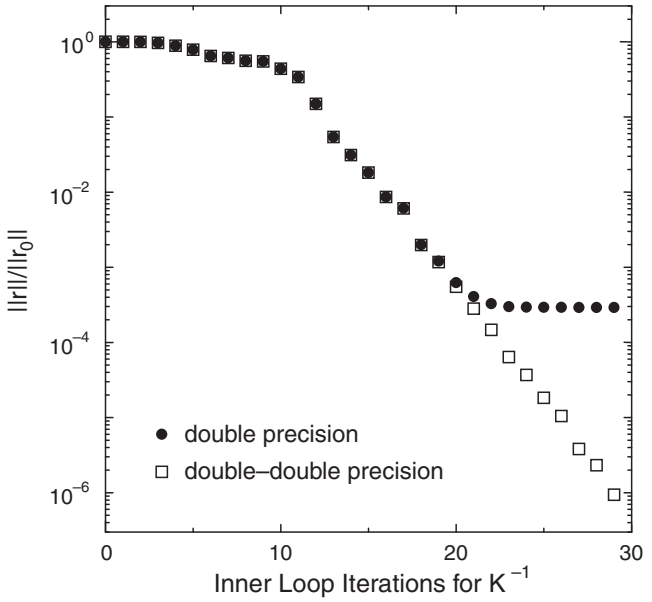

Fig. 4. Convergence history of the inner solve for K^{-1} with GCR + DD and GCR for the SINKER problem with a viscosity contrast of $\Delta\eta = 10^4$. The mesh size of this calculation was $64 \times 64 \times 64$.

that our mixed precision method does not change the observations regarding convergence of the MINRES method. The high precision calculation does not appear to improve robustness of the three terms recurrence in this problem.

Regarding the overhead related to the use of the mixed precision method, the difference in the CPU time between the GCR and GCR + DD methods is small (around 20% increase in Table 1), even though GCR + DD utilizes the more expensive double–double arithmetic. We can give the following two reasons for this small additional cost in computational time. First, by our mixed precision approach of Section 4, calculation with double–double precision is limited to within the Krylov kernel, which in itself only represents a rather small fraction of the total calculation cost. Secondly, the acceleration of arithmetic calculations required by the double–double precision is provided by the computer architecture. In order to see this phenomena, the performance of matrix–vector product calculations for double and double–double precision are shown in Table 3.

Table 3
Comparison of averaged calculation time [ms] and speed [Gflops] between double–double and double precision modules to compute $r_{dd} = f_d - K_{dd}u_d$ and $r = f - Ku$, respectively using a grid size of $64 \times 64 \times 64$.

Precision	Time [ms]	Speed [Gflops]
Double–double	10.168	45.38
Double	1.32	18.94

Although double–double arithmetic requires approximately 20 times more operations than the standard double precision, the CPU time required for the double–double precision module is less than 10 times that of the double precision module. The speed up of over a factor of two is attributed to the high computational intensity of DD arithmetic on vector computer architecture.

These observations encourage the use of GCR + DD method with GMG preconditioning as the suitable inner solver for K^{-1} in highly varying viscosity problems.

The performance of the inner solve for $L_s\bar{z} = r$, which appears in BFBt preconditioning calculation are also shown in Table 4. Our solution method for this scalar value problem gives results which are almost independent of the magnitude of the viscosity jump. We also note that solving this sub-problem twice requires significantly less time compared with that required for K^{-1} in the SINKER problem.

5.2.2. Performance of outer solve

In this subsection, we analyze the performance of the outer solver for the SINKER problem using a mesh resolution of $64 \times 64 \times 64$. Following the results of Section 5.2.1, the GCR+DD method with GMG is used as the inner solver for K^{-1}.

Firstly, we will discuss the performance by the SC approach. In Table 5, the number of SC outer iterations and CPU time (shown in brackets) required to reach satisfy our stopping condition are shown using both the BFBt and LV preconditioners.

The number of outer iterations required by the BFBt preconditioner with the tolerance parameter $\varepsilon^L = 10^{-3}$ are constant at eight for the cases from $\Delta\eta = 10^1$ to $\Delta\eta = 10^4$, indicating an almost perfect scalability with respect to the viscosity jump. In order to see the role of the diagonal scaling used within BFBt (explained in Section 3.2.1), the history of the outer solve residuals for the cases with and without scaling are presented in Fig. 5. The figure shows that the convergence rate is significantly improved by using diagonal scaling for problems with large $\Delta\eta$. This behavior is consistent with the observations made using a two dimensional finite element discretization [21].

On the other hand, the iteration counts obtained using the LV preconditioner also shows good scalability with respect to increasing viscosity contrast, although the required number of outer iterations is larger than that obtained using BFBt for varying viscosity problems.

The CPU time required to solve the problem using either SC + BFBt or SC + LV is similar for the cases in Table 5. This fact indicates several interesting features. The main difference between these two preconditioners lies on the requirement of the two inner solves for L_s^{-1} which are need by the BFBt approach. As we discussed in the previous subsection, our inner L_s^{-1} solution method shows good scalability w.r.t viscosity contrast. This means that compared with the solution obtained using the LV method, the use of BFBt preconditioning potentially has an advantage when solving problems with very large $\Delta\eta$ as the approach requires fewer outer iterations, each of which only incurs a small additional but almost $\Delta\eta$-independent cost for solving L_s^{-1} twice. In general, it is difficult to analysis this trade-off in terms of computational cost because the number of inner iterations needed to solve $Ku^* = f^*$ are different for each outer iteration. The required inner iterations depends not only on the property of the matrix K, but also on the right hand side vector f^*. In Table 5, we numerically confirm the existence of a cross over point in terms of computational cost when $(\varepsilon^S = 10^{-7}, \varepsilon^K = 10^{-7})$ with $\Delta\eta = 10^5$ Here we observe that fewer outer iterations, with an expensive inner solve for K^{-1} lead to an overall faster solution time when using BFBt in comparison to the LV preconditioner.

Next, we discuss the performance on the FC approach, in which a low accuracy solution for the inner solve K^{-1} is allowed. In order to determine the optimal choice of the solution parameter ε_r^K, we compare the performance using the SINKER problem with $\Delta\eta = 10^3$ by using two tolerance parameters $\varepsilon_r^K = 10^{-6}$ and $\varepsilon_r^K = 10^{-3}$. The results are presented in Table 6. It is observed that the appropriate choices of the tolerance parameters for effective convergence are different for the BFBt and LV preconditioners. For BFBt preconditioning, using a strict tolerance $\varepsilon_r^K = 10^{-6}$ with fewer outer iterations (i.e. fewer inner

Table 4
Number of iterations and CPU time required to solve $L_s\bar{z} = \hat{h}_s$ (inner solver for BFBt) with $\varepsilon^L = 10^{-3}$ for the SINKER problem using a grid size of $64 \times 64 \times 64$.

Parameters	GCR
Iterations :(*CPU time* [*s*])	
$\Delta\eta = 10^1$	8 :(0.269)
$\Delta\eta = 10^3$	8 :(0.264)
$\Delta\eta = 10^5$	10 :(0.281)

Table 5
Number of SC outer iterations and CPU time required to solve the SINKER problem for a grid size of $64 \times 64 \times 64$. Here an asterisk (*) indicates stagnation of convergence for the inner solver for K^{-1}. Details of the stopping condition are given in text of Section 5.1.3.

Parameters	SC + BFBt $\varepsilon^L = 10^{-3}$	SC + LV
Iterations :(*CPU time* [*s*])		
$\Delta\eta = 10^0$ $\varepsilon^S = 10^{-6}$ $\varepsilon^K = 10^{-6}$	4 :(5.41)	1 :(1.826)
$\Delta\eta = 10^1$ $\varepsilon^S = 10^{-6}$ $\varepsilon^K = 10^{-6}$	8 :(13.15)	8 :(7.61)
$\Delta\eta = 10^2$ $\varepsilon^S = 10^{-6}$ $\varepsilon^K = 10^{-6}$	8 :(17.56)	11 :(14.99)
$\Delta\eta = 10^3$ $\varepsilon^S = 10^{-6}$ $\varepsilon^K = 10^{-6}$	8 :(25.41)	12 :(24.84)
$\Delta\eta = 10^4$ $\varepsilon^S = 10^{-6}$ $\varepsilon^K = 10^{-6}$	8 :(57.88)	11 :(.57.90)
$\Delta\eta = 10^5$ $\varepsilon^S = 10^{-6}$ $\varepsilon^K = 10^{-6}$	9 :(113.86)	11 :(101.78)
$\Delta\eta = 10^5$ $\varepsilon^S = 10^{-7}$ $\varepsilon^K = 10^{-7}$	10 :(139.4)	15 :(144.97)

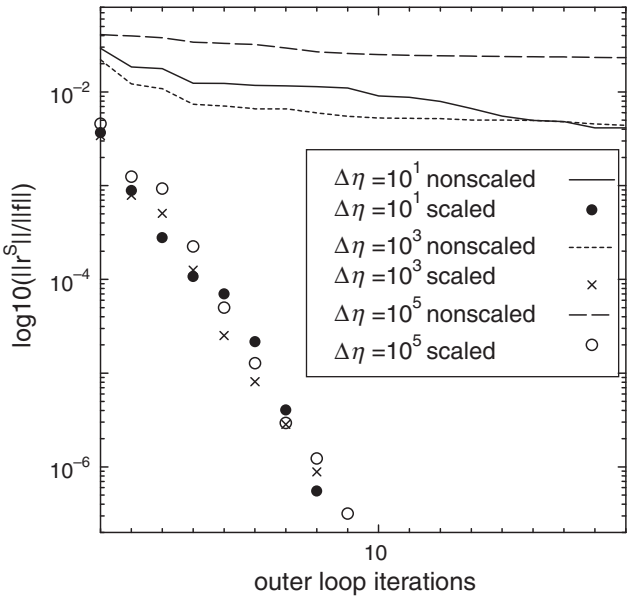

Fig. 5. Convergence history of the outer iterations of the SC approach utilizing the scaled and non-scaled BFBt preconditioner, for the SINKER problem with different viscosity contrasts $\Delta\eta$. The mesh size of this calculation was $64 \times 64 \times 64$.

Table 6
Dependence of tolerance parameters for FC outer iterations and CPU time required to solve the SINKER problem for a grid size of $64 \times 64 \times 64$ with $\Delta\eta = 10^3$. Here an asterisk (*) indicates stagnation of convergence. Stagnation was defined as having occurred if the Krylov iterations in outer solve exceeded 60.

Parameters	FC + BFBt $\varepsilon^L = 10^{-3}$	FC + LV
Iterations :(*CPU time* [*s*])		
$\varepsilon^S = 10^{-6}$ $\varepsilon^K = 10^{-6}$	9 :(24.67)	13 :(23.55)
$\varepsilon^S = 10^{-6}$ $\varepsilon^K = 10^{-3}$	37 :(71.68)	17 :(13.97)

solves for K^{-1}) produces faster result. On the contrary, using LV preconditioning method, many inner solves with a relaxed tolerance of $\varepsilon_r^K = 10^{-3}$ shows better performance w.r.t CPU time. From this observation, we use $\varepsilon_r^K = 10^{-6}$ and $\varepsilon_r^K = 10^{-3}$ as our tolerance parameters for the BFBt and LV preconditioner respectively.

Table 7 provides the performance of the two preconditioning methods used within the FC approach for the SINKER problem with various values of $\Delta\eta$. We observe that both preconditioning methods will converge with viscosity contrasts up to $\Delta\eta = 10^5$. Regardless of many outer iterations required, the combination of FC + LV exhibits the best performance in terms of total CPU time among all outer solve methods. This result indicates the advantage of using low accuracy solves for K^{-1} in the FC + LV method. In addition, we also observe that FC + LV method could solve the case $\Delta\eta = 10^{5.5}$, which FC + BFBt method could not solve. In this example, the stagnation of convergence in FC + BFBt comes from the stagnation of inner solve for K^{-1}.

Table 7
Number of FC outer iterations and CPU time required to solve the SINKER problem for a grid size of $64 \times 64 \times 64$. Here an asterisk (*) indicates stagnation of convergence for the inner solver for K^{-1}. Stagnation was defined as having occurred if the Krylov iterations in outer solve exceeded 60. Details of the stopping condition are given in text of Section 5.1.3.

Parameters	FC + BFBt $\varepsilon^K = 10^{-6}\varepsilon^L = 10^{-3}$	FC + LV $\varepsilon^K = 10^{-3}$
Iterations :(*CPU time* [*s*])		
$\Delta\eta = 10^0 \quad \varepsilon^S = 10^{-6}$	5 :(5.72)	3 :(1.812)
$\Delta\eta = 10^1 \quad \varepsilon^S = 10^{-6}$	9 :(13.78)	9 :(4.51)
$\Delta\eta = 10^2 \quad \varepsilon^S = 10^{-6}$	9 :(17.45)	14 :(7.42)
$\Delta\eta = 10^3 \quad \varepsilon^S = 10^{-6}$	9 :(24.67)	17 :(13.97)
$\Delta\eta = 10^4 \quad \varepsilon^S = 10^{-6}$	9 :(54.26)	21 :(37.60)
$\Delta\eta = 10^5 \quad \varepsilon^S = 10^{-6}$	10 :(111.16)	24 :(73.51)
$\Delta\eta = 10^{5.5} \quad \varepsilon^S = 10^{-6}$	*	18 :(85.88)

The low tolerance used in FC + LV with $\varepsilon_r^K = 10^{-3}$ helped to ensure that each the inner solve would not stagnate before the required residual was obtained.

As was found using the SC approach, the BFBt preconditioner results in fewer outer iterations in comparison to the LV preconditioner, indicating that a trade-off relation also exists when using the FC approach. However, in contrast to the SC approach, we could not find the cross over point in CPU time between these preconditioning methods. We suspect that the lack or robustness of our inner solve for K^{-1} may be preventing us from reaching such a cross over point.

5.3. Benchmark test of hot blob problem (BLOB)

In order to see the performance for the problem with higher but more diffusive variation of viscosity than SINKER problem, we deal with the rising hot blob (BLOB) problem demonstrated in [18]. We consider the computational domain of unit cube surrounded by free-slip wall boundary conditions. The body force and incompressibility are given by $f = (0, 0, \beta T)$ and $h = 0$, respectively, where the temperature field T is defined by $T = \exp\left(-\gamma\left(x_1^2 + x_2^2 + (x_3 - 0.3)^2\right)\right)$ with the constant parameters $\beta = 10^6$ and $\gamma = 200$. The temperature dependent viscosity $\eta = \exp(-\alpha T)$ is employed with the parameter for viscosity contrast α. In this subsection, the total $((\eta)_{\max}/(\eta)_{\min}$ on the discretized domain) and the locally highest viscosity contrast are denoted by $\Delta\eta_{global}$ and $\Delta\eta_{local}$ respectively. Here the local viscosity contrast $\Delta\hat{\eta} \in \Re^n$ is defined on each control volume coordinate I by $\Delta\hat{\eta}_I = \max[\hat{\eta}_{\text{neighbor cells of } I}] / \min\left[\hat{\eta}_{\text{neighbor cells of } I}\right]$, where neighbor of cells of I consists of 6 neighboring control volume cells of I and I itself. And the locally highest viscosity contrast is given by $\Delta\eta_{local} = \max_I(\Delta\hat{\eta}_I)$. This model setting produces a rising hot blob of material which we depict in Fig. 6. We solve the Stokes flow field for the BLOB problem with $\varepsilon^S = 10^{-6}$.

The inner solver for K^{-1} is equipped with GCR+DD method and is preconditioned using the GMG method. Compared with the SINKER problem, the viscosity profile of this model exhibits spatially diffusive variations. Therefore, we do not have the problem of stagnation during the inner solve for K^{-1}.

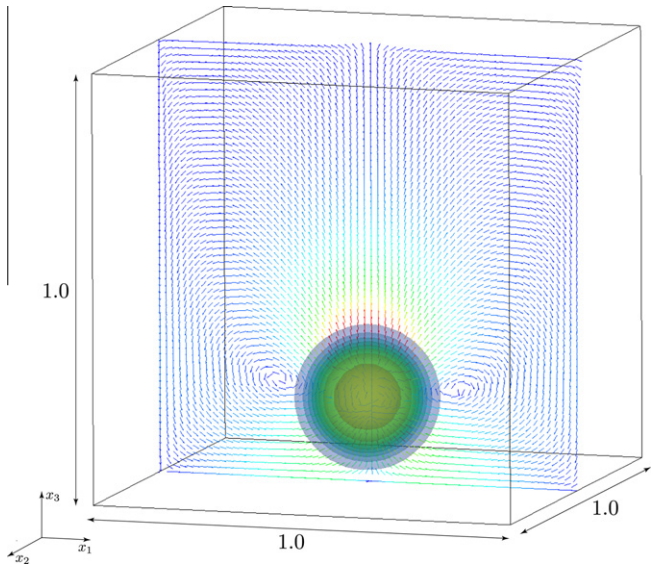

Fig. 6. Simulation setting of BLOB problem. Isosurface and vectors represent temperature field and computed flow respectively.

Table 8
Number of outer iterations and CPU time required to solve the BLOB problem for a grid size of $64 \times 64 \times 64$. Here an asterisk (*) indicates stagnation of convergence for the outer solve. Stagnation was defined as having occurred if the Krylov iterations in outer solve exceeded 60. Details of the stopping condition are given in text of Section 5.1.3.

Parameters	SC + BFBt $\varepsilon^L = 10^{-3}$	SC + LV	FC + BFBt $\varepsilon^L = 10^{-3}$	FC + LV	FC + BFBt + DD $\varepsilon^L = 10^{-3}$	FC + LV + DD
Iterations :(*CPU time* [*s*])						
$\alpha = 7.5$ $\varepsilon^S = 10^{-5}$ $\Delta\eta_{local} = 3.98e+0$ $\Delta\eta_{global} = 1.46e+3$	$\varepsilon^K = 10^{-6}$ 8 :(10.94)	$\varepsilon^K = 10^{-6}$ 12:(7.97)	$\varepsilon_r{}^K = 10^{-6}$ 9 :(11.49)	$\varepsilon_r{}^K = 10^{-3}$ 15 :(6.95)	$\varepsilon_r{}^K = 10^{-6}$ 9 :(12.45)	$\varepsilon_r{}^K = 10^{-3}$ 15 :(7.71)
$\alpha = 15.0\varepsilon^S = 10^{-5}$ $\Delta\eta_{local} = 1.58e+1$ $\Delta\eta_{global} = 2.14e+6$	$\varepsilon^K = 10^{-6}$ 13 :(15.80)	$\varepsilon^K = 10^{-6}$ 20 :(11.82)	$\varepsilon_r{}^K = 10^{-6}$ 14 :(18.32)	*	$\varepsilon_r{}^K = 10^{-6}$ 13 :(17.87)	$\varepsilon_r{}^K = 10^{-3}$ 22 :(11.13)
$\alpha = 22.5$ $\varepsilon^S = 10^{-5}$ $\Delta\eta_{local} = 6.3e+1$ $\Delta\eta_{global} = 3.11e+9$	$\varepsilon^K = 10^{-10}$ 21 :(30.66)	$\varepsilon^K = 10^{-10}$ 34 :(34.04)	$\varepsilon_r{}^K = 10^{-6}$ 20 :(24.04)	*	$\varepsilon^K = 10^{-6}$ 25 :(34.68)	*
$\alpha = 30.0$ $\varepsilon^S = 10^{-5}$ $\Delta\eta_{local} = 2.50e+2$ $\Delta\eta_{global} = 4.56e+12$	$\varepsilon^K = 10^{-14}$ 37 :(115.33)	$\varepsilon^K = 10^{-15}$ 54 :(80.97)	*	*	$\varepsilon_r{}^K = 10^{-6}$ 37 :(79.79)	*

Table 8 shows the outer iterations and CPU time required to solve the BLOB problem for different values of $\alpha$with $(\Delta\eta_{global}, \Delta\eta_{local})$. Here, the use of the mixed precision Krylov method for the outer solves are denoted by 'FC + BFBt + DD' and 'FC + LV + DD'.

First, as was the case with the SINKER problem, the trade-off relation is found between BFBt and LV preconditioners. The outer iterations of methods using the BFBt preconditioner always resulted in a smaller number than was obtained when the LV preconditioner was employed. This observation was common to both the SC and FC approaches.

From the point of robustness against α, we observe that FC approaches are more vulnerable to rounding error effects than the SC approach. Especially, we note that the FC + LV provides the worst result in terms robustness w.r.t viscosity contrast, even though it is the fastest solution method when it converges. In the case of BFBt, the extra calculation cost of applying the BFBt preconditioning seems to be justified given the robustness of the approach. Using the mixed precision within the outer solve is also found to improve the robustness of outer solve for large α. In practice, FC + BFBt + DD could solve the problem with $\alpha = 30.0$.

On the other hand, the SC approach could solve the problem up to $\alpha = 30.0$ in the double precision arithmetic without having stagnation problems. This observation supports the advantages of SC method in terms of its robustness, although it is typically slower than FC method.

Another interesting finding is that we have to use strict tolerance ε^K to satisfy the assumption $r_u \approx 0$, thereby ensuring that $r_S = -r_p$, for problems with a large α when the SC method is used. This assumption is important for a comparable performance analysis between SC and FC approaches by (40). For the small $\alpha_{problems}$, $\varepsilon^K = \varepsilon^S = 10^{-6}$ is sufficient to satisfy (38). However, when we solve the problem with $\alpha = 22.5$ and $\alpha = 30.0$ using $\varepsilon^K = 10^{-6}$, the obtained solution $\vec{x}_N$ was found to not satisfy (38). In Table 8, we indicate the tolerances required for these high αsimulations. These values were determined manually by performing several preliminary calculations with different values of ε^K and then verifying that (38) was indeed satisfied.

If convergence was achieved for a given problem with large α, the solution times are observed to be quite similar between the SC + BFBt, SC + LV and FC + BFBt + DD methods. However, when we take into account the memory required for each of the methods, we cannot justify employing the mixed precision FC + BFBt + DD method, because it requires significantly more memory than the other approaches, to obtain a similar performance to the SC approaches.

6. Concluding remarks

In this study, we examine the performance of several different types of preconditioned iterative methods to solve variable viscosity Stokes problems arising from a staggered grid, finite difference discretization. We assess the applicability of these methods by performing a series of numerical experiments in which we examine the dependence of the convergence rate and end-to-end CPU time required to obtain the solution, as a function of the viscosity contrast and mesh resolution.

For the solutions of sub-problems involving *K*, Arnoldi type Krylov method such as GCR, preconditioned with geometric multigrid (GMG) are found to be capable of solving models possessing a large, sharp viscosity contrast. The robustness of this solver was shown to be further enhanced by introducing a mixed precision arithmetic Krylov kernel. This approach consisted of combining double–double (DD) precision arithmetic for the operations matrix–vector product, dot product and norms required during a Krylov iteration, together with a standard double precision preconditioner. The collective approach was demonstrated to be scalable and robust with regards to the problem size and viscosity jump respectively.

The characteristics of the different outer solver schemes were also examined for problems involving variable viscosity. We employed a fully coupled (FC) and a decoupled (SC) approach using a local viscosity scaling (LV), or the scaled BFBt as preconditioners for the Schur complement. The performance analysis resulting from our SINKER and BLOB experiments provides some guidance as to which solver/precondition combination is appropriate to use for a given problem. In general, if the FC + LV converges for a target problem, it will invariably be the fastest method because the application of the LV preconditioner is cheap and the sub-problems involving K can be solved with a relaxed stopping condition. However, the FC + LV method is by no means the most robust solution strategy. The BFBt preconditioner was found to be a stronger preconditioner and would always yield lower iteration counts than the LV preconditioner. In addition, SC approach generally showed more robust convergence than FC approach in our BLOB experiment.

Our experiments also demonstrate that the usage of a mixed precision arithmetic using the double–double method improves the convergence of Krylov method of both inner/outer solvers over the standard double precision approach, without significantly increasing the overall calculation time. One of the merits of our mixed precision scheme is its ability to enable problems which would otherwise be rendered intractable by standard double precision arithmetic, to be solved. Moreover, if the rounding error of double–double precision arithmetic is insufficient to enable convergence for a given problems with larger viscosity contrast or larger problem sizes (very large problem size is also known to cause rounding error in Krylov subspace methods), it is straightforward to increase the precision of the arithmetic used by the Krylov kernel within the same solver design described here. In practice, arbitrary precision based on double precision arithmetic, for example the quad-double method [29], is available for such higher precision arithmetic [38].

In this study, the mesh resolution used in all out problems was limited in such that the jobs could be executed on single CPU. All of the algorithms utilized in this study are amenable to parallel implementation. We are currently implementing this solver in a multiple CPU environment. Our parallelized solver is intended to be used to study the global scale dynamics of a self consistent, coupled plate-mantle system.

Acknowledgements

We thank Masanori Kameyama, Takashi Nakagawa and Akira Kageyama for fruitful discussion and support of this study. This work was performed in the Institute of Geophysics ETH Zurich, during the research abroad program of the Japan Agency for Marine-Earth Science and Technology. All of the numerical calculations presented in this paper were performed by Earth Simulator of Japan Agency for Marine-Earth Science and Technology. Finally, we would like to express our gratitude to the nonymous reviewers of their useful comments and suggestions which helped us to improve the original version of the paper.

Appendix A

Arithmetic operations for addition and multiplication based on Dekker [39] and Kunth [40] are as follows:
Additional operation of $a = b + c$ in double–double precision

$$sh = b\%hi + c\%hi, \tag{A.1}$$
$$v = sh - b\%hi, \tag{A.2}$$
$$eh = (b\%hi - (sh - v)) + (c\%hi - v), \tag{A.3}$$
$$eh = eh + b\%lo + c\%lo, \tag{A.4}$$
$$a\%hi = sh + eh, \tag{A.5}$$
$$a\%lo = eh - (a\%hi - sh). \tag{A.6}$$

Multiplication operation of $a = b \times c$ in double–double precision

$$p1 = b\%hi^* \ c\%hi, \tag{A.7}$$
$$t = 134217729.0^* b\%hi, \tag{A.8}$$
$$ah = t - (t - b\%hi), \tag{A.9}$$
$$al = b\%hi - ah, \tag{A.10}$$
$$t = 134217729.0^* c\%hi, \tag{A.11}$$
$$bh = t - (t - c\%hi), \tag{A.12}$$
$$bl = c\%hi - bh, \tag{A.13}$$
$$p2 = ((ah^* bh - p1) + ah^* bl + al^* bh) + al^* bl, \tag{A.14}$$
$$p2 = p2 + (b\%hi^* c\%lo), \tag{A.15}$$
$$p2 = p2 + (b\%lo^* c\%hi), \tag{A.16}$$
$$a\%hi = p1 + p2, \tag{A.17}$$
$$t = a\%hi - p1, \tag{A.18}$$
$$a\%lo = (p1 - (a\%hi - t)) + (p2 - t). \tag{A.19}$$

In our implementation, we skip (A.16) when b is a double precision number ($b\%hi, 0$).

References

[1] G. Schubert, D.L. Turcotte, P. Olson, Mantle Convection in the Earth and planets, Cambridge University Press, 2001. pp. 940.
[2] R.G. Gordon, Diffuse oceanic plate boundaries: Strain rates, vertically averaged rheology, and comparisons with narrow plate boundaries and stable plate interiors, in: M.A. Richards, R.G. Gordon, R.D. Van der Hilst (Eds.), History and Dynamics of Global Plate Motions, Geophysical Monograph 121, American Geophysical Union, Washington, DC, 2000, pp. 143–159.
[3] H. Schmeling, A. Babeyko, A. Enns, C. Faccenna, F. Funiciello, T. Gerya, G. Golabek, S. Grigull, B.J.P. Kaus, G. Morra, S. Schmalholz, J. van Hunen, A benchmark comparison of spontaneous subduction models – towards a free surface, Phys. Earth Planet. Interiors 171 (2008) 198–223.
[4] M. Furuichi, M. Kameyama, A. Kageyama, Three-Dimensional Eulerian method for large deformation of viscoelastic fluid: toward plate-mantle simulation, J. Comput. Phys. 227 (2008) 4977–4997.
[5] M. Furuichi, M. Kameyama, A. Kageyama, Validity test of a Stokes flow solver by fluid rope coiling: toward plate-mantle simulation, Phys. Earth Planet. Interiors 176 (2009) 44–53.
[6] G. Stadler, M. Gurnis, C. Burstedde, L.C. Wilcox, L. Alisic, O. Ghattas, The dynamics of plate tectonics and mantle flow: from local to global scales, Science 329 (5995) (2010) 1033–1038.
[7] M.A. Jadamec, M.I. Billen, Reconciling rapid 3-D mantle flow and surface plate motions near the Eastern Alaska slab edge, Nature 465 (2010) 338–341.
[8] P.J. Tackley, 1993, Effects of strongly temperature-dependent viscosity on time-dependent, three-dimensional models of mantle convection, Geophys. Res. Lett. 20 (20) (2009) 2187–2190.
[9] P.J. Tackley, S. Xie, Stag3D: a code for modeling thermo-chemical multiphase convection in Earth's mantle, in: K.J. Bathe (Ed.), Proceedings of the Second MIT Conference on Computational Fluid and Solid Mechanics, Elsevier B.V, Amsterdam, 2003, pp. 1524–1527.
[10] P.J. Tackley, Modelling compressible mantle convection with large viscosity contrasts in a three-dimensional spherical shell using the Yin-Yang grid, Phys. Earth Planet. Interiors. 171 (2008) 7–18.
[11] L. Moresi, V. Solomatov, Numerical investigation of 2D convection with extremely large viscosity variations, Phys. Fluids 7 (1995) 2154–2162.
[12] R.A. Trompert, U. Hansen, The application of a finite volume multigrid method to three-dimensional flow problems in a highly viscous fluid with a variable viscosity, Geophys. Astrophys. Fluid Dyn. 83 (3) (1996) 261–291.
[13] C. Auth, H. Harder, Multigrid solution of convection problems with strongly variable viscosity, Geophys. Res. Lett. 137 (3) (1999) 793–804.
[14] M. Albers, A local mesh refinement multigrid method for 3-D convection problems with strongly variable viscosity, J. Comput. Phys. 160 (1) (2000) 126–150.
[15] M. Kameyama, A. Kageyama, T. Sato, Multigrid iterative algorithm using pseudo-compressibility for three-dimensional mantle convection with strongly variable viscosity, J. Comput. Phys. 206 (2005) 162–181.
[16] M. Kameyama, A multigrid-based mantle convection simulation code and its optimization to the Earth Simulator, J. Earth Simul. 4 (2005) 2–10.
[17] S. Zhong, A. McNamara, E. Tan, L. Moresi, M. Gurnis, A benchmark study on mantle convection in a 3-D spherical shell using CitcomS, Geochem. Geophys. Geosyst. 9 (2008) Q10017.
[18] C. Burstedde, O. Ghattas, G. Stadler, T. Tu, L.C. Wilcox, Parallel scalable adjoint-based adaptive solution for variable-viscosity Stokes flows, Comput. Methods Appl. Mech. Eng. 198 (2009) 1691–1700.
[19] C. Burstedde, O. Ghattas, G. Stadler, E. Tan, T. Tu, L.C. Wilcox, S. Zhong, Scalable adaptive mantle convection simulation on petascale supercomputers, in: Proceedings of ACM/IEEE SC08, 2008.
[20] S.V. Patankar, Numerical Heat Transfer and Fluid Flow, McGraw-Hill, New York, 1980.
[21] D.A. May, L. Moresi, Preconditioned iterative methods for Stokes flow problems arising in computational geodynamics, Phys. Earth Planet Interiors. 171 (2008) 33–47.
[22] H.C. Elman, Preconditioning for the steady-state Navier–Stokes equations with low viscosity, SIAM J. Sci. Comput. 20 (1996) 1299–1316.
[23] H.C. Elman, D.J. Silvester, A.J. Wathen, Finite Elements and Fast Iterative Solvers, Oxford University Press, Oxford, 2005. p. 353.
[24] H.C. Elman, V.E. Howle, J. Shadid, R. Shuttleworth, R. Tuminaro, Block preconditioners based on approximate commutators, SIAM J. Sci. Comput. 27 (5) (2006) 1651–1668.
[25] D. Silvester, H. Elman, D. Kay, A. Wathen, Efficient preconditioning of the linearized Navier–Stokes equations for incompressible flow, J. Comput. Appl. Math. 128 (2001) 261–279.
[26] P.P. Grinevich, M.A. Olishanskii, An iterative method for the Stokes-type problem with variable viscosity, SIAM J. Sci. Comput 31 (5) (2009) 3959–3978.
[27] T. Geenen, M. ur Rehman, S.P. MacLachlan, G. Segal, C. Vuik, A.P. van den Berg, W. Spakman, Scalable robust solvers for unstructured FE geodynamic modeling applications: solving the Stokes equation for models with large localized viscosity contrasts, Geochem. Geophys. Geosyst. 10 (2009) Q09002, doi:10.1029/2009GC002526.
[28] D.H. Bailey, High-Precision Software Directory. <http://www.crd.lbl.gov/~dhbailey/mpdist/>.
[29] Y. Hida, X. Li, D. Bailey, Algorithm for quad-double precision floating point arithmetic, in: Proceedings of the 15th IEEE Symposium on Computer Architecture, 2001, pp. 287–302.
[30] H. Uzawa, Iterative methods for concave programming, in: K.J. Arrow, L. Hurwicz, H. Uzawa (Eds.), Studies in Linear and Nonlinear Programming, Stanford University Press, Stanford, CA, 1958, pp. 54–165.
[31] M. Benzi, G.H. Golub, J. Liesen, Numerical solutions of saddle point problems, Acta Numer. (2005) 1–137.
[32] A. Ramage, A.J. Wathen, Iterative solution techniques for the stokes and Navier–Stokes equations, Int. J. Numer. Methods Fluids 19 (1) (1994) 67–83.
[33] U. Trottenberg, C. Oosterlee, A. Schuller, Multigrid, Academic Press, New York, 2001. p. 316.
[34] S.P. Vanka, Block-implicit multigrid solutions of Navier–Stokes equations in primitive variables, J. Comput. Phys. 65 (1986) 138–158.
[35] A.J. Wathen, D.J. Silvester, Fast iterative solution of stabilised Stokes systems I: using simple diagonal preconditioners, SIAM J. Numer. Anal. 30 (1993) 630–649.
[36] G.L.G. Sleijpen, H.A. vander Vorst, J. Modersitzki, Differences in the effects of rounding errors in Krylov solvers for symmetric indefinite linear systems, SIAM J. Matrix Anal. Appl. 22 3 (2001) 726–751.
[37] H. Kotakemori, A. Fujii, H. Hasegawa, A. Nishida, Implementation of fast quad precision operation and acceleration with SSE2 for iterative solver library, IPSJ Trans. Adv. Comput. Syst. 1(1) (2008) 75–84 (in Japanese).
[38] J.R. Shewchuk, Adaptive precision floating-point arithmetic and fast robust geometric predicates, Discrete Comput. Geom. 18 (1997) 305–363.
[39] T. Dekker, A floating-point technique for extending the available precision, Numer. Math. 18 (1971) 224–242.
[40] D. Knuth, The Art of Computer Programming, Seminumerical Algorithms, first ed., vol. 2, Addison Wesley, Reading, Massachusetts, 1998.